Scalable Disambiguation of Institutions for the Web of Science

Tobias Backes
GESIS Cologne
tobias.backes@gesis.org

Daniel Hienert
GESIS Cologne
daniel.hienert@gesis.org

Philipp Mayr
GESIS Cologne
philipp.mayr@gesis.org

ABSTRACT

Determining address string equality is crucial to attribute publications or identify authors' affiliations – important prerequisites for higher-level research questions. While data sets for institution resolution are no smaller than in other large-scale scholarly resolution tasks like author disambiguation, the hierarchical nature of the institutions themselves leads to additional difficulties. This 'true' hierarchy coalesces with the hierarchy of partial and ambiguous representations. In this project, we explore the opportunities of ordering parsed address strings into the subset relation and performing iterative block merging subsequently. We test our model in the Web of Science – one of the largest collections of scientific metadata – and evaluate its performance against a large gold standard over its German subset. The results underline the vagueness of the presented task, flaws of the gold annotation but also the potential of our approach to answer specific questions concerning identity or equality of different affiliation strings.

1 INTRODUCTION

Institution resolution concerns the search for indicators of *identity* or *equality* and is a crucial task in semi-curated digital libraries where a wide range of inputs leads to a noisy mapping between institutions and their address strings. It is a prerequisite for various tasks, in particular determining researcher mobility or measuring institution-specific performance indicators. The heterogeneity of affiliation data leads to problems like (a) missing information, (b) deviations and variations and (c) temporal snapshots. For example: (a) a researchers department is *inst:GESIS,dep:WTS*, reported is only *inst:GESIS*; (b) the institute is *GESIS Leibniz Institute* but on one record it reads *GESIS Leibnitz Institute*; (c) it used to be *ZUMA*, but now it runs under *GESIS*. More formally, we distinguish *necessary* and *sufficient* conditions for equality, e.g. (i) *same email domain* is sufficient for equality, but not necessary (one can have multiple domains), (ii) disregarding name changes, *university* is necessary, but not sufficient (there are different institutions under one university), (iii) *similar terms*, are neither sufficient, nor necessary, but nevertheless a strong indicator. Sufficient conditions provide *evidence*. Necessary conditions tell us what must be separated.

Avoiding quadratic complexity. In large collections, scaling is critical. In this context, *blocking* refers to prior separation of mentions into partitions based on a small, distinctive set of features, assuming that whatever falls into separate blocks is sufficiently unlikely to co-refer – and direct comparison can be avoided. The blocking procedure itself must avoid pairwise comparison and therefore implements a bucket sorting algorithm.

The transitivity problem. Sufficient conditions define a transitive relation (if a and b are equal, and b and c are equal, then so are a and c), while necessary conditions do not [11]. The difficulty is that both blocks (separating what must be apart) and clusters (grouping what must be together) are ultimately disjoint partitions, and therefore equivalence classes of two equivalence relations – which are transitive by definition. Blocking tries to separate mentions with contradictory surface forms, (e.g. different universities), but matching is not transitive: *inst:GESIS,dep:WTS* matches with *inst:GESIS*, which matches with *inst:GESIS,dep:CSS*, while *GESIS,WTS* and *GESIS,CSS* are contradictory. This can be modelled with the help of the subset-relation, which is transitive, but not symmetric.

A lattice of blocking keys. The subset relation defines a *(semi-) lattice* over mentions based on sets of blocking properties of their representations. For example, here it holds GESIS $\rightarrow$ GESIS,WTS and GESIS $\rightarrow$ GESIS,CSS, while GESIS,WTS $\nleftrightarrow$ GESIS,CSS. The symmetric transitive closure of this relation (the respective graph's weakly connected components) can be used to obtain an equivalence relation which creates a blocking that separates no matching names but may group contradictory ones. If these blocks get too large due to high connectivity in the underlying relation, the latter must be modified before taking the closure. Most institution resolution approaches implicitly do so, e.g. when normalizing to *top-level only* by removing all edges to more specific representations.

Gradual improvement. So far, it was unclear which level of detail to consider to determine equality. One solution was proposed in [1] and takes the entropy of the frequencies of a mention's direct specifications into account. A better fit is *progressive* resolution, where the notion of a fixed blocking relation is abandoned. Instead, more entity pairs are compared in each iteration.

Hierarchical modeling. Besides reducing complexity, blocking offers the opportunity to model the relationship between different surface forms in a logical and visually accessible way. It unveils hierarchies present in the actual institutions as well as in their fragmented representations. First, adequate blocking keys and means to extract them are identified. Then, the lattice is build and sparsified by removing inadequate or irrelevant types. By instantiating the types in the lattice with observed mentions, the semilattice is built which includes different representations, their entailment and the estimated entailment likelihood for progressive block merging.

Lattice-based institution resolution. In this work, we present a method for large-scale hierarchical institution resolution. It builds on the blocking relation and combines it with a partition hierarchy to exploit both the underlying graph structure and additional frequency information. In section 2, we explain our formal model. In sections 3, 4 and 5, we describe how we apply it to institutions by parsing address strings into representations, ordering them into a semilattice, separating them by minimal elements and finally obtaining different levels of equality. In section 6, we explain our experiments as well as their results. In section 8, we discuss relevant literature. Finally, we describe the output computed in the project and discuss and conclude our findings.

