

Toponym Resolution on Historical Serial Sources

Dennis Sen

September 26, 2016



CAU Kiel

Department of Computer Science
Research Group for Communication Systems
Master's Thesis, Summer Semester 2016
supervised by Dipl.-Inf. Dr.-Ing. Jesper Zedlitz
and Prof. Dr.-Ing. Norbert Luttenberger

Abstract

Various challenges are present in the resolution of toponyms from historical serial sources, where problems range from spelling errors to incorrect information.

In this thesis, such problems are identified, design decisions for a resolution system are made, applicable heuristics from all over literature are gathered and complemented, the implementation of such a system is described, the implemented system's result is evaluated towards actual problems for toponym resolution, and the system is compared to other available systems regarding employed heuristics.

It is found that Data Matching heuristics, in combination with heuristics from various works towards toponym resolution, are capable of solving such problems; depending on the set of data, above 90% recall, precision, and accuracy can be reached when allowing an error tolerance of 20km.

To evaluate the system, gold standard files were produced on lines of historical serial sources dated 1870 and 1914, spanning 800 cases. It is assumed by the author that the system result is, in many cases, actually of higher quality than this identification by hand.

Contents

1	Introduction	3
1.1	Problem Statement	3
1.2	Thesis Plan	7
2	Previous and Related Work	7
3	Gazetteers	9
4	Heuristics	10
4.1	Heuristics for Data Matching	11
4.2	Heuristics for Toponym Resolution	13
4.3	Summary	19
5	Implementation	21
5.1	Program Flow	21
5.2	Input, Output, Gazetteer Access	34
5.3	Data Format	38
5.4	Requirements and Software	39
6	Evaluation	41
6.1	Assumptions	41
6.2	Gold Standard Construction	42
6.3	Evaluation Modes and Usage Scenarios	43
6.4	Evaluation Results	45
6.5	Comparison to other Toponym Resolution Systems	52
7	Conclusions	56
8	Appendix	57
8.1	Glossary	57
8.2	Acknowledgements	58
8.3	Further Referenced Contents	58
8.4	System Results as Tables	64

1 Introduction

In archives, serial sources (often *lists*) of toponyms (*placenames*) are found. They might - for example - be explicit in location hierarchy tables, contained in specific columns of person-related lists denoting a residential place, or are implicitly contained in the running text of travel diaries.

To extract the toponyms from such sources, one might write them down by hand; on larger documents, OCR is used to extract the text. Via geo-tagging, toponyms in these texts or tables can be found. These, however, are just a reference by name; while a knowing reader might understand them well, no geo-coordinate for further automatic processing is attached.

Especially in long lists with tens of thousands of entries - sometimes even millions -, further studies would take intense manual labor to make the information more accessible, wasn't it for *toponym resolution*: The automated resolution of a placename to a geo-coordinate in the spirit of the Digital Humanities.

Such enriched data can be used for a variety of purposes. Depending on where the toponyms were extracted from, surrounding information like family names might be used to visualize a distribution on a map, dates might be used to draw routes or show movement, hierarchical information can be utilized to develop the completeness of gazetteers.

The conceptualisation, development, and evaluation of such a toponym resolution system is documented in this thesis.

1.1 Problem Statement

Assuming a serial source of lines containing locality descriptions is given, a fitting geo-coordinate is desired for each line. Such a locality description may contain multiple toponyms: "Edinburgh, Scotland" for example consists of two toponyms in an implied hierarchy.

In this case, the coordinates of the hierarchically lowest object are desired, while hierarchically superior information can be utilized to find the correct place. While "Edinburgh, Scotland" is a line rather easy to be identified, a broad range of problems may occur, leading to more difficult scenarios.

In this very chapter, such problems will be outlined to explain the need for a sophisticated approach to toponym resolution.

First and foremost, the problem of *placename ambiguity* needs to be addressed. As Overell collects from various sources, depending on gazetteer

and region at least 10% and up to 57% of placenames contained in examined systems are ambiguous. [20]

Were placenames evenly distributed in serial sources, this would directly result in 10% to 57% of toponyms not being identifiable by a simple look-up, as multiple places would match, resulting in the so-called *referent ambiguity*[20][26].

Depending on the source, even worse situations might arise: For example, in a list that contains places of residence, cities like Berlin will more frequently occur. As Leidner finds, there exist at least 70 Berlins. [16, p. 25] The same problem is in effect for many places with a greater number of inhabitants; the toponyms of populous places tend to be re-used, amplifying the problems due to *referent ambiguity*.

Another problem arises when *multiple names* are used to refer to a certain location. Spelling variations and variant forms of placenames are a problem that needs to be accounted for. [22] While gazetteers often do contain multiple name variants per place, completeness is not guaranteed. The more alternative names exist, the more potential placename ambiguity arises, too.

Additionally to spelling variants, spelling mistakes further increase the problem. Such mistakes may also have been introduced during OCR. For "Edinburgh", a simple string compare will not suffice. In the worst case, the misspelled word refers to another location than the originally intended one.

Rupp - who also worked on historical sources - suspects this is a major problem, as their methods result in "an adequate precision, but disappointing recall [...]. Our suspicion was that variation in spelling and in the actual placenames would be the root cause of this." [22, p. 61]

So far outlined are the problems of ambiguity and spelling. Furthermore, lines often contain abbreviations and annotations that have to be dealt with.

For example, "E., Scotland" is an abbreviation that might occur; on first glance, Edinburgh could be expected, but there's also the town of Eldin that this line might refer to.

Annotations occur in varying forms; while some of them translate to a proximity ("nearby", "send post via"), others imply a hierarchy ("in") or a correction ("Edinburg (corr.: Edinburgh)"). In other cases, *structural ambiguities*[20][26] arise: "Frankfurt On The Main" may refer to the town Frankfurt that happens to be on the river Main, or it might have become a valid toponym on its own. Furthermore, type information can be supplied

within the line. ("Village Berlin")

In historical sources, where human interpretation was the expected use case, patterns do not necessarily always refer to the same relationship. As mentioned, "A, B" will in many documents refer to a hierarchy of places. In long documents with multiple authors, it can in some parts also enclose the meaning of "nearby". Misinformation leads to further inconsistencies; on writing down information in an interview-like manner, people will respond in different patterns and may give wrong information (un-)intentionally.

Wieczorek - who worked on geo-referencing natural history collection data - specifies 9 commonly found types of "locality descriptions" in his documents. [27] Four of them are also applicable to the problem at hand:

Dubious lines contain question marks, annotations like "presumably" and other hints of uncertainty. Lines that *can not be located* contain information that does not fit any place; for example, person names could have falsely ended up here during or after OCR. *Demonstrably inaccurate* lines contain "irreconcilable inconsistencies" ("Edinburgh, Schleswig-Holstein"). *Named Place* lines finally refer to an identifiable position.

Four more of his types deal with offsets ("6 km NW Mombasa"), one type with explicitly given coordinates. These will not be dealt with as such cases are not contained in the sources worked on.

In the works of Beaman, a list of examples and challenges is presented, dealing with almost the same type of problem. He refers to toponym hierarchies as "topological nesting" and further identifies the challenges in historical placenames and change over time. [3, p. 48]

In summary, the following properties of lines have to be considered during toponym resolution on historical serial sources as outlined:

- Referent Ambiguity ("Which Berlin is it?")
- Structural Ambiguity ("Is the river name part of the placename?")
- (Spelling) Variations and Errors ("Edinburgh? Edina?")
- Abbreviations ("What is 'E.'?")
- Annotations ("Kronshagen *nearby city* Kiehl {*korr.: Kiel*}".)
- Data Quality (Is the information itself consistent?)

...	Kaldenkirchen, Rheinl.
Kalde{n}kirchen	Kaldenkirchen, Rheinland
Kaldenkirchen, Düsseldorf	Kaldenkirchen, Rheinld.
Kaldenkirchen, Gummersbach	Kaldenkirchen Rheinl., Kempen
Kaldenkirchen, Kempen	Kaldenkirchen, Rheydt
Kaldenkirchen. Kempen	Kaldenkirchen, Rhld.
Kaldenkirchen, Kempen a. Rh.	Kaldenkirchen, Rhld., Kempen
Kaldenkirchen, Kempen i. Rh.	Kaldenkirchen, Schleswig
Kaldenkirchen, Kempen i. Rhld.	Kaldenkirchen, Schwaben
Kaldenkirchen, Kempen (nicht Bauchem)	Kaldenkirch, Kempen
Kaldenkirchen, Kempen Rh.	Kaldenkirchen, Kempen
Kaldenkirchen, Kempen, Rhein	Kalden, Kreis Cassel
Kaldenkirchen, Kempen, Rhld.	Kalden, Kreis Kempen
Kaldenkirchen, Kempten	Kalden, Kreis Schlochau
Kaldenkirchen, Keppen	Kaldenrek
Kaldenkirchen, Kmepen	Kalden, Schlochau
Kaldenkirchen, Kreis Kempen	Kaldensundheim, Eisenach
Kaldenkirchen, Kr. Kempen	Kaldenwertheim, Dermbach
Kaldenkirchen, M.-Gladbach	Kalderfeld, Waltershausen
Kaldenkirchen, Mörs	Kalderlah, Gifhorn
Kaldenkirchen, Posen	Kaldern
Kaldenkirchen, Pr.	...

Table 1: extract of a list containing birth places of WWI soldiers; note that when looking up "Kaldenkirchen, Kempen" and "Kaldenkirchen, Rheinland" in three gazetteers (GOV, Nominatim and GeoNames), no fitting places were found; in summary, two churches and an arboretum were returned on 2016-07-26

Due to these occurring problems, accuracy of toponym resolution usually does not exceed 85%. [9] In other words, usually about one in seven decisions is either a wrong identification (where no or another place is considered correct) or a missing identification (where a place considered as correct exists).

The extract from a historical serial source in Table 1 is used to exemplify the situation at hand. In the following chapter, the Thesis Plan is presented.

1.2 Thesis Plan

Following this chapter, previous and related work concerning the described problem is presented. This happens on an abstract level, as heuristics and solutions to the present problems are discussed later on when needed.

Subsequently, gazetteer principles are discussed to identify what information is to be expected from such a system that supplies the very foundation of toponym resolution.

Briefly, the design decisions required for a toponym resolution system are discussed. Based on this, heuristics from literature and other systems - and their suitability to the problem - are described. These heuristics are complemented by methods from the field of Data Matching and domain-specific solutions. In connection with these an implementation is presented; both program flow and deployment will be outlined.

Finally, an evaluation takes place: The implemented system is tested against a variety of data sets. The main focus in this evaluation are recall, precision, and accuracy: As many lines as possible are desired to be identified; misidentifications are to be kept to a minimum; decisions overall should be correct. The system is briefly compared to other systems regarding employed heuristics.

2 Previous and Related Work

Toponym resolution - the term itself was coined by Leidner in 2004 [16] - is not a new topic. Larson quotes a source about recognition of the problem from 1990, where a related process was called "geographic indexing" and was utilized to search a variety of media. [14]

A variety of strategies has been tried to solve the problem of toponym resolution. Depending on where the gathered toponyms stem from, different methods are employed.

An active field of research are news texts; for example, Leidner limits his works to "a collection of present-day news text". [16] Many other works are also based on newspaper corpora. [7][8][9]

Another source worked on is the aforementioned locality data of natural history museums. [3][27] Also, (historic) route descriptions are of relevance, especially in recent years. [6][18][21]

Serial sources as discussed here are used when it comes to evaluating such systems or constructing online services. [1][2][11] Then there's work on historical texts and the extracted placenames from such sources. [4][12][22] Mind that the references here are meant to give an overview and do not aim

for completeness on the respective topics. In addition to these categories, a lot of work is built upon generic unstructured text.

The geo-tagging strategies used in most of these sources are not directly relevant for this work; at the stage of toponym resolution, they are considered done. However, almost all aim to also supply a geo-coordinate, namely practice toponym resolution of different degrees of complexity.

For example, the online service *GeoWhiz*¹ expects a list of toponyms and uses properties like population size and place closeness to build and rank sets of places. The main focus of this work is to deal with *Referent Ambiguity*.

Other works have other core concepts and heuristics to solve the problem, as mentioned frequently for a certain domain. However, none of the systems found address all of the specific problems for historical serial sources. Often, perfect but ambiguous input is expected, especially neglecting the possibility of low input quality.

This is understandable considering the sources used; for instance, news text corpora will usually have a very high typing quality; in other cases, spelling problems might have been corrected after OCR or are simply ignored due to their rarity. Problems that are mentioned and highly relevant - as seen in Table 1 - do not occur in their data.

A domain that does recognize the varying quality of data is the domain of *Data Matching*, where different entries in one or more databases are connected when describing the same real-world object. The Data Matching Process according to Christen[10] is a basic foundation of this work. While some re-interpretation is necessary, basic principles hold for the domain of toponym resolution. This will be addressed in its own sub-chapter in the heuristics section.

¹URLs to referenced services are deposited in the Appendix, Table 8

3 Gazetteers

Gazetteers ”can be defined as geospatial dictionaries of geographic names” [13] and are of fundamental importance for toponym resolution, as all resolution aims to look up the geo-location in such a system or database.

Hill describes as core components of gazetteer entries the following:

- a name (could have variant names also)
 - a location (coordinates representing a point, line, or area)
 - a type (selected from a type scheme of categories for places/features)
- [13, p. 1]

She further explains how these attributes are in praxis refined in more complex representations. For example, names might contain information about pronunciation and time period of validity; the latter also applies for geometrical representation, which can furthermore contain their accuracy of measurement.

Also, the notion is made that gazetteers may also contain information about hierarchy, so a place may have a ”parent”. Further information about a place may always be given.

Many such systems exist and are available freely to any internet user. To focus on one of such systems, however, will limit the usability of a developed software.

In an open data context, the availability of gazetteers is expanding [...]. However, whatever the resource is selected for geocoding, the problem that usually arises is its completeness. [18]

For example, the historical gazetteer GOV focuses on Australia, the USA, and large parts of Europe, but does not contain much information on Italy, Portugal, and the British Isles. The historical gazetteer *Regnum Francorum Online* does have information on Portugal and Italy; however, the focus is on Late Antique and Early Medieval Europe.

Each of the systems would be expected to yield a different quality of results on different historical sources. To use multiple sources, ”Historical Gazetteer System Integration” [5] is developed to unify multiple databases. Olteanu describes the matching of ”geographical databases” with Data Matching strategies. [19] This is a development that needs to be accounted; in the section *Implementation*, the input format construction will refer to this situation. Also, the system will be designed with interfaces to make it attachable

to various gazetteer systems with relatively low effort.

With this abstract concept of gazetteers and the problem in mind, the next step towards the implementation of a toponym resolution system is defining a set of applicable heuristics.

4 Heuristics

To address the challenge of toponym resolution, a broad variety of heuristics has been developed. In this chapter, heuristics fitting to historical serial sources are identified.

To identify such fitting heuristics, the nature of retrieval for the problem at hand is discussed first.

Larson defined an "information retrieval/data retrieval spectrum" on which he explains the nature of "geographic information retrieval". [14] His broader view will now be used to further emphasize the focus of this work.

In this, some re-interpretation will take place to match the narrow topic to the rather broad categories.

	Information Retrieval	Data Retrieval
Retrieval Models	Probabilistic	Deterministic
Indexing	Derived from Contents	Complete Items
Matching / Retrieval	Partial or "best match"	Exact Match
Query Types	Natural Language	Structured
Results Criteria	Relevance	Any Match
Results Ordering	Ranked	Arbitrary

Table 2: Larson's categories, brought to a table

Retrieval Model: Both kinds apply. On the one hand, deterministic retrieval is expected for the query "Edinburgh, Scotland"; on the other hand, when dealing with queries that contain misspelling, the system has to ponder which place is the most fitting. Whenever uncertainty arises, a probabilistic approach has to be taken.

Retrieval Indexing: Only data derived from the gazetteer contents is used, primarily the placenames. Per definition, all gazetteer entries (should) contain a geo-coordinate, but in the present case, it can't be used for matching; instead, it is a vital part of the result.

Matching: The Matching aims for an "Exact Match"; however, after seeing the data, it quickly becomes obvious that partial or best matches will have to be used on many occasions.

Query Types: When interpreting a line of the serial source as a query, they are of a mostly structured nature. Examples have shown that they may also contain natural language in parts. This category heavily tends towards the structured nature, though, often presenting toponyms separated by a comma and having a limited vocabulary for natural language.

Results Criteria: In this case, types of places can be used as a result criterion. For example, when knowing that a serial source contains the residence of people, no non-residence buildings need to be considered, even though they may perfectly match by name. Therefore, relevance of results exists.

Results Ordering: As not only direct matches, but also spelling variants and mistakes have to be considered, a rating and therefore ranking of similarity between lines and places will take place.

To be noted is that these are, in part, design decisions. For the given problem, they largely come from necessity regarding the problem's nature.

In conclusion, the process will heavily tend to the information-side of retrieval; while the data-side sometimes is what we wish or aim for, it does neither reflect the input quality nor the ambiguities that need to be accounted for.

4.1 Heuristics for Data Matching

As seen, the quality of lines is a core problem of historical serial sources. As this problem does not occur on most data sets used in toponym resolution, not many approaches have been made to address it.

Well-fitting to this challenge are the methods of Data Matching, which already have been employed in gazetteer integration. When re-interpreting the historical serial source as a database of very low quality - so low that each entry is but a string containing a semi-formal naming and duplicates commonly occur -, then Data Matching seems to be the natural solution.

Such techniques have already been used on structured historical documents about persons [10, p. 21], so productive usage on structurable lines of placenames seems plausible.

Christen describes the following steps of Data Matching:

Data Preprocessing: Preparing the input to achieve better comparability. Expanding of abbreviations, correction of wrong spelling, normalization of variations, and for the present problem, separation of toponyms, fall into this category. *From a retrieval point-of-view, this handles the mixture of natural and structured language to generate a structured query.*

Data Matching Indexing: Christen describes the matching of two databases. In our case, a low-quality database (the serial source) is matched against a high-quality database (the gazetteer). The basic principle holds: All probable cross-database pairs that may result in a match are gathered. *Fits the chosen Information Retrieval Indexing principle by focusing on toponyms. Also, Results Criteria can be used in this step to prune unwanted results without further comparison.*

Record Pair Comparison: Of such indexed pairs, usually a sameness value is computed. As our serial source does only contain a few fields and has considerably less information than the gazetteer-provided places, this value does less express sameness and more how well a line fits a place, and will here be called the comparison value. Multiple perfect matches will occur on the line "Neustadt" without implying that those multiple matches refer to the same location. *In calculating a comparison value, the probabilistic nature of the problem is reflected. A Results Ordering becomes available in ranking the indexed places to a line by such a comparison value.*

Classification: The comparison value is used to either choose a match, mark potential matches, or declare non-matches of indexed places to a line. *This results in either an exact match or a list of partial/best matches. In the ordering list, the one or multiple best are chosen if above a threshold.*

Clerical Review: In a clerical review, potential matches are revised manually. Not all decisions can be safely done without human interaction, but the system can provide a reviser with the most relevant lines and suggestions. While this is not practiced within this thesis, data for a clerical review is generated to allow for productive usage of the system.

