WEB-BASED AFFILIATION MATCHING

(Research Paper)

David Aumueller, Erhard Rahm

University of Leipzig, Germany
{aumueller, rahm}@informatik.uni-leipzig.de

Abstract: Authors of scholarly publications state their affiliation in various forms. This kind of heterogeneity makes bibliographic analysis tasks on institutions impossible unless a comprehensive cleaning and consolidation of affiliation data is performed. We investigate automatic approaches to consolidate affiliation data to reduce manual work and support scalability of affiliation analysis. In particular, we propose to set up a reference database of affiliation strings found in publications. A key step in this task is the matching of different affiliation strings to determine whether or not they match. For affiliation matching we investigate web based similarity measures utilizing the cognitive power of current search engines. They determine the similarity of affiliations based on how the URLs in the result sets of affiliation web searches overlap. We evaluate the effectiveness of affiliation matching based on URL overlap as well as for the combined use with the Soft TF-IDF similarity measure.

Key Words: Data Quality, Information Quality, Entity Resolution, Measures, Metadata and IQ

INTRODUCTION

Authors of scholarly publications state their affiliation in highly heterogeneous ways. This heterogeneity makes bibliographic analyses on affiliation data challenging, for example to determine the total number of papers or citations of papers from specific institutions or geographic regions such as a city or country.
It would also be insightful to analyse where publications to particular research topics come from, and to determine the change over time in origins of research on certain topics. A prerequisite to such analysis is a cleaning and consolidation of affiliation data, in particular to determine different variants of the same affiliation (affiliation matching). Some bibliographic service organizations such as SCOPUS by Elsevier or ISI Web of Science perform a consolidation of affiliation data for journal papers, but presumably with a high degree of manual effort. Furthermore, they do not capture most conference and workshop papers thereby missing a large amount of research results. To reduce the manual effort and support affiliation analysis of arbitrary sets of papers we aim at a largely automatic approach to affiliation consolidation. In particular, we want to set up a reference database of cleaned affiliation data and matching variants. Such a reference database is not yet available but highly desirable as it allows retrieving a consistent version to an affiliation input string. In this paper we present methods to bootstrap the creation of such a reference database of affiliation data, in particular we present a web based affiliation matching technique.

Background

Affiliations are stated on papers in various forms denoting the same real world institution across papers. To be able to aggregate papers by single real word entities, the different variants need to be matched. Affiliation matching is a challenging case of entity resolution where duplicates include acronyms, abbreviations and multiple long forms, as well as the usual misspellings. For illustration consider the matching affiliation strings “M.I.T.” and “Massachusetts Institute of Technology”, or the various campuses of the “University of California”, e.g. “UC Santa Cruz”, or “UCLA”. A matching approach via common approximate string measures is likely to yield only unsatisfactory results. Hence, we propose the incorporation of web search results in matching the heterogeneous affiliation strings and present two

variants of the so-called URL overlap measure.

Related Work

There has been a large amount of recent work on entity resolution and approximate string matching, e.g. [5] and [7] survey current approaches. More specific work has been published on using web search results for entity resolution and data cleaning in various domains. The linkage of short to long forms of textual descriptions for real world entities (e.g. MIT as the short form for Massachusetts Institute of Technology) by querying web search engines and extracting their relationship from within the result snippets is studied in [15]. The authors link short to long form upon the co-occurrence of the other form in the snippet of the search result to the queried form. Instead of snippets we will consider the complete URLs of search results. Similarly, [6] and [15] experimented with comparisons of parts of URLs, i.e. host/domain name frequencies, for web based linkage but did not take full URLs into account. We argue that the whole URL is important for discriminating web search results in comparing query strings. For example, two given queries may both result in hits to Wikipedia – does that mean the query terms are related? Regarding only the domain name (here, for both query strings it would be wikipedia.org) we cannot infer any relatedness between the two query strings whereas a high relatedness can be assumed when the full result URLs are the same. [3] and [8] use page counts of web search results in judging the semantic similarity of two words (or queries) and adapt several measures to these counts, e.g. WebOverlap or Google Distance. [11] presents an approach about querying users about the correctness of matches. Such user feedback is common in the Web 2.0 era and could also be promising for curation of our affiliation database.