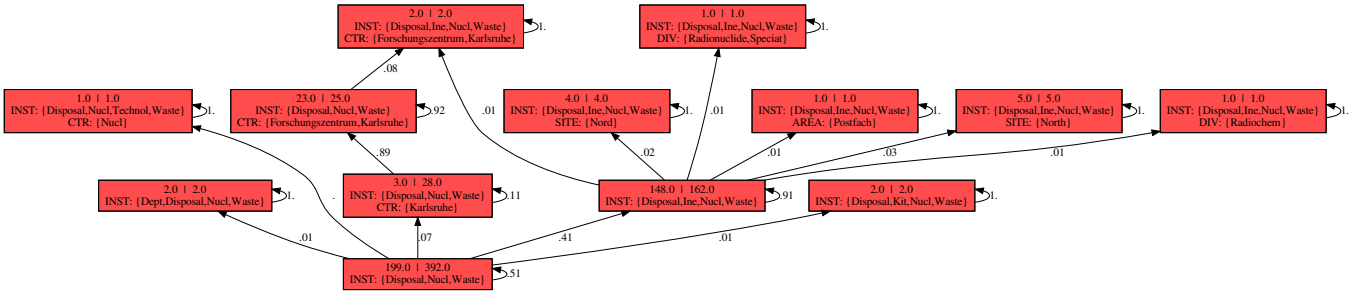

Figure 1: The semilattice for *Institute for Nuclear Waste Disposal* – build and colored using information from the test data.

2 THE GENERAL MODEL

The overall goal is to minimize the number of non-coreferring address pairs in the same block (high bPrec) and the number of coreferring pairs in different blocks (high bRec). For institutions however, a single level of coreference is not sufficient as they have a hierarchical structure. For example, two addresses might corefer on the university level, but belong to different chairs. We strive to model such hierarchies in our approach and in addition we can iteratively merge the most similar representations with the goal of simplifying the structure and resolving synonyms.

Representation hierarchies. In our model, each address mention x is *represented* by a set R_x of key-value pairs $(a, v) \in A \times V$:

$$R_x = \{(a, v) \mid x \text{ has value } v \text{ for key } a\}$$

e.g. $\{(INST, Disposal), (INST, Nucl), (INST, Waste)\}$ in Figure 1. For each representation R_x, one can obtain its *type*

$$T_{R_x} = \{a \mid x \text{ has some value for key } a\}$$

by considering only the keys, i.e. $\{INST\}$ or $\{INST, INST, INST\}$. The subset relation $<$ defines a *lattice* over the power set $\mathcal{P}(A)$ – that is over types. For example $\{INST, INST, INST\}$ is a subset/-generalization of $\{INST, INST, INST, CTR\}$. We can encode world knowledge by removing certain types from this lattice (e.g. *no chair without university*) [1]. We say a representation is *licenced* by the lattice, if its type is in the latter. The subset relation also defines a *directed acyclic graph* over the representations R_x and their licensed generalizations. For example, *INST:{Disposal,Nuclear,Waste}* is a generalization of *INST:{Disposal,Kit,Nuclear,Waste}*. We can view this semilattice as a graph with one node for each representation R_x. Each node 'contains' all the institution mentions $x_1, \ldots, x_k$ with this representation. Edges implement the *covering relation* $\lessdot$ as known from lattice theory. Each node holds two values: (1) the *observation-count* #, giving the number of times the representation has been observed and (2) the *carry-count* $\check{\#}$, giving the number of times one of its specifications has been observed. Using # and $\check{\#}$, we define conditional probabilities as edge weights:

$$p(R_x|R_{x'}) = \frac{\check{\#}(R_x)}{\check{\#}(R_{x'})}, p(R_x|R_x) = \frac{\#(R_x)}{\check{\#}(R_x)}$$

so that the probability of R_x given itself is 1 iff R_x has no observed specifications. As a node's carry count is the sum of all its specifications' (and it's own) observation-count, $\check{\#}(R_x) \geq \#(R_x)$, and $\check{\#}(R_{x'}) \geq \check{\#}(R_x)$ for any $R_x > R_{x'}$.

Logical reasoning. Our model uses both *deductive* and *inductive* reasoning. Deductive, top-down reasoning is implicit in the notion of the carry-count: if we observe an institution mention with a certain representation, we know that we also observe all its generalizations and we add its observation count as carry-count to all of them. Edges are strong if the origin is seen often as a generalization of the target (not so much on its own or as generalization of other nodes). Inductive, bottom-up reasoning models uncertainty about the completeness of a representation. If we observe a mention with a certain representation, we can guess that this representation is, with some probability, only an observed generalization of a more accurate 'true' representation: we hypothesize missing information. This is one of the two central operations in our model: *discounting*. Here, we guess that a certain percentage δ of a node's observation-count should be credited to more specific nodes. We use the straightforward rule that a node's discount mass $\delta \cdot \#(R_{x'})$ is distributed among its direct specifications (its covers) $R_x \gtrdot R_{x'}$ proportional to the current probability distribution:

$$\Delta_{R_x \leftarrow R_{x'}} = \delta \cdot \#(R_{x'}) \cdot \frac{p(R_x|R_{x'})}{\sum_{R_{x''} \gtrdot R_{x'}} p(R_{x''}|R_{x'})}$$

As each node $R_{x'}$ with nonzero observation has a nonzero probability $p(R_{x'}|R_{x'})$, and therefore also distributes observation mass to itself, we can choose $\delta = 1$. Until the end of the turn, a node's gains $\Delta_{R_x \leftarrow R_{x'}}$ are 'buffered in the edge' and only then added to its observation-count to prevent it from being propagated further.

Collapsing structures. In each iteration, we can *merge* all nodes/blocks $R_x, R_{x'}$ that are connected by edges with weights/conditionals $p(R_x|R_{x'})$ above a threshold t_b. This is the second central operation of our blocking method and the final purpose of discounting. An example is shown in Figure 4, where after some iterations, a number of strongly related representations for the *L3S* have been combined. Merging two nodes, the result contains the mentions of both. The observation counts are added and the carry count is updated w.r.t. the new set of specifications. At the end of each iteration, we perform discounting as described above. As a result, the observation mass gradually moves up the lattice. A welcome side-effect is that edge weights in the entire graph grow – although more dominantly in some edges. Therefore, the next iteration is likely to find edges exceeding t_b that had not done so before. As it can happen that all observation mass ends up in the most specific nodes before all nodes are merged into a single large block, we also lower t_b by a small amount.