Evaluation: After duplicates have been identified, the quality of these decisions has to be evaluated to infer accuracy, precision, and recall of the process. To the system, an automated evaluation system is attached; however, it has to be provided with gold standard data to actually work.

For the clerical review, when automated learning processes fail or are not available, left-over potential matches are still pending for further inspection and decision-making. For example, this can be done by providing a user with a minimap and all available information as Samet proposes. With this technique, he declares to "enable toponym resolution with an effective 100% rate of recall". [23]

As shown, the principles of Data Matching can be applied to our case of toponym resolution and fit the design decisions. Of the problems described, these steps deal with spelling variations and errors, abbreviations, annotations, and partly with structural ambiguity when it comes to identifying for which part multiple interpretations are available.

In the latter case, when multiple alternatives stand for resolution due to structural ambiguity, all of them have to be tried. In the best case, only one of the alternatives produces results. In the worst case, a ranking of results as discussed has to further evaluate if and which result is to be preferred.

However, only little use has been made so far of toponym-specific properties. How to utilize those is subject to the next chapter.

4.2 Heuristics for Toponym Resolution

In 2007, Leidner identified 17 heuristics for toponym resolution, naming them /H0/ to /H16/. However, not all of them apply to the domain of historical serial sources, and additional heuristics are required to handle problems not accounted for in these.

In this chapter, heuristics are discussed and chosen for the implementation when seen fitting to the system's design.

From Leidner's taxonomy, some heuristics are not applicable for serial sources as they require a discourse context or textual neighbours. Lists, however, are often alphabetically sorted, reducing all meaning of the neighbourhood to spelling likeness. The heuristics /H2/, /H4/, /H9/, and /H14/ are not further reviewed due to these circumstances.

In toponym resolution, a *geographical focus* is a point or polygon derived from unambiguous toponyms that is used to decide on ambiguous toponyms. When a majority of toponyms refer to a certain region, for the ambiguous toponyms the system will decide to choose the ones that are closest to such a geographical focus. A related strategy is to identify all toponyms in such a way that they return a minimal bounding area; distances to that bounding area are to be kept minimal. Larson showed that even the "simplest regional

approximations [rectangles] provide significant improvements”. [15, p. 11]

However, to apply this strategy to historical serial sources with thousands or millions of lines seems pointless when a far easier strategy is applicable here. For historical documents and especially serial sources, usually a scope is known or can be inferred quickly. Leidner’s /H8/ describes the process of ”thinning out the gazetteer” to such a scope, effectively reducing the data to a region that’s known to hold all the content. The aforementioned strategy will in many cases just produce a bounding polygon resembling the already known scope, so its appliance may as well happen in advance.

With this decision, /H8/ is in use, whereas /H5/ (minimality) and /H10/ (focus) are excluded.

The heuristic /H7/ is named ”Ignore small places”. While it was shown that this approach is ”helpful in some applications”[16, p. 105], it is not desirable for our system as toponyms in the problems at hand often refer to small places.

A ”meta-heuristic”[16, p. 107] is /H11/, which gives higher weight to more frequently occurring toponyms when it comes to calculating - for example - a geometric focus. As no such calculation is planned, this heuristic can not be used. Frequency does play a vital role in the clerical review, however, as more frequent lines are more beneficial to the overall recall when evaluated, and will therefore be given to a reviewer first.

/H6/ states that, if a toponym fits to a capital, that one shall be returned. This will often implicitly be done by /H15/ (explained later), but not used as a rule on its own. Finally, /H16/ describes a preference order when multiple gazetteer systems are used. As this system does not use multiple gazetteers, but a local database (possibly developed from gazetteer integration), no such decision has to be made at this level.

Left for use (and partly discussion) are the heuristics /H0/, /H1/, /H3/, /H8/, /H12/, /H13/, and /H15/. They are listed, named, and described in Table 3.

/HX/	Heuristic's Name	Usage
/H0/	assign unambiguous referent	Partly applicable. A single toponym can be handled this way, but when hierarchy information is available, it has to be taken into account. (as in [25] and /H1/)
/H1/	"contained-in" qualifier following	A frequent pattern that is vital to the interpretation of each line. "Edinburgh, Scotland" shall only return Edinburghs in Scotland. (Note that this condition is weakened to "nearby" when no "in" can be found.)
/H3/	largest population	Applicable on a complete source. When "Berlin" is mentioned as a single toponym, the largest could be preferred when its population is larger than all others by a factor. (Only used for suggestion ranking to lines of clerical review here, since historical population data is often incomplete.)
/H8/	focus on geographic area	A document containing places in France does not need to consider places in other countries. Reduce the database accordingly. (System accepts a polygon describing the allowed area.)
/H12/	prefer higher-level referents	When a place and a subordinate place share a name, the administratively higher object is chosen when no other feature allows further distinguishing them. (Originally meant in a broader sense, where an administratively higher element would always be preferred, even when not in a relation; this does not hold here due to the broadness of data.)
/H13/	feature type disambiguator	When a type is supplied, favour places of that type. The types of places are used for other ranking and selecting operations, too, as will be described throughout the thesis.
/H15/	default referent	Decide a default referent for certain toponyms. Used for single toponyms that are difficult to automatically decide; "this heuristic draws directly on human intuitions about the salience". [16, p. 108]

Table 3: Applied heuristics from Leidner's taxonomy

Grover et al. made "Use of the Edinburgh geoparser for georeferencing digitized collections".[12, title] In this, a toponym resolver was used; after a gazetteer lookup, a variety of heuristics was used to decide on one of the gathered places. While most of them are covered by Leidner, the feature type is used in a different sense: Some types are preferred to other types. "For example, we prefer populated places to 'facilities'." [12, p. 3883]

This goes beyond the heuristic Leidner found, which used types only when one was suggested by the source document.

A similar approach was taken by Volz et al., who attributed weights to types and showed that using them in ranking results may greatly enhance precision up to a point where "only misspellings and other noise in the data prevent 100% precision". [25, p. 5]

The importance of using types to distinguish results quickly becomes obvious. While some types can be excluded from the beginning, the importance of other types on certain patterns can be learned on identified lines. As this work is mainly about lists with thousands of lines, such a learning approach seems promising. Volz et al. plan very similarly "a dynamically adjustment of the ontology weight values". [25, p. 7]

This learning is analogous to how a geographical focus would have been inferred: After a first run, information is gained from the unambiguous results, and utilized to distinguish the ambiguous results. This heuristic will be called "Machine Learning", and is in use.

Another approach to using types has been made by Adelfio and Samet. They utilize geographic containers, population sizes, and feature types to attach a category to each result and return matches category-wise as an online service, *GeoWhiz*. [1][2] This tool explicitly is a "place list disambiguator" as stated on the website; however, its focus is on resolving referent ambiguity of present-time names for relatively small lists.

The core principle still is a possible solution to the problem of route placename resolution; whereas it does not draw a route, it will find the closest aggregation of route marks; an example is shown in Figure 16 in the Appendix. However, placename ambiguity often happens close-by [7] and "[...] the improvement by adding the local context [...] is small". [8] Therefore, whether this technique actually resolves the problem would need testing.

For historic routes, multiple works in recent years tried to solve the problem with differing strategies. Blank and Henrich used distance measures both geographically and in the placenames, showing one of the rare works where imperfect placenames are considered. However, their system only

interpreted 40% of placenames correctly. [6]

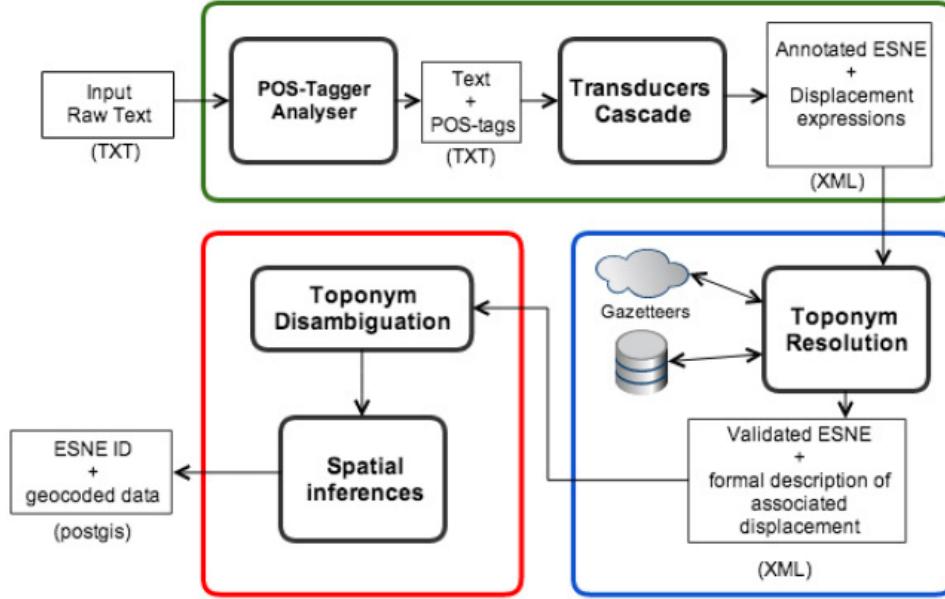


Figure 1: Historic Route Resolver Architecture by Moncla et al. [18]

A main problem again is the presence of referent ambiguity, that especially happens close-by as previously mentioned. [7][8] A solution has been presented by Moncla et al., considering both referent and structural ambiguity and even including defining an area for non-identified toponyms helpful for later completion of gazetteers. [18] This system decides matches with spatial inference. However, it does not yet take into account "variants of toponym names and types in other languages, abbreviations, etc.". [18, p. 191] As such problems are handled via Data Matching heuristics in this work anyway, using their strategy on such processed lines is promising.

Especially when reviewing their system's architecture (Figure 1), a connection of the systems seems a plausible approach, where the Toponym Resolution System (black box in blue box) would be the system developed here, rendering the later "Toponym Disambiguation" superfluous. Moncla et al. report additionally identified toponyms on route data of roughly a third of the original data set by spatial inference. While the strategy of spatial inference is not used here, it shows that the constructed system's output could be beneficial to other systems and problems as well. To acknowledge this, later chapters will describe how to deploy it for use in other projects.

From the insight that referent ambiguity often happens close-by, another heuristic is developed: When all best matches lie within a limited range, a representative shall be chosen according to the most dominant type. This method is part of the Machine Learning, as the most dominant type is decided by unambiguous results. When the “wrong” location is chosen, it does not influence the result on larger scales. To maintain correctness, the point-radius format for georeferencing according to Wieczorek will be chosen, consisting of a point and a radius of uncertainty. [27] In his work, multiple representations for insecurities are investigated. Alternatively, a shape could be returned, using as corners the geo-coordinates of all fitting places within a certain distance.

Another heuristic not found in literature (to the best of my knowledge) will be the “Skips” heuristic. When confronted with a hierarchy of place-names, the case occurs that the hierarchically lowest toponym can not be resolved; this may happen due to the wrong writing of a lesser known place or incompleteness of the gazetteer. In such a case, this toponym shall be skipped and the rest of the hierarchy evaluated. This way, a less precise, but still meaningful marker can be supplied, and the non-identified toponym can be noted to this marker as unfound, possibly resulting in the completion of the gazetteer after manual investigation.

However, before this skipping, it is tested whether the hierarchy can be resolved by searching not in a parent-child relationship, but by searching nearby. When “A in B” does not exist in the gazetteer, maybe “A nearby B” results in a hit. This may be due to gazetteer incompleteness (missing connections) or the information originally meaning nearby, as many inhabitants of rather unknown places will give their residences’ placename in relation to the next big city. This heuristic will be called “Proximity Search” here, and since nearby connections show up in the result, the heuristic does further work towards gazetteer completion.

Should the proximity search not result in indexed places, finally misspellings are considered, first for the hierarchically lowest toponym - as it is potentially less known and therefore more prone to misspelling -, then for all toponyms of a line.

On a completely different note, Berman finds that not only geographical, but also chronological information is of great value to toponym resolution. [4] This property has (also due to the nature of research documents) largely been ignored. For historical documents, a point in time or timespan is

usually known. To reduce the database to names, types and ultimately places of that time (period) will likewise help to lower referent ambiguity and also remove irrelevant options from a return list, simplifying both automatic decisions and the clerical review. However, for such a design, the used gazetteers need to supply time frame information for their places and place attributes.

4.3 Summary

To grant an overview over the employed heuristics, they are now visualized as a tree with multiple categories.

Buscaldi describes three main categories of toponym disambiguation - "map-based", "knowledge-based", "data-driven / supervised". [9] Leidner employs "world knowledge" with the sub-categories "population" and "spatial distribution", "linguistic knowledge" with the sub-categories "local", "statistical", "discourse-level", and "minimality-based". [16, p. 111] Not all of these categories are used in this work, and not all heuristics used in this work fall into these categories. However, they provide a sound base vocabulary to be utilized, and from it, the following representation is built.

The inferred categories are "World Knowledge", "Linguistic Knowledge" and "Data-Driven/Human Knowledge", where the first focuses on information that can be read from a gazetteer, the second references all string manipulation operations, and the third encompasses all machine learning and human knowledge.

Mind that the placetype heuristic is used both *restrictive* and *favouring*; while in some parts it prunes results, in others it creates an ordering. Other heuristics are purely used as a guideline to modify and structure data; the heuristics work in various ways and stages. Their concrete uses will be described in detail in chapter 5.2.

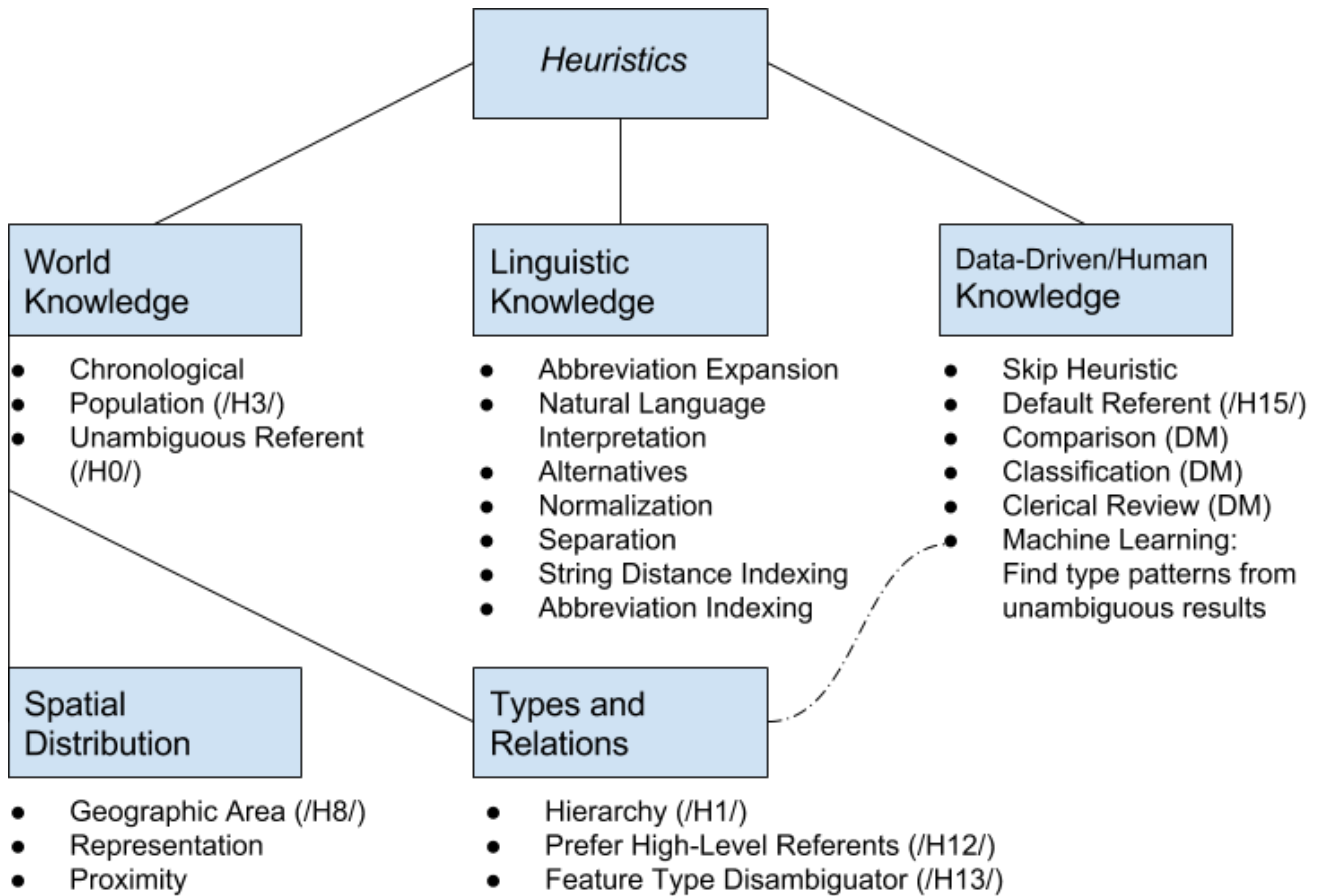


Figure 2: Heuristics for Toponym Resolution on Historical Serial Sources; the dotted line marks that Machine Learning relies on Types and Relations

5 Implementation

To implement the selected heuristics, a program flow must be defined. Also, parallelizable parts of the program must be identified to lower the run-time when millions of lines are supplied to the system.

Further, toponym resolution on historical serial sources is a very specific domain. For a broader approach, the system’s architecture will be held rather abstract to allow for easier expansions should new related use cases arise or for when a different approach in one of the modules is desired.

This chapter presents the practical decisions made.

For a more technical viewpoint, please consider viewing the JavaDoc attached to the implementation. The services written specifically for the GOV are only partially described in this thesis, and some run-time lowering mechanisms that do not influence results are not mentioned, but all developed classes and methods are documented and commented.

5.1 Program Flow

The program flow in many parts resembles the Data Matching Process as presented by Christen. [10, p. 24] The main differences are that not two databases are matched against each other, but one serial source is matched against a database; also, a learning step and additional data sources are introduced.

In Figure 3, the process is visualized with a simplified flow chart. Following this, each step is described, both in which problems are dealt with and which heuristics are employed to do so. A piece of pseudo-code is attached to clarify the order of events within each step.

