The Dissertation Committee for Michael Adrian Speriosu
certifies that this is the approved version of the following dissertation:

Methods and Applications of Text-Driven Toponym Resolution with Indirect Supervision

Committee:

Jason Baldridge, Supervisor

Katrin Erk

David Beaver

Matthew Cohen

James Scott

Methods and Applications of Text-Driven Toponym Resolution with Indirect Supervision

by

Michael Adrian Speriosu, B.S., B.A., M.A.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2013

Dedicated to Jamie.

Acknowledgments

First and foremost, I would like to thank my adviser Jason Baldridge for being the primary guiding force for my research from the moment I spoke to him over the phone about the possibility of coming to UT Austin until the moment he signed off on the final version of this dissertation. Jason always managed a perfect balance between rigorous encouragement and understanding empathy, having been through the process himself and maintaining awareness of the best ways to help me toward my goals. Taking classes with Jason and working with him as a teaching assistant only added to the benefits I received from him and which can never be repaid. I would also like to thank my other committee members Katrin Erk, David Beaver, Matt Cohen, and James Scott. Katrin always had a knack for adding new perspectives to problems Jason and I were working on, and helped me revamp papers and presentations alike in ways that greatly improved their usefulness to others. David encouraged me to add rigor and structure to my love of language in his classes, and his suggestions contributed to this dissertation's attempts to connect to the bigger picture. Matt was invaluable as an ambassador into the world of the humanities, helping me understand what was and wasn't useful for scholars in his area and showing me meaning in my work I would otherwise have missed. James pushed me to question the assumptions behind the mathematics of my approaches and to trim as much fat as possible in their use.

I thank Jochen Leidner for writing the single most important publication to this one in his dissertation, as well as for preparing the desperately needed TR-CoNLL corpus. I thank Scott Nesbit for providing the equally vital annotations to the Perseus Civil War and 19th Century American Collection. I thank Ben Wing for writing most of the code under the hood of the TextGrounder document geolocation system used in the TRIPDL and TRAWL resolvers. Fieldspring would be a much less useful software package without the early organization and programming done by Travis Brown, Taesun Moon, and others who worked on TextGrounder. I thank Rob Turknett and Brandt Westing for familiarizing me with Processing and helping me run visualizations at the Texas Advanced Computing Center.

There were many others at UT and elsewhere who contributed to the completion of this work, via collaborations on related projects, discussions about my research, and general conversations about computer science, linguistics, geography, and life, including but not limited to Ben Wing, Stephen Roller, Travis Brown, Taesun Moon, Nikita Sudan, Sid Upadhyay, Elias Ponvert, Joey Frazee, Carissa Miller, Nick Gaylord, Yinon Bentor, Dan Garrette, Chris Brown, Leif Johnson, Weiwei Ding, Matt Lease, Ray Mooney, Philip Resnik, Arto Anttila, Tom Wasow, Matthew Adams, Gabriel Recchia, Sandra Markarian, and Juwon Lee. I owe the relief that came after more than a few moments of panic to Linguistics Department staff members Ben Rapstine, Leslie Crooks, and Gina Pollard.

I thank Jamie Neville, Gabriel Recchia, Tina Lin, Randi Neville, and

Larry Huang for proofreading various parts of this dissertation; all remaining errors are entirely my fault.

My girlfriend and best friend Jamie Neville has been the most supportive, loving, and beautiful person in my life ever since the day I met her. Our dogs Darwin and Tails also provided emotional support in the way only animals can. My parents Virgil Speriosu and Michelle Watté have always been there for me, showing me the road to success but letting me walk—and stumble—along it in my own way.

No ants were harmed in the writing of this dissertation.

Methods and Applications of Text-Driven Toponym Resolution with Indirect Supervision

Michael Adrian Speriosu, Ph.D.
The University of Texas at Austin, 2013

Supervisor: Jason Baldridge

This thesis addresses the problem of toponym resolution. Given an ambiguous placename like *Springfield* in some natural language context, the task is to automatically predict the location on the earth's surface the author is referring to. Many previous efforts use hand-built heuristics to attempt to solve this problem, looking for specific words in close proximity such as *Springfield, Illinois*, and disambiguating any remaining toponyms to possible locations close to those already resolved. Such approaches require the data to take a fairly specific form in order to perform well, thus they often have low coverage. Some have applied machine learning to this task in an attempt to build more general resolvers, but acquiring large amounts of high quality hand-labeled training material is difficult.

I discuss these and other approaches found in previous work before presenting several new toponym resolvers that rely neither on hand-labeled training material prepared explicitly for this task nor on particular co-occurrences

of toponyms in close proximity in the data to be disambiguated. Some of the resolvers I develop reflect the intuition of many heuristic resolvers that toponyms nearby in text tend to (but do not always) refer to locations nearby on Earth, but do not require toponyms to occur in direct sequence with one another. I also introduce several resolvers that use the predictions of a document geolocation system (i.e. one that predicts a location for a piece of text of arbitrary length) to inform toponym disambiguation. Another resolver takes into account these document-level location predictions, knowledge of different administrative levels (country, state, city, etc.), and predictions from a logistic regression classifier trained on automatically extracted training instances from Wikipedia in a probabilistic way. It takes advantage of all content words in each toponym's context (both local window and whole document) rather than only toponyms.

One resolver I build that extracts training material for a machine learned classifier from Wikipedia, taking advantage of link structure and geographic coordinates on articles, resolves 83% of toponyms in a previously introduced corpus of news articles correctly, beating the strong but simplistic population baseline. I introduce a corpus of Civil War related writings not previously used for this task on which the population baseline does poorly; combining a Wikipedia informed resolver with an algorithm that seeks to minimize the geographic scope of all predicted locations in a document achieves 86% blind test set accuracy on this dataset.

After providing these high performing resolvers, I form the groundwork

for more flexible and complex approaches by transforming the problem of toponym resolution into the traveling purchaser problem, modeling the probability of a location given its toponym's textual context and the geographic distribution of all locations mentioned in a document as two components of an objective function to be minimized. As one solution to this incarnation of the traveling purchaser problem, I simulate properties of ants traveling the globe and disambiguating toponyms. The ants' preferences for various kinds of behavior evolves over time, revealing underlying patterns in the corpora that other disambiguation methods do not account for.

I also introduce several automated visualizations of texts that have had their toponyms resolved. Given a resolved corpus, these visualizations summarize the areas of the globe mentioned and allow the user to refer back to specific passages in the text that mention a location of interest. One visualization presented automatically generates a dynamic tour of the corpus, showing changes in the area referred to by the text as it progresses. Such visualizations are an example of a practical application of work in toponym resolution, and could be used by scholars interested in the geographic connections in any collection of text on both broad and fine-grained levels.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The relevance of geographic aspects of data has undergone a dramatic increase in recent years. The ability to process large amounts of data is allowing for research related to geography not possible before, in areas such as history (Scheidel et al., 2012), ethnography, and analysis of news and web content (Gelernter and Mushegian, 2011). Such studies ask questions connected with geography, such as how territories of various peoples grow and shrink over time (Guldi, 2009), how different people refer to features of landscape (Derungs and Purves, 2013), how news spreads through traditional and social media (Teitler et al., 2008; Sankaranarayanan et al., 2009; Starbird and Palen, 2011), and how dialects evolve in space and time. These questions require some form of geographic labels on the data.

Some of the data being generated today comes with automatic, high-quality geographic metadata (called **geotags**) such as the coordinates derived via GPS that are associated with tweets and digital pictures, for those users with GPS-equipped devices who allow this functionality. In other cases, less precise information is available, such as a Twitter user's freeform location, e.g. *Brooklyn* or *Justin Bieber's heart* (Hecht et al., 2011). Many Wikipedia

articles contain human-provided geotags on pages referring to geopolitical entities and landmarks. An ongoing body of research seeks to automatically assign a location (or distribution over many locations) to arbitrary documents (Eisenstein et al., 2010, 2011; Wing and Baldridge, 2011; Roller et al., 2012).

This thesis is concerned with the automatic disambiguation of explicit placenames such as *York* and *Portland* in arbitrary textual contexts. High accuracy on this task is essential for research questions like those mentioned above, as well as for software engineering problems connected to text and geography. Problems in information retrieval are also concerned with geographic references in texts to be stored and retrieved, on the level of the document as well as the explicit placename. Depending on what one's broader goals are, quickly and automatically identifying which *Washington* is being referred to in a historical or contemporary text is often as important as obtaining an overall geographic label for the document containing the word or analyzing the locations of tweets or images related to that document. The methods I introduce achieve high performance on this task by taking advantage of evidence sources usually ignored in the literature but without relying on direct supervision, expensive hand-gathered metadata such as population counts, or oversimplified heuristics.

1.1 Task

Toponym resolution is the task of disambiguating a toponym (placename) to a geographic location based on that toponym's context.

Definitions

A **toponym** is a place name such as *Japan* or *Springfield*, and may refer to a single location on earth or be ambiguous in its referent. For example, *Paris* may refer to the capital of France or to any number of cities in the United States, Canada, and other countries.[1]

In this thesis, a **location** is defined as a point or set of points on the earth's surface. In the case of a single point, a coordinate pair in latitude and longitude is sufficient to encode the location. A contiguous set of points, also called a **region**, may take at least two forms. It may be a simple bounding box, definable by two corner coordinates, or have arbitrary borders such as true geopolitical entities have. Some locations, e.g. the United States, are made up of multiple regions due to holdings separate from a mainland. The **administrative level** of a location is its level in the geopolitical hierarchy, e.g. city, state/province, or country.