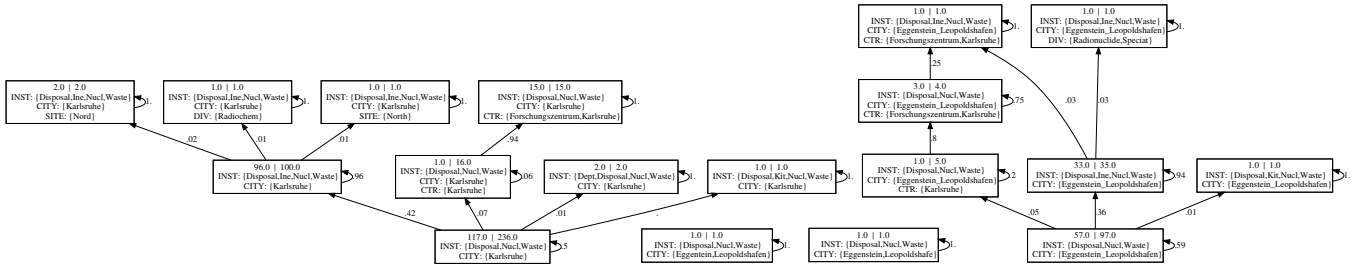

Figure 2: The semilattice for *Institute for Nuclear Waste Disposal* – built with additional city information from the WoS.

3 ADDRESS PARSING

In this section, we give details about the information extraction process that we apply to the affiliation/address strings available for most authors in the Web of Science. Our solution is specific to this collection, as the addresses have been entered and normalized in a certain way that is – to some extend – homogeneous. This is a parsing problem, as the task is to assign labels to all components of the address, identifying both groups and their labels. Generic address parsers fail to recognize the generalities and peculiarities of the WoS affiliation strings. As shown in Figure 3, the following general steps are applied:

(1) Obtain components by splitting string on commas
(2) Identify standard address components (city, postcode, street, country) by regular expressions, comparison with WoS city/country fields as well as look-up in *geonames.org* [5]
(3) Classify remaining components by keyword occurrence
(4) Detect phrases based on component-specific cooccurrence statistics and use the four most specific terms/phrases

After removing meaningless parts, the central method is a keyword classifier for the remaining components. The latter is defined in a separate file and uses over a hundred rules of the following kind:

$$Inst \rightarrow institute$$

means upon *Inst*, remove *Inst* and use the rest as *institute*.

$$Fachhsch \rightarrow polytechnic[FH]$$

means upon *Fachhsch*, replace by *FH* and use as *polytechnic*. A hierarchy of classes is also used, e.g. if both "Univ" and "Hosp" are found in the same component, classify as *hospital*. In addition, we experiment with suffix-matching as well to answer in particular for German compounds like *Krebszentrum*. This shows mixed results, for example *Post fach* is frequently parsed as a research *area*. Table 1 shows the current types/classes, their approximate hierarchy and sample keywords. While some language and convention-specific adoptions are probably required, most address strings obey international standards. Even in the German subset most addresses are in English. We note that the last point (4) in the above enumeration is optional. Instead of representing each classified component (i.e. *institute*) by up to four extracted phrases/terms, we can simply use its (normalized) string. While this is certainly nice in terms of presentation, it fails to deal with any kind of variation (e.g. *GESIS Leibniz Institute* vs. *GESIS Leibnitz Institute*). In addition to term extraction and phrase detection, additional means of normalization (i.e. stemming or translation) could be applied. This is indeed a major focus for future work, as in our experience, any improvement in the parsing step leads to considerably less complexity in the resulting graphs. Consider Figure 2, where the typos *Eggentein* and *Leopoldshafe* lead to isolated representations. One of the main contributions of our work is to offer a scalable framework that allows to iteratively adopt and improve the address parser to capture world knowledge and quickly view the consequences to the overall disambiguation output. For example, with a few modifications to the parser, we find that adding the reliably annotated city information from the WoS to the representations offers useful hints for the disambiguation of very general representations like *CLI:{Univ}*.

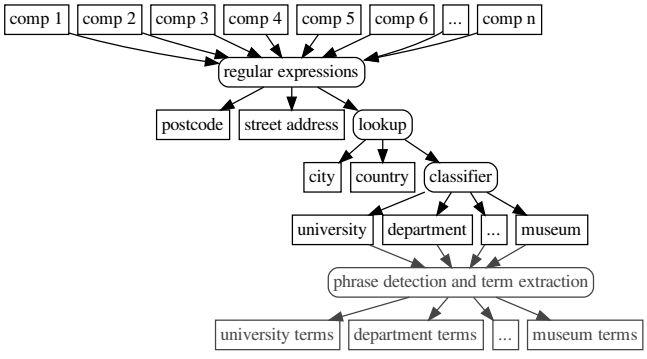

Figure 3: The pipeline used to parse address strings and obtain affiliation representations. Optional part grayed-out.

type/class	lvl	example	type/class	lvl	example	lvl	explanation
university	0	Univ	college	1	Sch		
polytechnic	0	Fh	collection	2	Bib	0	top level
academy	0	Acad	chair	2	Ls		
agency	0	Agcy	lab	2	Labor	1	below top
association	0	Soc	institute	2	Inst		
company	0	Ltd	division	2	Dept	2	in between
faculty	1	Fac	site	2	Campus		
clinic	1	Infirm	area	2	Fach	3	bottom level
factory	1	Werk	community	3	Panel		
center	1	Ctr	city	4	-	4	unspecified

Table 1: Types of institution components currently used, corresponds to the classifier labels. Including approximate hierarchy levels and example keywords that trigger assignment.