In the pseudo-code, the word “chain” is used to describe a sequence of specific objects; for example, a toponym chain may consist of three toponyms, where the first is the one we’re looking for, while the next two provide hierarchical information. Such a toponym chain comes with a type chain supplied by the data that denotes favoured types for each toponym-corresponding place. A line of the serial source holds a toponym chain and a type chain, although, when no type information is provided, the type chain may consist purely of wildcards. A place chain is a sequence of places (by ID), and place chains whose placename chain fits the toponym chain and whose placetype chain fits the type chain of a line are desired.

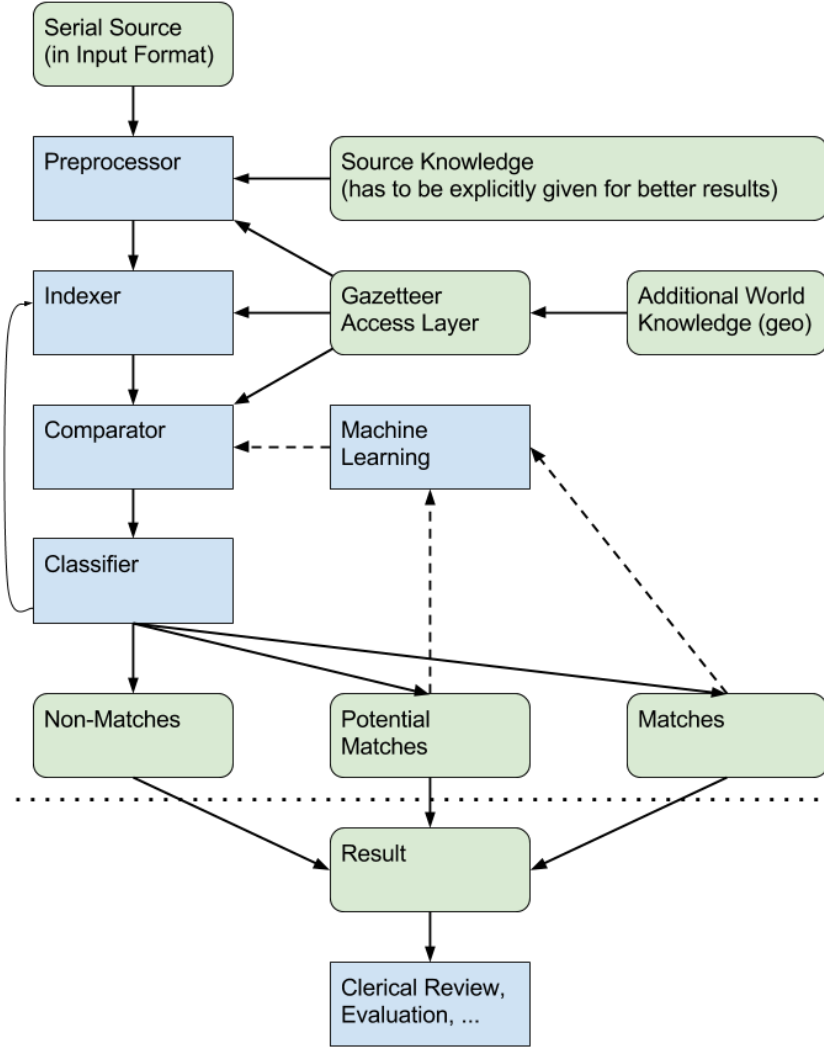


Figure 3: Program flow of the toponym resolution system. While blue boxes denote processing steps, green boxes describe data, and arrows indicate the direction of data flow.

The skips heuristic is denoted by the back arrow from Classifier to Indexer, and the dotted lines imply a re-iteration over data after machine-learned type pattern weights have been gathered. Only when this step is complete, the Result is formed, and can be processed further.

More details about the data during processing follow.

Preprocessor Gets lines from the serial source and source knowledge in form of common abbreviations, rules on natural language parts, and rules on how to filter certain characters. It may also make use of the gazetteer’s typing system to identify type names. The Source Header is the header of the serial source that may contain type and time information. While the preprocessor can operate on the given default values, results will improve when providing problem-specific information.

Problems: Structural Ambiguity. Abbreviations. Annotations.

Heuristics: Abbreviation Expansion. Natural Language Interpretation. Normalization. Separation. Alternatives.

Data: Source Knowledge; Source Header; Serial Source

Result: Lines prepared for indexing (toponyms, types, timespans)

```

for line  $\in$  Serial Source do
    filter/replace unwanted characters(Source Knowledge);
    interpret annotated information(Hardcoded Rules);
    expand abbreviations(Source Knowledge);
    for column  $\in$  line do
        separate toponyms(column);
        find type-denoting information(separated column);
        add column type to first toponym(Source Header);
    end
    find structural ambiguities(Source Knowledge);
    set alternative names(Source Knowledge);
    if line has no timespan then
        use default timespan(Source Header);
    end
end

```

Algorithm 1: Preprocessor

Indexer All places in the gazetteer that the given line might refer to are gathered. It is attempted to find perfect matches; should no such be found, string distances and proximity search are allowed to find places alike.

In the pseudocode, a “toponym chain” describes a path through all toponyms of a line, where one alternative per toponym is chosen. For example, the line $[A, [B, C], D]$, where three toponyms are found, and the second is either B or C , the toponym chains $[A, B, D]$ and $[A, C, D]$ exist. Left-over abbreviations are interpreted, too, where “Abc.” is read as regular expression “A*b*c*”.

Problems: Spelling Variations and Errors. Abbreviations. Data Quality.

Heuristics: Hierarchy. String Distance Indexing. Abbreviation Indexing. Default Referent. Proximity Search. (Skip Heuristic Reentry Point.)

Data: Preprocessed Lines; Gazetteer Access (GA); Default Referents

Result: Places are indexed to lines

```

for  $line \in \text{Preprocessed Lines}$  do
  if only a default referent is left to be processed then
    | index default referent(Default Referent);
  else
    | alternatives = build all toponym chains(line);
    | search for toponym chains as hierarchy(alternatives, GA);
    if nothing indexed to line then
      | proximity search for toponym chains(alternatives, GA);
    end
    if nothing indexed to line then
      | search for toponym chains with string variance on
      | child(alternatives, GA);
    end
    if nothing indexed to line then
      | search for toponym chains with string variance on
      | all(alternatives, GA);
    end
  end
end

```

Algorithm 2: Indexer

Comparator Before Machine Learning, the Comparator uses toponym string distance, type matching, and geographical proximity of places to, for example rivers, to compute a toponym, type, and geo comparison value. After Machine Learning, the type patterns fitting the place chain are used to decide between the best-valued places.

Mind that in both cases, the toponym chain is “spread over” the place hierarchy. For example, assume toponym chain $[A, B, C]$ exists. The Place P is indexed, and P is in the place hierarchy $[P, Q, R, S, T]$. It may happen that A corresponds best to P , B to R , C to T . In such a case, $[A, B, C]$ is compared to $[P, R, T]$, while Q and S are skipped. When speaking of placetypes, this shortened hierarchy types are meant. The Comparator also acknowledges proximity hits from a proximity search, interpreting such as if they were a hierarchy.

The Comparator also considers whether the place hierarchy, place-names, and placetypes hold during a line’s timespan. While place-names and placetypes always have to be valid during a line’s timespan, the hierarchy can be more loosely interpreted, as actual hierarchy and proximity are mixed, making the data more fuzzy anyway. For example, in the year 1870, a person might have given $[A, B]$ as residence, even though A was only introduced as a child of B in 1875. In that case, the proximity will still hold, and it still is a meaningful information, especially since B may also be interpreted as a region. For such cases, some points are subtracted; per default, 5% of the toponym points. This percent value can be freely configured by a user.

Problems: Spelling Variations and Errors. Data Quality.

Heuristics: Feature Type Disambiguator. Comparison (DM).

Data: Lines with indexed places (IL); Gazetteer Access (GA);
 Additional World Knowledge (AWK)
Result: Line-Place pairs have a comparison value
for *line* \in *IL* **do**
 | **for** *indexed place* \in *line* **do**
 | | rate accordance: line toponyms to placenames(GA);
 | | rate accordance: line types to placetypes(GA);
 | | rate proximity: place position to geo reference(AWK);
 | **end**
end

Algorithm 3: Comparator (Before Machine Learning)

Data: Lines with indexed places (IL); Gazetteer Access (GA);
 Type Pattern Weights (TPW)
Result: Line-Place pairs have a type pattern relevance value
for *line* \in *IL* **do**
 | **for** *indexed place* \in *line* **do**
 | | rate relevance: placetypes(TPW, GA);
 | **end**
end

Algorithm 4: Comparator (After Machine Learning)

Classifier Based on the comparison values, the Classifier decides the match status of a Line-Place pair, where the most important information is the toponym value, followed by type, then geo proximity. Should all best fitting lines be in a hierarchy, the top level object is chosen. To resolve close-by ambiguities, representation attempts will be performed. After Machine Learning, it will be simply checked if one of the places has a best type pattern relevance above a threshold in comparison to all other candidates; if none is dominant, a subset of best-fitting lines is formed, and on this, the normal choosing mechanisms are performed. The determined bins [*no_match*, *possible_match*, *match*] are also used later to determine lines for a clerical review.

Problems: Referent Ambiguity.

Heuristics: Unambiguous Referent. Prefer High-Level Referents. Representation. Classification (DM).

Data: Compared Lines; Thresholds

Result: Decision on a match

```
for line  $\in$  Compared Lines do
    sort places in bins according to comparison value:
        no_match, possible_match, match;
    if bin match has a single best-fitting place then
        | mark place as match(best-fitting place);
    else if bin match has multiple best-fitting places in a place
        hierarchy then
        | choose top-level place(best-fitting places);
    else if bin match has multiple best-fitting places within
        representation radius then
        | determine a representative place(best-fitting places);
    end
end
```

Algorithm 5: Classifier (Before Machine Learning)

Data: ML-Compared Lines; Thresholds

Result: Decision on a match

```
for line  $\in$  ML-Compared Lines do
    bin = collect places with value good enough;
    if bin has a single place then
        | mark place as match(best-valued place);
    else if bin has multiple places in a place hierarchy then
        | choose top-level place(best-valued places);
    else if bin has multiple places within representation radius then
        | determine a representative place(best-valued places);
    end
end
```

Algorithm 6: Classifier (After Machine Learning)

Machine Learning In the Machine Learning step, decided matches are used to learn about type patterns in the document worked on. Should a type pattern have a sufficiently clear dominance, it will be used in the iteration to decide on potential matches.

A type pattern consists of a type chain $[\cdot, \cdot, \cdot]$ with all elements being types and a value that resembles its weight. For example, when of all matches that took place on toponym chains of the size three 90% share the type chain $[Q, R, S]$, then the type pattern is $([Q, R, S], 0.9)$, where 0.9 is the weight. As places may have multiple types, the indexed places to a toponym chain may match multiple type patterns. Then, their weight sum is used.

The skips heuristic is also employed. Assuming a toponym chain $[A, B, C]$ could not be resolved on A , and $([Q, R, S], 0.8)$ is a type pattern, then for $[B, C]$ all type patterns are shortened by 1, so $([R, S], 0.8)$ is used to reflect on the toponym chain having three entries originally.

Problems: Referent Ambiguity.

Heuristics: Machine Learning.

Data: Classified Lines

Result: Type Patterns

patterns = map<type chain, weight>;

for *line* \in *Classified Lines* **do**

if *line* has a match **then**

for *type chain* \in *line's match place chain* **do**

if *type chain* \in *map* **then**

 add weight 1 to pattern;

else

 add new pattern to patterns with weight 1;

end

end

end

end

normalize weights so that for each length, the sum of weights is 1;

Algorithm 7: Machine Learning

Clerical Review Setup In the clerical review, lines that the automatic resolver could not decide on are evaluated with human decisions. Lines that have potential matches can offer their candidates. Lines with no matches will have to be researched manually. Either way, in documents with millions of lines, the lines with a higher frequency should be evaluated first to gain increased returns for the same time invested.

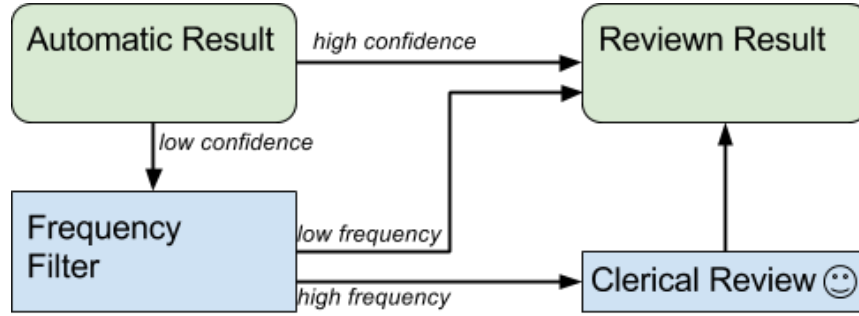


Figure 4: Choosing the most promising lines for a Clerical Review.

Speaking Design Pattern, this prodecure employs *High Confidence Switching* and a *Magic Filter* as described by Lofi et al.[17], meaning only data the system is not confident in will be propagated to a manual review and such data will be sorted by frequency, possibly cutting off all lines below a certain threshold. See Figure 4.

Problem: System was not able to identify a line.

Heuristics: Clerical Review (DM). Population.

Data: System Result(SR)

Result: List of promising lines for human inspection

forall the most frequent unresolved lines on skips level $0 \in SR$ do

if line has multiple places in bin match then

 | list matches in order of population size;

else if line has multiple places in bin potential_match then

 | list potential matches in order of population size;

end

 write lines to clerical review file in order of frequency;

end

Algorithm 8: Clerical Review Setup

Finally, an actual human goes through this list and fills in his decisions in the System Result. Currently, there is no user interface for this.

Evaluation The quality of the results should be evaluated, as the quality for toponym resolution may heavily vary depending on the source. [9] This problem-inherent property requires a project-wise evaluation to guarantee quality measures. For an automated evaluation, a gold standard is required that contains a manually resolved subset of the problem.

Problem: The system’s result quality needs to be inferred.

Solution: Comparison with gold standard.

How a gold standard should be constructed will be discussed later on. Mind that a match in the system result on a higher skips level will always result in the distance deciding if it’s *TruePositive*, *ErrorPositive*, or *FalsePositive*. The case *ErrorPositive* is not originally contained in binary classification, but added to distinguish between system mistakes in more detail; a normal binary classification could only decide whether a line is identifiable, but not what the correct identification is; this extension handles misidentifications.

Should the case occur that the system result is on a higher skips level, but the gold standard is empty, then this line can not be interpreted, as the gold standard does only contain information on skips level 0 (which equals not using the skips heuristic) and therefore the quality of the system result is completely unknown in these cases.

The sorting of lines into the extended binary classification bins is shown in Algorithm 9. More on Evaluation, the interpretation of *ErrorPositive* results, and the significance of distances in chapter 6.

Data: System Result(SR); Gold Standard(GS)

Result: Recall, Specificity, Precision, Accuracy, F-Measure (possibly
for various kilometer tolerances to the Gold Standard place)

```
for lineGS ∈ Gold Standard do
    find corresponding lineSR(lineGS, System Result);
    compare result in lineSR to result in lineGS;
    if agreement then
        if both matched nothing then
            | note as TrueNegative;
        else if both matched the same then
            | note as TruePositive;
        end
    else if only GS has a match then
        | note as FalseNegative;
    else if only SR has a match then
        | note as FalsePositive;
    else if disagreement then
        | note distance to line;
    end
end
if evaluating with distances then
    for various kilometer tolerances do
        for noted distances in lines do
            if distance within tolerance then
                | add line to TruePositive for this tolerance;
            else if distance above tolerance then
                | add line to ErrorPositive for this tolerance;
            end
        end
    end
end
else
    | note lines with distances as ErrorPositive;
end
compute quality measures for classification of each distance;
```

Algorithm 9: Evaluation (simplified: no skips mentioned, these will be handled by the different evaluation modes described in chapter 6)

Not all heuristics have been used in the previous listing of system parts. The left-over heuristics (Geographic Area (/H8/), Chronological) are used in generating the *GazetteerAccess*, practically reducing the entries to a relevant subset of places and their attributes. This task can both be solved by the system itself, when sufficient data is present, or must be done in a preparation step by the user. Usually, the reduction of a data-set in time and space is a rather trivial problem that relies on simple filters and only depends on the data provided by a gazetteer. While the system is constructed for connection to any gazetteer and does work on the bare minimum of toponyms without any further information, result quality will decrease a lot, should a gazetteer not provide sufficient information to disambiguate places with such parameters.

The noted "Additional World Knowledge" is in this case a set of coordinates for major rivers in the region of the German Empire. Depending on what the source contains, other information (i.e. names of mountains, landscapes, ...) that does not occur in the gazetteer might be useful to distinguish toponyms.

For parallelisability, each line is worked on independently of all other lines most of the time. Preprocessor, Indexer, Comparator, and Classifier are able to process all lines in parallel, sometimes with waiting points to learn from partial results. While the Comparator does rely on the Machine Learning step in the second iteration, the required information is gathered during Classification with minimal locking to avoid race conditions and can then be used in parallel for all left-over lines again.

Due to this, the program will linearly keep up in speed with the supplied hardware. Further details on this will be discussed in chapter 5.4.

5.2 Input, Output, Gazetteer Access

In this chapter, the system’s interfaces and overall connection to the outside world are discussed. As already seen in Figure 1, a toponym resolver has three main connections: It has an input, the to-be-resolved lines; an output, the connection between such information and a place; and requires a gazetteer that holds a set of places as described and possibly further information that will be subsumed as gazetteer information here.