The context surrounding a toponym can take various forms. It might be the entire text of the document containing the toponym, or a subset such as all words within a certain window of the toponym. It might also include prior knowledge about the author or time of writing. It could include structural knowledge such as that provided by links in a corpus such as Wikipedia, or information about the author's friends or followers if the toponym is in a social

[1] Throughout this thesis, I use italics when referring to a (potentially ambiguous) toponym such as *Springfield*, but do not use italics when referring to a particular location such as the Springfield in Illinois.

media message such as a tweet or Facebook status. The resolvers I present in this thesis take advantage of textual context on three levels:

Local context is defined as some number of words *or* some number of sentences around a toponym, including the toponym itself.

Document context includes all text in the same document as a toponym. As I will discuss, a document may be comprised of a newspaper article, a Wikipedia page, an entire book, or a contiguous subset of a book, depending on the corpus used and the treatment of that corpus.

Often there is a desire to resolve all toponyms in an entire corpus, such as a collection of books with a common author, theme, or place or time of writing, or a subset of all Wikipedia pages. In these cases, knowledge about the distribution of toponyms in the entire **corpus context** may be used to aid in toponym resolution.

System Components

The typical framework for performing toponym resolution involves several parts. The first step is to detect toponyms in free text, if they have not been provided by a human annotator. This can be achieved with a **named entity recognizer (NER)**, which typically identifies names of people, places, and organizations using contextual information as features for a machine learned classifier trained on annotated text. The identified place names indicate which words and phrases constitute toponyms. Throughout this thesis, I will refer to this process as **toponym identification**, not to be

confused with toponym resolution.

Next, a **gazetteer** must be used to obtain lists of candidate locations for each toponym. For example, a gazetteer might indicate that *Paris* can be used to refer to cities in France, Texas, Arkansas, and so on, with latitude/longitude coordinates given for each such candidate location. Many gazetteers also include other information for each location such as population, elevation, a unique identification number, and geopolitical hierarchy information (e.g. that a particular Paris is in Lamar County, Texas, United States).

Once the toponyms and candidate locations have been obtained for a document or corpus, a **resolver** selects a candidate location for each toponym. Some simple resolvers, e.g. one that always chooses the candidate location with the greatest population, select a particular location for all instances of a toponym *type*. Other more sophisticated resolvers take some form of context into account, and are capable of selecting different candidate locations for separate *tokens* of the same toponym type.

Core Thesis

The primary thesis of this work is that toponym resolvers with accuracy high enough to be used in practical applications can be built with mild or indirect supervision by taking advantage of annotations already made by humans for other purposes. In other words, the expensive and time consuming process of hand labeling training instances specifically for this task is not necessary to obtain robust toponym resolvers. In some of my best performing resolvers,

I utilize document location labels on Wikipedia articles as indirect supervision at training time. At prediction, I combine document-level evidence, local contextual evidence, and knowledge about likely geographic distributions of sets of locations. I lay ground work for framing toponym resolution as the traveling purchaser problem (TPP), using a heuristic construction algorithm and ant colony optimization to solve the TPP instances corresponding to each document, resulting in the joint resolution of all toponyms in a document. The results I present outperform several resolvers representative of the state of the art on a small dataset of news articles and a new, larger dataset of books.

1.2 Motivations

Improving the State of the Art

Leidner (2008) presents a thesis similar in many of its goals to this one. He provides an overview of several toponym resolution methods from previous literature, constructs the TR-CoNLL dataset, and analyzes the performance of previous and novel methods on the corpus. Prior to his work, there was little to no quantitative evaluation data on which to test toponym resolution methods. He also introduces a toponym resolver based on two heuristics: that repetitions of a toponym in one discourse segment likely refer to the same location (the concept of one sense per discourse, borrowed from word sense disambiguation; see e.g. Yarowsky (1995)) and that resolutions of multiple toponyms in one discourse segment that cover a smaller area of the globe are typically preferable to those that cover a larger area of the globe. This thesis

in many ways builds on the foundation laid by Leidner.

Much existing work in toponym resolution relies on some form of heuristics or hand-built rules. Some attempts use simple rules based on information contained entirely in the gazetteer such as population or administrative level (city, state, country, etc.), resolving every instance of the same toponym type to the same location regardless of context (Ladra et al., 2008). Many efforts use other toponyms in a context (local or whole document) and look for potential containment relationships, e.g. *London* and *England* occurring in the same paragraph, or even as the bigram *London, England* (Li et al., 2003; Amitay et al., 2004; Zong et al., 2005; Clough, 2005; Li, 2007; Volz et al., 2007; Jones et al., 2008; Buscaldi and Rosso, 2008; Grover et al., 2010; Loureiro et al., 2011). Others rely on finding unambiguous toponyms first and then disambiguating other toponyms based on geopolitical relationships with or distances to the unambiguous toponyms (Ding et al., 2000). Many favor resolutions of toponyms within a local context or document that cover a smaller area of the globe over those that are more dispersed (Smith and Crane, 2001; Rauch et al., 2003; Leidner, 2008; Loureiro et al., 2011; Zhang et al., 2012). Ireson and Ciravegna (2010) make use of an ontology in the form of Yahoo! GeoPlanet, but consider a toponym's context to be other toponyms occurring as tags applied by the author of that toponym and other users in that author's social network.

A major problem with such approaches is that information about nearby toponyms' resolutions is required to resolve toponyms, either explicitly in the

form of unambiguous toponyms or multi-toponym phrases or implicitly via the global distribution of candidate locations of toponyms in context. This reliance often results in poor coverage, when the required information simply isn't present in the context, and tends to fail in documents that mention locations neither nearby geographically nor sharing a geopolitical relationship. In this thesis, I present resolvers that use all types of words in both local and document context as features in the disambiguation of toponyms. The connection between non-spatial words and locations has been successfully exploited in data-driven approaches to document geolocation (Ding et al., 2000; Serdyukov et al., 2009; Cheng et al., 2010; Eisenstein et al., 2010, 2011; Kinsella et al., 2011; Wing and Baldridge, 2011; Adams and Janowicz, 2012; Roller et al., 2012; Sadilek et al., 2012; Dias et al., 2012) and other tasks (Hao et al., 2010; Pang et al., 2011; Intagorn and Lerman, 2012; Hecht et al., 2012; Louwerse and Benesh, 2012; Adams and McKenzie, 2013). The importance of spatially relevant words like *downtown* that are not explicit toponyms has been shown by e.g. Hollenstein and Purves (2012). My approach takes advantage of the clues that prior efforts rely on without being restricted to using only those clues.

Few probabilistic toponym resolvers exist in the literature. Li (2007) builds a probability distribution over candidate locations for each toponym, but does so in a way that still relies on nearby toponyms that could refer to geopolitical entities that contain that toponym. When such toponyms are absent, the probability distribution used is one built completely by hand, giv-

ing a constant probability mass to each administrative level. Speriosu et al. (2010) use probabilistic topic modeling where each topic is associated with a grid cell on the earth's surface. Toponyms are constrained to be in one of the topics containing a candidate location of that toponym according to a gazetteer. While this approach is probabilistic and does not rely on hand-built heuristics, its performance is low as it only uses the distributional properties of the corpus and a gazetteer as knowledge sources. I present resolvers in this thesis that are probabilistic in a data-driven way, take advantage of several knowledge sources (though never training data annotated explicitly for this task), and achieve results that represent a new state of the art.

Toponym resolvers that use machine learning are also rare in the literature. Framing toponym resolution as a traditional classification task, each toponym type can have a classifier associated with it where the set of labels is the set of candidate locations and features can be extracted from local and/or document contexts. This need for a separate model for every toponym type led Leidner (2008) to state that machine learning approaches to toponym resolution might face daunting tractability challenges. Despite this, Smith and Mann (2003) train a Naive Bayes classifier for each toponym type, using unambiguous mentions like *Springfield, MA* as training instances. However, the need for training instances with such a specific form presents the same coverage problems as many approaches that do not use machine learning. Overell and Rüger (2008) and Overell (2009) escape this problem to an extent by extracting training instances from Wikipedia that use nearby toponyms as

features, but do not capture information that non-toponym words in a nearby context may be contributing to a toponym's disambiguation. Garbin and Mani (2005) extract training instances from unlabeled text but use heuristics like preferring capital cities over other cities to guess at the correct labels for these instances. The resolvers I present that use machine learning do not require specific strings of toponyms in training material and do not apply a simplistic resolution heuristic in the prediction of labels, instead taking advantage of geotags on Wikipedia articles containing toponyms whose locations could be close to those geotags.

There have been limited prior efforts to perform toponym resolution using text other than toponyms themselves. Roberts et al. (2010) use relationships learned between people, organizations, and locations from Wikipedia to aid in toponym resolution when such named entities are present, but do not exploit any other textual context. Mani et al. (2010) and Qin et al. (2010) use local context words in machine learning approaches, but do not take advantage of non-local words in the document and require hand-labeled training data. My approach makes use of all words in local and document context and requires no explicitly labeled toponym tokens.