A domain attracting much work incorporating web search is the disambiguation of person names [10], especially author names [13], [16]. For instance, for ranking authors by citation counts the disambiguation is crucial not to throw multiple authors' counts together in case of common names. Not only using web search engines but any kind of secondary source is the topic of [12], with their primary example being geocoding or local/spatial search services as secondary source. We considered Google local search [9] to retrieve spatial information but observed only low coverage using our affiliation strings. Data cleaning of heterogeneous strings such as product names or affiliation strings was recently studied in [1]. The authors propose a generic rule based approach and test it on 100 strings restricted to the academic domain. We also use domain specific rules or patterns for cleaning the raw affiliation strings, executed via a scripting language instead of a cleaning framework. However, we consider a much larger dataset with a correspondingly higher degree of heterogeneity (for examples see Table 1 below).

This paper extends our preliminary work [2] on affiliation string matching that only considered a test set of 150 strings whereas the current evaluation uses a much larger dataset of roughly 20 times more instances. Furthermore, we explicitly consider the location information for affiliation matching and discuss the whole workflow for setting up a reference database of affiliations. Also, we introduce new variants of the URL overlap measure, including a binary one, and consider the results of two web search engines. The citation analysis [14] evaluated the originating institutions and countries of database publications in five venues over a period of ten years. This study relied on a manual determination and classification of the affiliation information for the first author only. Our approach aims at an automatic determination of the author affiliations to enable more comprehensive evaluations with little human effort.

Contributions

The main contributions in this paper gather around the web-based matching of affiliation strings. We present methods to establish an initial reference database of affiliation strings, including investigations into variants of a web-based similarity measure for matching and clustering affiliation string variants as well as propositions of incrementally enriching the database in a pay as you go fashion. This aims at complete coverage for looking up affiliation strings in future bibliographic evaluations.

Organization of the paper

The next section gives an overview of the workflow used to set up an initial affiliation database. For this

we investigate the matching problem of heterogeneous affiliation variants. We discuss the incremental affiliation aggregation into institution clusters which is needed to query the database in future bibliographic evaluations.

METHODS

We distinguish three main workflows for establishing, extending, and using the affiliation reference database:

Initial database setup:

data acquisition → pre-processing → matching → clustering → curation

Incremental database extension:

new instances → fuzzy look-up → add to cluster/create cluster → curation

Use of reference database:

look-up

In the initial workflow, a first version of the reference database is created based on a larger set of publications of interest. An ETL-like process (extract, transform, load) is performed to extract the affiliation strings from the papers, to pre-process and match them, and to store them in a consolidated form in the database. All variants of the same affiliations form a cluster and are stored for lookup. The second workflow performs an incremental extension of the affiliation data in the reference database for yet unmatched affiliations in a set of papers. This step is typically applied for a set of papers for which an affiliation analysis is planned to ensure that the corresponding affiliations are covered by the reference database. The final workflow utilizes the cleaned affiliation data in the reference database for lookup to retrieve consistent versions of affiliation strings of interest.
In the following we explain the main steps of each phase. The insights of the comparison of matching results lead to a one step matching+clustering approach.

Affiliation strings

For the initial version of the reference database we collected affiliation data from over 20.000 computer science papers as well as referenced publications thereof. After extracting the affiliations of all authors, the number of distinct affiliation strings amounted to about 20.000. We observed that affiliation strings are fairly unstructured, containing usually at least an institution name, often pre- or suffixed with departmental information, and usually a location, which sometimes includes the exact street address, sometimes only a city name. More than half of our variants include a location. Affiliation strings that do not mention a location often are mere acronyms or company names, e.g. “MIT”, “WPI”, “IBM Research”, or “Microsoft Research”. Further information given may include the job position of the author and/or the email address. In this paper we concentrate on the identification of the ‘main’ institutions and their location(s) on city level. Other organizational compartments such as departments, divisions, groups, and also postal or street addresses are ignored here. Observations showed that affiliation strings roughly consist of the following pattern, with asterisk denoting any repetition of the group in parenthesis and question mark denoting a single possible occurrence:

```
<compartment>(, <compartment>)*(, <location>)?
```