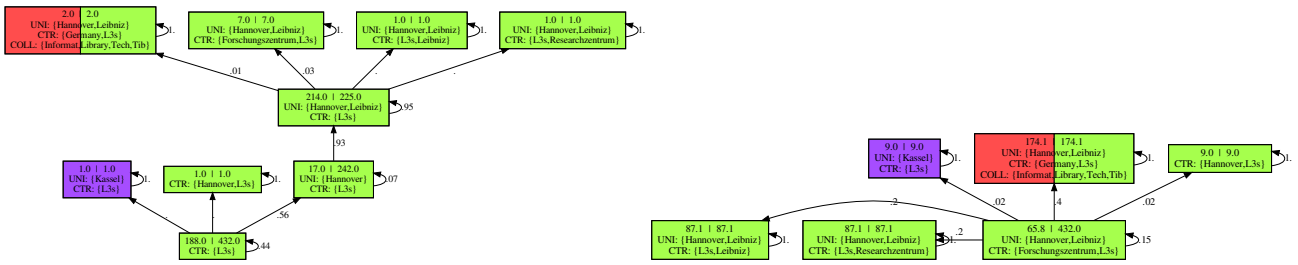

Figure 4: The semilattice for *L3S Center* in the test data. On the right after a number of iterations of merge and discount.

4 COMPLEMENTARY MINIMAL ELEMENTS

While our directed acyclic graph (DAG) of institutional representations is highly connected and there are very large connected components, a number of special nodes with zero in-degree stand out. These nodes form minimal elements of overlapping semilattices and optimally constitute top-level organisational units (i.e. *UNI:{Heidelberg}*). The respective representations (a) are observed at least once, (b) have no observed generalizations and (c) are licensed by the lattice. We do not include unobserved minimal elements in our graph. In practice, over the WoS, all major universities are observed at least a few times in their general form and therefore constitute a minimal element. Observed illegal minimal elements like *DIV:{Neurol}* cannot be ignored, so they are separated as single-node graphs. The union of all semilattice decompositions includes all mentions, although some will appear in more than one of these 'meta-blocks'. Decomposing the DAG is crucial to obtain blocks that can be independently processed while maintaining all edges. In addition, we can expect that correct minimal elements amount to top-level institutions and therefore have 100% Recall against the top-level annotation. Table 2 presents legal and illegal minimal elements used. Some of them are legal but not unambiguous. For example *UNI:{Munich}* could refer to *LMU* or *Technische Universität*. While these cases are usually the result of unsatisfying address parsing, they present an interesting test-bed where top-level precision is not 100%. Details are discussed in the results section.

lower bound	carries	observed	%	legal	init P	end P
Uni Heidelberg	170816	13464	7.9	•	99	96
Uni Munich	341727	13326	3.9	•	99	48
RWTH Aachen	96732	11277	11.7	•	91	95
Humboldt Uni	93762	9782	10.4	•	94	53
Univ Cli	476473	4941	1.0	•	59	4
Max Planck Inst	352246	1834	0.5	•	97	2
Neurol Div	75201	1385	1.8	○	94	3
Med Div	230147	1001	0.4	○	95	3
Phys Inst	360336	951	0.3	•	98	2
Theory Div	2451	701	28.6	○	96	14
BMW Div	838	572	68.3	○	100	98
Tech Uni	417359	397	1.0	•	99	22

Table 2: Some lower bounds with number of addresses equally or more specific (*carries*), observation frequency, their ratio (%), legality, as well as initial and final Precision.

5 LEVELS OF EQUALITY

Previously, we have described how our model can obtain and collapse a hierarchy of affiliation representations. An essential remaining question is how to use this hierarchy in the context of resolving institutions. We find that this question is underspecified: While it can be broken down to whether two pairs of affilition references are 'identical' or 'equal', it is unclear what constitutes this equality: Are two different departments of the same institute equal? Is the university equal to one of its faculty chairs? In fact, it can be expected that the definition of equality depends on the usecase at hand. Therefore, we aim at leaving the concrete definition of equality to the user and offer different answers for whether two addresses are equivalent. A single set of identifiers defining a fix partitioning is inadequate in this context. We note that it is infeasible to output a Boolean equality value for each pair of mentions as this has quadratic complexity and propose the following solution: For each state (iteration) of the institution hierarchy, obtain a sparse mention:representation mapping R. In the strict equality case, each mention is mapped only to its own representation. In the most general case, a mention is mapped to each representation in the semilattice that is more or equally general than its own representation. In Figure 4 on the left, a mention with the representation *CTR:{Hannover,L3s}* has as generalization *CTR:{L3s}* at 1 level higher, while another mention represented as *UNI:{Hannover,Leibniz},CTR:{L3s,Leibniz}* has the same generalization at 'distance' 3. In the most general case, all addresses are equivalent for which there is a *meet* or *infimum* in the semilattice. Stricter senses of equality can for example be obtained by adding only representations to R that are n levels more general than the address representation. In Figure 4 on the left, *CTR:{Hannover,L3s}* has a distance of 3 (the maximum of their distances to the infimum *CTR:{L3s}*) to *UNI:{Hannover,Leibniz},CTR:{L3s,Leibniz}*. Moreover, we can say that the equality of two addresses is defined by their infimum – in the above example, the two mentions are equivalent in that they both belong to *L3s center*. These equivalences are resolved to some extend by the iterative merging of nodes. In Figure 4 on the right, the distance between the two mentions has been reduced to 1. It is not unjustified to suspect that counting the number of edges for a distance is not appropriate when edge weights are available. This clearly needs to be investigated in future work, however we currently depend on the categorical nature of distance to enable efficient queries (see section 7). Previous work [9, 10] behind our gold data has only considered an unrestricted top-level sense of equality. Our work subsumes and refines this notion.