Entity disambiguation systems such as those of Kulkarni et al. (2009) and Hoffart et al. (2011) disambiguate references to people and organizations as well as locations, but these systems do not take into account any features or measures unique to geography such as physical distance. Here I demonstrate the utility of incorporating distance measurements in toponym resolution sys-

tems. Furthermore, I argue that locations are fundamentally different from other named entities such as people and organizations. Locations are much more grounded in a measurable, immutable reality than people or organizations: they refer to very specific areas of the globe, and these areas rarely move. This constancy can be acknowledged and exploited in algorithms that try to disambiguate references to them. Another kind of constancy is that the set of places with a given name is well known, rarely changes, and is catalogued in gazetteers. There are many people named *Michael Brown*, only a few of them are famous enough to be entered into a catalog such as Wikipedia, the set of Michael Browns changes frequently as people die and are born, and no Michael Brown has any characteristics as true and unchanging in a precisely measurable way as latitude and longitude. One might frame the set of disambiguation tasks as a spectrum from those with clearly, rarely changing possible referents (toponym resolution) at one extreme, to those whose referents are endlessly debatable in both number and nature (word sense disambiguation), with the disambiguation of people and organization names somewhere in the middle.

Towards True Grounding of Language

The need to ground natural language geographically is one component of a more general need to automatically connect text to entities that are not text. In some fields, such as robotics, a form of connecting language to non-language is already in use. For example, a robot might connect a phrase in

a user's command such as "the table" to a representation of a real wooden table in the room. A true connection between natural language and the real world, such as humans possess in their representation of meaning, is at least arguably necessary for a computer system that is able to converse and interact with humans in a natural way. However, questions regarding the nature of such a representation, let alone how one might implement it in a real system, remain largely unanswered and are presumably extremely hard to answer.

Making predictions about which parts of text refer to particular parts of the world (and which parts of the world they refer to) is a task that is a step towards the complete understanding of text, and one that is within reach of current research. If the true connection of language to non-language is **grounding**, then the simpler task of connecting particular words and phrases to places on the globe (or times in history) might be called **pseudogrounding**. Since this task includes correctly resolving composite phrases such as "a settlement thirty miles west of the mouth of the Nile," toponym resolution is an even simpler subset of pseudogrounding that only deals with explicit toponyms.[2] The results I show in this thesis demonstrate that performance on toponym resolution is now at a usable level for applications such as those I discuss below.

[2]See Mani et al. (2010) for a description of the SpatialML system for annotating both toponyms and composite spatial phrases in a consistent, machine-readable format.

Applications of Toponym Resolution

Hill (2006) emphasizes the importance of connecting informal references to places (toponyms) to formal references to places (coordinates or sets of coordinates). Petras (2004) makes the conservative estimate that half of the world's stored knowledge, both printed and digital, has geographic relevance. Skupin and Esperbé (2011) suggest that geographic information pervades many more aspects of humanity than previously thought. Allowing for the querying and exploration of pieces of that knowledge in a truly geographically informed way requires more powerful tools than a keyword-based search can provide, in part due to the ambiguity of toponyms. Robust toponym resolution plays an essential role in the automated geographic indexing and searching of information.

There are many other, more specific ways the automatic resolution of ambiguous toponyms in text could be practically useful. For example, there are systems in place that anonymously read text a user has entered on the Internet such as emails and present content such as advertisements that are tailored to that user based on the topics of his or her text. Content could also be tailored based on an estimated location for that user based on his or her text, including any toponyms used.

Many cases of identity theft involve a perpetrator operating significantly far away from the victim. If this perpetrator is entering text on a computer or mobile phone while using the stolen identity, toponym resolution could help quickly identify the perpetrator's location (despite potential

attempts to disable any automated location services like GPS), adding a geographic component to automated defenses like those of Chou et al. (2004).

By running a toponym resolver on a collection of text, a summary of the geographic facets of that text can be presented to a user such as a historian or scholar of literature, e.g. by using a visualization like the ones I present here. Toponym resolution also plays an essential role in automated geographic indexing and information retrieval. These aspects are useful for historical research that combines age-old geographic issues like territoriality with modern computational tools (Guldi, 2009), ethnographic analyses of the kinds of language used to refer to geographic entities (Derungs and Purves, 2013), studies of the effect of historically recorded travel costs on the shaping of empires (Scheidel et al., 2012), and systems that convey the geographic content in news articles (Teitler et al., 2008; Sankaranarayanan et al., 2009) and microblogs (Gelernter and Mushegian, 2011).

1.3 Contributions

This thesis presents strong results in toponym resolution with several methods, using corpora with manually annotated resolved toponyms only for evaluation. Results are also given for a collection of books that is new to this task and much larger than previously published evaluation sets (Crane, 2000). I examine evaluation methods previously used for toponym resolution and other geographically relevant tasks, and argue for a set of metrics that is appropriate for toponym resolution and that gives a more complete picture

of performance than many previously used metrics. I introduce the concept of framing toponym resolution as the traveling purchaser problem, discussing the motivations for doing so and presenting methods that lay the ground work for this novel approach. Finally, several methods for automatically generating visualizations from such an annotated corpus are presented that are useful for error analysis for those doing future work in toponym resolution, and for data exploration for those studying large corpora of historical or current texts.

Datasets

In addition to outperforming previously published results on a small collection of news articles called TR-CoNLL (roughly 200,000 words, about 6000 of which are toponyms), I present strong results on a much larger collection of books that is new to the task of toponym resolution: the Perseus Civil War and 19th Century American Collection (roughly 58 million words, about 240,000 of which are toponyms).

Resolvers

Unsupervised resolvers I present two toponym resolvers that require very little prior knowledge and perform at a level many percentage points better than chance. Both require only a corpus (with either manually annotated toponyms, or as raw text if a named entity recognizer is used) and a gazetteer that lists candidate locations for each toponym (other information, such as population and geopolitical hierarchies, are not needed). One resolver,

MinDist, simply tries to resolve toponyms within each document to locations nearby on the globe. The other, SPIDER, also does this but computes a notion of global prominence for each location, choosing more prominent locations more often than MinDist. Resolvers requiring such little prior knowledge but of comparable performance have not been previously published.

Noisily supervised resolvers Several resolvers are presented that incorporate more knowledge than MinDist and SPIDER while still not relying on hand-annotated training instances of the same type as the evaluation instances used. The TRIPDL resolver uses a predicted location for each document (as output by a supervised document geolocation system) to make predictions for the toponyms in that document. WISTR also uses this knowledge, but it creates training instances at the toponym level by searching for toponyms in a separate training corpus with candidate locations close to location labels on the documents containing them. The TRAWL resolver combines document and local context information in a confidence-driven way, also taking into account administrative level. Both WISTR and TRAWL use machine learning to construct a separate model for each toponym word type, taking advantage of indirect training material.

Toponym resolution as the traveling purchaser problem Two of the strongest resolvers I present are the WISTR and SPIDER resolvers (the latter being particularly strong when initialized with WISTR's output). These

two resolvers represent two extremes on a spectrum. WISTR focuses exclusively on a toponym's local textual context, disambiguating it based on training instances extracted from Wikipedia and ignoring how the set of predicted locations for a document is distributed on the earth. The SPIDER resolver ignores textual context and does not require any supervision, but rather tries to predict a set of locations in a small geographic region. WISTR performs better than SPIDER on the TR-CoNLL corpus of international news articles (large geographic score), while the opposite is true on the CWar corpus of American Civil War writings (small geographic scope).

Ideally, a single resolver would perform best on all corpora. Towards that end, I develop two resolvers based on framing toponym resolution as the traveling purchaser problem (TPP), which explicitly model the individual resolution power of WISTR and the spatial minimality of SPIDER as two components of a cost function to be optimized for each document. My TRACO resolver simulates populations of ants with properties that adapt and evolve to prefer different kinds of documents and corpora as they solve TPP. These efforts form a foundation for future, adaptive toponym resolvers. They also yield interesting outputs aside from resolved toponyms, such as ordered geographic tours through documents and patterns of evolution that reflect differences in the kinds of text they are given.

Visualizations

Several automatically generated visualizations are presented that demonstrate what one might do with a corpus that has been geographically grounded, e.g. via toponym resolution. I present visualizations to be viewed in Google Earth as well as those that are standalone programs, focusing on the advantages of the latter. Principles for creating effective graphical views of data are discussed, along with how they apply to these visualizations. A major goal in this section is to demonstrate how such visualizations can aid research in the humanities, illuminating patterns related to geography in historical and literary texts that might otherwise elude a scholar who only read the texts or a portion thereof.

There is now far more machine readable text than ever before, whether in the form of digitized historical texts or text generated by Internet users, and such data is ripe for study in the fields of history, linguistics, sociology, economics, and literature, to name a few. Scholars and educators in the humanities need tools that aid in the digestion of large corpora of texts, harnessing the ability of computers to process large amounts of data and display trends in that data that might otherwise not be obvious. The visualizations I present here are examples of such tools.

Static visualizations Some of the visualizations I present are static in that they give an overall picture of which parts of the globe are referred to in a collection of texts. Going beyond the simple polygons giving the outer borders

of the location of a piece of text presented in Leidner (2008), the visualizations I present show how many times each location was mentioned, allow the user to click on a location to bring up the portions of the text that refer to that location, and allow the user to show or hide any subset of the full collection loaded into the visualization. Zooming and panning capabilities allow the visualization to be informative on displays of many different sizes.

Dynamic visualization I also present a visualization that takes the user on a tour of the texts in question, showing how the areas of the globe referred to by the text shift as the text progresses. This allows for automatically generated stories that walk the user through the narrative of the texts in a dynamic way that is alluded to but not implemented by Leidner (2008).

1.4 Software

The FIELDSPRING software package implements the methods described in this thesis. Source code and instructions for replicating results, running on new data, and producing visualizations are available at the GitHub repository at `github.com/utcompling/fieldspring`.