The order of compartments in the affiliation strings may reflect the organizational hierarchy, but sub-compartments are not consistently mentioned before or after the parent compartment. As e.g. variants having the string “Uni” or “Inst” before the string “Dep” are less prevalent than the other way round (the ratio roughly is 1:10), we can say that usually the hierarchy of compartments is reflected left to right from low to high. The following examples in Table 1 illustrate the number and order of compartments in affiliation strings, with the main institution emphasized — identified location attributes (zip, city, state, country) not counted.

a)

Brown University		
Department of Computer Science, Box 1910, ***Brown University***, *Providence, RI*	3	left to right
Brown University, Department of Computer Science, *Providence, RI*	2	*right to left*
Dept. of Computer Science, ***Brown University***, Box 1910, *Providence, RI*	3	left to right
Computer Science Department, ***Brown University***, *Providence, RI*	2	left to right

b)

Humboldt University Berlin		
Institute of Pedagogy and Informatics, Faculty of Philosophy IV, ***Humboldt University Berlin***, *10117 Berlin, Germany*	3	left to right
Institute of Information Systems, Faculty of Economics, ***Humboldt University Berlin***, *10178 Berlin, Germany*	3	left to right
Humboldt-Universität zu Berlin, *Berlin, Germany*	1	

c)

IBM Research		
Department of Computer Science, B2-250, ***IBM Almaden Research Center***, 650 Harry Road, *San Jose, CA 95120, USA*	4	left to right
IBM Research Division, Almaden Research Center, Department K53/802, 650 Harry Road, *San Jose, CA*	4	*right to left*

Table 1. Exemplary affiliation strings illustrating heterogeneity of affiliation string compartments

Pre-processing

As many of the collected affiliation strings consist of very detailed information, such as email addresses, postal addresses, job positions etc., some pre-processing has to be done prior matching for entity resolution. The goal of preprocessing is to transform an input affiliation string to the two attributes for the name of the main institution and the city name. We clean the strings using a series of heuristics, starting with removing email addresses using a regular expression pattern. Note that the domain name of the email addresses may be used as an indicator for institution or location, but neither too many of the extracted affiliation strings do contain addresses (~10 %) nor do they always reflect the institution (other email service providers).
Explanations of further heuristics follow.

Locations in affiliation strings

We use a large database of 3 million city names to extract locations from affiliation strings by mere checking whether the location name is contained within the affiliation string. To not run into ambiguities of city names we iteratively search for variations of concatenations of city, region, and country with decreasing specificity. Further we favour cities with larger populations in case of cities with the same name. That way we were able to match locations roughly to 60 % of the collection. The identified locations get stored in their own attributes, city, region, and country. The primary goal of this step, though, is not to identify a location to each affiliation string but to be able to remove the location from the affiliation strings. Note that through clustering affiliations, locations can be assigned also to strings that

do not mention one in the first place. With removing the location from the affiliation strings we remove possibly pre- or suffixed zip codes as well, using a regular expression pattern that matches digits before and after the identified location as literal. As the city name often identifies universities (city name pre- or suffixed to "University [of]") we have an exception pattern in place when removing locations such that we do not delete parts possibly identifying an institution.

Organizational compartments

The processed affiliation strings so far still may contain multiple organizational compartments (departments, divisions, groups) and addresses. After splitting affiliation strings by comma into parts returning a list of organizational compartments we apply heuristics that include key terms indicating top most compartments in the organizational hierarchy. For instance, if a compartment contains the term "University" or "Corporation" this compartment is the one of interest in our scope, i.e. the one that represents the main institution. Also, compartments containing acronyms of three or more capital letters are judged to be more important compartments, whereas the ones containing a series of digits are regarded as being of low importance (also addresses and other noise) and are removed. If no distinctive term is present, we fall back to regarding the rightmost compartment of the remaining string as most important, as the observed prevalent order of compartments suggests.