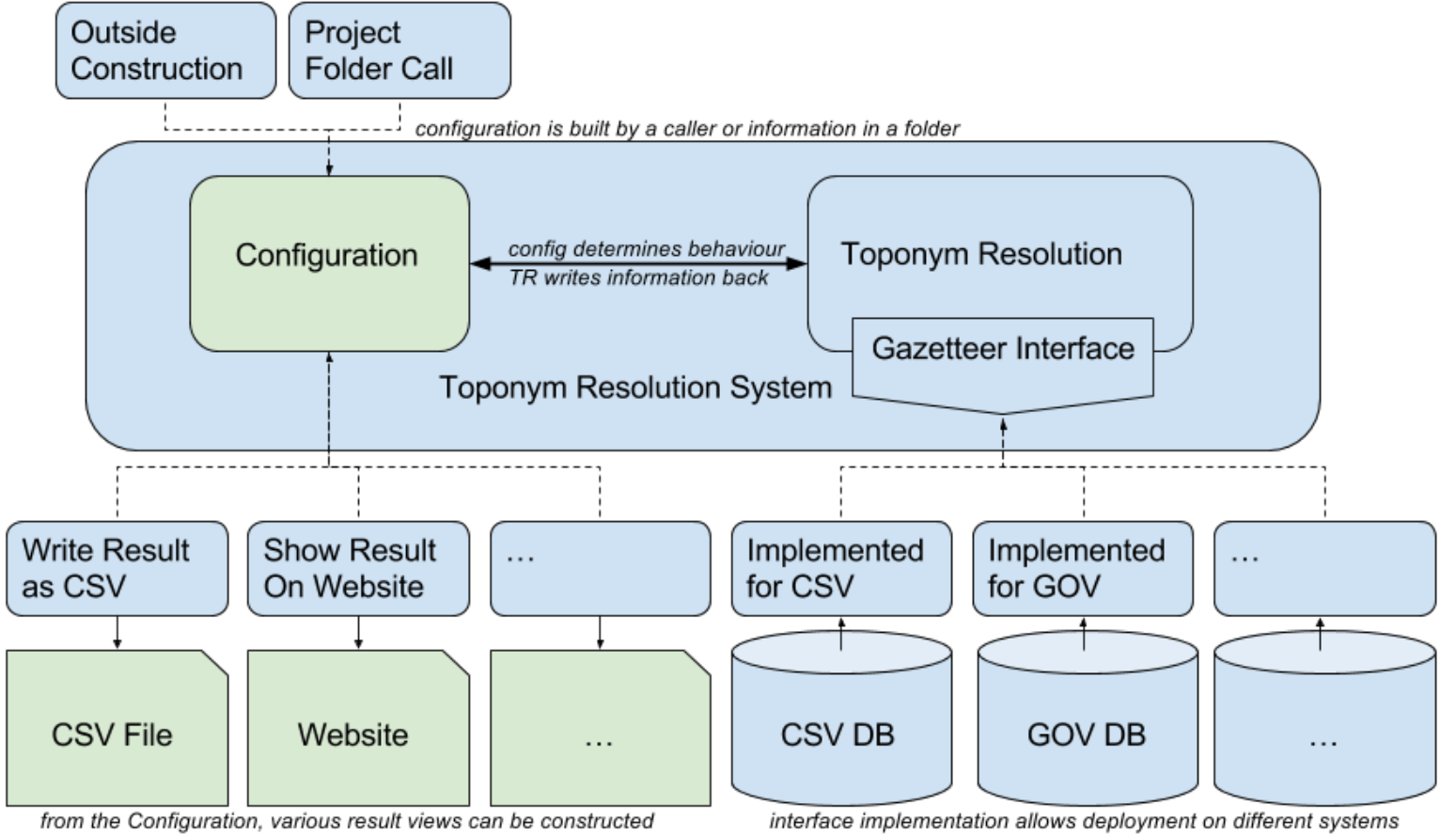


Figure 5: Input, Output, Gazetteer; dotted lines signal alternatives

The here used solution for these necessities is given in Figure 5. In this representation, the modules Preprocessor, Indexer, Comparator, Classifier, Machine Learning, Clerical Review Setup, and Evaluation are all encapsulated as “Toponym Resolution”. Their gazetteer access is realized by an

implementation to an attached Gazetteer Interface that describes all the methods required by the system. Exchanging the gazetteer is as simple as re-implementing the Gazetteer Interface for a new environment.

For this, two implementations are given: First, the system is capable to build an In-RAM-Database on its own from CSV files. While this does need excessive amounts of RAM, such information can be constructed with knowledge of the configuration's requirements, thus minimizing its size and removing filter steps for types, locations, and timespans, saving time. Second, an implementation using the GOV's database is given that uses the actual website database and services provided by (and partially re-written of) the GOV.

For Input and Output, a Configuration object encapsulates all resolution and control data. This object contains all thresholds and behavioural instructions as well as input, line-place pairs, match states, and so on. With it, a detachment of data and logic takes place, also allowing for multiple configurations (and attached tasks) to be evaluated by the system at the same time.

Regarding Input, such a Configuration can either be constructed outside of the system or can be constructed by the system itself when given a folder containing the necessary data. An access for a website user of the GOV is attached that uses the Outside Construction to resolve a request.

When using the website, four access points (see Figure 12 in Appendix) for a resolution process are given:

Quick Search: The user can write a single line and ask the system to resolve it. It will be resolved in a broad default search environment and return a map and list of fitting places. (see Figures 13, 14)

Advanced Search: The user can write a single line, give further parameters, and ask the system to resolve it. Parameters are *restrictive* (region) or *favouring* (preferred type or type group), where timespan fits in both categories (see 5.1). It will be resolved in the parameter-according changed default search environment and return a map and list of fitting places.

File Resolution: The user can upload a CSV file with multiple lines, give further parameters, and ask the system to resolve it. Parameters are *restrictive* (region, type groups, toponym language), where timespan again is both *restrictive* and *favouring*. The file will be resolved in the parameter-according changed default search environment and return a map and

list of fitting places, where for each match that was found, one marker respectively one list entry is set. The user also has the option to download a result file with GOV ID and coordinates attached. (see Figure 15)

Project Resolution: The user can start a project folder evaluation that works just as a regular project folder evaluation, and does not return a view, but simply writes the CSV files to a result folder. This access point is supposed to be used for large files or configuration-intense projects. Currently, access is restricted to administrators.

All figures regarding these are attached at the end of the document.

For the CSV result files, the system writes the information inside the Configuration to files. Three files are gathered from this: First, a complete result file that contains all indexed places and their rating per line as well as a choice and the place chain for the choice. Second, a simplified version of this containing only matches and nothing where no decision was made. Third, a Clerical Result Setup file that contains suggestions to the most promising lines.

A project folder can contain the following files to determine behaviour:

- alternatives.csv** In this, toponym strings and their alternatives are listed. For example, some files may refer to the region “elsaß”. However, it is not part of the GOV, which instead contains “ober-elsaß” and “unter-elsaß”. To make such lines resolvable, the two rows “elsaß|ober-elsaß” and “elsaß|unter-elsaß” are added, where ‘|’ denotes the column separator. (In the actual project, ‘\t’ is used.)
- dbTypes.csv** Contains allowed types, type groups, or type classes, with one entry per line. Places without such types will be pruned from returns.
- defaultReferents.csv** Contains rows of “toponym|GOV ID”, where the default referent heuristic is used with the toponym.
- gold.csv** Contains rows of “Line Number|GOV ID” referring to the line numbers in the lines.csv, where the first line to be resolved has ID 1.
- inlineTypes.csv** Contains rows of “word|one or multiple types and type groups separated by a semicolon” where words are in-line occurring type descriptors. While the GOV actually contains a typename to typenumber system, it is not guaranteed to fit on historical serial sources, where one

typename might actually refer to multiple GOV types and names are not guaranteed to overlap. For example, many lines will simply note “Kreis” (district) where the GOV has a more sophisticated differentiation to *Kreis*, *Kirchenkreis*, *Landkreis*, *Stadtkreis*, *Reichskreis*, *Amt* (*kreisähnlich*), *Kreishauptmannschaft*, *Kreisdirektion*, *Kreisgebiet*, *Ritterkreis*, and *Kreis* (*mittlere Verwaltungsebene*). Only considering the one type that’s actually named “Kreis” may be insufficient. To use all, create an entry “Kreis|Kreis;Kirchenkreis;...” or note a subset or type group that is supposed to fit.

While some of them are more, others less likely to be meant, a document-specific list of plausible connections from typenames to GOV types will improve quality of type favouring. For the default search environment, the GOV types and their typenames are used naively. As types are very specific, the system allows half the type points for sharing a type group with the actual type. To gather an actually good type vocabulary for present-time users is a problem of its own.

- lines.csv A file containing the lines to be resolved, formatted as described in the following chapter.
- polygon A file containing a string describing the borders of a region. Only places inside the region are used.
- substitutions.csv A file containing rows of “original|substitution” where the Preprocessor tries all rules on each line of input and replaces all originals with their substitutions. This can, for example, be used to extend abbreviations that commonly occur and are safe to expand. Also, irrelevant phrases or words can be removed with this, and common spelling mistakes in a document can be corrected.

Whenever a restrictive file is missing or empty, that is interpreted as “no restriction”. For example, an empty polygon means the whole world, and allowing no types is the same as allowing all types, else the system could not act at all.

In addition to these files, the “additional world knowledge” as used before exists, but due to its globally valid nature, is not project-bound. To this, the following file belongs:

- geoRef.csv A mapping “GeoName|Longitude|Latitude”, where a *GeoName* is for example a river name and can occur as many times as desired (for example to describe the path of a river). Right now, only rivers are gathered here, with their coordinates being read from OpenStreetMap.

5.3 Data Format

For the implementation, a more tangible description of the problem has to be made in the form of a structured input of the system. A table of the format as in Table 4 is the most complex input the program is able to read so far. Toponyms may contain wildcards (“?”, “*”) and explicit notions of fuzziness (“~”); those work on all website searches.

Frequency	Timeframe	[types]	more columns as before
[integer]	[Timeframe]	[String]	more columns as before
Frequency	Timeframe	group_8	group_1
5	2000-01	Edinburgh	Scotland
7	1000 to 1999-04-01	Kiel	Schleswig-Holstein, Deutschland

Table 4: A specification of the input and an example

Mind that the example already describes a more complex input. In this case, information is already split and frequencies are given. Usually, lines will not be pre-aggregated, and in many cases, no information split into multiple columns is given. Furthermore, not even types or type groups² might be supplied by the user. In the most simple case, a simple string like “Edinburgh, Scotland” is the full input.

Should the input be as structured as in the example, the hierarchy is assumed to be rising from left to right. Systems like *nominatim* also try different orders. As it will usually be possible to determine the order on historical serial sources, the toponyms can be fed in the right order, and the additional heuristic of trying toponyms in a backwards order is not needed here.

Mind that the second example column also contains multiple toponyms. In such cases, the type favouring is only applied to the first toponym.

This input format has been chosen as it allows for pre-aggregation (if not given, “1” is used for each line), considers that typing information might already be present (if not given, no type is favoured) and that a source might already be split in multiple columns (if not given, just one column will suffice).

Due to most of the fields being optional, simple requests can easily be translated into such an input format, presenting a standard approach to all

²8 is for place of residence, 1 for political administration; GOV type groups.

inputs.

In the program, for the second example line, the structure is broken up and translated into the following fields during pre-processing:

```
Line
(Line Number 2) (Frequency 7) (Timespan 1000-01-01 1999-04-01)
toponyms: "kiel", "schleswig-holstein", "deutschland"
types: "group_8", "group_1", unknown
```

With `unknown` being the wildcard, a type favouring for each toponym is described. Furthermore, multiple types might be supplied for a column. Also, multiple toponyms might apply for a position when ambiguities arise. In such cases, not a simple string, but a list of strings fills a position, each entry representing an alternative for the position.

For example, the input "Elsaß" is ambiguous. "Elsaß" as a region does not exist in the GOV, and therefore, "Ober-Elsaß" and "Unter-Elsaß" are used to try the two regions that might apply. A minimal input "Stemlisberg, Elsaß", where the column type, timespan, and frequency are empty, would lead to the following Line.

```
Line
(Line Number 1) (Frequency 1) (Timespan unbounded)
toponyms: "stemlisberg", ["ober-elsaß", "unter-elsaß"]
types: unknown, unknown
```

This situation describes that both alternatives for the second position will have to be tried. In this case, "Stemlisberg" in "Ober-Elsaß" will be found. A comparable situation arises when structural ambiguities are found. An array ["`placename on the river`", "`placename`"] to describe toponym alternatives is constructed then.

Such structured CSV input can be uploaded to the website for resolution or put in a project folder for resolution in a more complex environment. The column header "Timespan" can be replaced with an actual timespan (like "x to 2015") that will be used as default then. On the website search, this default can only be given by the user via the input form.

For all searches on files, for each match the dominant decision strategy is named in the result, coming from a pool of decisive heuristics.

5.4 Requirements and Software

The implementation of the presented system is done in a Java1.8 IntelliJ IDE project in a Grails Environment, and has been nested into the original

system code of the GOV itself.³ A `readme.md` is provided to help navigating folders. It is based on a Data Matching program I have built in the previous semester that now employs logic from a more sound theory, as prepared in the previous chapters. All necessary system parts and a few example projects are supplied. For use on new problems, additional files may be required for good results, as they may contain new vocabulary.

The CSV version of the Gazetteer Access uses great amounts of RAM. 20+GB of RAM were taken on a document with 1.3 million lines and about 350.000 places in the database. While this amount of RAM is not required, it certainly helps with lowering run-time. Per default, such use of Java is not expected by the IntelliJ IDE; the program should be run with the VM option `-Xmx` announcing how much RAM you have to spare and `-XX:-UseGCOverheadLimit`, else the system might falsely detect an overflow during certain object creation stages.

The GOV version of the program requires access to an elasticsearch database and the services of the GOV. As of writing this, due to a problem in the connection of IntelliJ IDE and Java1.8, the project can not be started from within the IntelliJ IDE for this mode, and instead has to be started via command line. Files containing the required command lines are supplied. Mind that the elasticsearch database itself requires RAM in greater amounts⁴, the lowest valid number on team elastic's page being 4GB, with keeping the same amount free for other database-relevant processes. The Grails Cookbook advises to use 1GB⁵ for the grails application itself. The Toponym Resolution process can use up multiple GBs of RAM, completely depending on the problem size. For the online features, a limit to the uploaded file's length should be found and enforced.

On the available system with 8GB RAM, with elasticsearch running on 1GB, grails running on 1GB, Mozilla Firefox and the IntelliJ IDE being opened, an uploaded file of 200 lines took below 2 seconds per line to evaluate, resulting in the file being resolved in about two minutes. This time will likely decrease with sufficient amounts of memory, and it has been noticed that database response times are likely to decrease over time.

However, the main concern of this work is not optimization (although parallelizable parts have been parallelized), but the quality of the result. In the following chapter, the evaluation is described.

³<https://gitlab.genealogy.net/dse/gov>; evaluated version tag: *evaluationVersion*

⁴according to <https://www.elastic.co/guide/en/elasticsearch/guide/current/heap-sizing.html>

⁵<http://grails.asia/step-by-step-tutorial-on-how-to-host-your-grails-application>

6 Evaluation

Buscaldi describes the problem of evaluating toponym resolution as following:

The lack of a rigid evaluation procedure and a standardised test-set has represented a major issue in the evaluation of toponym disambiguation methods. Most of the initial works published calculated their results under particular conditions and using small test-sets that makes reproduction of their results difficult. In some cases, the same method applied to different test-sets obtained very different results [...].” [9]

He goes on to name existing test corpora; however, they do not fit the domain of historical serial sources. To evaluate the quality of results, four gold standard files have been crafted from two actual historical serial sources, covering 800 lines in total, gathered by two different strategies. With this varied data, a broader view on the system result’s quality is presented.

While a clerical review is both planned and advised for a real toponym resolution, it is not taken into account for the evaluation, as it is concerned with finding out how well the automatic parts fulfill their assigned jobs. A clerical review would also return results akin to the gold standard, as both are produced in a comparable fashion.

In the following, premises are formulated and assumptions made to describe with which guidelines the gold standard was formed and how its evaluation results are to be interpreted. Usage Scenarios are defined and evaluation results for them presented. Finally, comparisons to other freely available toponym resolution systems are made; however, comparability is rather low, as will be described in its own chapter.

6.1 Assumptions

The results are evaluated towards the chosen database. Whenever the database does not contain a location even though it exists, this is not interpreted as a fault in the program. Instead of the ”real truth”, whatever that may exactly be, the ”database truth” is taken as measure, as the Toponym Resolution System is tested, not the quality or completeness of the used gazetteer.

Therefore, this evaluation is not concerned with missing data, but with how well the presented heuristics and their implementation work on what is available. To gain a better base of data, methods of gazetteer integration

and further gazetteer development need to be done, which is not the focus of this work. A step towards gazetteer development is taken by implicitly attaching toponyms that could not be identified to a higher hierarchy level during the skips heuristic.

Note that these assumptions have been inspired by the assumptions made by Doerr. [11] They are only used to evaluate the system in itself, as the local database truth might differ from other system’s database truth. Inter-system comparability will be discussed after the evaluation of the system in itself.

6.2 Gold Standard Construction

Problem 1 (P1) is a roughly typed list of two columns dating back to 1870 and denoting the residencies of missed soldiers, where the first column is named “Ort” (place), the second “Kreis” (district). For the system, it is assumed that the first denotes a living space, the second a political entity somewhere around the district level; these assumptions are used as CSV column types. All project folders begin with *project_1870*. The data is rather clean with a low amount of abbreviations and natural language; sometimes, multiple toponyms are located in the same column. It holds about 60.000 different lines.

Problem 2 (P2) is an untyped list of one column dating back to 1914 and denoting the birthplaces of WW1 soldiers. It is assumed that the lines point to either a living space or a political entity; this assumption is used by reducing database returns to these types. All project folders begin with *project_wk1*. The data is of mixed quality, most lines hold multiple toponyms, and about 1.2 million different lines are contained.

Choosing Method 1 (CM1) is by frequency: more common lines are more likely to be chosen, which leads to representative data set for the problem; however, no line can be chosen multiple times. Due to this, “Berlin” is very likely to occur, but it can’t occur more than once.

Choosing Method 2 (CM2) is by problem: each different line has the same chance to be chosen, again, without duplicates. This results in a more difficult subset of the problem, as less common lines are more likely to contain mistakes, and tend to contain smaller places. Lines that fit the “Default Referent” heuristic are less likely to be chosen.

For each pair $\{P, CM\} \in \{P1, P2\} \times \{CM1, CM2\}$ 200 lines from P were chosen by CM , resulting in 800 lines overall. For each of these lines, I tried to find a corresponding place in the GOV’s database fitting assumptions regarding type, and noted it as gold standard. I did not use the skips

heuristic in this manual linkage: If the first toponym was not identifiable, the line was left empty. For the construction, I used human intuition and no strict set of rules: When a place seemed to be a sufficiently plausible fit, I used it.

To illustrate why human intuition was used instead of any strictly defined procedure, an example identification is described: $\{P1, CM2\}$ contains the line “Smolary, Chodziesen”. The second toponym (also referred to as “Chodgiesen” in another line) is very likely to refer to the place with toponym “Powiat chodzieski” in the database, where “powiat” is polish for *county* (*Landkreis*) and “Chodzieksi” is sufficiently close to “Chodziesen”, especially when considering polish-german language differences and the fact that this county actually contains a place named “Smolary”.

6.3 Evaluation Modes and Usage Scenarios

Three evaluation modes are defined to test the system, each with its own way to interpret the system result towards the gold standard data. These evaluation modes correspond to usage scenarios, that, too, will be described in the following list. In all cases, an extended binary classification will be used to gather quality measures, as the usual binary classification does not fully correspond to the problem at hand. Normally, a binary classification describes a property as either present or absent; in this case, however, mismatches may occur that do not naturally fit into any of the categories.