Chapter 2

Data

Some of the material in this chapter also appears in Speriosu and Baldridge (2013).

2.1 Gazetteer: GeoNames

The gazetteer used to produce the lists of candidate locations can be considered data, since different available gazetteers may have more or less coverage and more or less structural/hierarchical data such as which county/state/province/country a city is in, and so on.

GeoNames is a freely available gazetteer containing over eight million entries worldwide. Each entry contains a name (sometimes more than one) for a location, and a coordinate in latitude and longitude. Entries also include information about type (*CITY*, *STATE*, etc.), which country a location lies in, and an administrative code that can give a more middle administrative level, e.g. which U.S. state a city is in. The following is an excerpt from GeoNames, showing some of the most important fields for cities named Paris in France, Texas, and Arkansas:

2988507	Paris	48.853	2.348	P	FR	A8	2138551	2011-11-04
4717560	Paris	33.660	-95.555	P	US	TX	25171	2011-05-14
4125402	Paris	35.292	-93.729	P	US	AR	3532	2011-05-14

The fields shown are, respectively, ID number, name, latitude, longitude, feature class (P for populated place), country code, code for an intermediate administrative level (A8 for Île-de-France, TX for Texas, and AR for Arkansas), population, and date of last modification.

Ideally, locations that are typically considered to have greater areas than a single point would be represented as a complete and accurate border. Then, in any resolver that computes the distance between two locations, the distance between a large location such as a country and any city within that country would be correctly considered zero. However, such accurate border data may not be freely or easily available, and processing it may be mathematically complex in the general case. One promising freely available tool for working with geographic data that includes polygonal location information is OpenStreetMap (Haklay and Weber, 2008). Built from the submissions of users, OpenStreetMap could be used as an advanced gazetteer as well as a mechanism for viewing the output of toponym resolvers. While I explore the use of some such tools in Chapter 4, I leave integration with OpenStreetMap to future work.

GeoNames gives the locations of regional items like states, provinces, and countries as single points. This is clearly problematic when one seeks connections between words and locations: e.g. we might learn that many

words associated with the USA are connected to a point in Kansas. To get around this, I represent regional locations as a set of points derived from the gazetteer. Since regional locations are named in the entries for locations they contain, all locations contained in the region are extracted (in some cases over 100,000 of them) and then k-means (Hartigan and Wong, 1979) is run to find a smaller set of spatial centroids. These **representative points** act as a tractable proxy for the spatial extent of the entire region. k is set to the number of 1° by 1° grid cells covered by that region. Figure 2.1 shows the points computed for the United States.[1] A desirable property of this representation is that it does not involve region shape files and the additional programming infrastructure they require.

Formally, each location l has an initial set of points P_l contained in it according to the gazetteer. $||P_l|| = 1$ for all cities, as I do not consider toponyms at the sub-city level in this thesis. For each l where $||P_l|| > 1$, k_l is the number of 1° by 1° grid cells containing at least one point in P_l. Then the k-means clustering algorithm is run, resulting in a set of representative points R_l that is typically much less than P_l. For cities, $P_l = R_l$ is the single point representing the location. For regional locations, $||R_l|| = k_l$.

When the distance between two locations is computed, the great circle distance (i.e. the distance along the earth's surface, as a car would drive on a perfectly straight road) between the closest pair of representative points – one

[1]The representation also contains three points each in Hawaii and Alaska not shown in Figure 2.1.

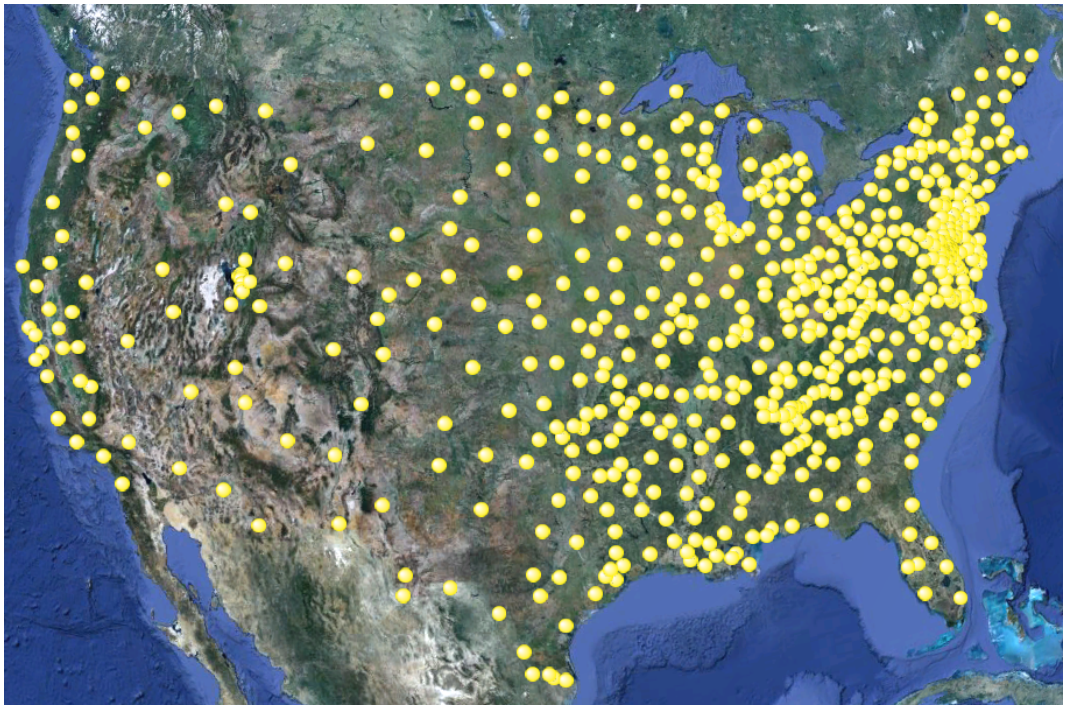


Figure 2.1: Points representing the United States.

from each location – is used. Thus, the distance between two locations with only one representative point each is simply the great circle distance between those two points. The distance between a location with many representative points and one with a single representative point is the distance from the single point location to the representative point of the other location closest to that point. Note that this means that the distance from a city to the country containing that city will often be nonzero though likely not large unless that city is in an otherwise sparsely populated area of that country, far away from a centroid computed in the k-means clustering for that country.

Formally, the distance between locations l and q, with representative point sets R_l and R_q respectively, is defined as

$$dist(l, q) = min_{l_i \in R_l, q_i \in R_q} dist(l_i, q_i)$$

where the great circle distance between any two single points $x = (x.lat, x.lon)$ and $y = (y.lat, y.lon)$ in latitude and longitude coordinates is defined as

$$dist(x, y) =$$
$$arccos(sin(x.lat)sin(y.lat) + cos(x.lat)cos(y.lat)cos(y.lon - x.lon)).$$

This yields a distance in radians, which must be multiplied by a constant to convert to a common distance measure if desired, e.g. 6372.8 km.

2.2 Toponym Resolution Corpora

TR-CoNLL

The TR-CoNLL corpus was compiled by Leidner (2008) and contains 946 news articles from REUTERS published between August 22 and August 31, 1996 inclusive. It is about 204,000 words. The articles range in length from a few hundred words to several thousand words. The corpus has manually annotated named entities due to its previous use in the 2003 Conference on Natural Language Learning (CoNLL) shared task on named entity recognition. Each annotated toponym in the corpus was further annotated with lists of candidate locations by Leidner (2008) using a combination of the GNIS gazetteer of the U.S. Geographic Survey, the GNS gazetteer of the National Geospatial Intelligence Agency (NGA), and the CIA World Factbook (WFB).

Figure 2.2 an example of a document in the TR-CoNLL corpus, with manually identified toponyms shown in bold and each sentence beginning on a new line. The document has several noteworthy properties relevant to toponym resolution that are typical of many texts. First, it contains toponyms like *Springfield* and *Washington* with more than one reasonably likely candidate location given no context. In at least one case, *Washington*, the referenced location is not the candidate location with the greatest population. (At the time of writing, the United States capital has a population of about 620,000, while that of Washington state is about 6.8 million.) Second, it contains several instances of referenced locations that are contained within other referenced locations: St. Albans, Roxbury, and Montpelier are all contained within Ver-

Amtrak Train Derails, Three Injured – Officials.
MONTPELIER, **Vt**.
1996-08-27
At least three people were injured when an Amtrak passenger train slammed into an empty logging truck and derailed Tuesday, officials said.
The Vermonter, which runs between **St. Albans**, **Vermont**, near the Canadian border and **Washington, D.C.**, collided with the truck at 7:51 A.M. EDT near the rural town of **Roxbury** some 15 miles southeast of the state capital **Montpelier**, Amtrak spokeswoman Maureen Garrity said.
Vermont Central Hospital spokesman Dan Pudvah said two of the injured were treated there – the truck driver, who was suffering from multiple trauma injuries, and a passenger.
Pudvah said he understood other people with minor injuries were being treated at the scene.
Garrity said a train conductor was also injured.
The train's engine and its six cars derailed but were still standing, state police said.
The exact number of passengers on the train was not known.
"We had 70 reservations for the train, but that doesn't mean there were 70 passengers aboard," Garrity said.
Uninjured passengers were to be taken by bus to **Springfield**, **Massachusetts**, where they will be put aboard another train to continue their journey to **New York City** and **Washington**, Garrity said.
She said the train was travelling at 54 MPH when it crashed into the truck, which was crossing the tracks onto a dirt road in the rural area bordering the Northfield Mountains.