Pre-processing results

After these pre-processing steps we extracted from the affiliation strings the top-most compartment as attribute 'institution' and an optional according location, yielding e.g. "University of Leipzig" in "Leipzig, Germany" or "IBM Research Laboratory" in "San Jose, CA", "Univ. of California Riverside" in "Riverside, CA", or just "UCSC". In our test set the number of these distinct institution strings was reduced by 50 % of the original number of distinct affiliation strings; roughly 25 % of the original affiliation strings already have at least one correspondence, i.e. share the same cleaned variant. The average (and maximum) lengths of the initial strings was reduced from 80 (max. 336) to 25 (max. 63). The so achieved affiliation or organization strings could be categorized in respect to matching challenge. Examples include common substitutions (dept, department), company short forms (HP, Hewlett-Packard), added or missing parts (IBM, IBM Research), and international equivalents (university, Universität). We wanted to avoid the establishment of synonym tables for such cases realizing that web search engines that we will harvest do have such tables already in place.

Matching affiliation strings

Matching affiliation strings is a special case of matching entities/instances described by one or several attributes (e.g., institution and location). To reduce the search space for matching, so-called blocking strategies based on a predefined key are common (cf. [3], [7]), i.e. splitting the entirety of the relation into disjoint partitions. Regarding affiliation matching the search space could be restricted to blocks of same geographic regions instead of comparing each affiliation variant with each other of the whole set. However, we did not yet incorporate blocking to avoid reducing recall at the initial stage.

Common match approaches

To judge the similarity of two given strings various string measures are available to quantify the similarity via a single value, usually in the 0..1 range, with 1 being identical and 0 having no common substring. One distinguishing feature of the various string measures is how the substrings are derived that the algorithm regards as atomic. Parts are either of fixed length (based on character or n-grams) or of variable length (token based) which are derived by splitting the string at specified characters, usually punctuations. A recent comparison [5] of string distance measures identified a combination of character- and token-based measure, the Soft TF-IDF measure, as most effective in name matching tasks. Roughly, in TF-IDF two strings are treated more similar the more common tokens are counted which are weighed by frequency in each string and across the whole collection. The soft variant introduces a sub-measure for

attenuating the restriction of equality in 'common' tokens by allowing slight variations.

The URL overlap similarity measure

The URL overlap similarity measure is a new web-based approach taking web search results for each affiliation string into account. Each affiliation string is queried against a web search engine and the result sets are collected. Search results are sets of top-k ranked items containing a URL, title, snippet, etc. To determine the similarity of two affiliation strings we consider to what degree the URLs in the search result sets overlap. The rationale for similarity being, the more URLs overlap across two result sets, the more related the two (query) strings are. We consider two variations of URL overlap similarity. In the simple binary variant, which we call *simURLbase(k)*, we assume that two affiliations match if there is a non-empty URL overlap in the first k search result items; for k=1 this means the first result URL of the two affiliations must match, i.e. be identical.

simURLbase(k) = 0 for no overlap in the first k search result items, 1 otherwise (default k of 1).

The more fine-grained variant, *simURLdist(k)*, considers all overlapping URLs and a distance factor within the compared result sets. The distance is the difference between ranks of the first overlapping URL, i.e. with the lowest rank each. Normalizing the URL overlap within the range 0..1 is achieved by dividing the size of the URL overlap ω by the maximum number k of retrieved URLs, i.e. similarity = ω / k. The inverse of the distance (1/δ) is added to this value. Instead of using the harmonic mean of the two numbers to normalize the similarity value into range 0..1 we incorporate weighting factors. In this measure we propose to judge the number of overlapping URLs more important than the distance between the first overlapping pair which is reflected by weighing the first twice as important as the latter by default.

simURLdist(k) = [α · (**ω** / *k*) + β / (1 + δ)] / (α + β), whereby …

ω: number of overlapping URLs,
α, β: weighting factors (default of 2 and 1 respectively),
δ: distance between minimum rank of first overlapping URL, i.e. min(rank of URL u in result set for query string1) – min(rank of URL u in result set for query string2),
k: number of retrieved/examined top-k search result items

For example, consider three strings A, B, C, to which each web search result yields 3 items, i.e. URLs (Table 2):

a)

	web search results		
rank	**A**	**B**	**C**
1	url1	url1	*url2*
2	*url2*	url4	**url3**
3	**url3**	*url2*	url5

b)