The Evaluation Algorithm (see Algorithm 9) sorts the lines into extended binary classification fields as described in Table 5.

	Positive	Negative
True	Both, Acceptance Condition Met	Neither
False	System Result, No Gold Result	No System Result, Gold Result
Error	Both, Acceptance Condition Not Met	—

Table 5: Extended Binary Classification, described by whether the System Result and Gold Standard hold an ID for a line

Mode 1: *ID Perfection* describes a sorting where the Acceptance Condition is described by a perfectly matching ID, meaning the exact same place was chosen. Whenever the system used the skips heuristic, this is interpreted as a Negative, since the system actively noted that the base toponym could not be identified. *Usage Scenario: Further information to the actual place is desired. In such cases, the exact ID must fit, and*

even two places in the same spot won't fit if the ID differs. While this is not the original target of Toponym Resolution, the system's quality for such a need is also evaluated.

Mode 2: *Distance without Skips.* Instead of the ID, the geo-positions of the chosen IDs are compared. The Acceptance Condition is met when they lie within a specified distance to each other. Skips is a Negative, just like above. *Usage Scenario: The user wants to draw a map (or somehow only use the geo-position). In such cases, depending on the resolution of the map or other limiting factors, a certain distance to the correct location is tolerable, as it's either not noticeable or irrelevant to the problem.*

Mode 3: *Distances with Skips* works like Mode 2, but accepts the position of places gathered with the skips heuristic. By comparing Mode 2 and Mode 3, it will be shown that the skips heuristic is a beneficial heuristic as it provides meaningful pointers.

This is the very same fashion Tobin et al. evaluate their system in; their “exact evaluation” works on ID sameness as Mode 1 does, their “proximity evaluation” works on geo-coordinates as Mode 2 and 3 do. Their proximity distances are 5km, 25km, 50km, and 100km; however, they did not use the variant of binary classification as described here. [24]

For Mode 2 and 3, the usage scope plays a vital role: A map of a continent, a map of a special region or hierarchical element (nation, district, community) or a single marker with high resolution could be desired. Then there's also always the possibility that we're working on route data, which could correspond to any of those levels.

To comply with these needs, deviations of falsely identified places are computed to declare how influential errors are to the result. For each of the user scopes, a differing level of inaccuracy seems acceptable. To evaluate how well the system works for various allowed distances between gold standard and system result position, these allowed distances are tested: 0km, 1km, 5km, 10km, 20km, 30km, 50km, 100km, and 200km.

Quality Measurement Formulae are changed according to the Extended Binary Classification:

Recall: $TruePositive / (TruePositive + FalseNegative + ErrorPositive)$

Precision: $TruePositive / Positive$

Accuracy: $True / Population$

6.4 Evaluation Results

For all data sets, the project folder files as specified in chapter 5.2 were constructed. The searches were geographically limited to Europe, known occurring inline types were linked to types in the GOV, and the search was limited to the types as described in the problem sets. For substitutions, a shortened file from the previous semester was used that was written for *P2*. In a later step, these rule files will be removed to show their impact on the system result. Furthermore, the partaking of decisive heuristics on the overall result will be pointed out. Charts here are meant to give a rough impression; for exact numbers, see chapter 8.4.

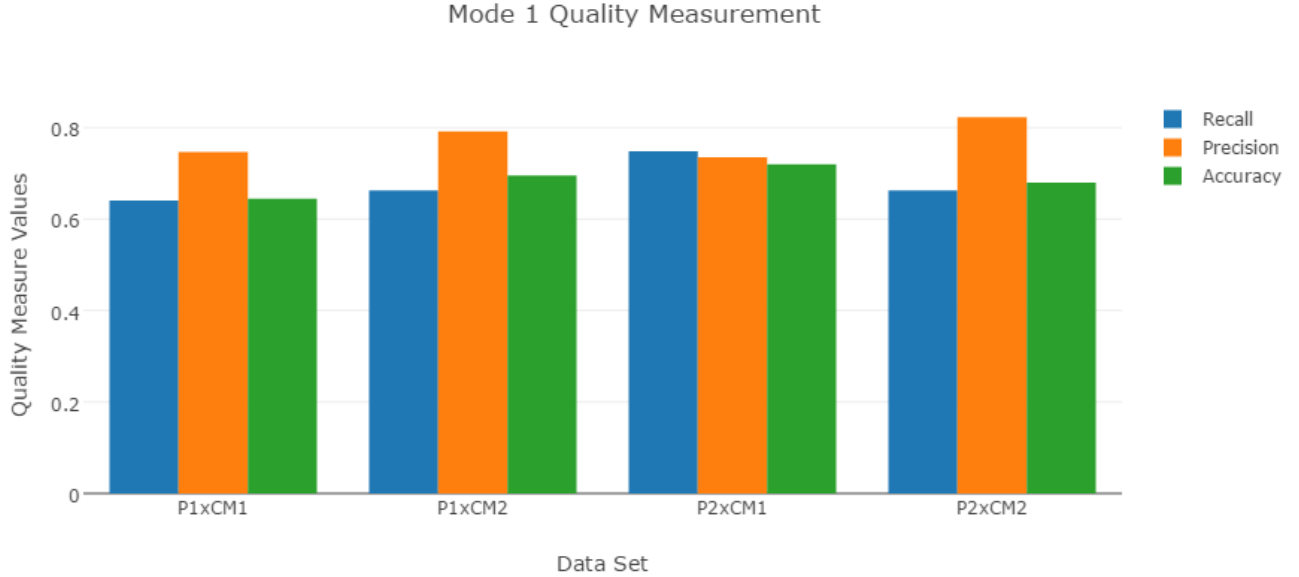


Figure 6: Quality Measurement Values for the Mode 1 Evaluation

For the evaluation of Mode 1, results are rather poor, as can be seen in Figure 6. Partly, this is a result of how the gold standard was constructed, which is by intuition. When having to decide between multiple places close-by, I frequently made a choice different to the program by my personal preference for what type seems most relevant. The system, relying on actual data about unambiguous type frequencies within the problem, decided otherwise. Results drastically improve on allowing a distance of 20km

around the position the gold standard points to, as can be seen in Table 6. Furthermore, in all scenarios the usage of Mode 3 increased the recall rate, although the impact of the skips heuristic varies.

	Mode 1	Mode 2	Mode 3
$P1 \times CM1$	0.6402	0.7831	0.8617
$P1 \times CM2$	0.6629	0.7966	0.8418
$P2 \times CM1$	0.7485	0.8947	0.9006
$P2 \times CM2$	0.6629	0.7314	0.8171

Table 6: Recall Comparison, 20km distance allowed on Mode 2, 3

To further distinguish Mode 2 and 3, the various quality measures are compared for each data set in Figure 7. Usually, recall and accuracy both improve, while precision drops. $P2 \times CM1$ is the only exclusion from this rule. When aiming for high recall, Mode 3 is definitely useful; when aiming for high precision, its use is questionable. Overall, the results improve.

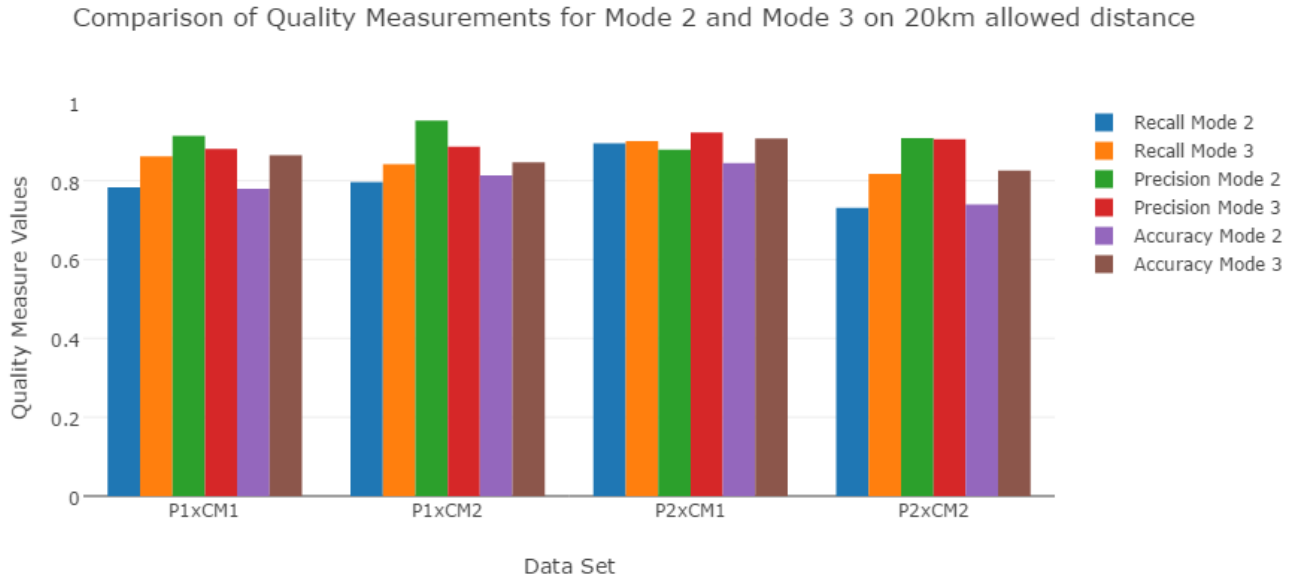


Figure 7: Quality Measurement Values for comparing Mode 2 and Mode 3

From this, it is gathered that the skips heuristic is usually helpful and enabling it returns the best results. In Figure 8, the development of quality measures over allowed distance is displayed.

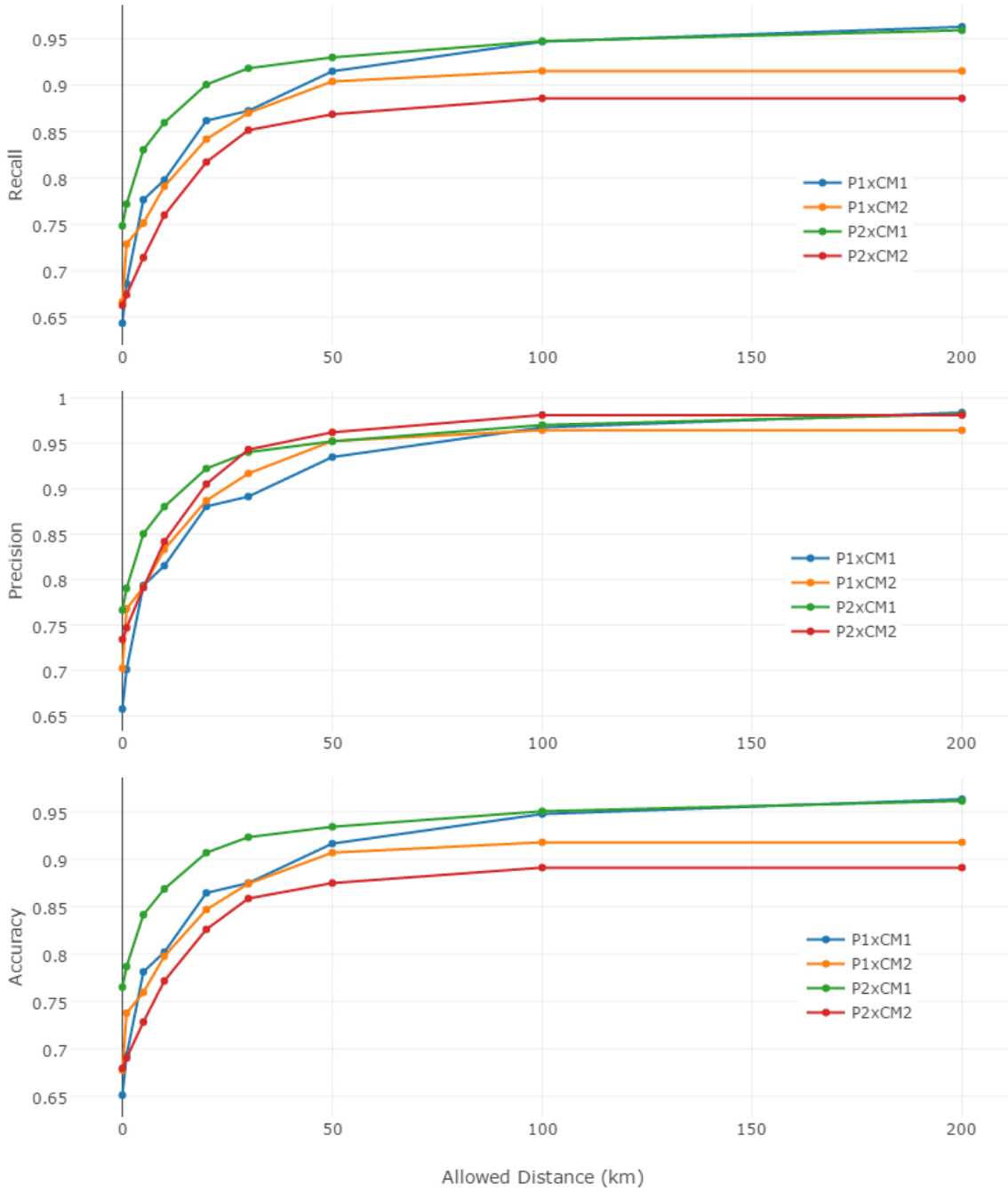


Figure 8: Mode 3 Quality Measure Improvement over allowed distance

To be seen is that mistakes especially happen close-by, and even allowing a distance of 5km can drastically improve recall by about 13%. This reflects both how I have misjudged type importance and how toponym ambiguity happens especially close-by, as previously mentioned.

On these data sets, maps with a tolerance of about 20-30km are the most promising use case. As the data comes from large sets spanning the German Empire, this value seems to suffice. However, there will still be outliers: When viewing precision, about 1 in 10 to 1 in 20 markers will extend this tolerance; when viewing accuracy, the generally expected result of 85% is usually met and most of the time exceeded. Allowing a distance of 50km (and so on) would further increase the quality measures, but a map zoomed out far enough to allow for 50km to be a tolerable distance is probably not the usual use case; the message being that mistakes, when they happen, happen close-by and are acceptable for some uses.

Again, mind that the distances are purely relying on what I thought the correct place is. As the data is rather fuzzy and usually multiple of the close-by places would fit, reconstructing the gold standard with knowledge of the preferred types would instantly better the results.

To prove this point, a small data set of about 80 lines has been generated from the gazetteer data, where from the district Plön, various communities and villages were chosen. When leaving type information in, the system result was perfect. When removing type information, a few mistakes occurred due to toponym ambiguity and favouring of types that were more dominant in the data; however, all were within 5km distance, and it could be argued that they were more plausible results than what was in the gold standard. The system has also been tested on various other sets of data, but without a gold standard or surprising results. Please refer to the project files for further information on these.

The achieved result quality on these data sets depends, as previously mentioned, on supplying rule files. It has been evaluated which files did contribute how much to the system result, and the results are shown in Figure 9. For this, single rule files were removed, and finally all of them.

At first glance surprisingly, removing the bounding polygon on its own does not influence the result in any data set, even though supplying a bounding polygon has been described as a vital part to reduce toponym ambiguity in literature. This is due to two effects: First, single toponyms are resolved by the Default Referent heuristic (meaning “Berlin” has a fixed solution) and therefore, no problems occur on many single toponyms; second, toponyms

that occur in a hierarchy are implicitly bound to a region as a relation “A, B” will with a much lower chance occur multiple times in the world than just “A”, and, since the data is completely in german, the chance of repeating toponym hierarchies is even lower. This might be a different situation for english toponyms when considering they can occur in Great Britain, the US, Canada, Australia, and possibly other regions as well.

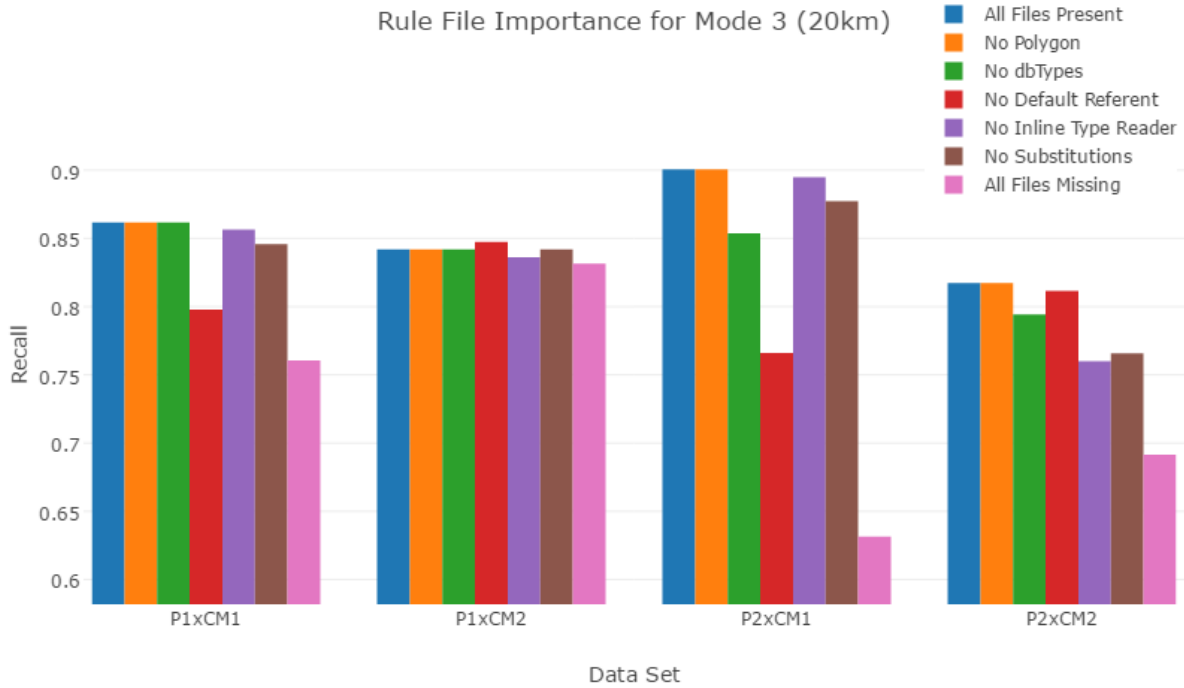


Figure 9: Rule File Importance

Removing the file *dbTypes* that is used to prune places by type from query returns did not matter in *P1* at all, but had a heavy impact of 5% on $P2 \times CM1$. Why this did not influence *P1* is investigated later with the data in Figure 10.