Figure 2.2: A document from the TR-CoNLL corpus, with manually identified toponyms shown in bold and each sentence beginning on a new line.

mont, and Springfield is contained within Massachusetts. Third, repetitions of the same toponym (possibly using alternate names, e.g. *Washington, D.C.* versus *Washington* and *Vermont* versus *Vt*) refer to the same location; the idea of "one sense per document" is upheld. Fourth, there is a clear region that is relevant for the document: the Eastern Seaboard of the United States, with a focus on the state of Vermont and nearby areas. Finally, there are numerous non-toponym words and phrases in the document that could be used to help a human or machine disambiguate toponyms in it, such as *Amtrak*, *train*, *logging*, *Vermonter*, *Canadian border*, *Vermont Central Hospital*, *Northfield Mountains*, names of people associated with particular areas, and other words whose geographic relevance may be less obvious.

For the purposes of this thesis, the TR-CoNLL corpus was divided into a development and a test portion. No training portion was created because the corpus is used only for unsupervised methods or methods noisily supervised with separate data. After sorting the documents in TR-CoNLL by document ID ascending, every third document was placed into TRC-TEST, beginning with the third. The remaining documents were placed into TRC-DEV. That is, the first and second documents were put into TRC-DEV, the third into TRC-TEST, the fourth and fifth into TRC-DEV, the sixth into TRC-TEST, and so on. This results in the development set (631 documents) being roughly twice the size of the test set (315 documents).

There were several systematic types of errors in the original TR-CoNLL corpus, such as coordinates being swapped for some locations and some lon-

gitudes being zero or the negative of their correct values. I repaired many of these errors, though some more idiosyncratic mistakes remain. These fixes and instructions for how to apply them to the original corpus are available at the Fieldspring GitHub page at `github.com/utcompling/fieldspring`.

The Perseus Civil War and 19th Century American Collection (CWar)

This corpus contains 341 books written primarily about and during the American Civil War, is about 58 million words, and is part of the Perseus Digital Library project (Crane, 2000). The corpus underwent an initial automated process where toponyms were identified with a named entity recognizer and resolved using simple rules. These annotations were then corrected by hand. It includes biographies of Civil War leaders like Ulysses S. Grant and Abraham Lincoln as well as personal war memoirs written during the war by soldiers, women, children, and others living in the early United States. Other documents are less closely related to the Civil War but involve other aspects of 19th century America.

Figure 2.3 shows an excerpt from one such document, *Louis Agassiz: A Life in Science*, written by Edward Lurie in 1960. The biography contains many samples of the writing of Agassiz, a Swiss scientist who spent time as a professor at Harvard University in the mid-1800s. The excerpt mentions *Springfield* and *Hartford*, and context indicates that the cities referred to are those in Massachusetts and Connecticut respectively, though the word *Connecticut* refers in this case to the river rather than the state. The references

Thus American science lacks the scope which is characteristic of higher instruction in our old **Europe**.
Objects of art are curiosities but little appreciated and usually still less understood.
On the other hand, the whole population shares in the advanced education provided for all...
From **Springfield** the railroad follows the course of the Connecticut as far as **Hartford**, turning then directly toward the sea-coast. The valley strikingly resembles that of the Rhine between **Carlsruhe** and **Heidelberg**.

Figure 2.3: An excerpt from *Louis Agassiz: A Life of Science*, from the CWar corpus.

to the European cities Carlsruhe[2] and Heidelberg are made clearer by the reference to Europe itself in nearby context. In this case, the author is principally referring to a region of the United States, though parallels are drawn to locations thousands of kilometers away.

In this thesis, CWar is divided into development (CWar-dev) and test (CWar-test) sets in the same way as TR-CoNLL, resulting in 228 documents for development and 113 documents for test. States and countries are not annotated in CWar, so I do not evaluate end-to-end using NER plus toponym resolution for it as there are many (falsely) false positives.

[2]This is the old spelling of Karlsruhe.

2.3 Document Geolocation Corpus: GeoWiki

The GeoWiki dataset consists of over one million English articles from a February 11, 2012 dump of Wikipedia. Each article has a gold location annotation in the form of a point in latitude/longitude coordinates. The corpus was divided into training (80%), development (10%), and test (10%) sets at random and performed various preprocessing to remove markup in exactly the same manner as Wing and Baldridge (2011). The training portion is used in this thesis as training data for resolvers that utilize a document-level location prediction as part of toponym resolution, and to extract training instances for the WISTR resolver described in section §3.1. I also extract data from Wikipedia's link structure in two ways, both to extract additional training instances and to estimate the probability of traveling from one group of locations to another, based on link frequencies, for use in the traveling purchaser problem (TPP) resolvers.

2.4 Toponym Frequency and Ambiguity

In order to understand the scope of the problem of toponym resolution, it is useful to examine how often toponyms occur as well as how ambiguous they are. Table 2.1 gives statistics for each corpus used in the evaluation of toponym resolution methods. The quantities shown are number of documents, number of word tokens, number of word types, number of toponym tokens, number of toponym types, average ambiguity (number of candidate locations) per toponym token, and maximum ambiguity for any toponym. The statis-

Corpus	docs	toks	types	toks_{top}	types_{top}	amb_{avg}	amb_{max}
TRC-DEV	631	136k	17k	4356	613	15.0	857
TRC-DEV-NER	-	-	-	3165	391	18.2	857
TRC-TEST	315	68k	11k	1903	440	13.7	857
TRC-TEST-NER	-	-	-	1346	305	15.7	857
CWAR-DEV	228	33m	200k	157k	850	29.9	231
CWAR-TEST	113	25m	305k	85k	760	31.5	231

Table 2.1: Statistics of the corpora used to evaluate toponym resolution performance. The columns correspond to corpus name, number of documents, number of word tokens, number of word types, number of toponym tokens, number of toponym types, average ambiguity (number of candidate locations per toponym token), and maximum ambiguity.

tics are based on NER-identified toponyms where indicated and gold-standard toponym identifications elsewhere.

The task of this work is restricted to the disambiguation of words and phrases that refer to geopolitical entities like cities, states, and countries by proper name. The statistics from Table 2.1 indicate that such references occur about 7 times per news article (one in every 33 words) in TR-CoNLL and about 700 times per book (one in every 240 words) in CWAR. According to GEONAMES, the average number of possible referents per toponym in TR-CoNLL is about 14, while this number is about 30 for CWAR. Clearly, the frequency and ambiguity of toponyms varies heavily by domain.

A far more difficult task is that of attempting to resolve geographically any kind of text that refers to a point or area of the globe. Even the task of identifying all such references automatically is challenging, as the kinds of

clues that might indicate that a string of text is such a reference have a much greater variety than the clues used to identify the proper names of places. Even using part of speech or syntactic annotations does not make the problem easy. The highly productive and metaphoric qualities of language mean that simplistic methods like counting the number of prepositions that could be (but aren't necessarily) locative like 'in' or 'at,' even within some syntactic context, are not likely to provide accurate statistics. A human can determine that "my old place" is a geographic reference while "a place in my heart" is not, but building a system to reliably recognize the difference is a full fledged task unto itself.

The SpatialML annotation scheme and corpora have paved the way for such tasks, hand-annotating about 6000 arbitrary locative references in 428 news articles and weblogs, about 1000 references in 100 medical documents about emerging diseases, about 3500 references in 121 news releases from U.S. Immigration, and about 4000 references in 298 other arbitrary documents (Mani et al., 2010). Assuming the annotations are correct and complete, this comes to an average of about 16 geographic references per document, where each document is about the length of a news article. Comparing this to the 7 toponyms per news article in TR-CoNLL, one can make the very rough claim that a little less than half of all geographic references in the news domain are toponyms.

Chapter 3

Heuristic and Text-Based Methods

Some of the material in this chapter also appears in Speriosu and Baldridge (2013).

3.1 Resolvers

Baseline Resolvers

Random The RANDOM resolver simply selects a candidate location at random for each toponym, resulting in an average case accuracy equal to the inverse of the average ambiguity of toponyms in the corpus. In general, such a random resolver (which includes knowledge from a gazetteer) will still perform at a level much higher than one which selects a point on the earth's surface (or even a point guaranteed to be on land) completely at random. This is true not only because there is a chance that exactly the correct location will be selected, but also because sets of locations with the same name are typically autocorrelated: the set of Springfields forms a cluster of locations in a much tighter arrangement than a uniform distribution over the earth's surface is likely to produce (Brunner and Purves, 2008). I do not consider a resolver that makes no use of a gazetteer in this work.

The runtime of RANDOM is $O(||T||)$, where $||T||$ is the number of toponyms in the corpus.

Population The POPULATION resolver always selects the candidate location with the greatest population for each toponym. Naturally, it can only be used when such information is available, and is more effective when the population data used was taken from a time period close to the time period to which the corpus is relevant. Population data is also an extremely work-intensive knowledge source to gather, given the difficulty of accurately counting the number of individuals living in any geopolitical entity.

The POPULATION resolver is a strong baseline in many cases. For example, if an author mentions *Chicago* in a modern text, she is extremely likely to be referring to the large city in Illinois. In other cases, however, POPULATION is guaranteed to perform quite poorly. When a toponym has several candidate locations with reasonably large populations, e.g. *Portland* or *Springfield*, POPULATION will often predict incorrectly. It in effect only knows about one Springfield, and always predicts it given its name. The cases that are most interesting from a toponym resolution perspective are those for which the level of uncertainty about the geographic referent of a word or phrase is high. POPULATION assumes that toponyms are in fact *un*ambiguous, thus effectively disregarding the task altogether. So, while I include results from POPULATION using population data from GEONAMES, I do not consider it a true competitor as a toponym resolver despite its good performance in terms

of overall accuracy, as it is guaranteed to behave incorrectly in the very cases that motivate the task. Furthermore, I will show that for a corpus whose time of writing and/or time period being discussed differ from the time the population data was gathered by 100-200 years, methods that do not take population data into account far outperform POPULATION.

