Type-Based Categorization of Relational Attributes

Babak Ahmadi
Fraunhofer IAIS
babak.ahmadi@iais.fraunhofer.de

Marios Hadjieleftheriou
AT&T Labs Research
marioh@research.att.com

Thomas Seidl
RWTH Aachen University
seidl@informatik.rwth-aachen.de

Divesh Srivastava
AT&T Labs Research
divesh@research.att.com

Suresh Venkatasubramanian
University of Utah
suresh@cs.utah.edu

ABSTRACT

In this work we concentrate on categorization of relational attributes based on their data type. Assuming that attribute type/characteristics are unknown or unidentifiable, we analyze and compare a variety of type-based signatures for classifying the attributes based on the semantic type of the data contained therein (e.g., router identifiers, social security numbers, email addresses). The signatures can subsequently be used for other applications as well, like clustering and index optimization/compression. This application is useful in cases where very large data collections that are generated in a distributed, ungoverned fashion end up having unknown, incomplete, inconsistent or very complex schemata and schema level meta-data. We concentrate on heuristically generating type-based attribute signatures based on both local and global computation approaches. We show experimentally that by decomposing data into q-grams and then considering signatures based on q-gram distributions, we achieve very good classification accuracy under the assumption that a large sample of the data is available for building the signatures. Then, we turn our attention to cases where a very small sample of the data is available, and hence accurately capturing the q-gram distribution of a given data type is almost impossible. We propose techniques based on dimensionality reduction and soft-clustering that exploit correlations between attributes to improve classification accuracy.

1. INTRODUCTION

Many business practices generate vast amounts of data on a daily basis. The data is often generated in a distributed fashion, spanning wide geographical areas. Data analysts need to peruse the data and perform a variety of analytical tasks for business process optimization purposes, finding current trends, predicting future developments and, generally, identifying and solving problems. It is not uncommon for businesses to maintain relational databases that contain hundreds of tables, and in the order of tens of thousands of attributes. In some cases tables and attributes are created automatically, in other cases manually, but it is not uncommon for the schema and important attribute level meta-data of such ungoverned data collections to be fuzzy, unavailable and in general overly complicated and difficult to understand. Furthermore, it is often the case that only a sample of the data is made available for analysis purposes, due to practical considerations regarding the size of the data. Therefore, robust tools for data analysis are necessary in order to understand, identify, clean and process the collected data.

As a concrete example, many business processes within AT&T generate data logs that are necessary for assessing the success or failure and the quality of the given process. Such data logs are generated, for example, from network traffic monitoring tools (e.g., ICMP, BGP), from customer care services (e.g., statistics from call logs and recorded call sessions), from the sales department, etc. To complicate matters even more, the various subsidiaries of AT&T have their own, autonomous business processes that may do data collection independently (e.g., AT&T Mobility and YellowPages.com). AT&T Labs receives several gigabytes of data feeds per day from a variety of sources within the company. The data consists of tens of thousands of relational attributes that are usually only sampled and, in many cases, only with very small sampling ratios in order to cope with the overwhelming flow of information. The main purpose of collecting the data is to distribute the feeds to various groups within the labs that will perform analytical tasks for identifying and solving problems. It is easy to see that the data analysis task becomes daunting, especially since a large portion of the data might contain inconsistencies due to diverse data collection specifications, errors and incompatibilities.

Previous work on mining the structure of the data has concentrated on quickly identifying attributes containing similar values [13]. In this work we concentrate on categorization of relational attributes based on their data type. Assuming that attribute type/characteristics are unknown or unidentifiable, we analyze and compare a variety of techniques for classifying the attributes based on the semantic type of the data contained therein (e.g., router identifiers, social security numbers, email addresses). Notice that we are not trying to identify similar attributes with respect to the actual

EDBT 2009, March 24–26, 2009, Saint Petersburg, Russia.

content of the data columns, which is an orthogonal problem. Instead, we would like to cluster attributes that contain data of similar semantic type, irrespective of how much actual data overlapping, two columns exhibit. To that end, we consider a variety of *type-based attribute signatures* and measure the ability of a signature to capture the underlying data type characteristics of an attribute. By computing the similarity of two signatures we can identify the similarity of their data types. The signatures can subsequently be used for other applications as well, like clustering and index optimization/compression. We also assume that only a *small sample of the data* contained in each attribute is available for building the signatures, and test the ability of signatures to capture data type information under varying sample sizes. The usefulness of data type categorization is self-evident. An analyst confronted with unknown/obscure data that nevertheless exhibit some structure (e.g., cryptic router, link, session, customer identifiers) can potentially identify attributes by placing them into the appropriate groups. A data exploration tool can cluster attributes by data type as a roll-up operation. A data indexing framework can create separate indexes per data type to improve performance (if the data type of the query can be identified, only the relevant indexes need to be explored).

By considering all data as strings and decomposing strings into q-grams, we can represent each attribute as a high dimensional vector in the q-gram feature space. Representing attributes as q-gram vectors is useful, since q-grams tend to follow distributions that are characteristic of the particular data. The q-gram distribution has been shown to give information about the language of the text, the underlying topic and even authorship [2, 22]. We will show experimentally that q-gram distributions can also give information about the semantic type of the data. Clearly, one can use these q-gram vector signatures directly for type-based categorization (any similarity/distance function can be used for classification purposes). But, these signatures consume large amount of space, and are shown to work poorly under very small samples, failing to capture properly the actual q-gram distribution of the attributes. Hence, we also consider a variety of techniques for reducing the size of q-gram vector signatures, that can capture underlying similarities and dissimilarities between pairs of signatures (we construct signatures using global dimensionality reduction and clustering strategies for that purpose), thus enabling accurate classification even under very small sampling ratios.

The rest of the paper is organized as follows. Section 2 presents the formal problem setting and notation. Section 3 presents a variety of signature techniques along with analysis of their utility on type-based categorization under various settings. Section 4 presents various optimizations. Section 5 presents an empirical comparison of the proposed signatures with real data. Finally, Section 6 presents related work and Section 7 concludes the paper.

2. BACKGROUND

2.1 Problem Setting

Assume that the data collection consists of a large set of relational attributes $C = \{c_1, \ldots, c_n\}$. Our goal is to associate with each attribute a signature that best describes the semantic data type of the data contained therein. Ultimately, we would like to use the signatures for classification, clustering, indexing, and other applications. Irrespective of data type, we view data entries as strings and decompose them into q-grams, representing each attribute as a high dimensional vector in the q-gram feature space. Let s be a string. We denote the set of q-grams of s with $Q_q(s)$ (e.g., $Q_3(`Babak') = \{`Bab', `aba', `bak'\}$ is the set of 3-grams of string 'Babak'). Let the known q-gram universe Q (as portrayed by the attribute samples that are available for processing) be of size $d = |Q|$. The attribute samples are represented as an $n \times d$ matrix X, where every element x_{ij} is the frequency of appearances of q-gram j in attribute i. We call the rows of X the q-vectors of attributes c_i. Depending on the size of the samples, the number of attributes, and the type of attributes, d (which is the size of the distinct union of all q-grams) can potentially be extremely large. For example, using 3-grams d could be in the order of 50^3, assuming a 50 character alphabet.