	URLbase(k) with given k		
k	**(A, B)**	**(A, C)**	**(B, C)**
1	1	0	0
2	1	1	0
3	1	1	1

c)

	simURLdist(3) with weighting factors α=2, β=1		
	(A, B)	**(A, C)**	**(B, C)**
overlap ω	**2**	**2**	**1**
distance δ	**0**	**1**	**2**
	[α·(**2**/3) + β/(1+**0**)] / (α+β)	[α·(**2**/3) + β/(1+**1**)] / (α+β)	[α·(**1**/3) + β/(1+**2**)] / (α+β)
similarity	(4/3+1/1)/3=(7/3)/3= 7/9 ≈ **0.78**	(4/3+1/2)/3=(9/6)/3= 1/2 = **0.5**	(2/3+1/3)/3 = 1/3 ≈ **0.34**

Table 2. Illustrating c) simURLdist(k) and b) simURLbase(k) for a given web search result (a)

The number of possible values for simURLdist(k) is limited by the value of k which influences both the possible number of overlapping URLs and the possible rank differences. Instead of taking the rank differences into account we could have used a standard set overlap measure for result URLs, such as Jaccard. However, we would then not use the search engine's knowledge which typically ranks results higher which are more similar or more related to the queried terms (here, affiliation string). Using a simple Jaccard measure would result in the same similarity for matching (A,B) and (A,C) in the example because both pairs have an overlap of two URLs. By contrast, our URL overlap measure returns a better match for (A,B) which share the same first result URL.

Matching result evaluation

For being able to evaluate automatic approaches of creating a reference database we first need at least a subset of affiliation string clusters known to be correct. For our gold standard we used the last ten years of SIGMOD affiliations and extended this set by interactively matching random affiliation strings from our 20k data set. That way we set up a test database of 2450 variants clustered into 670 institutions with locations (many institutions maintain multiple locations). In our test set most (90 %) affiliation strings contain locations that we were able to identify also using the above mentioned pre-processing. Two attributes are used for matching: the extracted institution compartment and the extracted location. As location extraction was done via mention matching across a location database, we match this attribute using a binary measure on string equality only. If no location is present for an institution (not extracted from affiliation string), we cannot restrict its correspondences on location, thus it may match with any other string (left to other matcher). The other measures we experimented with are Soft TF-IDF and URL overlap for matching the institution compartment. To determine which combination of matching approaches is most promising for automatically setting up an affiliation database we evaluated the different settings in terms of precision, recall, and F-measure. With precision the reliability of the found correspondences are judged whereas by recall the share of real correspondences that is found is nominated. F-measure is in its most common variant the harmonic mean between both precision and recall. For each experiment we count the true positives, i.e. correctly identified correspondences, as well as the false positives, i.e. false correspondences, derived in comparison to the manually established mappings in our test set. The measures are calculated as follows.

precision	= |true positives| / (|true positives| + |false positives|)
recall	= |true positives| / |real correspondences|
F-measure	= 2 · precision · recall / (precision + recall)

Soft TF-IDF, URL overlap, and combinations

We used soft TF-IDF for matching the pre-processed affiliation strings (institutions). For the URL overlap matcher we concatenated the two attributes institution and location if available. For the combination of matchers we consider the union (and intersection) of the matching results of the single matchers, i.e. we either count a match as positive when both (or either one of the) match similarities are above an individual threshold (respectively). Although location was present already in the query string for the URL overlap measure, we investigate on combinations with exact location matching as well, as web search results do not need to be that restrictive by themselves. As search engines we queried both Google and Yahoo using their API (see [9], [16]), retrieving the maximum of items per request provided, which in the case of Google is 8, in the case of Yahoo 50. Thus, the time needed for gathering web search results depends directly on the number of distinct strings to compare. For comparison of search engines and to test the simple URL overlap measure simURLbase(k), we run evaluation test series taking only the first retrieved item into account. Results are shown in Table 3 below.