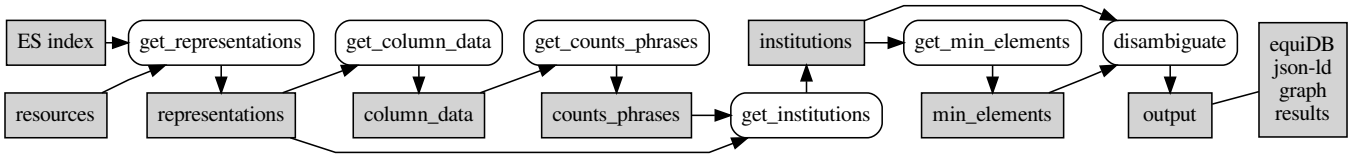

Figure 5: The workflow of our approach to produce outputs like evaluation, visualizations or equivalence level DBs

6 EXPERIMENTAL EVALUATION

6.1 Data

In the KB project 'Efficient Retrieval of Web of Science data with Elasticsearch', WoS data was imported from XML documents into an Elasticsearch index enabling efficient retrieval on 58 million WoS documents between 1980 to 2019. There are 233, 791, 889 authors with an average of 4 per document and 94, 592, 361 addresses with an average of 1.6 per document. In the KB project 'Institution Coding as a Basis for Bibliometric Indicators' [10], WoS publications with 6, 513, 669 German author addresses were matched to 2124 top-level institutions which we use as gold standard. We imported all institutions and related addresses into another index.

6.2 Evaluation Measures

As explained in section 5, there are as many correct answers to institution resolution as there are interpretations of equality. However, our gold standard expects only one such answer, namely the most general. Consequently, the task of our evaluation is two-fold: (1) How well can we match the annotation on the German subset of the WoS and (2) what is the range of feasible and sensible configurations for other interpretations of equality. While (1) can be quantified, (2) can only be studied qualitatively by looking at excerpts. For (1), we use pairwise Precision, Recall and F1: We record for each iteration precision and recall of current the current node assignment as percentage of correct returned pairs (TP) in returned pairs (P) or in correct pairs (T), respectively. P is the number of mention pairs that are currently contained by the same node (remember that nodes can be merged and their mentions afterwards belong to the same node). T is the number of mention pairs that have the same id in our gold annotation (that is they are said to belong to the same top-level institution). TP is the number of mention pairs that are in the same node and have the same id.

6.3 Experimental Setup

In the following, we describe the workflow (see Figure 5) that is applied to conduct experiments and produce the different outputs.

Preprocessing. We have loaded the WoS as well as the Bielefeld annotation and the associated addresses into an *Elasticsearch* index. We also use a *geonames.org* database as an additional resource. Not depicted are parser configurations. Next, we download addresses, split them into components, classify and normalize the latter (*get_representations*). From the output, we extract the strings for each of the different classes like *university* or *institute* (*get_column_data*) to compute term frequencies and perform phrase detection. We use these to get institution representations like *Uni:{Hannover},CTR:{L3s}*. As described in section 4, it is much advisable to find a way to split the data into chunks for job-wise processing. In theory, the institution representations could be used directly in the *disambiguate* script, but in practice, we add the following intermediate processing: On the partially ordered set of institution representations, we apply an improved version *get_min_elements* of the algorithm described in [2] for detecting *min_elements*, which are stored as queries for the *institutions* database and can be supplemented with their carry and observation count for further analysis (see Table 2). Then, we can call *disambiguate* for each minimal element with the respective query and run our actual method from previous work on the subset returned by the query.

Graph-building and iterative node-merging. Given a minimal element, we first build the semilattice under it, like in Figures 1, 2 and 4 on the left. We note the difference between Figures 1 and 2, where the first is used in the evaluation against the Bielefeld annotation. It lacks the city information from the WoS and adds institution IDs, which are visualized by a color-coded histogram for each node. Next, we start iterations of discount and merge as described in section 2. This leads to a gradual conflation of the semilattice: more and more nodes share the same node/representation. As shown in Figure 6, for each iteration, we can output (a) a *DOT*[1] graph visualization like Figures 1,2,4, (b) *JSON-LD* for the maximum spanning tree of this graph, (c) a row for the evaluation results of this job and (d) additional edges for the global iteration-specific *equiDB* database (see section 5). In this work, we do not determine the best time to stop iterative conflation. Instead, we show the development of precision and recall over all iterations. These evaluation measures are obtained for each minimal element and aggregated by adding for each iteration T, P and TP over all jobs. Time is also measured but not displayed due to server-load dependent variations.