2.2 Q-qram distributions

As already mentioned the q-gram distribution conveys important information about the underlying data. Figure 1 shows the distribution of 3-grams of a variety of attributes taken from the business listings data of YellowPages.com [1], like business names and query terms. These graphs show the occurrence tallies of 3-grams in the data entries of a particular attribute on the y-axis, where the universe of 3-grams is lexicographically sorted on the x-axis (with 3-grams containing special characters having the highest priority, then numerals, then letters). We can clearly see that the q-gram distributions of different data types are completely different as expected (e.g., IP addresses and business names), but even for similar data types (like business names and search terms) the distributions exhibit certain similarities but a lot of dissimilarities as well.

We expect signatures built on q-grams to be able to convey significant information regarding the type of the data. A straightforward signature for type-based similarity is to use the q-vectors associated with the attributes. The similarity of two attributes can then be computed with respect to the set resemblance of the corresponding q-vectors (i.e., the intersection over the union of the q-gram sets). Resemblance is a measure of content similarity between attributes, and attributes of similar content naturally have similar data types. A variety of other signature alternatives fall under the same category. For example, one can use the q-vectors for computing cosine similarity (or any other set similarity measure) to similar effect [2].

In our experimental evaluation we show that for large sample sizes, q-vector signatures capture the expected q-gram distribution of a given data type accurately and result in very high classification accuracy. This is a good indication that indeed the q-gram distribution can be a good type-based similarity signature. Nevertheless, for small sample sizes, where it is harder to capture the underlying q-gram distribution accurately, more sophisticated techniques are needed in order to achieve classification of high quality. Resemblance of q-gram sets inherently cannot capture similarities and dissimilarities between attributes, mainly because it does not take advantage of hidden structure and correlations between q-grams and attributes. In that respect, it is a local

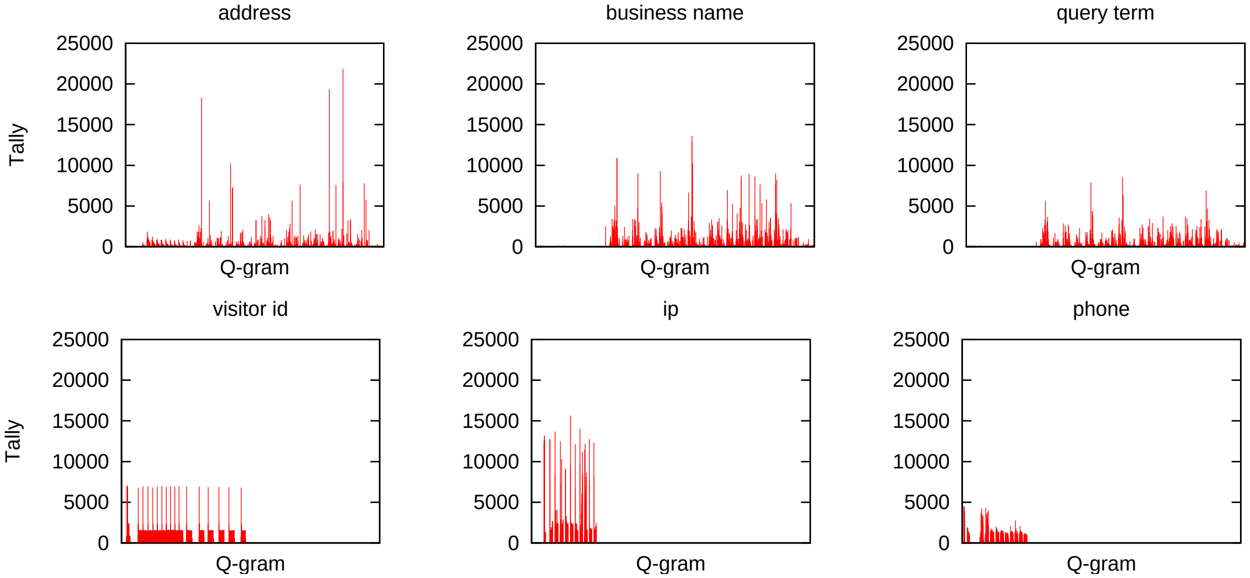

Figure 1: Q-gram distribution of various attributes.

approach that builds attribute signatures independently for each attribute. Hence, under the small sample assumption, we need to consider global approaches that take advantage of correlations between attributes. In addition, type-based categorization is context dependent. For example, one could argue that English and Chinese names belong to either the same data type for a particular application (e.g., a customer database) or very different data types in other settings (e.g., categorizing information by language). It is essential to be able to design techniques that take into account context dependent criteria as well. Global approaches are well suited for our purposes.

3. TYPE-BASED SIGNATURES

3.1 Min-hash

Since the q-gram set of an attribute can be very large, q-vectors become excessively large, and computation of similarity very expensive. To alleviate the problem one can *estimate resemblance* by using min-hash signatures [8, 5]. Min-hash signatures essentially select a random subset of the q-grams in a principled way in order to give guarantees on the estimation accuracy obtained. The resulting signatures have very small size and can be used to estimate resemblance accurately and efficiently. Computing signatures on updates (data updates as well as inserting or deleting attributes) is straightforward.

To compute a min-hash signature of size $k \ll d$, given a q-vector Q, we use k independent random permutations of set Q and keep the smallest q-gram value under each permutation. For practical purposes we can use k independent hash functions from a family of universal hash functions [6] and keep the q-gram value that correspond to the smallest hash value under each hash function. Formally, let the independent hash functions be $h_1, \ldots, h_k$. The k-dimensional min-hash signature of Q is defined as:

$$\mathcal{S} = \{min_{s \in Q} h_1(s), \ldots, min_{s \in Q} h_k(s)\}. \quad (1)$$

The signatures can be used to estimate the resemblance between two q-gram sets. Resemblance is defined as the size of the intersection of the two sets over the size of the union:

$$r(Q_1, Q_2) = \frac{|Q_1 \cap Q_2|}{|Q_1 \cup Q_2|}. \quad (2)$$

Using min-hash signatures, resemblance is estimated as a fraction of values two signatures have in common:

$$\hat{r}(Q_1, Q_2) = \frac{1}{n} \sum_{i=1}^{k} \delta_i, \delta_i = \begin{cases} 1, & \text{if } \mathcal{S}_1^i = \mathcal{S}_2^i \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

It has been shown that reliable estimates of resemblance can be obtained from such signatures [8, 5]. Min-hash enables classification of very high accuracy when sample sizes are sufficiently large, and the q-gram distribution of an attribute can be captured accurately. Nevertheless, similarly to q-vectors, performance deteriorates for very small sample sizes.