The runtime of POPULATION is $O(||T|| \cdot maxamb(T))$, where $||T||$ is the number of toponyms in the corpus and $maxamb(T)$ is the maximum ambiguity (number of candidate locations) of any toponym.

Unsupervised Algorithmic Resolvers

Two essentially unsupervised resolvers I present in this thesis are the MINDIST and SPIDER resolvers. Both work on the intuition that nearby toponyms in text (where "nearby" means "in the same document" here) tend to refer to nearby locations on the earth, reflecting the proximity intuitions assumed by many previous efforts in a more general way (Smith and Crane, 2001; Rauch et al., 2003; Leidner, 2008; Loureiro et al., 2011; Zhang et al., 2012). Neither makes use of any outside knowledge or labeled data other than a gazetteer.

MinDist MINDIST selects, for each toponym, the location that minimizes the total distance to some resolution (i.e. some selection of candidate locations) of all other toponyms in the same document. MINDIST captures the intuition that if one mentions an unambiguous city in Illinois and then men-

```
for doc ∈ corpus do
  for top_i ∈ doc do
    overallmin ← ∞ // Initialize min. distance for this toponym's candidates
    for cand_a ∈ top_i do
      totaldist ← 0 // Reset total distance for this candidate to other toponyms'
      for top_j ∈ doc do
        if top_i ≠ top_j then
          min ← ∞ // Initialize min. distance between two particular locations
          for cand_b ∈ top_j do
            dist ← distance(cand_a, cand_b)
            if dist < min then
              min ← dist // Found shorter distance from cand_a to top_j
          totaldist ← totaldist + min // Update total distance for candidate
      if totaldist < overallmin then
        overallmin ← totaldist // Found a new optimal candidate
        cand_opt ← cand_a
    top_i.predict(cand_opt) // Predict the optimal candidate for this toponym
```

Figure 3.1: MinDist pseudocode.

tions *Springfield*, he or she is likely (though not guaranteed) to be referring to Springfield, Illinois. More generally, MinDist favors resolutions that clump all resolved toponyms within a document to locations near each other. While this intuition is not necessarily true in general, the results I present show that it nevertheless performs much better than Random. Figure 3.1 gives pseudocode for the MinDist algorithm.

Figure 3.2 gives an example of MinDist resolving the toponyms A, B, and C, all of which are mentioned in one document. The algorithm disambiguates each one so that it minimizes the distance to the closest candidate location of each other toponym. The result is a tight clumping of predicted locations (A_1, B_2, and C_1). In general, however, the selection of a particular

toponym's location based on some candidate location of another toponym does not guarantee that the latter location is selected. When the algorithm finds B_2 to be the closest B to A_1 while selecting a location for A, this does not *require* that B_2 also be selected for B (though it happens to be selected in this and many other cases); the location for B is selected separately.

MinDist does poorly in cases where a toponym like *Paris* occurs in the same document as *Dallas*, and the resolver chooses Paris, Texas, even though the author was talking about a flight from the large Texan city to the large French city, not the small Texan town. MinDist has no sense of which locations are more likely a priori to be referred to.

The runtime of MinDist is $O(T_{max}^2 \cdot ||D|| \cdot maxamb(T)^2)$, where T_{max} is the maximum number of toponyms in any document in the corpus, $||D||$ is the number of documents in the corpus, and $maxamb(T)$ is the maximum ambiguity of any toponym, since each candidate location of each toponym must be compared in distance to each candidate location of each other toponym in the same document.

SPIDER One straightforward way to incorporate prior knowledge about locations is by using Population. However, as mentioned above, Population is not always applicable, requires large amounts of human intervention, and ignores the ambiguity inherent in toponyms. Thus, it is desirable to have an unsupervised (i.e. lacking specific information about particular locations other than coordinates) resolver still capable of capturing the *prominence* of differ-

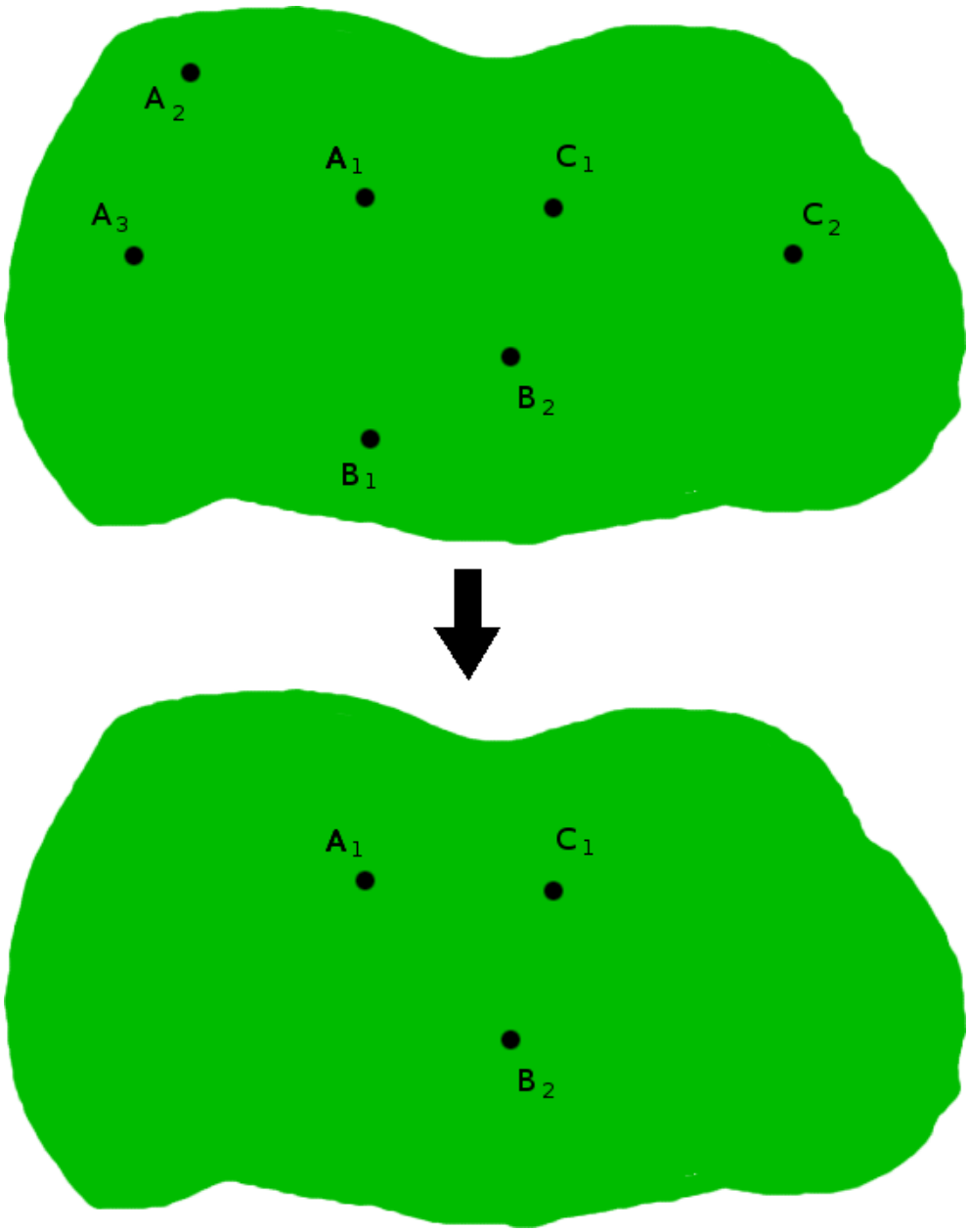


Figure 3.2: Example of MinDist resolving toponyms in one document. The set of predicted locations A_1, B_2, and C_1 form a tight clump.

ent locations in some way. That is, while the intuition behind MinDist that nearby locations tend to get mentioned nearby in text is sometimes true, in other cases a location may be referred to because of its thematic relevance, because a long trip is being described, or other reasons. A resolver that captures at least some of the cases where this intuition fails should improve results.

Leidner (2008) describes two general and useful *minimality* properties of toponyms:

- **one sense per discourse**: multiple tokens of a toponym in the same text generally do not refer to different locations in the same text
- **spatial minimality**: different toponyms in a text tend to refer to spatially near locations

Many toponym resolvers exploit these (Smith and Crane, 2001; Rauch et al., 2003; Leidner, 2008; Grover et al., 2010; Loureiro et al., 2011; Zhang et al., 2012). Here, I define SPIDER (Spatial Prominence via Iterative Distance Evaluation and Reweighting) as a strong representative of such textually unaware approaches. In addition to capturing both minimality properties, it also identifies the relative prominence of the locations for each toponym in a given corpus.