	Similarity measure(s)	thresholds t1×t2	precision	recall	F-measure
1	Soft TF-IDF	0.7	28.6 %	27.2 %	27.9 %
2	Soft TF-IDF, same location	0.5 × 1	84.0 %	30.0 %	44.2 %
3	simURLbase(1), same location	1 × 1	93.9 %	61.9 %	74.6 %
4	simURLbase(1) Google	1	93.0 %	70.6 %	**80.3 %**
5	simURLbase(2) Google	1	88.2 %	74.7 %	80.9 %
6	simURLbase(3) Google	1	82.6 %	77.8 %	80.1 %
7	simURLbase(4) Google	1	80.6 %	80.0 %	80.3 %
8	simURLbase(8) Google	1	73.8 %	84.0 %	78.6 %
9	simURLdist(8) Google	0,3	89.5 %	77.7 %	**83.2 %**
10	simURLdist(8) Google, same location	0 × 1	86.1 %	64.0 %	73.4 %
11	simURLdist(3) Google	0,5	91.3 %	73.4 %	81.4 %
12	simURLbase(1) Yahoo	1	94.7 %	61.9 %	74.9 %
13	simURLbase(1), same location	1 × 1	96.0 %	54.6 %	69.5 %
14	simURLdist(8) Yahoo	0.2	86.6 %	73.6 %	79.6 %
15	simURLdist(50) Yahoo	0.2	87.4 %	73.6 %	79.9 %
16	simURLdist(50) Yahoo, same location	0.1 × 1	85.8 %	70.8 %	77.6 %
17	1 union 4	0.5 × 1	95.6 %	58.7 %	72.7 %
18	2 union 9	0.9 × 0.3	87.1 %	78.2 %	**82.4 %**
19	2 intersection 9	0.5 × 0.1	77.5 %	35.9 %	49.1 %
20	2 union 10	0.9 × 0.3	91.2 %	68.7 %	78.4 %
21	2 union 16	0.8 × 0.1	84.8 %	72.1 %	78.0 %
22	cross product union same location	0 × 1	23.3 %	87.9 %	36.8 %
23	same location only	1	60.1 %	73.6 %	66.2 %

Table 3. Thresholds, precision, recall, and F-measure for highest F-measure per (combined) measure

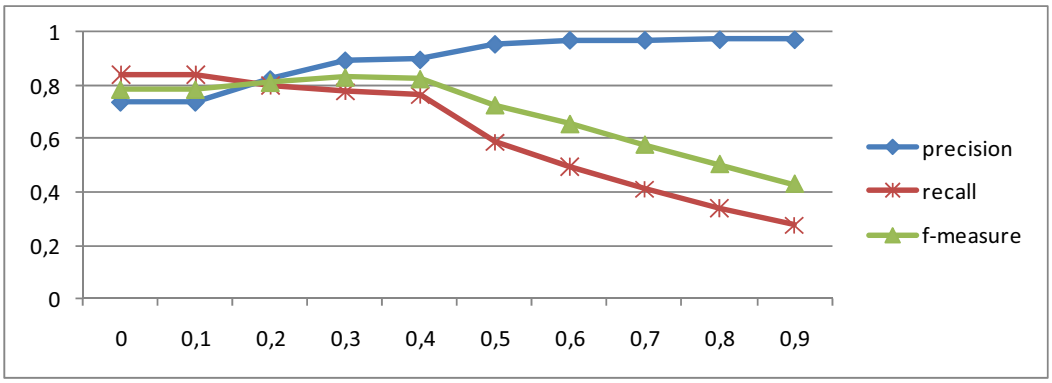

Figure 1. The URL overlap measure with increasing threshold for series 9 (simURLdist(8) Google)

Figure 1 shows precision, recall, and F-measure of URL overlap similarity as of test series 9 (simURLdist(8) Google) with increasing threshold on the x-axis. Maximum F-measure was achieved at a threshold of 0.3.

Discussion

The distance-based URL overlap measure alone, simURLdist(8), achieved the highest value for F-measure of 83% which is considered a very good result. Interestingly, restricting the approach to the very first item of the web search results, simURLbase(1) measure, achieves already an F-measure of 80%. The advantage of this measure is its simplicity and ease of calculation. In particular, there is no need to find a suitable similarity threshold as opposed to simURLdist and illustrated by Figure 1. Taking more than k=8 URLs into account does not significantly affect results, as illustrated by the Yahoo test series 12 and 13. Use of Google search results consistently outperformed the use of Yahoo for both the simURLbase and simURLdist measures.