3.2 Dimensionality Reduction and PCA

By considering signatures in the vector space model we can apply any dimensionality reduction scheme for revealing potential statistical patterns in the data while reducing dimensionality and producing more compact and meaningful signatures. A simple class of dimensionality reduction techniques is based on orthogonal linear transformations. An example is Principal Component Analysis (PCA) [10]. PCA reduces the dimensionality of the data by retaining only the biggest principal components. The principal components are defined by a new coordinate system which takes as its basis the eigenvectors corresponding to the largest eigenvalues of a related covariance matrix. This coordinate system is in a sense a linear transformation of the original coordinates into a space that reflects the largest pairwise variance between dimensions on the first axis, the second largest pairwise variance on the second axis, and so on and so forth. Essentially, one can extract signatures from the data that take into consideration correlations between the original dimensions. This is a departing point from local approaches that compute attribute signatures independently. By assuming a closed data space we can compute signatures that not only reflect similarities between attributes, but also dissimilarities and, in general, hidden data trends.

We use PCA to perform dimensionality reduction on the q-vector matrix X. Clearly, there are two dimensions over which we can perform dimensionality reduction. We would like to either express each attribute as a reduced set of q-gram combinations, or represent q-gram membership into a reduced, uncorrelated set of attribute combinations. The first approach reduces the dimensionality of the q-gram space d, while the second approach reduces the dimensionality of the attribute space n.

For the first approach we use PCA as follows. First, the data in X is normalized by subtracting the column means from the data in order to center at zero. Then, the $d \times d$ covariance (or correlation) matrix $C = X'X$ is computed (where X' is the transpose of X). The eigenvectors and eigenvalues of the covariance matrix are calculated, and the k eigenvectors corresponding to the k largest eigenvalues are selected. Denote the $d \times k$ eigenvector matrix with V. The columns of V form the basis of the new coordinate system of reduced dimensionality. Now, we project the data contained in X to the new coordinate system by computing the $n \times k$ matrix $Z = XV$. We take the rows of Z to be the type-based signatures of the attributes.

For the second approach, first we normalize the data in X once again and then compute the $n \times n$ covariance matrix $C = XX'$. Then we compute the eigenvectors and eigenvalues of C and keep the k eigenvectors corresponding to the top-k eigenvalues. Denote the $n \times k$ eigenvector matrix with V. Notice that one can use the rows of the covariance matrix C as type-based signatures for the attributes. Each row of C contains the covariances of a specific data attribute with all other attributes, in a sense representing an attribute with respect to its similarities and dissimilarities with other attributes. The covariances can be used in a sense to gauge the similarity of two distributions with respect to their similarities and differences across a variety of other distributions. Furthermore, we can project the $n \times n$ covariance matrix onto the coordinate space defined by the eigenvector matrix V, to reduce the size of the matrix to its principal components. The resulting $n \times k$ matrix $Z = CV$, gives the attribute signatures in the reduced space. We refer to this version of PCA as PCA^T (for transpose). Further details about PCA are beyond the scope of this paper and can be found in [20]. Irrespective of how we compute the type-based signatures, we estimate type-based similarity with respect to the vector similarity of signatures. Any vector similarity/distance function can be used for that purpose. Examples include the Euclidean distance, cosine similarity, and Hellinger distance.

Essentially, PCA identifies the most prominent patterns, relationships among the original dimensions of the data by exposing the projections of the data with the greatest variance. The drawback of PCA is that first, computing the eigenvectors of the covariance matrix can be very expensive depending on the size of n or d, and second, in order to handle data updates all signatures need to be recomputed. One possible way of handling new attributes is to use the existing eigenvectors to compute a signature, and postpone complete recomputation until a significant amount of the data changes.

3.3 Soft-clustering and fuzzy c-means

Another approach for discovering underlying relationships in the data is clustering. In our application we would like to cluster attributes based on q-gram membership. Then, each attribute cluster represents a different data type. More specifically, we are interested in creating a signature for each attribute that conveys the probability of the attribute to belong to a particular data type. For that reason we focus our attention on soft-clustering algorithms, i.e., algorithms that allow data points to belong to multiple clusters with varying probabilities (as opposed to hard clustering techniques that associate each data point with only one cluster). A representative soft-clustering approach is fuzzy c-means [15].

Fuzzy c-means tries to assign each data point to multiple clusters with varying probabilities. The closer a point is to the centroid of a cluster, the larger the membership probability. The algorithm is initially given a target number of clusters k. Each data point is randomly assigned membership probabilities for each cluster. Let p_{ij} be the membership probability of attribute c_i to cluster j, t_j be the centroid of cluster j, and $||\cdot||$ be any distance function. The centroids of the clusters are computed as the mean of all points weighted by the membership coefficients:

$$t_j = \frac{\sum_{i=1}^{n} p_{ij}^{m} c_i}{\sum_{i=1}^{n} p_{ij}^{m}}, m \geq 1. \tag{4}$$

The algorithm follows an iterative procedure that tries to minimize an objective function based on any distance function that measures distance of points from the cluster centroids. In that respect the algorithm is very similar to k-means, and converges to a local minimum, depending on the initial distribution of points into clusters [4]. The objective function minimized is:

$$\sum_{i=1}^{n}\sum_{j=1}^{k} p_{ij}^{m} ||c_i - t_j||^2, m \geq 1. \tag{5}$$

After every iteration the algorithm recomputes the cluster centers using Equation (5), and membership probabilities using:

$$p_{ij} = \frac{1}{\sum_{l=1}^{k} \left(\frac{||c_i - t_j||}{||c_i - t_k||}\right)^{\frac{2}{m-1}}}. \tag{6}$$

Similar to PCA, when new attributes arrive, fuzzy c-means needs to be re-run to compute the new signatures. Contrary to PCA fuzzy c-means cannot assign temporary signatures to newly arriving attributes.

Let X be the $n \times d$ data matrix. By giving X as input to fuzzy c-means the algorithm will return an $n \times k$ soft-clustering matrix Z, where z_{ij} represents the probability of attribute i belonging to cluster j. We use the rows of Z as the attribute signatures. We expect attributes of similar data types to have similar cluster distributions. As before, we measure signature similarity using any vector similarity/distance function. We refer to this approach as FCM.