Removing Default Referents was especially harmful to data sets collected by *CM1*, as more frequent data contained more single toponyms resolved in that list; for many of those, no other heuristic was able to solve the problem, resulting in a loss of over 12% in recall for $P2 \times CM1$. Interestingly, using Default Referents has actually lowered recall in $P1 \times CM2$, as a false place was noted that could actually be resolved by the system itself correctly.

Removing the Inline Type Reader only affected $P2 \times CM2$; more rare lines from $P2$ are more likely to contain type information, as “Berlin” was more frequent than “Stadt Berlin” (City Berlin). Removing the substitution file had the biggest impact here, too, as these lines were also more likely to hold abbreviations of types that were expanded by the substitution file; without the file, the system tried to interpret the abbreviations as toponyms. For example, “Frstt.” is expanded to “Fürstentum” (principality) and then read by the Inline Type Reader; when the file went missing, the system tried to find a toponym fitting to “F*r*s*t*t*”.

Only one data set was mostly resistant to removing all files. Usually, removing all rule files leads to a worse result than the sum of deterioration due to removing single files. In some cases, one heuristic could fill in for another.

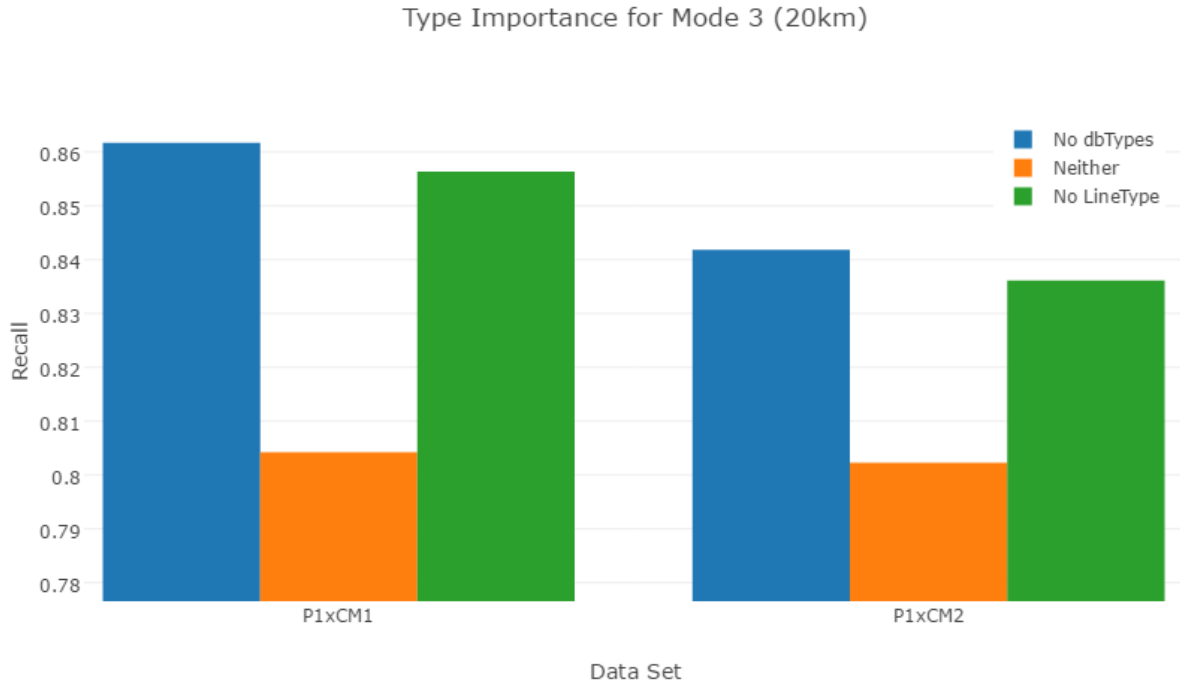


Figure 10: Type Importance

To further investigate why removing the dbTypes filter file did not influence $P1$, the line type header (denoting that *group_8* and *group_1* are the expected type groups of base toponym and its parent) has been removed experimentally. As can be seen in Figure 10, both dbTypes file and LineType

manage to keep the recall up; however, when both are missing, the quality drops by about 4% to 6% recall value. As *P2* has no type header due to dbTypes covering all knowledge completely, the removal is more influential on it.

In Figure 11, the relevance of heuristics for the overall result is displayed. Mind that this is partially due to their order as described in the Implementation section. For example, Machine Learning would potentially decide more if less was decided in the previous steps, and so on.

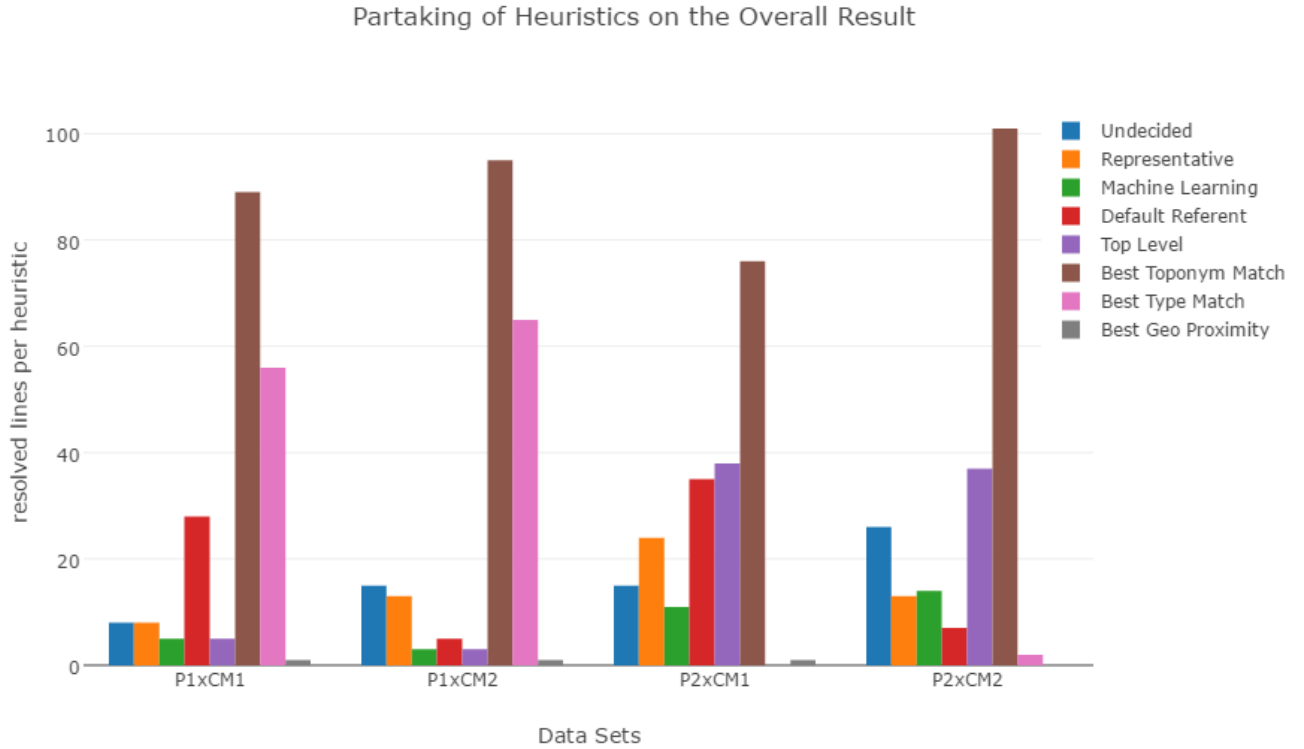


Figure 11: Heuristic Partaking

The varying relevance of types and default referents in different data sets becomes obvious once more. The difference for the Top Level heuristic (“Prefer High-Level Referents (/H12/)”) is due to *P1* favouring places of type group 8; the name might repeat on higher levels, but they are usually of type group 1 and therefore ranked lower, hence Top Level is used less.

The rarity of the Geo Proximity heuristic is due to hardly any of the lines in the data sets containing geo references to rivers. On resolving the

full data behind *P1*, which is the problem containing about 60.000 lines, Geo Proximity was decisive 398 times, which is about 0.7% of all decisions made on the full resolution.

Another evaluation has been done on data about Danzig from 1927, where the region was restructured, and where the GOV's data is currently being developed. This problem shall be called *P3*. The gold standard file has been given by a person working on it that claimed all 331 lines in the file were identifiable. In Mode 3, with 20km distance allowed, above 93% recall, precision, and accuracy were achieved. However, since the data is centered around Danzig, this is hardly surprising, since the covered area is rather small anyway. On 5km allowed distance, only 72% were achieved for the three values, and Mode 2 only came up to 78% for 20km allowed distance, since 56 *FalseNegative* cases arise due to the non-identification of the first toponym. In Mode 1, 229 of the 331 lines are shown to be ID-perfect matches between system and reviewer, which equals 69% agreement.

On inspecting the actual data, quite some disagreement comes due to doublets, false typing of places, and missing connections between places, since the region is currently being developed. The difference between system and user result can now be employed to develop the completeness of the area; reviewing disagreement often reveals problematic parts of the model.

6.5 Comparison to other Toponym Resolution Systems

As noted multiple times, the comparison of the presented system to others is a difficult project. First and foremost, toponym resolution systems are usually attached to a database and not available on their own. To actually compare the here presented system to others in its quality in detail, other toponym resolvers would have to be attached to the same database as the here constructed system, or the other way around.

Due to gazetteer incompleteness, different outcomes are to be expected: An ambiguous toponym in one database might be unique in the other, or not even present. Furthermore, gazetteers might simply disagree on where a place holding a toponym is located, especially since both only reflect a part of the world and may both be correct regarding the real world within their respective subset. As Tobin finds, a gazetteer's focus on a certain time may also be problematic:

"It is not ideal to georeference historical collections using contemporary gazetteers because spelling changes may affect look-up success and because administrative boundaries change over time." [24]

Additionally to be noted is that system comparison is also not ideal due to the varying focus of various systems. The *toponym resolution system developed here* (TRSDH) is especially made for historical serial sources, which can't be said of most of the other systems viewn in the following; to measure them all on such sources will lead to rather biased results.

Due to all these restrictions, an actual comparison of the resolver part of the systems regarding quality measures like recall or precision is not within reach. However, the systems can still be tested for which heuristics are employed and what kind of input they are able to read.

All of the systems used for comparison are linked by URL in Table 8, and are solely referred to by name in Table 7, where a comparison between systems is gathered. This focuses on the most common visible heuristics, where uncommon heuristics and features are additionally noted per system.

How results were gathered and additional information about systems is given separately in the running text.

Geobrowser	Hierarchy	Fuzziness	Type	Bounding	Other
GOV+TRSDH	Yes	Yes	Yes	Yes	many; see this thesis
DARIAH Geobrowser	No	No	No	No	animatable view various historical background maps
GeoNames	Yes	Yes	Yes	Country Continent	interactive map
GeoWhiz	Yes	No	No	No	classification of sets of places by various attributes
Nominatim	Yes	No	Yes	No	results ranked, shown on map
Google Maps	Yes	Yes	No	No	completion suggestions with hierarchy
Yahoo Maps Beta	Yes	Yes	No	Map View	completion suggestions with hierarchy shows alternatives on demand
Bing Maps	Yes	Yes	No	No	completion suggestions with hierarchy

Table 7: (Historical) Gazetteers' featured heuristics

The *DARIAH-DE Geobrowser* is a tool for “researchers of the humanities and cultural studies for the analysis of space-time relations of data”⁶. Its Datasheet Editor reads only single toponyms and returns Longitude, Latitude, and a GettyID to the user. A timespan can be specified, but it seems to be only used for further services allowing the display of maps over time, not for the resolution itself. As it only accepts single toponyms, no use

⁶<https://de.dariah.eu/geobrowser>, accessed 2016-09-13, translated

can be made of hierarchical information, and no spelling likeness is taken into account. A clerical review user interface exists that allows to decide between multiple results.

While this, too, is a tool for toponym resolution on historical data, the resolution itself is limited to perfect singletons, and the focus seems to be on displaying clerically reviewn results.

Experimentally, the first columns of $P1 \times CM1$ and $P1 \times CM2$ were copied into the system, which largely contain single toponyms. 116 *answers* (in large parts with multiple options) were returned on the 400 requests made, which is an answer rate of about 29%. The *TRSDH* returned 373 *decisions*, which is a rate of about 93%. However, the DARIAH Geobrowser results would have to be further clerically reviewn due to obvious mistakes: Places were chosen far outside of the German Empire, Königsberg was marked in the middle of today’s Germany, Mainz within Munich, and Neustadt was arbitrarily located in Bremen. Since the system does not use hierarchy information, the problem falls back to the problem of toponym ambiguity for perfectly spelled toponyms that is resolved by a user, and the first return value seems arbitrarily chosen.

GeoNames does accept hierarchical information and returns an ordered list of results; an extended search is available that allows restricting results to a continent, a country, a feature class, and allows for enabling a fuzzy search. However, neither wildcards nor abbreviations can be used, and the fuzzy search does not always work on misspellings, but on mixing up data field strictness. When searching for “Kiel, Schleswig-Holsteinx”, where the added “x” is an artificial spelling error, nothing is returned in fuzzy mode. When searching for “Heidkater” where the added “r” is the artificial error, the correct location named “Heidkate” is returned among others; “Heidkater, Schleswig-Holstein” does not return any results again, even though the hierarchy is correct and previous results showed “Heidkate” to be in Schleswig-Holstein. Due to this design of fuzzy searches, the toponym resolution system would not be suited to resolve the files worked on here, as fuzziness on toponyms and hierarchy are required to work at the same time.

For using the aforementioned service *GeoWhiz* that employs place properties to return a set of common places, all problems have been converted to a list of comma-separated hierarchies so that the system can read them; however, it reads neither abbreviations, nor is it capable of searching fuzzily. All 800 lines were sent in 200-lines batches, but only 11% (91 / 800) of the lines were identified overall, where the *TRSDH* reaches 92% (735 / 800).

Toponym imperfection aside, this difference also stems from many middle administrative levels simply missing from the database of *GeoWhiz*; these levels are very common in the files.

Nominatim does take into account hierarchical information, but does not allow for toponyms to be imperfect. As return, a website with attached list of results is given, where results are ordered by type relevance. The type can also be given by the user, allowing to search for the “Village Mühlheim”; “City”, on the other hand, does not work. The search field’s type vocabulary is limited. The webservice offers to resolve more complex requests via URL query, where a city can be specified, and is able to resolve requests to institutions and restaurants, streets with and without number, and regions. Its inability to search fuzzily or for abbreviations still makes it less usable for the problem at hand, as does its focus on contemporary information. The service does accept wildcards, though, so that abbreviations may be pre-processed and then fed to the system.

Regnum Francorum Online is a very specific historical gazetteer, its database is simply not comparable to the data worked on. To be mentioned is that the toponym resolver seems to be a simple lookup of a provided toponym, and its focus is more on displaying a map of the time it focuses on.

Yahoo Maps Beta, *Google Maps*, and *Bing Maps* show comparable behaviour. All of them are able to read hierarchies and automatically apply fuzziness to the search words when nothing was found, much like the *TRSDH* does. According to tooltips, *Yahoo Maps Beta* is supposed to be able to show results with the Map View as bounding area, but no such behaviour could be observed. Interestingly, both *Bing Maps* and *Yahoo Maps Beta* sometimes work according to the skips heuristic: When searching for “shfhadfhdfah, Schleswig-Holstein”, both skip the keyboard mashing and return Schleswig-Holstein, while *Google Maps* is more strict and returns the information that no such object was found. *Bing Maps* seems to furthermore return places nearby, as “Heidkate, Kiel” will return two buildings containing the name “Heidkate” outside of Kiel, while “Heidkate, München” returns empty results; interestingly, the skips heuristic is not used in this case, and how the system overall works it not entirely clear.

None of the services interprets abbreviations or wildcards, and no timespans are available. While those services are still the most promising ones due to their mostly fitting fuzziness behaviour, their contemporary focus reduces usability.

Concluding from these comparisons, none of the presented systems is well usable for the task at hand, and due to their scope it would be a surprise if they were. Systems not meant for historical sources tend to either not feature fuzziness and placetypes at the same time, or lack other properties to satisfactorily resolve such a source; other services are meant for very specific scopes, present one focus heuristic, or are more about display of data than actual resolution. *GeoWhiz* was the only service to make use of a problem’s seriality by finding common characteristics of places.

Historical serial sources are a very specific topic, and most systems are simply not constructed with such a topic and the problems and opportunities it brings in mind.

For the data sets at hand, the gazetteer data employed by the services will not suffice with the *TRSDH* attached either, since it covers large parts of the eastern German Empire that are restructured and renamed nowadays. While this work focused on how the resolution can be improved, the underlying gazetteer data is of critical importance for an actually useful result.

7 Conclusions

The Toponym Resolution System presented here is able to work on historical serial sources as specified with satisfactory results. Due to its specific orientation and testing, it is difficult to compare it with other systems, but it has been gathered that other systems do not support the mixture of heuristics that was required to achieve a high result quality. It does hold up and exceed generally noted values for accuracy within a range of distance error that is inherent to the problem and that should be tolerable for most uses.

With the four varied data sets has been shown that even in related problems great differences in active heuristics and required files occur. While for some tasks the default mechanisms work on their own, for others files have to be supplied by a user to specify filtering and working rules. Due to this, the system is less of a complete solution, and more of a tool that requires careful configuration for specific cases and that comes with tools to evaluate and improve its results as well as enhance the used gazetteer’s completeness.

Writing additional files pays off greatly on big sets of data, where one rule may lead to the correct resolution of multiple thousands of lines, and the use of heuristics from all over literature allows for configuration for a broad variety of problems.

8 Appendix

8.1 Glossary

Gazetteer: Gazetteers "can be defined as geospatial dictionaries of geographic names".[13] Entries in such dictionaries contain names, geo-coordinates and location types. Further content might be supplied. [13]

Also known as *geographic database*.

Indexing: Indexing in this document refers to two concepts. In the *retrieval sense*, it describes which elements of an entry are provided to look it up. In the *data matching sense*, it describes finding all pairs of entries possibly referring to the same real-world object. When not noted otherwise, the second sense is meant.

Matching: Matching is the decision of whether two indexed entries actually correspond. In other sources, it may refer to the whole process of Data Matching.

Referent Ambiguity: When a toponym matches multiple places perfectly (meaning multiple places share a name), referent ambiguity arises, as it is not trivially (or at all) accessible which place the toponym refers to.

Serial Source: A source that is characterized by having entries in a list-like fashion.

Structural Ambiguity: On some strings, it is not immediately clear whether it is a toponym or a part of the string is just additional information. "Frankfurt On The Main" is a toponym, "Hamburg On The Elbe" is a toponym with additional location information.