[1] https://en.wikipedia.org/wiki/DOT_(graph_description_language)

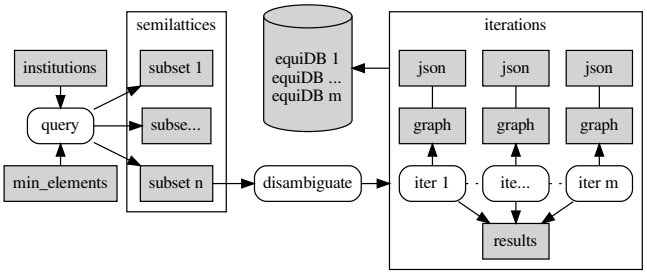

Figure 6: Output details: one subset per minimal element, one result progress per subset, one graph+json output per subset and iteration – and one equivalence DB per iteration

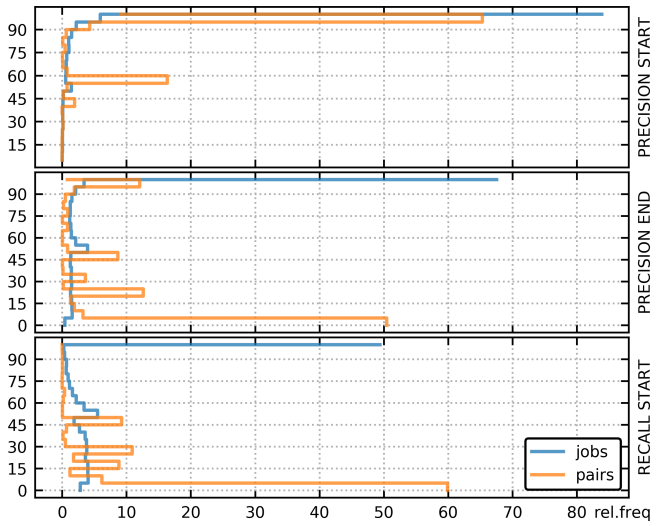

Figure 7: A histogram over size-5 bins of precision in the first and last block merging iteration as well as initial recall.

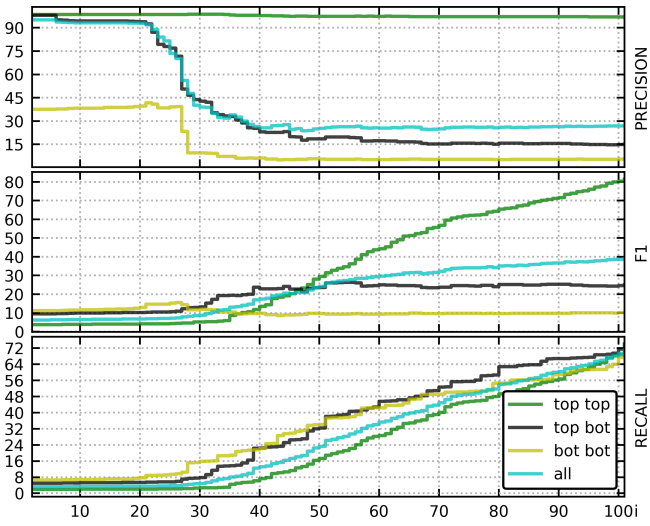

Figure 8: Results over all iterations for three major classes from Table 3 and all minimal elements.

6.4 Results

Classes of minimal elements. Due to the iterative process of block merging, precision and recall measurements are taken per iteration and we compare not static results, but their *development*. With continued collapsing, recall must increase and precision decreases. Here, our method should gain more than it looses: recall should increase faster than precision decreases. As our gold standard only supports top-level institutions and we currently evaluate recall only within jobs, minimal elements used for separating the data have a large impact on the final result. Therefore, we compare different classes of jobs based on their initial and final precision. These measurements in the first and last iteration do not depend on our block merging method, but only on address parsing and the resulting minimal elements. As shown in Table 3, we distinguish nine different classes of jobs, whereof three (those where precision increases) are empty. From the remaining six, we investigate *top-top*, *top-bottom* and *bottom-bottom*, as those involving *medium* are just less significant versions of these. Top precision is set to be any value $> 85\%$ and bottom precision to be $< 50\%$. We see that counting the number of jobs, 73% of them belong to the *top-top* class. However, as depicted in Figure 7, these account for much less mention pairs (which are counting towards evaluation) as large jobs contain more than one top-level institution and final precision is low.

	*top top**	*top med*	*top bot**	*med top*	*med med*	*med bot*	*bot top*	*bot med*	*bot bot**
jobs	73%	10%	9%	0%	4%	4%	0%	0%	1%
pairs	15%	3%	61%	0%	0%	19%	0%	0%	2%
init P	>.85	>.85	>.85	.5-.85	.5-.85	.5-.85	<.5	<.5	<.5
end P	>.85	.5-.85	<.5	>.85	.5-.85	<.5	>.85	.5-.85	<.5

Table 3: Classes of lower bounds based on precision in initial and final iteration of block merging – with percentage of jobs and pairs per class. For selected classes*, see Figure 8.

Class-based performance observations. In Figure 8, we view average performance of all jobs as well as for the three above mentioned classes. It can be seen that on average, block-merging is unable to reliably identify different top-level institutions, as comparing values for the same iteration on the x–axis, precision decreases much more quickly than recall increases. This performance over all jobs is dominated by the top-bottom class as it makes for 61% of all pairs (see Table 3) and *all* and *top-bot* show a similar development. The top-top class sees an obvious increase in recall at constant precision. Without a more detailed hierarchical gold standard, we cannot assess the quality of these merges. Interestingly, in the bot-bot class, precision slightly improves before dropping, suggesting that a number of correct merges are done first.However, there are much less jobs of this kind, so it is unclear whether this is due to chance. We summarize that as expected, precision decreases and recall increases on average. F1 also increases, but this is due to precision and recall approaching each other and bears little additional information. It seems that more or larger blocks are merged that contain mentions of different top-level institutions than the opposite is the case. This is different from our experiences with smaller visualized examples like Figure 4. In Figure 9, we view separate performance for two jobs of each class. The largest jobs on the left have roughly the same performance as their class in Figure 8 – that is if there is any improvement, it is in jobs with low initial precision. For the smaller jobs on the right, our method seems to work better. The 95% largest top-bot minimal element *INST:{Pharmacol}* sees recall increase faster than precision drops. Regarding minimal element job separation as such, we note that most jobs contain only one top-level institution (in Figure 7, end precision of 100% has a relative frequency of almost 70%). This suggests a good separation in many cases. However the problematic cases are both hard to disambiguate towards the top-level annotation and contribute very decisively to the overall performance due to their size.