Alternatively, we can run fuzzy c-means on the transpose data matrix X'. The resulting $d \times k$ soft-clustering matrix Z represents the membership probabilities of q-grams into clusters. Given the initial data matrix X, the projection $Y = XZ$ yields an $n \times k$ matrix Y that in every row contains the probabilities of an attribute consisting of a mixture of q-gram clusters, with respect to the q-grams contained in that attribute. We can use the rows of Y as the type-based signatures. We refer to this approach as FCM^T.

3.4 Information bottleneck

The Information Bottleneck method (IB) [25] is a non-linear transformation approach that is related both to dimensionality reduction techniques and soft-clustering algorithms. Given the empirical joint distribution of two variables, the idea behind IB is to compress one variable (by creating clusters) in a way that best preserves mutual information. The mutual information is the *relative entropy* between a joint distribution of two variables and the product of the marginal distributions:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}, \qquad (7)$$

The entropy of a variable is a measure of uncertainty. For example, a fair coin has maximum entropy (50% heads or tails), while a Bernoulli trial with success rate $p \in \{0,1\}$ has zero entropy (no uncertainty). Mutual information measures the information that the two variables share, i.e., how much knowing one variable reduces the uncertainty about the other. If the two variables are independent, then knowing one does not give any information about the other.

In our case one variable is the data type of an attribute and the other variable is the distribution of q-grams for a particular data type. Let C be the set of attributes and Q the q-gram universe. We would like to compress C by producing clusters T, while preserving the mutual information $I(T;Q)$ as much as possible. Without compression, where $T = \{\{c_1\},\ldots,\{c_n\}\}$, clearly $I(T;Q) = I(C;Q)$. IB tries to find an optimal clustering by minimizing the reduction of $I(T;Q)$, as non-trivial clusters are formed. Using IB we get a soft-clustering where each attribute is assigned membership probabilities in clusters of T. Since every attribute is now represented as a vector of cluster membership probabilities, where cluster selection is driven by information theoretic measures, we can use these vectors as our type-based signatures. We say that attributes with similar vectors (similar cluster membership probabilities) most probably belong to similar data types, as expressed by the mutual information between attributes and q-grams. Once again, any vector similarity/distance function can be used for that purpose.

An important step of the IB algorithm is the addition of background context. Background context is essential as it helps identify structures within the data which without the addition of background noise, would remain unnoticeable. Background context is added during the algorithm, and removed afterwards. Since data are represented as distributions, appropriate backgrounds are random distributions from the space of all distributions. Further technical details with respect to the addition of background noise, depending on the application at hand, appear in [11].

Similar to PCA, IB is a global strategy that considers both similarities and dissimilarities between the data. In that respect, by clustering the data it discovers underlying structure and correlations between attributes and q-grams. Notice that inherently IB considers context dependent criteria — the diversity of attributes that are input to the algorithm significantly affect the notion of similarity/dissimilarity between attributes. For example, given an attribute that represents phone numbers, attributes corresponding to names will appear more similar to each other than phone numbers. But in the absence of a field with phone numbers, attributes containing names might as well be very dissimilar to each other. Another strength of IB is that it can produce a soft-clustering with the addition of background noise. Essentially, background noise can provide the necessary context for exposing inherent similarities and dissimilarities in the data, depending on the application at hand. Further details about IB can be found in [25]. Similarly to PCA and fuzzy c-means, IB cannot handle incremental updates. When new attributes arrive all signatures need to be recomputed.

4. OPTIMIZATIONS

The advantages of using global approaches for constructing signatures are obvious, but clearly these techniques come at a cost of increased computation cost due to their exponential nature. Here, we present an optimization that can significantly reduce computation cost, while at the same time it can potentially improve accuracy. In the foregoing discussions we have not taken into consideration the semantic properties of the q-grams. The document frequency of q-grams can potentially play a significant role in computing appropriate type-based signatures. Consider, for example, an attribute containing HTML addresses, which will have a high occurrence of q-grams from the string 'http://', or a field of IP addresses that will have high occurrence of q-grams containing numerals and dots. The existence of such q-grams alone could be enough to positively identify the data type of an attribute. Intuitively, a good type-based signature should give importance to the most frequent q-grams. For that purpose we can assign weights to q-grams, and use those weights when computing signatures. Assume that the weight of a q-gram is the Inverse Document Frequency (idf), i.e., the inverse of the number of times the q-gram appears in a given attribute (we compute q-gram idfs on a per attribute basis). Idf weights follow a Zipfian distribution with the vast majority of the q-grams having high idf and a very small minority having very low idf. The biggest advantage of restricting processing only to low idf q-grams is that by drastically reducing the input space, the algorithms become considerably faster. In addition, as was evident by our experimental evaluation, restricting the input space to the low idf q-grams helps improve accuracy by focusing on more informative q-grams, and reduce the inevitable noise present in the very high dimensional q-gram space. One could claim that the set of high idf q-grams (the rarest q-grams) could also be a good type-based signature for particular data types. This is actually true and reflected in the fact that the q-gram distribution as a whole proves to be a good classifier. Notice that the high idf q-grams essentially comprise almost 100% of the q-grams (but with

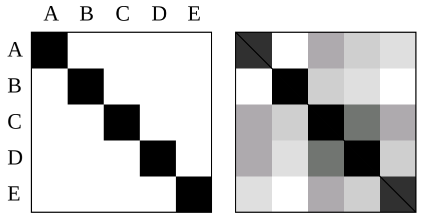

Figure 2: On the left, a 25×25 baseline matrix with 100% similarity between all pairs of attributes belonging to the 5 data types. On the right, a similarity matrix constructed using min-hash.

very low frequency). The problem with high idf signatures is that they are not robust under small sample sizes and random projections, since it is almost impossible to consistently capture the underlying q-gram distribution of a data type, given a number of attributes of a particular data type that contain highly varied data, and hence highly diverse high idf q-gram sets.

5. EXPERIMENTAL EVALUATION

5.1 Setup

For our experimental evaluation we used an Intel Xeon CPU 3.20GHz with 4 GB of RAM, running Windows Server 2003. All code was written in C++ and compiled both with g++ and VC++. We use two real datasets, the DBLP citation database [23] and the business listings (BL) from Yellow-Pages.com [1]. From the DBLP database we compiled 20 attributes containing 1000 author names each, from 5 classes of names: Chinese, Japanese, Greek, French and German. From the BL database we compiled 500 attributes containing 100000 entries each, from 5 different classes: business names, query terms, addresses, phones, and IP addresses. The two datasets are fundamentally different, since one contains only data from one basic data type (first and last names), while the other is more diverse, containing various unrelated fields.