Soft TF-IDF performed poorly and did not exceed 44% F-measure. This is on the one hand due to the heterogeneity of affiliation variants, e.g. “MIT” and “Massachusetts Institute of Technology” do not at all correspond in (Soft) TF-IDF matching (threshold>0), but also on the other hand because of ambiguous institution names such as “Northeastern University” (which refers not only to the NEU in Boston, MA, but to many others[1]), or simply to similar names of various universities, e.g. “Univ. of Torino” and “Univ. of Toronto”, or “MIT” and “RMIT”.

The considered combinations of Soft TF-IDF and URL overlap did not improve F-measure, but can achieve the highest single recall or precision values (max. recall of 88.9% in test 16, max. precision of 99.8 % in test 18; note, these threshold combinations are not shown in the table above).

Clustering affiliation strings

To support a lookup for all affiliation variants we keep clusters of matching variants in the reference database. One selected and cleaned variant is treated as a representative of an affiliation (cluster) and typically includes the location information which can be determined as long as one affiliation string variant mentions it.

The clusters could be derived in a separate merge step after all match correspondences between affiliations have been determined. Instead, we determine an initial set of clusters together with the matching in one go. This is especially feasible for the simple URL overlap measure simURLbase(1). We simply group all affiliations on the first returned URL of their web search results to obtain clusters of all matching affiliations. That way, each affiliation string is only clustered into one cluster. While the result is not yet perfect (~80% F-measure, cf. Table 1) it is a good first clustering for refinement by merging

[1] http://en.wikipedia.org/wiki/Northeastern_University_(disambiguation)

clusters or removing unfitting variants. Following the previous example of strings A, B, C of Table 2, we end up with two disjoint clusters, namely cluster *url1* containing both strings A and B, and cluster *url2* containing query string C. Using that approach on the test set of 2450 strings we obtained 914 clusters of which 340 have more than 1 variant.
As the database constructed so far (workflow 1) still contains errors, further processing to refine clusters as well as a manual validation and curation are needed to ensure high quality of the reference database. We set up an interactive user interface for manual inspection, in particular to remove variants from clusters, merge clusters, and add new variants. We offer the user to merge clusters on the basis of several heuristics to determine the similarity of an affiliation string with the affiliation variants of existing clusters. In particular, we consider common URL parts, e.g. subdomain, as well as the maximal string similarity for all variants per cluster.

Incremental database extension
To use the affiliation database for bibliographic analyses (workflow 3) the goal is to merely look up the institution via an input affiliation string. Thus, the set of affiliation strings gathered for the intended bibliographic analysis is run through the affiliation database first off to extend the collection if needed (workflow 2). Generally, the same pre-processing takes place as in workflow 1 to extract and transform affiliation strings. If the so identified main institution of the affiliation string is not found in the database it is checked whether the considered affiliation should be added as a new variant for an existing cluster or whether it represents a new institution leading to a new cluster. Several matching techniques are used to find the best fitting clusters for the affiliation string under consideration. First, a web search is performed for the affiliation string and it is checked whether there is already a cluster for the retrieved top URL. Furthermore, we use the same search strategies as for merging clusters, i.e. we compare the URL domain, the name of the main institution, and its location to the corresponding values of affiliation clusters and their variants. For determining the string similarity of the main institution standard approaches such as Soft TF-IDF or DBMS-specific approaches such as fuzzy match in MSSQL can be used. To preserve a high quality of the reference database the user interface proposes top k clusters for each new affiliation string – the user can decide whether to accept the top match, choose another offered cluster, search for other, un-presented clusters, or create a new one. A threshold can be configured for candidates with large enough similarity values not to show up during user interaction, but to cluster such affiliation strings automatically, flagged with timestamp and uncertainty (similarity value) for later curation.