SPIDER resolves each toponym by finding the location for each that minimizes the sum distance to *some* candidate location of *all* other toponyms in the same document. On the first iteration, it tends to select locations that

clump spatially: if *Paris* occurs with *Dallas*, it will choose Paris, Texas even though the topic may be a flight from Texas to France. Further iterations bring Paris, France into focus by capturing its prominence across the corpus. The key intuition is that most documents will discuss Paris, France and only a small portion of these mention places close to Paris, Texas; thus, Paris, France will be selected on the first iteration for many documents (though not for the *Dallas* document). SPIDER thus assigns each candidate location a weight (initialized to 1.0), which is re-estimated on each iteration. The adjusted distance between two locations is computed as the great circle distance divided by the product of the two locations' weights. At the end of an iteration, each candidate location's weight is updated to be the fraction of the times it was chosen times the number of candidates for that toponym. The weights are global, with one for each location in the gazetteer, so the same weight vector is used for each token of a given toponym on a given iteration.

For example, if after the first iteration Paris, France is chosen thrice, Paris, Texas once, and Paris, Arkansas never, the global weights of these locations are $(3/4)*3=2.25$, $(1/4)*3=.75$, and $(0/4)*3=0$, respectively (assume, for the example, there are no other locations named *Paris*). The sum of the weights remains equal to the number of candidate locations. The updated weights are used on the next iteration, so Paris, France will seem "closer" since any distance computed to it is divided by a number greater than one. Paris, Texas will seem somewhat further away, and Paris, Arkansas infinitely far away. The algorithm continues for a fixed number of iterations or until the

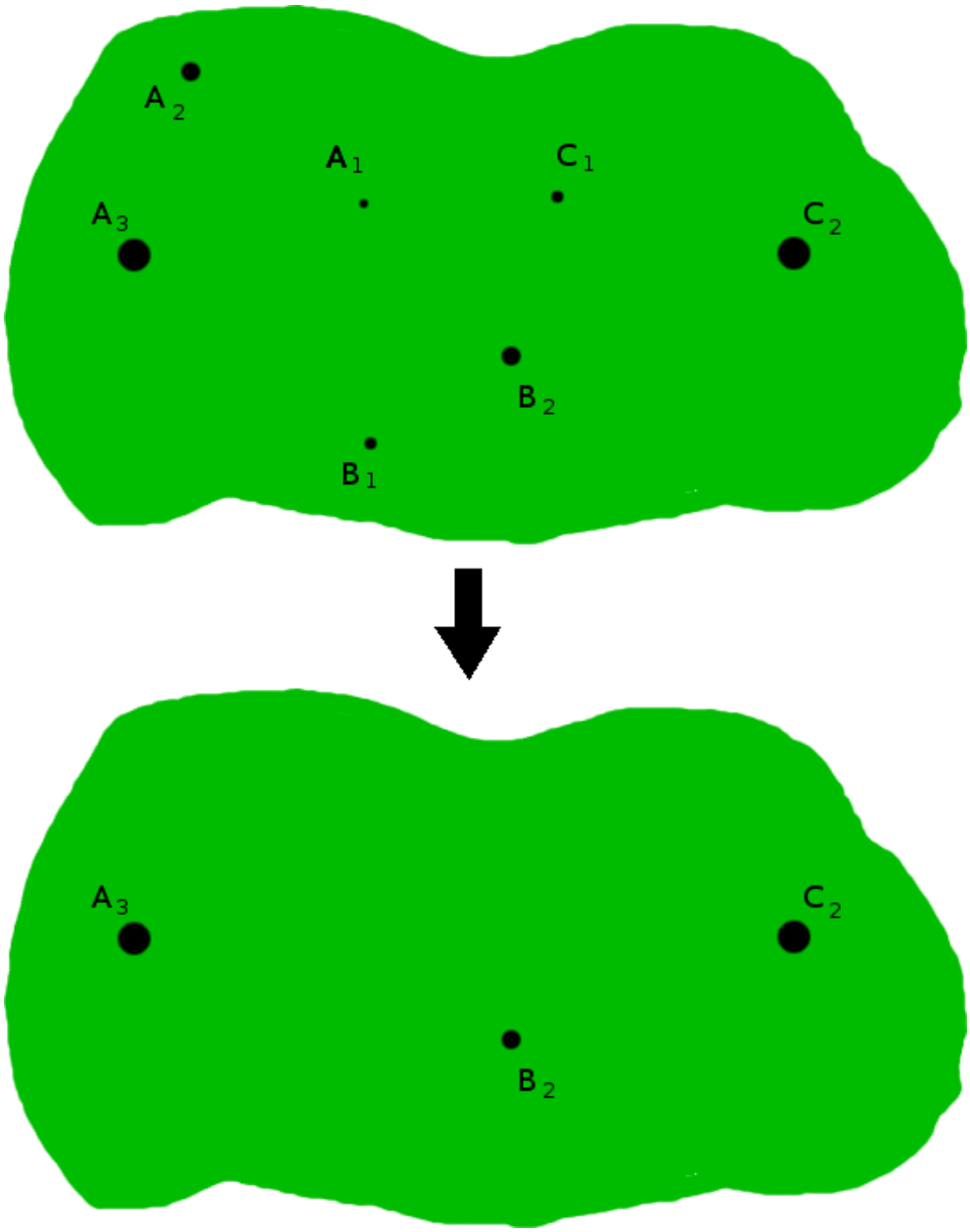


Figure 3.3: Example of SPIDER resolving toponyms in one document on a particular iteration. The sizes of the candidates represent their weights, which reflect the frequency of their being selected on previous iterations. The set of predicted locations A_3, B_2, and C_2 do not form as tight of a clump as in Figure 3.2.

```
for doc ∈ corpus do
  for top ∈ doc do
    for cand ∈ top do
      cand.weight ← 1.0 // Initialize weights
for iter ∈ {0...NUMITERATIONS} do
  for doc ∈ corpus do
    for top ∈ doc do
      top.count ← 0 // Reset toponym counter
      for cand ∈ top do
        cand.count ← 0 // Reset candidate counter
  for doc ∈ corpus do
    for top_i ∈ doc do
      overallmin ← ∞ // Initialize min. distance for all candidates of top_i
      for cand_a ∈ top_i do
        totaldist ← 0 // Reset candidate's total distance to other toponyms'
        for top_j ∈ doc do
          if top_i ≠ top_j then
            min ← ∞ // Initialize min. distance between two given locations
            for cand_b ∈ top_j do
              dist ← distance(cand_a, cand_b)/(cand_a.weight * cand_b.weight)
              if dist < min then
                min ← dist // Found shorter distance
            totaldist ← totaldist + min // Update total distance for candidate
        if totaldist < overallmin then
          overallmin ← totaldist // Found a new "closer" candidate
          cand_opt ← cand_a
      top_i.predict(cand_opt) // Predict the optimal candidate for this iteration
      cand_opt.count ← cand_opt.count + 1 // Increment candidate counter
      top_i.count ← top_i.count + 1 // Increment toponym counter
  for doc ∈ corpus do
    for top ∈ doc do
      for cand ∈ top do
        cand.weight ← (cand.count/top.count) * top.ambiguity // Reweight
```

Figure 3.4: SPIDER pseudocode.

weights do not change more than some threshold. Here, I run SPIDER for 10 iterations; the weights have generally converged by this point. Figure 3.3 gives an example of the SPIDER algorithm on a particular iteration, where changed weights cause the set of predicted locations to change, versus the simpler MinDist algorithm. Figure 3.4 gives the pseudocode for the SPIDER algorithm.

When only one toponym is present in a document, SPIDER simply selects the candidate with the greatest weight. When there is no such weight information, such as when the toponym does not co-occur with other toponyms anywhere in the corpus, it selects a candidate at random.

SPIDER captures the intuition that if a location tends to get selected often in a corpus, it is likely to be a more prominent location than its competing locations. Its increasing weight can cause it to be selected even more often in future iterations, while still allowing a less prominent location to be selected if the evidence in its document (many toponyms with candidates near it) is strong enough. Thus, SPIDER still relies on the intuition that texts tend to refer to nearby places, but it relaxes that constraint while adding the intuition that texts tend to refer to the same locations multiple times in a corpus.

SPIDER is a heuristic algorithm not based on the optimization of an objective function. Its worst case running time is $O(T_{max}^2 \cdot ||D|| \cdot maxamb(T)^2)$, where T_{max} is the maximum number of toponyms in any document in the corpus, $||D||$ is the number of documents in the corpus, $maxamb(T)$ is the maximum ambiguity of any toponym, and i is the number of iterations. The

algorithm is essentially MinDist repeated i times, with some added computations that are not further dependent on the size of any inputs.

Text-Driven Resolvers

The text-driven resolvers presented in this section all use local context windows, document context, or both, to inform disambiguation.

TRIPDL In this resolver, I use a document geolocator trained on GeoWiki's document location labels. Others—such as Smith and Crane (2001)—have estimated a document-level location to inform toponym resolution, but mine is the first I am aware of to use training data from a different domain to build a document geolocator that uses all words (not only toponyms) to estimate a document's location. I use the document geolocation method of Wing and Baldridge (2011), which discretizes the earth's surface into 1° by 1° grid cells and assigns Kullback-Leibler divergences to each cell given a document, based on language models learned for each cell from geolocated Wikipedia articles. I obtain the probability of a cell c given a document d by the standard method of exponentiating the negative KL-divergence and normalizing these values over all cells:

$$P(c|d) = \frac{\exp(-KL(c,d))}{\sum_{c'} \exp(-KL(c',d))}$$

This distribution is used for all toponyms t in d to define distributions $P_{DL}(l|t,d)$ over candidate locations of t in document d to be the portion of $P(c|d)$ con-

sistent with the t's candidate locations:

$$P_{DL}(l|t,d) = \frac{P(c_l|d)}{\sum_{l' \in G(t)} P(c_{l'}|d)}$$

where $G(t)$ is the set of the locations for t in the gazetteer, and c_l is the cell containing l. TRIPDL (Toponym Resolution Informed by Predicted Document Locations) chooses the location that maximizes P_{DL}.