We extracted random samples of various sizes from all columns and created signatures using the representative techniques discussed herein. For all experiments involving samples, we extract 10 independent random samples and report averages over 10 runs. Since the datasets are pre-labeled, we assume that the expected classification of columns into data types forms perfect clusters, assigning 100% similarity between all pairs of attributes from the same type. We represent the perfect expected answer using a baseline square matrix that has 100% similarity on its diagonal, for all combinations of attributes belonging to the same class (e.g., a baseline matrix of 25 attributes with 5 attributes from each class is shown in Figure 2). Then, using the attribute signatures, we compute pairwise similarities between all attributes and create a similar square matrix. We measure the accuracy of a particular signature scheme as the similarity of the computed square matrix with the baseline matrix. The similarity between the two matrices is computed using cosine similarity.

For our evaluation we vary a number of parameters. First, we vary the size of the sample drawn from each attribute, from 100% down to 1% of each column. Second, we vary the set of q-grams we consider when building the signatures. We use all q-grams extracted from the samples and only the low idf q-grams (the idf being computed as the inverse of the total number of times a q-gram appears in a particular attribute sample). In all cases, the low idf q-grams are defined as the bottom 10% q-grams when sorted by idf. We also vary the size of signatures produced, as the number of coefficients per signature. The q-vector size is the total number of distinct q-grams per sample (assuming that the q-vector matrix X is stored as a sparse matrix), which, depending on the size of the sample, can vary from a few hundred q-grams up to a few thousand (depending also on the data type of the attribute). The reduced size signatures we produce have size from 5 up to 20 coefficients, depending on the algorithm used. For PCA^T and IB the upper limit on the number of coefficients is the total number of attributes (where no dimensionality reduction is performed for PCA^T, and every attribute forms its own cluster for IB). Finally, we also vary the similarity function used, and report results using the Euclidean distance, cosine similarity and set resemblance where appropriate. All in all, we compute seven signatures per attribute sample: q-vector, min-hash, PCA and PCA^T, FCM and FCM^T, and IB.

5.2 DBLP

Table 1 lists the classification accuracy of all signature techniques with respect to sample size, using all distance measures and signatures with 10 coefficients (except for q-vector that has size equal to the size of the q-gram universe by construction; e.g., 6133 coefficients for 100% sample). The distinct number of q-grams d contained in the sample universe is shown in parentheses in the table. The data for cosine similarity only are also plotted in Figures 4 and 5. As already mentioned, q-vectors and min-hash, by capturing the q-gram distribution accurately, work fairly well for large samples but deteriorate for very small samples. Not surprisingly, the accuracy of min-hash improves drastically when only the low idf q-grams per attribute are used to create the signatures. Since min-hash is a random projection approach, it benefits from reducing the input space to more informative q-grams. Still, for small samples, accuracy does not exceed 80%. The q-vector is unaffected by the set of q-grams used and performs better than min-hash, but has very large size. The average size of the q-vector matrix in a sparse representation for various sample sizes is shown in Figure 3. In comparison, the reduced signature matrix is at most $20 \times 20 = 400$ coefficients, which is smaller than the sparse q-vector matrix in all cases.

PCA performance is similar to that of min-hash. On the other hand, PCA^T has very high accuracy for Euclidean distance and cosine similarity, irrespective of the sample size. Notice that pruning the q-gram space to the low idf q-grams does not affect classification accuracy. Nevertheless, as we will see shortly, it helps reduce processing cost significantly. PCA^T computes signatures based on the covariance matrix of the transpose co-occurrence matrix, where each signature is a projection of the covariances of an attribute with all other attributes. Clearly, this signature seems to capture similarity and dissimilarity between attributes very accurately.

FCM and FCM^T do not give accurate results for signatures with 10 coefficients and small sample sizes. It is actually the

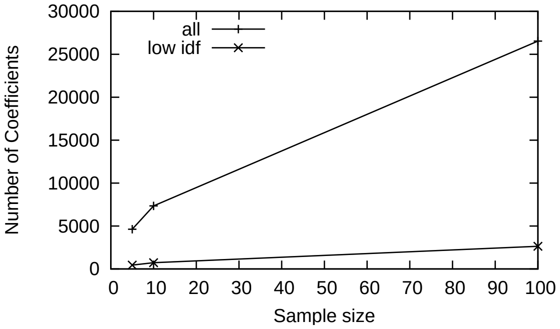

Figure 3: DBLP: Average size of the q-vector matrix for various sample sizes.

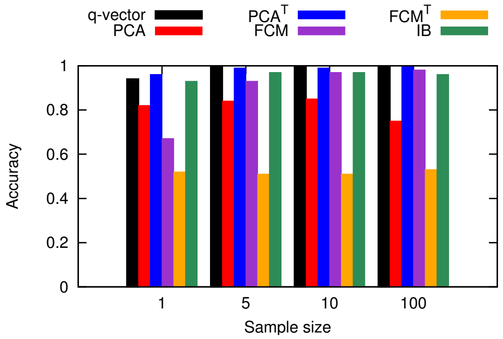

Figure 5: DBLP: Accuracy versus sample size (cosine similarity; low idf q-grams)

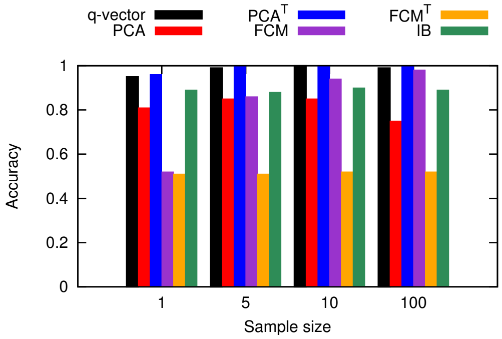

Figure 4: DBLP: Accuracy versus sample size (cosine similarity; all q-grams)

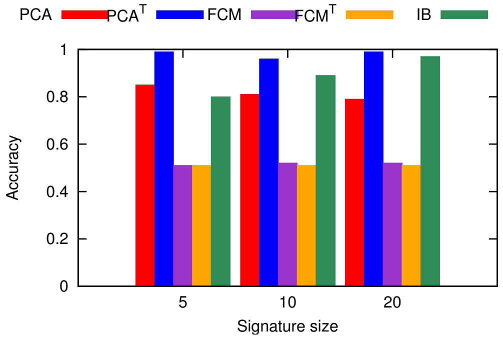

Figure 6: DBLP: Accuracy versus signature size (cosine similarity; all q-grams)

case that FCM appears to be sensitive to the target number of clusters k used (10 in this case). The closer the number of cluster is to the true number of data type attributes, the better the performance of the algorithm. Nevertheless, since we assume that the number of data types is initially unknown, this is a big disadvantage associated with this type of clustering algorithms.

Finally, we can see that IB is virtually unaffected by the reduction in the sample size, giving excellent accuracy across all metrics and exhibiting very stable behaviour. Notice that the DBLP dataset can be considered a fairly difficult dataset for classification purposes, since all the attributes essentially represent similar information. The good behaviour of IB can be attributed to the fact that it is tailored to preserve as much mutual information as possible when populating the soft-clusters. Notice also that considering only low idf q-grams has a positive impact on accuracy.