Toponym: A placename. [16, p. 25]

Toponym Resolution: "I define the task of automatic Toponym Resolution (TR) as computing the mapping from occurrences of names for places as found in a text to a representation of the extensional semantics of the location referred to (its referent), such as a geographic latitude/longitude footprint." [16, p. 3] In this thesis, a toponym is linked to a gazetteer entry; ultimately, such an entries geographic position is used.

Also known as *geo-coding*, *geographical indexing*, *geographic information retrieval (GIR)*.

TRSDH: For lack of a better name, the *toponym resolution system developed here*. This abbreviation is used where it's referred to frequently.

8.2 Acknowledgements

Software related graphics have been made in *Google Draw*, plots and bars have been generated with <http://plot.ly>, tables have partially been formatted with <http://tablesgenerator.com>. Thank you for saving me time!

8.3 Further Referenced Contents


Geobrowser	Address
Bing Maps	http://www.bing.com/mapspreview
DARIAH	http://geobrowser.de.dariah.eu/edit/
GeoNames	http://www.geonames.org/
GeoWhiz	http://geowhiz.umiacs.umd.edu/
Google Maps	https://www.google.de/maps/
GOV	http://gov.genealogy.net/search/index
Nominatim	https://nominatim.openstreetmap.org/
Regnum Francorum Online	http://www.francia.ahlfeldt.se/index.php
Yahoo Maps Beta	https://maps.yahoo.com/beta/

Table 8: Referenced (Historical) Gazetteers (URLs)

To be noted is that both Bing Maps and Yahoo Maps Beta rely upon the gazetteer services provided by Here⁷. Also, Yahoo officially shut down the services in September 2011.

⁷<https://maps.here.com/>

Verein für Computergenealogie



genealogy.net compgen.de

Suchmöglichkeiten

Schnelle Suche

Benutzen Sie die Erweiterte Suche für mehr Optionen.

Allgemein

- Abmelden
- Sprache wechseln
- Über das GOV
- Mitmachen
- Forum für technische Fragen
- E-Mail an die Admins
- Letzte Änderungen
- Liste aller Objekttypen
- Qualitätssicherung
- Benutzerverwaltung
- Objekte zusammenfügen
- "Patenschaften" verwalten
- Kontaktmöglichkeiten verwalten
- Systemzustand
- Übersetzung

Seite mit erweiterten Suchoptionen

Angemeldet als firstname_130238 lastname_130238

Erweiterte Suche

Toponymzeile:

Beispiele: "Kiel, Schleswig-Holstein", "Kiel in S.-H."

Typpräferenz:

all types

Bevorzugte/r Typ/gruppe. Wird höher eingestuft.

Region:

worldwide

Nur Objekte innerhalb der gewählten Region werden berücksichtigt.

Zeit (Von):

Zeit (Bis):

Beispiele: "1914-01-01", "1914-01", "1914" - leer/unlesbar wird als offenes Ende interpretiert

Suche

.CSV Datei auswerten

Gesuchte Objekte müssen zu folgenden Typklassen gehören:

☒

(politische) Verwaltung

☒

Zivilverwaltung

☒

Wohnplatz

☒

geographische Typen

☒

Gericht

☒

Verkehrswesen

☒

Kirche

☒

Sonstige

Akzeptiere Ortsnamen in folgenden Sprachen:

Feld frei lassen, um Sprachbeschränkung zu deaktivieren. Um nach Sprache der Ortsnamen zu filtern, schreibe Sprachen getrennt mit ";" wie in ISO 639-3. Beispiel: deu;eng;spa würde nur deutsche, englische und spanische Ortsnamen interpretieren.

Nur Objekte innerhalb der gewählten Region werden berücksichtigt.

worldwide

Gebe Zeitrahmen wie in eingeschränkter Suche ein. Wird als Default benutzt.

Zeit (Von):

Zeit (Bis):

Auswerten:

Durchsuchen...

Keine Datei ausgewählt.

CSV hochladen

Anleitung:

Die CSV muss wie die folgenden Beispiele strukturiert sein. Die Kopfzeile gilt für alle Folgezeilen, diese wiederum beinhalten aufzulösende Informationen. Zur Erläuterung sind im ersten Beispieldokument Kommentare nach # eingefügt.

Herunterladen der kommentierten Beispieldatei. (Englisch)

Herunterladen der auflösbaren Beispieldatei.

Start Project Resolution

Enter project folder:

auswerten

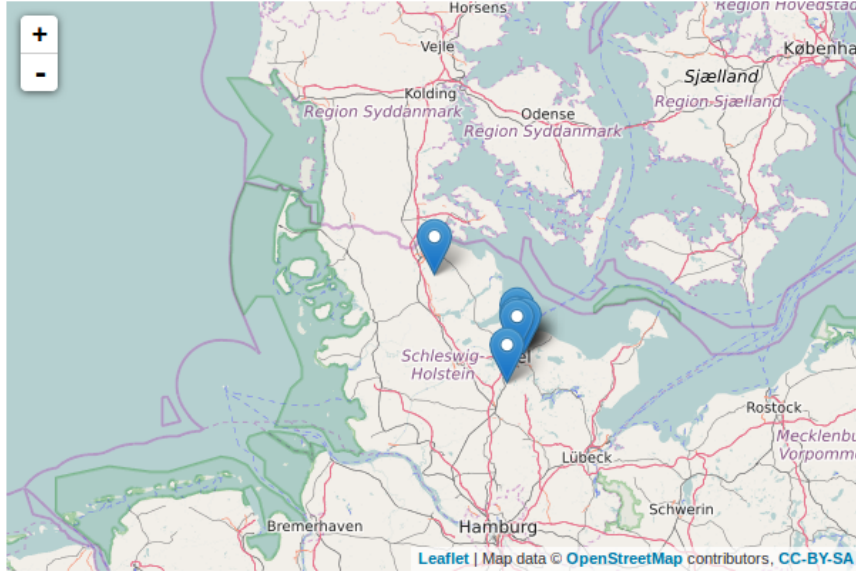
Warning: This function relies upon a properly configured folder. It is only visible for administrators.

Enter 'orte_wk1' to resolve content in ./toponymResolutionProjects/orte_wk1/

Figure 12: The Extended Search Page, when logged in as an administrator. It offers four access points as described in chapter 5.2; *Quick Search* is located in the left bar (*Schnelle Suche*), *Advanced Search*, *File Resolution*, and *Project Resolution* are located in the three separated fields to the right.

59

7 Einträge in 3,339 s gefunden.

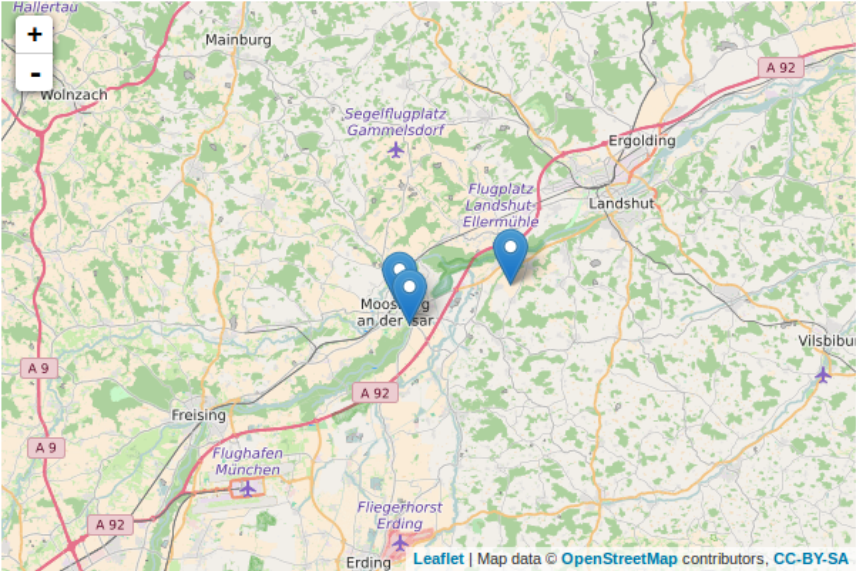


Ergebnisfilter:

	Name	Typ	Übergeordnete Objekte	Postleitzahl	GOV-Kennung	Toponymübereinstimmung ▼	Typübereinstimmung ▼	Geonähe ▼
	Kiel	[53] Stadtkreis	Schleswig-Holstein, Bundesrepublik Deutschland		adm_131002	1	1	0
	Kiel	[150] Stadt (Gebietskörperschaft)	Schleswig-Holstein, Bundesrepublik Deutschland	24103	KIEIJJO54BI	1	0.5	0
	Kiel	[51] Stadt (Siedlung)	Kiel, Schleswig-Holstein, Bundesrepublik Deutschland		KIEIJJO54BH	1	0.5	0
	Kiel, Bordesholm	[32] Kreis [36] Landkreis	Schleswig-Holstein, Bundesrepublik Deutschland		object_214377	1	0.5	0
	Kiel (Christ-König)	[26] Kirche	Kiel, Schleswig-Holstein, Bundesrepublik Deutschland		CHRNIGJO54CH	1	0.5	0
	Kiel-Holtenau	[26] Kirche	Holtenau, Kiel, Schleswig-Holstein, Bundesrepublik Deutschland		object_184896	0.9201	0.5	0
	Kiel	[29] Kirchspiel	Evangelische-Lutherische Landeskirche Schleswig-Holsteins, Evangelische Kirche in Deutschland		object_1044681	0.905	0	0

Figure 13: Example website output for searching “Stadtkreis Kiel, Bundesland Schl.-Holst.” (“City District Kiel, Federal State Schl.-Holst.”). As multiple places perfectly fit by name, the decision is made by Type Fit (column “Typübereinstimmung”, second from right).

3 Einträge in 2,484 s gefunden.



Ergebnisfilter:


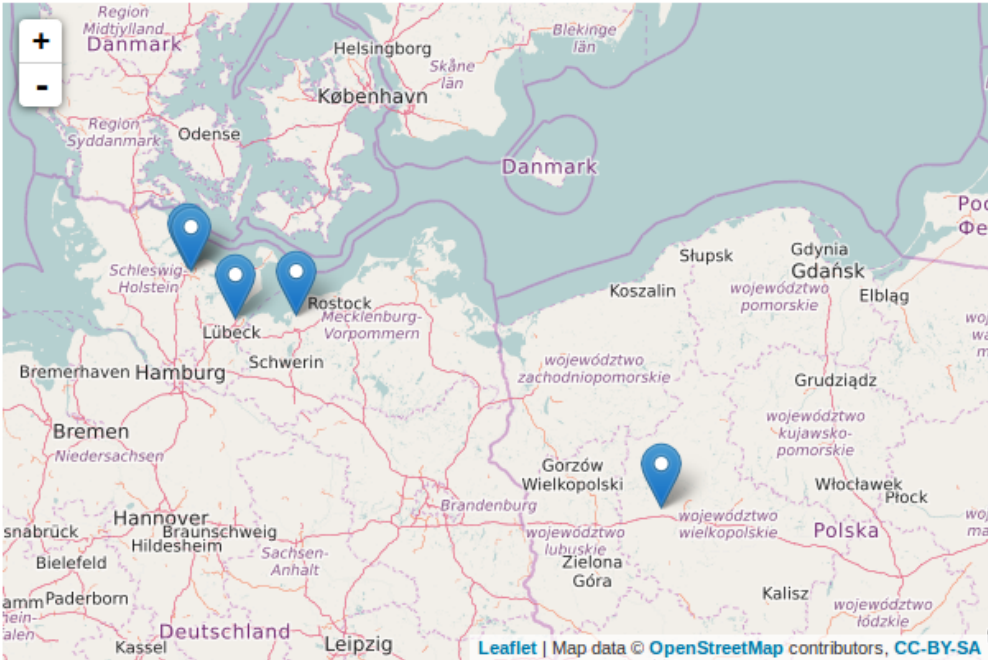
	Name	Typ	Übergeordnete Objekte	Postleitzahl	GOV-Kennung	Toponymübereinstimmung	Typübereinstimmung	Geonähe
	Moosburg an der Isar, Moosburg a.d.Isar	[150] Stadt (Gebietskörperschaft)	Freising, Oberbayern, Bayern, Bundesrepublik Deutschland	85368	object_300805	1	1	1
	Moosburg an der Isar, Moosburg a.d.Isar	[68] Hauptort	Moosburg an der Isar, Freising, Oberbayern, Bayern, Bundesrepublik Deutschland	85368	MOOURGJN58XL	1	1	0.7359
	Moosburg	[223] Landgericht (älterer Ordnung) [3] Amtsgericht	Landshut, München, Bayern, Bundesrepublik Deutschland		object_286655	1	1	0.379

Figure 14: Example website output for searching “Moosburg a. d. Isar, Oberbayern” (“Moosburg On The Isar, Oberbayern”). As multiple places perfectly fit by name and no type is specified, the decision is made by geographical proximity to the river “Isar” of that coordinates are known.

5 von 8 Zeilen sind in 12,689 s eindeutig identifiziert worden.
0 Zeile(n) wurden auf einer höheren Hierarchiestufe markiert.
[Herunterladen des kompletten Ergebnisses.](#)



Ergebnisfilter:

	Suchworte	Name	Typ	Übergeordnete Objekte	Postleitzahl	GOV-Kennung	Matchstrategie
	Duschnik, Posen	Duschnik	[29] Kirchspiel	Samter, Posen, Evangelische Kirche der altpreußischen Union		object_1070661	highest element of hierarchy
	Heidekaten	Heidekaten	[40] Ortsteil	Blowatz, Neuburg, Nordwestmecklenburg, Mecklenburg-Vorpommern, Bundesrepublik Deutschland	23974	HEITEN_O2401	best toponym match quality
	Kiel, Schleswig-Holstein	Kiel	[150] Stadt (Gebietskörperschaft)	Schleswig-Holstein, Bundesrepublik Deutschland	24103	KIEIELJO54BI	representative
	Kronshagen, Bordesholm	Kronshagen	[2] Amtsbezirk	Bordesholm, Schleswig-Holstein, Bundesrepublik Deutschland		object_1048469	highest element of hierarchy
	Ratekau, Lübeck	Ratekau, Rathekau, Ratkow	[66] Kirchdorf	Kaltenhof, Lübeck, Oldenburg, Deutsches Reich	23626	RATKAUJO53IW	best toponym match quality

Figure 15: Example output for uploading a CSV file, including amount of lines identified, used match strategy, and downloadable coordinates.

GEOWHIZ - PLACE LIST DISAMBIGUATOR

ENTER LIST OF PLACES

Flensburg
Schleswig
Kiel
Heiligenhafen
Neustadt
Lübeck

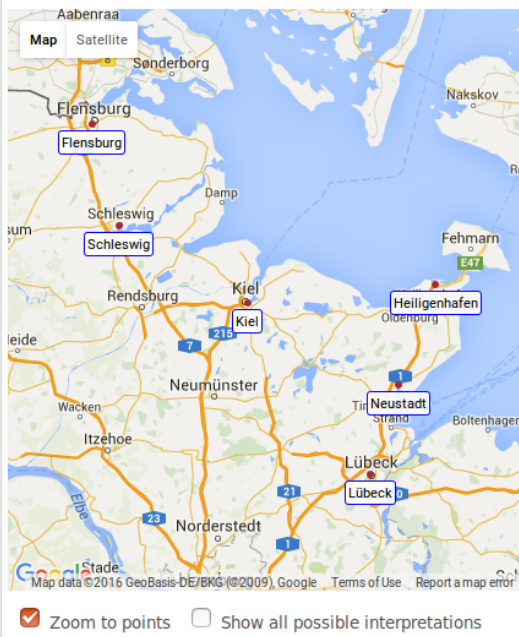
[Use sample list](#)

Submit

CATEGORY RESULTS

Category	Coverage	Ambiguity	Likelihood
populated places with population $\geq 1,000$ in Schleswig-Holstein, Germany	1.00	1.00	99.68%
cities with population $\geq 10,000$ in Schleswig-Holstein, Germany	0.83	1.00	0.32%
fourth-order administrative divisions with population ≥ 100 in Germany	0.67	1.00	0.00%
administrative regions with population $\geq 1,000$ in Germany	0.67	1.00	0.00%
places with population $\geq 1,000$ in Schleswig-Holstein, Germany	0.67	1.00	0.00%
populated places in North America	0.67	1.19	0.00%
spots, buildings, or farms around the world	0.50	2.71	0.00%
administrative regions with population $\geq 10,000$ in Germany	0.50	1.00	0.00%
farms in Lower Saxony, Germany	0.33	2.00	0.00%

MAP OF PLACES



CATEGORY TREE

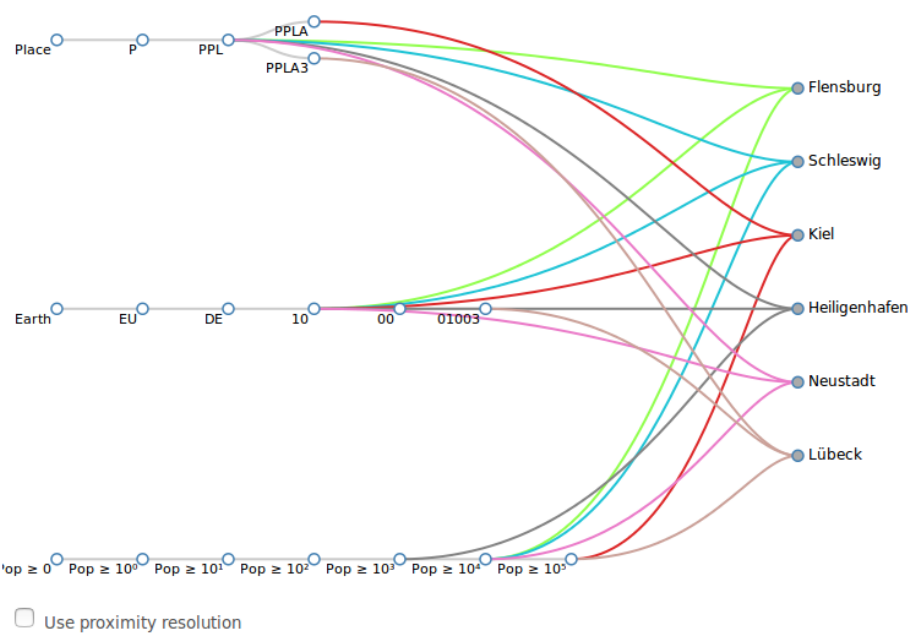


Figure 16: Screenshot of testing an example input on *GeoWhiz*, showing that the system is potentially able to identify fitting places for a serial source describing a route; when searching for Neustadt in GOV, 310 places are returned. Here, a plausible choice has been made. 2016-05-30.

8.4 System Results as Tables

In this chapter, all evaluation results are shown as tables. TP, EP, and so on, are abbreviations of the extended binary classification scheme fields. Evaluation formulae slightly differ due to the extension. In *Mode 3*, no FP occurred by design, hence the missing columns. For more information on how to interpret these tables, please read chapter 6.