RESULTS AND CONCLUSIONS

The presented approach of establishing an affiliation reference database supports an automatic matching and clustering of heterogeneous affiliation strings. We propose incremental extension of the database via string matching of new variants and user inspection to maintain high quality. For the approximate affiliation matching task we presented two URL overlap similarity measures: simURLdist and simURLbase. The simple but nevertheless very effective base measure simURLbase(1) is especially attractive as it avoids the need to find a suitable similarity threshold and it permits to match and cluster the affiliation strings in one go. In affiliation matching the proposed URL overlap measures achieve much higher precision and recall values than the common Soft TF-IDF measure which cannot sufficiently cope with the high heterogeneity of the strings, including abbreviations and acronyms. As URL overlap is based on the results of web search engines their background knowledge e.g. regarding synonyms and abbreviations is leveraged. On the other hand web search engines have to be seen as black boxes, of which search results are also likely to change over time, whereas Soft TF-IDF is robust and predictable.
Before matching the affiliation strings we had to perform several pre-processing steps, in particular heuristics to identify the part denoting the main institution. While the current approach works reasonably well, we plan to perform specific evaluations of the pre-processing effectiveness in the future. Further

possible extensions to the reference affiliation database include storing relationships to other institutions, e.g. branching or merging of companies, name changes over time, moving location, as well as identification of institution type – e.g. academic vs. industrial. So far we also left without further investigation affiliation strings that refer to multiple affiliations or institutions. Multiple affiliations are denoted e.g. by simply concatenating them via 'and', an ampersand, or a comma. These characters alone are not discriminative enough to separate multiple affiliations as they appear in single institutions names as well. In our database we flagged those specimens for later processing, e.g. checking whether multiple already known affiliation strings are mentioned.

We further plan to use the affiliation reference database for large-scale bibliographic analyses, e.g. to identify the main locations for specific research areas, and to study affiliation-specific impact numbers.

REFERENCES

[1] Arasu, A., Kaushik, R. A grammar-based entity representation framework for data cleaning. *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD),* 2009

[2] Aumueller, D. Towards web supported identification of top affiliations from scholarly papers. *Proc. German Database Conf. (Database systems in Business, Technology and Web (BTW 2009),* 2009

[3] Bollegala, D., Matsuo, Y., Ishizuka, M. Measuring semantic similarity between Words using web search engines. Proc. *WWW* Conf., 2007

[4] Christen, P., Goiser, K. Quality and Complexity Measures for Data Linkage and Deduplication. *Quality Measures in Data Mining.* Springer, 2007

[5] Cohen, W., Ravikumar, P, Fienberg, S. A Comparison of String Metrics for Matching Names and Records. *Data Cleaning and Object Consolidation, 19*(1), 2003

[6] Elmacioglu, E. et al. Web based linkage. *Proc. Web information and data management (WIDM)*, 2007

[7] Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S. Duplicate Record Detection: A Survey. *Knowledge and Data Engineering*, 2007

[8] Gligorov, R. et al. Using Google distance to weight approximate ontology matches. *Proc. WWW Conf.* 2007

[9] Google Inc. Google AJAX Search API <*http://code.google.com/apis/ajaxsearch*>

[10] Kalahnikov, D. V., Mehrotra, S., Chen, Z. Exploiting relationships for domain-independent data cleaning. *Proc. SIAM International Conference on Data Mining (SDM),* 2005

[11] McCann, R., Shen, W., Doan, A. Matching Schemas in Online Communities: A Web 2.0 Approach. *Proc. Data Engineering (ICDE),* 2008

[12] Michalowski, M., Thakkar, S., Knoblock, C. A. Automatically utilizing secondary sources to align information across sources. *AI Magazine*, Spring 2005

[13] Pereira, D. A. et al. Using web information for author name disambiguation. *Proc. Joint Conference on Digital Libraries (JCDL)*, 2009

[14] Rahm, E., Thor, A. Citation analysis of database publications. *SIGMOD Record*, Dec. 2005

[15] Tan, Y.F. et al. Efficient Web-Based Linkage of Short to Long Forms. *Proc. ACM Workshop on the Web and Databases (WebDB)*, Vancouver, 2008

[16] Torvik, V. I., Smalheiser N. R. Author name disambiguation in MEDLINE. *ACM Transactions on Konwledge Discovery from Data. 3(3)* July 2009

[17] Yahoo! Inc. Yahoo Search BOSS <*http://developer.yahoo.com/search/boss*>