Table 2 shows the cost of constructing the signatures for varying sample sizes. The times reported here do not include the time it takes to extract the q-grams and create the 20×6133 q-vector matrix that is the input to the algorithms. This time is reported as the q-vector time in the table, for completeness. Naturally, signature computation becomes faster as the sample size becomes smaller. PCA is very expensive for the 100% sample since it has to work with a very large 6133×6133 covariance matrix. Similarly, FCM^T suffers since it has to cluster a very large number of data points. On the other hand PCA^T (which uses a 20×20 covariance matrix) is by far the fastest algorithm. IB is significantly slower than PCA^T, but as fast as other algorithms and more stable and accurate.

Finally, Table 3 shows the accuracy of the signatures as a function of signature size, for 1% samples. The same results for cosine similarity are also plotted in Figures 6 and 7. There are three interesting observations here. First, for min-hash, PCA, PCA^T, FCM and FCM^T the size of the signature has no impact on performance for small samples. Second, the performance of IB improves as the signature size increases. Third, focusing on low idf q-grams does not affect accuracy, while it has a significant, positive impact on performance.

5.3 BL

We turn our attention to a fundamentally different dataset that contains a large number of unrelated attributes. For these experiments we use 25 attributes of 5 main classes. Every attribute contains 100000 strings. Once again, we average over 10 independent runs. Table 4 shows the classification accuracy of various signatures with respect to sample size. We show only the most efficient algorithms, since due to the larger size of this dataset ($d = 8731$ for 100% samples) PCA and FCM^T did not terminate in a reasonable time.

We observe here that the accuracy of all algorithms deteriorates sharply with decreasing sample sizes when all q-grams are considered. On the other hand, when we construct signa-

Table 1: DBLP: Accuracy vs sample size (signature size: 10).

			q-vector	min-hash	PCA	PCA^T	FCM	FCM^T	IB
q-grams	size	measure							
all	100%	Euclid	0.89	-	0.78	1	0.89	0.83	0.89
(6133)		cosine	0.99	-	0.75	1	0.98	0.52	0.89
		resemblance	0.84	0.81	-	-	-	-	-
(2832)	10%	Euclid	0.87	-	0.88	0.99	0.80	0.74	0.90
		cosine	1	-	0.85	1	0.94	0.52	0.90
		resemblance	0.79	0.76	-	-	-	-	-
(2117)	5%	Euclid	0.86	-	0.87	0.99	0.79	0.71	0.90
		cosine	0.99	-	0.85	1	0.86	0.51	0.88
		resemblance	0.75	0.71	-	-	-	-	-
(876)	1%	Euclid	0.81	-	0.83	0.91	0.74	0.68	0.90
		cosine	0.95	-	0.81	0.96	0.52	0.51	0.89
		resemblance	0.64	0.66	-	-	-	-	-
low idf	100%	Euclid	0.89	-	0.78	1	0.89	0.81	0.95
(745)		cosine	1	-	0.75	1	0.98	0.53	0.96
		resemblance	0.97	0.95	-	-	-	-	-
(308)	10%	Euclid	0.87	-	0.88	0.99	0.84	0.74	0.97
		cosine	1	-	0.85	0.99	0.97	0.51	0.97
		resemblance	0.92	0.91	-	-	-	-	-
(226)	5%	Euclid	0.85	-	0.87	0.99	0.79	0.70	0.96
		cosine	1	-	0.84	0.99	0.93	0.51	0.97
		resemblance	0.89	0.86	-	-	-	-	-
(75)	1%	Euclid	0.78	-	0.81	0.95	0.57	0.66	0.93
		cosine	0.94	-	0.82	0.96	0.67	0.52	0.93
		resemblance	0.84	0.79	-	-	-	-	-

Table 2: DBLP: Construction cost (secs) vs sample size (signature size: 10).

		q-vector	min-hash	PCA	PCA^T	FCM	FCM^T	IB
q-grams	size							
all	100%	2.84	0.02	23271	0.25	12.56	108.88	9.67
	10%	0.58	0.02	505.35	0.08	3.14	25.37	6.05
	5%	0.36	0.02	175.86	0.05	5.39	1.86	4.86
	1%	0.10	0.02	8.97	0.03	0.36	0.61	2.17
low idf	100%	0.13	0.02	10.22	0.03	1.49	8.11	1.17
	10%	0.05	0.02	0.39	0.02	0.77	0.59	0.53
	5%	0.03	0.02	0.14	0.02	0.69	0.38	0.41
	1%	0.01	0.02	0.02	0.02	0.22	0.08	0.14

Table 3: DBLP: Accuracy vs signature size (sample size: 1%).

q-grams	size	measure	min-hash	PCA	PCA^T	FCM	FCM^T	IB
all	5	Euclid	-	0.88	0.98	0.78	0.68	0.82
		cosine	-	0.85	0.99	0.51	0.51	0.80
		resemblance	0.62	-	-	-	-	-
	10	Euclid	-	0.83	0.91	0.74	0.68	0.90
		cosine	-	0.81	0.96	0.52	0.51	0.89
		resemblance	0.66	-	-	-	-	-
	20	Euclid	-	0.81	0.88	0.81	0.68	0.90
		cosine	-	0.79	0.99	0.52	0.51	0.97
		resemblance	0.64	-	-	-	-	-
low idf	5	Euclid	-	0.82	0.99	0.79	0.66	0.83
		cosine	-	0.83	0.98	0.73	0.51	0.83
		resemblance	0.79	-	-	-	-	-
	10	Euclid	-	0.81	0.95	0.57	0.66	0.93
		cosine	-	0.82	0.96	0.67	0.52	0.93
		resemblance	0.79	-	-	-	-	-
	20	Euclid	-	0.80	0.95	0.58	0.66	0.86
		cosine	-	0.82	0.96	0.52	0.51	0.95
		resemblance	0.83	-	-	-	-	-

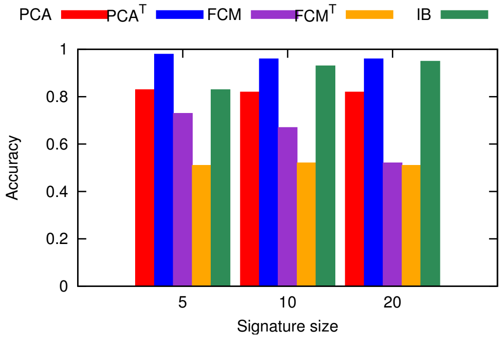

Figure 7: DBLP: Accuracy versus signature size (cosine similarity; low idf q-grams)

tures based only on the low idf q-grams, q-vector, min-hash and FCM have the same behaviour, but PCA and IB exhibit much smaller deterioration, with more than 84% accuracy in all cases.