Recall: $TP/(TP + FN + EP)$

Specificity: $TN/(TN + FP)$

Precision: $TP/(TP + FP + EP)$

Accuracy: $(TP + TN)/(EP + FP + FN)$

F-Measure: $2 * ((Precision * Recall)/(Precision + Recall))$

Data Set	Mode 1					TP	FP	EP	TN	FN
	Recall	Specificity	Precision	Accuracy	F-Measure					
$P1 \times CM1$	0.6402	0.7272	0.7469	0.645	0.6895	121	3	38	8	30
$P1 \times CM2$	0.6629	0.9545	0.7919	0.695	0.7217	118	1	30	21	30
$P2 \times CM1$	0.7485	0.5517	0.7356	0.72	0.7420	128	13	33	16	10
$P2 \times CM2$	0.6629	0.8	0.8227	0.68	0.7342	116	5	20	20	39
$P3$	0.6918	NaN	0.8327	0.6918	0.7558	229	0	46	0	56

Distance	Mode 2, $P1 \times CM1$					TP	FP	EP	TN	FN
	Recall	Specificity	Precision	Accuracy	F-Measure					
0km	0.6402	0.7272	0.7469	0.645	0.6895	121	3	38	8	30
1km	0.6825	0.7272	0.7962	0.685	0.7350	129	3	30	8	30
5km	0.7619	0.7272	0.8889	0.76	0.8205	144	3	15	8	30
10km	0.7725	0.7272	0.9012	0.77	0.8319	146	3	13	8	30
20km	0.7831	0.7272	0.9136	0.78	0.8433	148	3	11	8	30
30km	0.7831	0.7272	0.9136	0.78	0.8433	148	3	11	8	30
50km	0.8042	0.7272	0.9383	0.8	0.8661	152	3	7	8	30
100km	0.8148	0.7272	0.9506	0.81	0.8774	154	3	5	8	30
200km	0.8307	0.7272	0.9691	0.825	0.8946	157	3	2	8	30

Mode 2, $P1 \times CM2$										
Distance	Recall	Specificity	Precision	Accuracy	F-Measure	TP	FP	EP	TN	FN
0km	0.6667	0.9545	0.7973	0.6985	0.7262	118	1	29	21	30
1km	0.7288	0.9545	0.8716	0.7538	0.7938	129	1	18	21	30
5km	0.7514	0.9545	0.8986	0.7739	0.8185	133	1	14	21	30
10km	0.7740	0.9545	0.9257	0.7940	0.8431	137	1	10	21	30
20km	0.7966	0.9545	0.9527	0.8141	0.8676	141	1	6	21	30
30km	0.8079	0.9545	0.9662	0.8241	0.88	143	1	4	21	30
50km	0.8079	0.9545	0.9662	0.8241	0.88	143	1	4	21	30
100km	0.8136	0.9545	0.9730	0.8291	0.8862	144	1	3	21	30
200km	0.8136	0.9545	0.9730	0.8291	0.8862	144	1	3	21	30

Mode 2, $P2 \times CM1$										
Distance	Recall	Specificity	Precision	Accuracy	F-Measure	TP	FP	EP	TN	FN
0km	0.7485	0.5517	0.7356	0.72	0.7420	128	13	33	16	10
1km	0.7719	0.5517	0.7586	0.74	0.7652	132	13	29	16	10
5km	0.8246	0.5517	0.8103	0.785	0.8174	141	13	20	16	10
10km	0.8538	0.5517	0.8391	0.81	0.8463	146	13	15	16	10
20km	0.8947	0.5517	0.8793	845	0.8870	153	13	8	16	10
30km	0.9006	0.5517	0.8851	0.85	0.8928	154	13	7	16	10
50km	0.9064	0.5517	0.8908	0.855	0.8986	155	13	6	16	10
100km	0.9123	0.5517	0.8966	0.86	0.9043	156	13	5	16	10
200km	0.9240	0.5517	0.9080	0.87	0.9159	158	13	3	16	10

Mode 2, $P2 \times CM2$										
Distance	Recall	Specificity	Precision	Accuracy	F-Measure	TP	FP	EP	TN	FN
0km	0.6629	0.8	0.8227	0.68	0.7342	116	5	20	20	39
1km	0.6743	0.8	0.8369	0.69	0.7468	118	5	18	20	39
5km	0.7086	0.8	0.8794	0.72	0.7848	124	5	12	20	39
10km	0.7143	0.8	0.8865	0.725	0.7911	125	5	11	20	39
20km	0.7314	0.8	0.9078	0.74	0.8101	128	5	8	20	39
30km	0.7429	0.8	0.9220	0.75	0.8228	130	5	6	20	39
50km	0.7543	0.8	0.9362	0.76	0.8354	132	5	4	20	39
100km	0.7657	0.8	0.9504	0.77	0.8481	134	5	2	20	39
200km	0.7657	0.8	0.9504	0.77	0.8481	134	5	2	20	39

Mode 2, $P3$										
Distance	Recall	Specificity	Precision	Accuracy	F-Measure	TP	FP	EP	TN	FN
0km	0.6939	NaN	0.8358	0.6939	0.7583	229	0	45	0	56
1km	0.6939	NaN	0.8358	0.6939	0.7583	229	0	45	0	56
5km	0.7091	NaN	0.8540	0.7091	0.7748	234	0	40	0	56
10km	0.7485	NaN	0.9015	0.7485	0.8179	247	0	27	0	56
20km	0.7818	NaN	0.9416	0.7818	0.8543	258	0	16	0	56
30km	0.7970	NaN	0.9599	0.7970	0.8709	263	0	11	0	56
50km	0.8121	NaN	0.9781	0.8121	0.8874	268	0	6	0	56
100km	0.8182	NaN	0.9854	0.8182	0.8940	270	0	4	0	56
200km	0.8182	NaN	0.9854	0.8182	0.8940	270	0	4	0	56

Mode 3, $P1 \times CM1$									
Distance	Recall	Precision	Accuracy	F-Measure	TP	EP	TN	FN	
0km	0.6436	0.6576	0.6510	0.6505	121	63	4	4	
1km	0.6862	0.7011	0.6927	0.6935	129	55	4	4	
5km	0.7766	0.7935	0.7813	0.7849	146	38	4	4	
10km	0.7979	0.8152	0.8021	0.8065	150	34	4	4	
20km	0.8617	0.8804	0.8646	0.8710	162	22	4	4	
30km	0.8723	0.8913	0.875	0.8817	164	20	4	4	
50km	0.9149	0.9348	0.9167	0.9247	172	12	4	4	
100km	0.9468	0.9674	0.9479	0.9570	178	6	4	4	
200km	0.9628	0.9837	0.9635	0.9731	181	3	4	4	

Mode 3, $P1 \times CM2$									
Distance	Recall	Precision	Accuracy	F-Measure	TP	EP	TN	FN	
0km	0.6667	0.7024	0.6776	0.6841	118	50	6	9	
1km	0.7288	0.7679	0.7377	0.7478	129	39	6	9	
5km	0.7514	0.7917	0.7596	0.7710	133	35	6	9	
10km	0.7910	0.8333	0.7978	0.8116	140	28	6	9	
20km	0.8418	0.8869	0.8470	0.8638	149	19	6	9	
30km	0.8701	0.9167	0.8743	0.8928	154	14	6	9	
50km	0.9040	0.9524	0.9071	0.9275	160	8	6	9	
100km	0.9153	0.9643	0.9180	0.9391	162	6	6	9	
200km	0.9153	0.9643	0.9180	0.9391	162	6	6	9	

Mode 3, $P2 \times CM1$								
Distance	Recall	Precision	Accuracy	F-Measure	TP	EP	TN	FN
0km	0.7485	0.7665	0.7650	0.7574	128	39	12	4
1km	0.7719	0.7904	0.7869	0.7811	132	35	12	4
5km	0.8304	0.8503	0.8415	0.8402	142	25	12	4
10km	0.8596	0.8802	0.8689	0.8698	147	20	12	4
20km	0.9006	0.9222	0.9071	0.9112	154	13	12	4
30km	0.9181	0.9401	0.9235	0.9290	157	10	12	4
50km	0.9298	0.9521	0.9344	0.9408	159	8	12	4
100km	0.9474	0.9701	0.9508	0.9586	162	5	12	4
200km	0.9591	0.9820	0.9617	0.9704	164	3	12	4

Mode 3, $P2 \times CM2$								
Distance	Recall	Precision	Accuracy	F-Measure	TP	EP	TN	FN
0km	0.6629	0.7342	0.6793	0.6967	116	42	9	17
1km	0.6743	0.7468	0.6902	0.7087	118	40	9	17
5km	0.7143	0.7911	0.7287	0.7508	125	33	9	17
10km	0.76	0.8418	0.7717	0.7988	133	25	9	17
20km	0.8171	0.9051	0.8261	0.8589	143	15	9	17
30km	0.8514	0.9430	0.8587	0.8949	149	9	9	17
50km	0.8686	0.9620	0.875	0.9129	152	6	9	17
100km	0.8857	0.9810	0.8913	0.9309	155	3	9	17
200km	0.8857	0.9810	0.8913	0.9309	155	3	9	17

Mode 3, $P3$								
Distance	Recall	Precision	Accuracy	F-Measure	TP	EP	TN	FN
0km	0.6960	0.6960	0.6960	0.6960	229	100	0	0
1km	0.6960	0.6960	0.6960	0.6960	229	100	0	0
5km	0.7204	0.7204	0.7204	0.7204	237	92	0	0
10km	0.7964	0.7964	0.7964	0.7964	262	67	0	0
20km	0.9362	0.9362	0.9362	0.9362	308	21	0	0
30km	0.9635	0.9635	0.9635	0.9635	317	12	0	0
50km	0.9818	0.9818	0.9818	0.9818	323	6	0	0
100km	0.9878	0.9878	0.9878	0.9878	325	4	0	0
200km	0.9878	0.9878	0.9878	0.9878	325	4	0	0

Heuristics Partaking on Complete Result

Decision Heuristic	$P1 \times CM1$	$P1 \times CM2$	$P2 \times CM1$	$P2 \times CM2$	$P3$
Undecided	8	15	15	26	0
Representative	8	13	24	13	38
Machine Learning	5	3	11	14	46
Default Referent	28	5	35	7	0
Top Level	5	3	38	37	0
Best Toponym Match	89	95	76	101	232
Best Type Match	56	65	0	2	15
Best Geo Proximity	1	1	1	0	0

Influence of File Presence on Recall (Mode 3, 20km)

File Presence	$P1 \times CM1$	$P1 \times CM2$	$P2 \times CM1$	$P2 \times CM2$
All Files Present	0.86170	0.8418	0.9006	0.8171
No Polygon	0.86170	0.8418	0.9006	0.8171
No dbTypes	0.86170	0.8418	0.8538	0.7943
No dbTypes, No LineType	0.80423	0.8023	-	-
No LineType	0.85638	0.8362	-	-
No Default Referent	0.79787	0.8475	0.7661	0.8114
No Inline Type Reader	0.85638	0.8362	0.8947	0.76
No Substitutions	0.84574	0.8418	0.8772	0.7657
All Files Missing	0.76064	0.8315	0.6316	0.6914

Declaration

Herewith, I declare that this thesis has been completed independently and unaided and that no other sources other than the ones given here have been used.

Furthermore, I declare that this work has never been submitted at any other time and anywhere else as a final thesis.

2016-09-26, Kiel
Dennis Sen

References

- [1] Marco D. Adelfio, Hanan Samet. November 2013. *GeoWhiz: toponym resolution using common categories* in *SIGSPATIAL'13: Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. Pages 532-535. ACM New York, NY, USA.
- [2] Marco D. Adelfio, Hanan Samet. November 2013. *Structured toponym resolution using combined hierarchical place categories* in *GIR '13: Proceedings of the 7th Workshop on Geographic Information Retrieval*. Pages 49-56. ACM New York, NY, USA.
- [3] Reed S. Beaman, Barry J. Conn. 2003. *Automated geoparsing and georeferencing of Malesian collection locality data* in *Telopea 10*. Pages 43-52. [online; cited 2016-09-17] <http://dx.doi.org/10.7751/telopea20035604>
- [4] Merrick Lex Berman. March 2008. *Georeferencing Historical Placenames and Tracking Changes Over Time*. Presented at the Georeferencing Workshop, Harvard University. [online; cited 2016-09-17] http://www.fas.harvard.edu/~chgis/work/docs/papers/LexBerman_GeoRef_21Mar08.pdf
- [5] Merrick Lex Berman, Johan Åhlfeldt. 23rd of February, 2012. *Historical Gazetteer System Integration: CHGIS, Regnum Francorum, and GeoNames*. [online; cited 2016-09-17] http://www.fas.harvard.edu/~chgis/gazetteer/AAG_GazIntegration_23feb2012.pdf
- [6] Daniel Blank, Andreas Henrich. November 2015. *Geocoding place names from historic route descriptions* in *GIR '15: Proceedings of the 9th Workshop on Geographic Information Retrieval*. Article No. 9. ACM New York, NY, USA.
- [7] Tobias J. Brunner, Ross S. Purves. October 2008. *Spatial Autocorrelation and Toponym Ambiguity* in *GIR '08: Proceedings of the 2nd international workshop on Geographic information retrieval*. Pages 25-26. ACM New York, NY, USA.
- [8] Davide Buscaldi, Bernardo Magnini. February 2010. *Grounding Toponyms in an Italian Local News Corpus* in *GIR '10: Proceedings of the 6th Workshop on Geographic Information Retrieval*. Article No. 15. ACM New York, NY, USA.

- [9] Davide Buscaldi. June 2011. *Approaches to Disambiguating Toponyms in SIGSPATIAL Special: Volume 3 Issue 2, July 2011*. Pages 16-19. ACM New York, NY, USA.
- [10] Peter Christen. 2012. *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Springer Publishing Company, Incorporated.
- [11] Martin Doerr, Manos Papagelis. August 2007. *A Method for Estimating the Precision of Placename Matching in IEEE Transactions on Knowledge and Data Engineering, Vol. 19, No. 8*. Pages 1089-1101. IEEE.
- [12] Claire Grover, Richard Tobin, Kate Byrne, Matthew Woollard, James Reid, Stuart Dunn, Julian Bell. 19th of July, 2010. *Use of the Edinburgh geoparser for georeferencing digitized historical collections in Philosophical Transactions Of The Royal Society A: Mathematical, Physical and Engineering Sciences: Volume 368, No. 1925*. Pages 3875-3889. [online; cited 2016-09-17] <http://rsta.royalsocietypublishing.org/content/roypta/368/1925/3875.full.pdf>
- [13] Linda L. Hill. 17th of November, 2000. *Core Elements of Digital Gazetteers: Placenames, Categories, and Footprints in Research and Advanced Technology for Digital Libraries*. 4th European Conference, ECDL 2000 Lisbon, Portugal, September 18-20, 2000 Proceedings. Editors: José Borbinha, Thomas Baker.
- [14] Ray R. Larson. 1996. *Geographic Information Retrieval and Spatial Browsing in Geographic information systems and libraries: patrons, maps, and spatial information*. Papers presented at the 1995 Clinic on Library Applications of Data Processing. Pages 81-124. Editors: Linda C. Smith and Myke Gluck.
- [15] Ray R. Larson, Patricia Frontiera. June 2004. *Spatial Ranking Methods for Geographic Information Retrieval (GIR) in Digital Libraries in JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*. Pages 415-427. ACM New York, NY, USA.
- [16] Jochen Lothar Leidner. 2007. *Toponym Resolution in Text: Annotation, Evaluation and Applications of Spatial Grounding of Place Names*. Institute for Communicating and Collaborative Systems, School of Informatics, University of Edinburgh, Scotland.

- [17] Christoph Lofi, Kinda El Maarry. July 2014. *Design Patterns for Hybrid Algorithmic-Crowdsourcing Workflows* in *2014 IEEE 16th Conference on Business Informatics (Volume:1)*. Page 1-8. IEEE.
- [18] Ludovic Moncla, Walter Renteria-Agualimpia, Javier Nogueras-Iso, Mauro Gaio. November 2014. *Geocoding for texts with fine-grain toponyms : an experiment on a geoparsed hiking descriptions corpus* in *SIGSPATIAL '14: Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM New York, NY, USA.
- [19] Ana-Maria Olteanu. 21st of November, 2008. *A multi-criteria Fusion Approach for Geographical Data Matching* in *Quality Aspects in Spatial Data Mining*. Pages 47-55. CRC Press. Editors: Alfred Stein, Wenzhong Shi, Wietske Bijker.
- [20] Simon Overell. June 2011. *The problem of place name ambiguity in SIGSPATIAL Special: Volume 3 Issue 2, July 2011*. Pages 12-15. ACM New York, NY, USA.
- [21] Michael Piotrowski, Samuel Läubli, Martin Volk. February 2010. *Towards Mapping of Alpine Route Descriptions* in *GIR '10: Proceedings of the 6th Workshop on Geographic Information Retrieval*. Article No. 2. ACM New York, NY, USA.
- [22] C. J. Rupp, Paul Rayson, Alistair Baron, Christopher Donaldson, Ian Gregory, Andrew Hardie, Patricia Murrieta-Flores. October 2013. *Customising Geoparsing and Georeferencing for Historical Texts* in *2013 IEEE International Conference on Big Data*. Pages 59-62. IEEE.
- [23] Hanan Samet. 2014. *Using Minimaps to Enable Toponym Resolution with an Effective 100% Rate of Recall* in *GIR '14: Proceedings of the 8th Workshop on Geographic Information Retrieval*. Article No. 9. ACM New York, NY, USA.
- [24] Richard Tobin, Claire Grover, Kate Byrne, James Reid, Jo Walsh. February 2010. *Evaluation of Georeferencing* in *GIR '10: Proceedings of the 6th Workshop on Geographic Information Retrieval*. Article No. 7. ACM New York, NY, USA.
- [25] Raphael Volz, Joachim Kleb, Wolfgang Mueller. 2007. *Towards Ontology-based Disambiguation of Geographical Identifiers* in *Proceedings of the WWW Workshop I³*. Banff, Canada.

- [26] Nina Wacholder, Yael Ravin, Misook Choi. March 1997. *Disambiguation of proper names in text* in *ANLC '97: Proceedings of the fifth conference on Applied natural language processing*. Pages 202-208. Association for Computational Linguistics Stroudsburg, PA, USA.
- [27] John Wiecezorek, Qinghua Guo, Robert Hijmans. December 2004. *The point-radius method for georeferencing locality descriptions and calculating associated uncertainty* in *International Journal of Geographical Information Science*. Vol 18, Issue 8. Pages 745-767. Taylor and Francis Ltd.