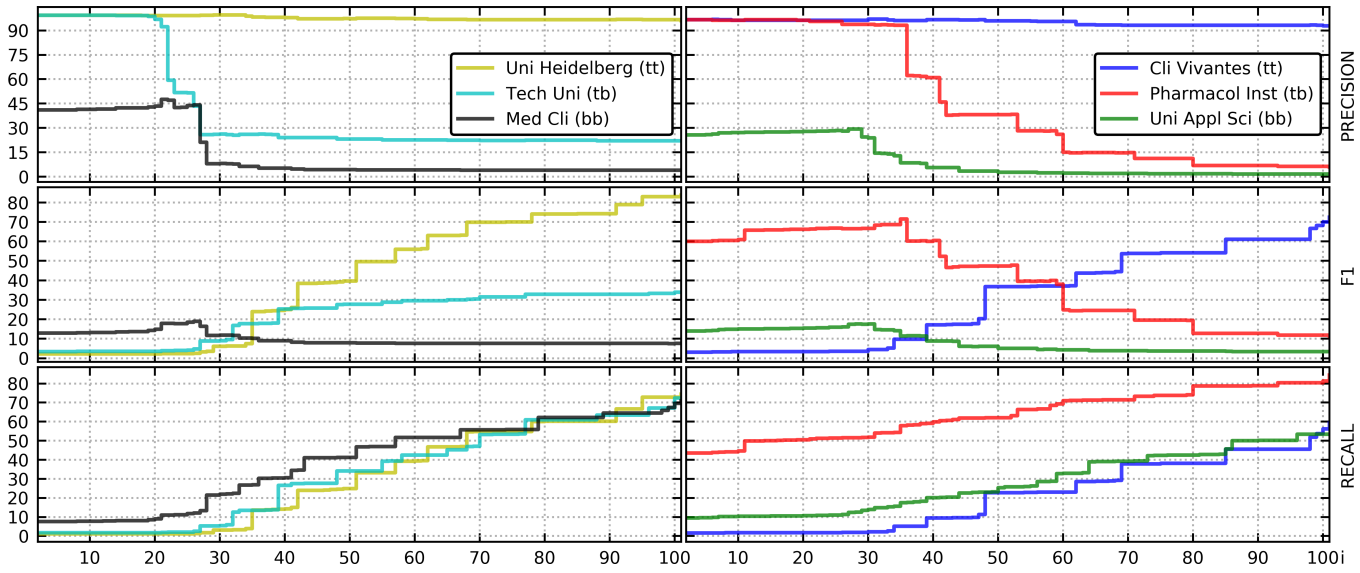

Figure 9: Results for two jobs – largest (left) and 95% largest (right) – from classes top-top (tt), top-bot (tb) and bot-bot (bb).

6.5 Discussion

Result interpretation. As a first error analysis, we find that the three chosen classes can be summarized as follows: (a) top-top: minimal element corresponds to an annotated top-level institution (e.g. *UNI:{Heidelberg}*), (b) top-bottom: the minimal element is too general but its specification are mostly correctly identified (e.g. *INST:{Pharma-col}*), (c) bottom-bottom: as the initial precision is already low, some serious problems in the parsing step have happened that have merged different addresses into the same representation (e.g. *CLI:{Med}*). Under the current evaluation, *top-top* does not say much about our method's performance except that the majority of delivered minimal elements make for good top-level institutions; *bot-bot* on the other hand asks for adjustments in the address parser, after which the few jobs of this kind should be reevaluated. As stated before, the most interesting case is *top-bot*, where multiple top-level institutions are contained in the same job and iterative block merging has the chance to first merge within these, before finally combining everything. It can be speculated that the main problem is the huge size of the dominating jobs.

Towards better block-merging performance. As stated above, with the current available gold annotation, we would like to see recall increase faster than precision declines in the *top-bot* cases. As first means for performance improvement, we need to prevent very large jobs either by tackling underspecification with additional information like *city* (cf. Figures 1, 2) or by taking the weight of edges into account when computing the minimal elements (i.e. by applying a threshold over edge weights). In addition, incremental improvement of the address parser promises decisive performance gains in downstream tasks, as this has been the case before.

Towards more concise evaluation. The evaluation presented here still has a number of flaws that should be addressed. Currently, recall is evaluated only within jobs / minimal elements, which assumes that there are no equivalent institutions annotated across job borders. Although this assumption is expected to be mostly true, it constitutes an unacceptable unclear point. Therefore, we should base our evaluation on addresses with the same top-level annotation, combining results from different jobs. We still need to determine a formula by which this can be done. In addition, our method has a number of parameters like the discounting parameter δ or by how much we reduce the block-merging threshold t_b after each iteration. These need to be tuned on held-out data or at least more than one setting needs to be used to investigate whether there is a significance influence of the respective parameter on performance. Also, a number of baselines need to be compared to the currently evaluated method, for example randomly merging blocks or merging blocks only based on initial, unchanged edge-weights (disabling discounting). Finally, we should find a hierarchical gold standard (i.e. the DFG *GERiT* database), find a way to map its entries to the WoS affiliation strings and evaluate the hierarchies created by our approach against it to obtain a better estimate of our method's ability to recreate true institution hierarchies.