Table 5 shows the signature construction cost as a function of sample size. Notice that concentrating on low idf q-grams only reduces costs drastically, while it also helps improve accuracy. Table 6 shows the accuracy of all signatures as a function of signature size. In this case, the bigger the signature size the better the performance, for all algorithms.

Finally, Table 7 presents a scaling experiment for increasing number of attributes. The signature size used is 10 coefficients and the sample size is 0.1%. We show scaling behaviour both for all and low idf q-grams. As expected, we can observe the exponential nature of the dimensionality reduction and clustering algorithms. Nevertheless, PCA^T, FCM and IB scale much better than PCA (which did not terminate in a reasonable amount of time for more than 100 attributes) and FCM^T. Finally, by using only low idf q-grams we can see that the algorithms scale much better for a large number of attributes, even on a low-end standalone server. FCM's performance exhibits some unexpected behaviour for low idf q-grams and 500 attributes. The algorithm requires a very large number of iterations to converge in this case on average.

5.4 Remarks

PCA^T gives excellent results and is fast to compute for small n, and when using low idf q-grams only. IB gives comparable results in all cases, especially for small sample sizes. The exponential nature of these algorithms becomes evident when scaling to large number of attributes, but restricting the input space only to low idf q-grams helps reduce the cost substantially and helps improve accuracy. Neither of these algorithms can handle incremental updates, although PCA^T can compute outdated signatures for newly arriving attributes and delay recomputation until a significant amount of the data changes. When large samples are available, min-hash signatures will work fairly well in practice, they are fast to compute and support incremental updates.

6. RELATED WORK

Database research has explored a variety of issues related to data cleaning in general, including finding duplicate values, record linkage [16, 18, 24], schema mapping [14, 7, 21], and finding primary keys and functional dependencies [19]. Work on schema matching is particularly relevant to our problem, since finding the data type of an attribute can be thought of as an instance of attribute matching. To the best of our knowledge, no techniques in schema matching literature have considered using the q-gram distribution to identify related fields. Existing work uses three primary approaches: 1. the presence of common data values; 2. existing schema information, meta-data, and available domain knowledge; 3. basic statistics over the data population, like averages and quantiles, which might work well for numerical data but will fail to give reasonable results for categorical data (like IP addresses), or for string values (like business listings). The first work to consider using data types in

Table 4: BL: Accuracy vs sample size (signature size: 10).

			q-vector	min-hash	PCA^T	FCM	IB
q-grams	size	measure					
all	1%	Euclid	0.85	-	0.98	0.91	0.86
(8731)		cosine	0.97	-	0.98	0.96	0.86
		resemblance	0.86	0.84	-	-	-
(6291)	0.1%	Euclid	0.77	-	0.92	0.59	0.74
		cosine	0.96	-	0.96	0.47	0.70
		resemblance	0.83	0.80	-	-	-
(2283)	0.01%	Euclid	0.57	-	0.72	0.64	0.70
		cosine	0.74	-	0.91	0.45	0.54
		resemblance	0.58	0.58	-	-	-
low idf	1%	Euclid	0.86	-	0.98	0.91	0.95
(1693)		cosine	0.99	-	0.98	0.98	0.95
		resemblance	0.92	0.89	-	-	-
(1210)	0.1%	Euclid	0.79	-	0.94	0.60	0.94
		cosine	0.96	-	0.96	0.56	0.94
		resemblance	0.80	0.79	-	-	-
(230)	0.01%	Euclid	0.59	-	0.85	0.65	0.89
		cosine	0.71	-	0.84	0.45	0.89
		resemblance	0.61	0.60	-	-	-

Table 5: BL: Construction cost (secs) vs sample size (signature size: 10).

		min-hash	PCA^T	FCM	IB
q-grams	size				
all	1%	0.04	0.55	22.61	19.39
	0.1%	0.02	0.29	8.05	12.74
	0.01%	0.02	0.06	1.52	1.97
low idf	1%	0.02	0.06	3.69	3.11
	0.1%	0.02	0.04	3.28	2.81
	0.01%	0.02	0.02	0.21	0.60

Table 6: BL: Accuracy vs signature size (sample size: 0.1%).

			min-hash	PCA^T	FCM	IB
q-grams	size	measure				
all	5	Euclid	-	0.98	0.59	0.65
		cosine	-	0.99	0.47	0.63
		resemblance	0.76	-	-	-
	10	Euclid	-	0.92	0.59	0.74
		cosine	-	0.96	0.47	0.70
		resemblance	0.80	-	-	-
	20	Euclid	-	0.91	0.54	0.73
		cosine	-	0.95	0.47	0.71
		resemblance	0.83	-	-	-
low idf	5	Euclid	-	0.99	0.68	0.77
		cosine	-	0.98	0.57	0.76
		resemblance	0.76	-	-	-
	10	Euclid	-	0.94	0.60	0.94
		cosine	-	0.96	0.56	0.94
		resemblance	0.79	-	-	-
	20	Euclid	-	0.93	0.45	0.93
		cosine	-	0.95	0.47	1
		resemblance	0.80	-	-	-

Table 7: BL: Construction cost (secs) vs number of attributes (signature size: 10, sample size 0.1%).

q-grams	attributes	min-hash	PCA	PCA^T	FCM	FCM^T	IB
all	100	0.06	68199.4	3.72	32.31	50.02	18.58
	200	0.11	-	15.89	71.33	53.02	52.52
	300	0.13	-	36.84	95.62	76.22	96.06
	400	0.17	-	68.91	131.45	1183.38	126.30
	500	0.20	-	118.45	144.42	2143.85	141.17
low idf	100	0.02	598.45	0.42	9.42	9.61	6.03
	200	0.03	3158.37	4.00	27.03	24.59	17.17
	300	0.03	-	8.74	46.88	47.72	31.02
	400	0.05	-	24.91	73.50	65.81	32.83
	500	0.08	-	41.78	321.56	100.39	58.55

a schema matching context is [12], where the authors construct signatures for validating rather than inferring schema matches. That work concentrates on information theoretic principles only (using IB). In the present work we consider a wider range of approaches. In that respect, the techniques presented here could prove useful in schema matching applications, and exploring this direction is left as future work.

Even though the focus has not been on type-based categorization of attributes, the problem of identifying fields containing similar data has been tackled before. In [13] the authors deal with databases that get disordered over time making it hard to understand the structure and the data contained therein. Disorder arises due to incomplete or missing documentation and meta-data, incremental changes to the database to model new structure, integration of multiple heterogeneous data sources, or simply lack of effort to correctly model the input data. The authors propose a data mining tool called Bellman for mining the structure of the database. They build various statistical summaries for determining keys in a table, finding heterogenous tables, finding (approximate) join paths between fields of different tables, and finding for the set of values of a given field another field that is textually similar or likely to be a subset of the first field. The authors also use min-hash [8, 5] for estimating attribute resemblance and q-gram sketches for substring similarity. Our work can be seen as a direct extension of Bellman for identifying the data type of attributes.

Another approach addressing the problem of finding similar columns can be found in [9]. The authors develop algorithms based on theory from association rule mining. They introduce hash-based signature schemes based on min-hash and Locality Sensitive Hashing (LSH) [17]. Locality-Sensitive Hashing (LSH) is a hashing based approach for efficient nearest-neighbour search in high dimensional spaces. The hashing functions are chosen in a way that the probability for a collision is higher for similar objects than for dissimilar ones. The goal is to distribute the objects into the different buckets such that similar objects are grouped together, i.e., located in the same or neighbouring buckets. The techniques focused on finding similar attributes that have significant data overlap. In our setting, if we view q-grams as transactions and the attributes we want to cluster correspond to itemsets, then this technique can be adapted to our problem. However, these signatures only work with 0 and 1 values, i.e., whether a q-gram appears in an attribute or not. In the case of relational attributes, two columns of different type might have a common character set and hence similar signatures. As already mentioned disregarding the q-gram distribution will hurt accuracy significantly.

The q-gram distribution has been shown to give information about the language of the text, the underlying topic and even authorship [2, 22]. Word clusters have also been used for document categorization purposes [3, 26]. For example, in [26] the authors propose a double clustering technique for clustering similar documents. First, an agglomerative information bottleneck method is used to derive word clusters from the corpus of documents. Then, each document is represented as a distribution of word clusters. Finally, a subsequent clustering step (using any hard clustering algorithm) is performed to cluster the documents using the new representation. This approach is shown to produce better quality results than existing clustering algorithms on a prelabeled dataset. The authors claim that representing documents as word clusters reduces the inevitable noise that exists in the document-word co-occurrence matrix that has very high dimensionality, similarly to the attribute-q-gram co-occurrence matrix in our scenario.

7. CONCLUSION

We analyzed and compared a variety of techniques that can be used for producing signatures for type-based similarity. We argue and show experimentally that the underlying q-gram distribution of a relational attribute forms a good quality type-based signature. If a large sample of the data attributes is available, then a simple min-hash scheme gives very good results. On the other hand, when only very small samples are available, then global strategies that compute signatures based on correlations between attributes are necessary for improving classification accuracy. Of course, the drawback of such techniques is that they cannot handle updates efficiently. In our evaluation we considered one representative algorithm from each general class of relevant algorithms, namely dimensionality reduction, soft-clustering based on partitioning, and soft-clustering based on information theoretic principles. In the future, we plan to test the utility of these signatures in a variety of applications, like schema matching and index optimization and compression.

Acknowledgements. The authors would like to thank Emmanuel Müller, Ira Assent and Ralph Krieger from RTWH

Aachen University for many useful discussions. Babak Ahmadi was partly supported by the Fraunhofer Attract project STREAM.

8. REFERENCES

[1] AT&T Inc. Business listings from YellowPages.com. proprietary.

[2] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, May 1999.

[3] L. D. Baker and A. K. McCallum. Distributional clustering of words for text classification. In *Proc. of the International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 96–103, 1998.

[4] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 1981.

[5] A. Broder. On the resemblance and containment of documents. In *Proc. of the Compression and Complexity of Sequences (SEQUENCES)*, pages 21–30, 1997.

[6] J. L. Carter and M. N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2):143–154, 1979.

[7] C. Clifton, E. Housman, and A. Rosenthal. Experience with a combined approach to attribute-matching across heterogeneous databases. In *In Proc. of the IFIP Working Conference on Database Semantics (IFIP DS)*, 1997.

[8] E. Cohen. Size-estimation framework with applications to transitive closure and reachability. *Journal of Computer and System Sciences*, 55(3):441–453, 1997.

[9] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J. D. Ullman, and C. Yang. Finding interesting associations without support pruning. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 13(1):64–78, 2001.

[10] W. Colley and P. Lohnes. *Multivariate data analysis*. John Wiley & Sons, 1971.

[11] B. T. Dai, N. Koudas, B. C. Ooi, D. Srivastava, and S. Venkatasubramanian. Rapid identification of column heterogeneity. In *Proc. of International Conference on Management of Data (ICDM)*, pages 159–170, 2006.

[12] B. T. Dai, N. Koudas, D. Srivastava, A. K. H. Tung, and S. Venkatasubramanian. Validating multi-column schema matchings by type. In *Proc. of International Conference on Data Engineering (ICDE)*, pages 120–129, 2008.

[13] T. Dasu, T. Johnson, S. Muthukrishnan, and V. Shkapenyuk. Mining database structure; or, how to build a data quality browser. In *Proc. of ACM Management of Data (SIGMOD)*, pages 240–251, 2002.

[14] A. Doan and A. Y. Halevy. Semantic-integration research in the database community. *AI Magazine*, 26(1):83–94, 2005.

[15] J. C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3:32–57, 1973.

[16] I. P. Fellegi and A. B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, 1969.

[17] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proc. of Very Large Data Bases (VLDB)*, pages 518–529, 1999.

[18] M. A. Hernández and S. J. Stolfo. Real-world data is dirty: Data cleansing and the merge/purge problem. *Data Mining and Knowledge Discovery*, 2(1), 1998.

[19] Y. Huhtala, J. Kärkkäinen, P. Porkka, and H. Toivonen. Efficient discovery of functional and approximate dependencies using partitions. In *Proc. of International Conference on Data Engineering (ICDE)*, pages 392–401, 1998.

[20] I. T. Jolliffe. *Principal Component Analysis*. Springer, second edition, October 2002.

[21] V. Kashyap and A. Sheth. Semantic and schematic similarities between database objects: a context-based approach. *The VLDB Journal*, 5(4):276–304, 1996.

[22] V. Keselj, F. Peng, N. Cercone, and C. Thomas. N-gram-based author profiles for authorship attribution. In *Proc. of the Conference Pacific Association for Computational Linguistics (PACLING)*, pages 255–264, 2003.

[23] M. Ley. DBLP database. http://dblp.uni-trier.de/xml.

[24] A. E. Monge and C. Elkan. The field matching problem: Algorithms and applications. In *Proc. of ACM Knowledge Discovery and Data Mining (SIGKDD)*, pages 267–270, 1996.

[25] N. Slonim, N. Friedman, and N. Tishby. Multivariate information bottleneck. *Neural Computation*, 18(8):1739–1789, 2006.

[26] N. Slonim and N. Tishby. Document clustering using word clusters via the information bottleneck method. In *Proc. of the International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 208–215, 2000.