7 AVAILABLE OUTPUT

As shown in Figure 6, we produce four different kinds of outputs: (a) evaluation results, (b) DOT graph visualizations, (c) JSON trees and (d) the equivalence database. While (a) and (b) are mainly used for testing and understanding, the JSON is enriched with *schema.org* types and made available for demonstration[2] and the SQLITE equivalence database can be used to find institution mentions related at 'distance' x to another mention m with the following query:

SELECT mentionIDIndex FROM generalizations WHERE level <= x AND repIDIndex IN (SELECT repIDIndex FROM generalizations WHERE level <= x AND mentionIDIndex = m);

8 RELATED WORK

Impact of Institution Ambiguity. Although there are certainly more examples of problematic uncertainties in institution identity, a prominent case is described by van Raan [12], where technical problems of institution resolution are listed as a reason to oppose the popular *Shanghai Ranking* of universities. A solution directly targeted to "research assessment" is presented by [7].

Resolution by Normalization. There are various works on the normalization of affiliation addresses to group identical institutional references. [3, 4, 8] Obviously, this does not answer the question of how to model different institutional hierarchies.

Institution Resolution in the WoS. Our work is meant to complement in most parts the work done by Rimmert et al. [9, 10]. The transformation step described by them can be considered parsing an institution mention (address) into a normalized (generalized) representation. The manual effort done in particular in [10] is used for evaluation purposes. Work by [6] aims at combining an entity linking and entity resolution approach to disambiguate references to institutions in the Web of Science. In the linking task, the Bielefeld group matches Wikidata [13] institutions with Web of Science addresses. In the resolution task, the Fraunhofer ISI group matches pairs of addresses. In the linking task, a number of strong assumptions are made – in particular "The underlying assumption of this approach is that every institution is mentioned in the WoS addresses at least on one occasion with a name variant appearing exactly this way [...] in wikidata". In the following, these assumptions are relaxed (OR-connected exact match as well as approximate match by Jaro-Winkler distance). Evidently, the linking approach fails as soon as an institution is not in Wikidata. In addition to presenting an inventory of different addresses, Wikidata entries and relations between them, [10] suggests features worth considering – in particular mail domains of author email addresses. As noticed in [9], hierarchy is a very important aspect in institution resolution and reason why pairwise address matching faces difficulties. This is noted both by the Bielefeld and the Fraunhofer ISI group.

Lattice-based Entity Resolution. Our work is based on previous work by Backes [1], including work under submission. The relevant aspects have already been described in detail in the introduction.

[2] https://search.gesis.org/InstDisambViz/

9 CONCLUSION

In this project, we have explored the use of progressive entity resolution for disambiguating institutional references. The approach is based on previous (partially unpublished) work and complements it in the following ways: (a) application to the institution domain, (b) usage of the blocking hierarchy to approximate true institution hierarchy – including visualization and an extended concept of equivalence, (c) partitioning by minimal elements and (d) tackling computationally challenging aspects of applying the workflow to the entire WoS. Due to the amount of work involved in handling the above points, not all aspects could be studied exhaustively. However, the current state of our work shows that the investigated approach based on partial orders of limited feature sets is promising in that it performs ER, addresses scaling issues, models the hierarchical structure of institutions and offers (visual) accessibility.

Address parsing works generally well, especially keyword-based type classification. So does phrase detection and term extraction for different classes. We are satisfied with the hierarchies created by our model, in particular with the general balance in the graphs that allows for visualization and even offers relationships like *'all cancer departments of the University of Cologne'*. Edge weighting and discounting seems to lead to reasonable merges if progressive structure collapsing is applied. Regarding technical aspects, we hope to find parallelized implementations for building the graph-DB and extracting minimal elements. In addition, we need to find a systematic way of dealing with or preventing oversize minimal elements. In future work, we will compare different baselines and hyper-parameter settings for block merging and evaluate against a hierarchical gold standard.

REFERENCES

[1] Tobias Backes. The impact of name-matching and blocking on author disambiguation. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 803–812. ACM, 2018.

[2] Constantinos Daskalakis, Richard M Karp, Elchanan Mossel, Samantha J Riesenfeld, and Elad Verbin. Sorting and selection in posets. *SIAM Journal on Computing*, 40(3):597–622, 2011.

[3] RE De Bruin and HF Moed. The unification of addresses in scientific publications. *Informetrics*, 1990.

[4] Carmen Galvez and Félix Moya-Anegón. The unification of institutional addresses applying parametrized finite-state graphs (p-fsg). *Scientometrics*, 69(2):323–345, 2006.

[5] GeoNames.org. Geonames data, 2019. Data retrieved from https://www.geonames.org/export/.

[6] Patricia Helmich. Kodierung internationaler institutionen - eine machbarkeitsstudie anhand von ausgewählten ländern. Technical report, Fraunhofer ISI, 2018.

[7] Shuiqing Huang, Bo Yang, Sulan Yan, and Ronald Rousseau. Institution name disambiguation for research assessment. *Scientometrics*, 99(3):823–838, 2014.

[8] Fernanda Morillo, Javier Aparicio, Borja González-Albo, and Luz Moreno. Towards the automation of address identification. *Scientometrics*, 94(1):207–224, 2013.

[9] Christine Rimmert. Institutional disambiguation for further countries-an exploration with extensive use of wikidata. Technical report, Bielefeld University, 2018.

[10] Christine Rimmert, Holger Schwechheimer, and Matthias Winterhager. Disambiguation of author addresses in bibliometric databases. Technical report, Bielefeld University, 2017.

[11] Tobias Backes. Effective Unsupervised Author Disambiguation with Relative Frequencies. In *Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries - JCDL '18*, pages 203–212, Fort Worth, Texas, USA, 2018. ACM Press.

[12] Anthony FJ Van Raan. Fatal attraction: Conceptual and methodological problems in the ranking of universities by bibliometric methods. *Scientometrics*, 62(1):133–143, 2005.

[13] WikiData.org. Wikidata knowledge base, 2019.