WISTR While TRIPDL uses an off-the-shelf document geolocator to capture the geographic gist of a document, WISTR (Wikipedia Indirectly Supervised Toponym Resolver) instead directly targets each toponym. It learns text classifiers based on local context window features trained on instances automatically extracted from GeoWiki.

To create the indirectly supervised training data for WISTR, the OpenNLP named entity recognizer detects toponyms in GeoWiki, and candidate locations for each toponym are retrieved from GeoNames. Each toponym with a location within 10km of the document location is considered a mention of that location. For example, the *Empire State Building* Wikipedia article has a human-provided location label of (40.75,-73.99). The toponym *New York* is mentioned several times in the article, and GeoNames lists a *New York* at (40.71,-74.01). These points are 4.8km apart, so each mention of *New York* in the document is considered a reference to New York City. Table 3.1 shows the top 50 most frequently extracted toponyms and their counts.

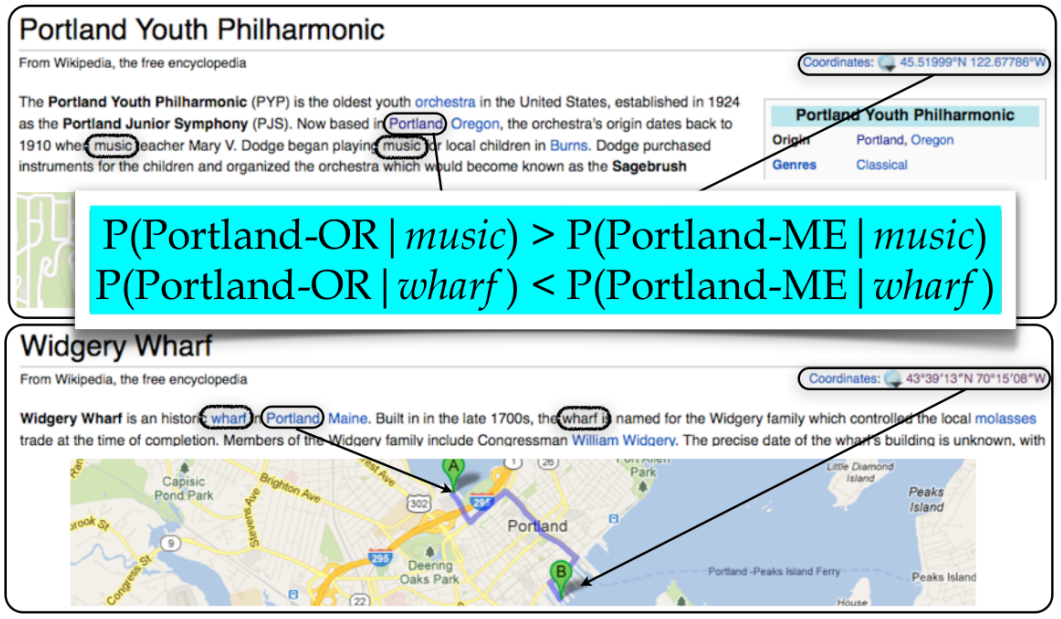


Figure 3.5: Example demonstrating WISTR's training instance extraction method. The *Portland Youth Philharmonic* Wikipedia article has a geotag very near Portland, Oregon, while that of *Widgery Wharf* has a geotag very near Portland, Maine. Words like *music* and *wharf* occurring in close proximity with *Portland* are features that help distinguish the two cities.

Next, context windows w of twenty words to each side of each toponym are extracted as features. The label for a training instance is the candidate location closest to the document location. I extract 1,489,428 such instances for toponyms relevant to the evaluation corpora. These instances are used to train logistic regression classifiers $P(l|t, w)$ for location l and toponym t. To disambiguate a new toponym, WISTR chooses the location that maximizes this probability.

Figure 3.5 demonstrates what WISTR's training instance extraction algorithm does with instances of the word *Portland* on the Wikipedia pages

for the *Portland Youth Philharmonic* and *Widgery Wharf*, two entities located in Portland, Oregon and Portland, Maine respectively. According to GeoNames, there are two locations named *Portland* near each of these articles' geotags. Thus, any mentions of *Portland* in either article get extracted as training instances, with each label assumed to be the nearby city. Context words like *music* and *wharf* are features that help distinguish the two Portlands from one another (and from other Portlands) in WISTR's logistic regression classifiers.

Few probabilistic toponym resolvers like WISTR exist in the literature. Li (2007) builds a probability distribution over locations for each toponym, but still relies on nearby toponyms that could refer to regions that contain that toponym and requires hand construction of distributions. Other learning approaches to toponym resolution (e.g. Smith and Mann (2003)) require explicit unambiguous mentions like *Portland, Maine* to construct training instances, while my data gathering methodology does not make such an assumption. Overell and Rüger (2008) and Overell (2009) only use nearby toponyms as features. Mani et al. (2010) and Qin et al. (2010) use other word types but only in a local context, and they require toponym-labeled training data. My approach makes use of all words in local and document context and requires no explicitly labeled toponym tokens.

The runtime of WISTR's evaluation phase is $O(||T_E||)$, where $||T_E||$ is the number of toponyms in the evaluation corpus once the classifiers have already been trained. The runtime of the training phase is $O(||T_{TR}||{\cdot}maxamb(T_{TR}))$,

where $||T_{TR}||$ is the number of toponyms in the training corpus and $maxamb(T_{TR})$ is the maximum ambiguity of any toponym in the training corpus.

LISTR I also introduce LISTR (LInk Supervised Toponym Resolver), which functions identically to WISTR, except that its training instances are extracted from toponyms in GeoWiki that are themselves links to geotagged Wikipedia articles, with the constraint that the coordinate of the destination page must be within 10 km of a candidate location of the toponym according to the gazetteer. This candidate is assumed to be the label for the training instance. If more than one candidate is within 10 km of the link destination's geotag, the closest such candidate is used. Redirects are processed before the final destination of the link is examined. I extract 703,122 training instances relevant to the evaluation corpora in this way.

I consider LISTR to be a more directly supervised version of WISTR, since it only receives training instances from explicit links in Wikipedia, which are essentially manually disambiguated toponyms. For this reason, I provide some results with LISTR but do not focus on it in this thesis. Besides this major difference from WISTR, the LISTR training instance extraction technique misses many of the instances extracted by WISTR, since often only a subset of the instances of a given toponym are made explicit links in a Wikipedia article (often only the first such occurrence). However, the potential for Wikipedia links to point to locations far from the page in which they are mentioned (or to appear in pages without geotags at all) means that LISTR

also extracts some instances that WISTR does not extract. Thus, taking the union of the training sets extracted by both techniques (WISTR+LISTR) is an interesting exercise in combining relatively direct supervision and relatively indirect supervision. I also experiment with combining the two training sets in a way that limits the LISTR component's contribution: only including training instances extracted from links for toponym types for which WISTR did not extract any training instances. This only allows LISTR to make a prediction over candidates for a toponym about which WISTR has nothing to say. Both WISTR and LISTR back off to TRIPDL's document-level distribution when attempting to resolve an unknown toponym, and this approach (which I call WISTR+LISTR$_{Back}$) normally uses only the WISTR training instances, backs off to the LISTR training instances if no WISTR instances exist, and finally backs off to TRIPDL's distribution if no training instances exist at all.

As with WISTR, the runtime of LISTR's evaluation phase is $O(||T_E||)$, where $||T_E||$ is the number of toponyms in the evaluation corpus once the classifiers have already been trained. The runtime of the training phase is $O(||T_{TR}|| \cdot maxamb(T_{TR}))$, where $||T_{TR}||$ is the number of toponyms in the training corpus and $maxamb(T_{TR})$ is the maximum ambiguity of any toponym in the training corpus.

TRAWL I bring TRIPDL, WISTR, and standard toponym resolution cues about administrative levels together with TRAWL (Toponym Resolu-

tion via Administrative levels and Wikipedia Locations). The general form of a probabilistic resolver that utilizes such information to select a location $\hat{l}$ for a toponym t in document d may be defined as

$$\hat{l} = \arg\max_l P(l, a_l|t, d).$$

where a_l is the administrative level (country, state, city) for l in the gazetteer. This captures the fact that countries (like Sudan) tend to be referred to more often than small cities (like Sudan, Texas). The above term is simplified as follows:

$$\begin{aligned} P(l, a_l|t, d) &= P(a_l|t, d)P(l|a_l, t, d) \\ &\approx P(a_l|t)P(l|t, d) \end{aligned}$$

where I approximate the administrative level prediction as independent of the document, and the location as independent of administrative level. The latter term is then expressed as a linear combination of the local context (WISTR) and the document context (TRIPDL):

$$P(l|t, d) = \lambda_t P(l|t, c_t) + (1-\lambda_t)P_{DL}(l|t, d).$$

λ_t, the weight of the local context distribution, is set according to the confidence that a prediction based on local context is correct:

$$\lambda_t = \frac{f(t)}{f(t)+C},$$

where $f(t)$ is the fraction of training instances of toponym t of all instances extracted from GeoWiki. C is set experimentally; C=.0001 was the optimal value for TRC-dev. Intuitively, the larger C is, the greater $f(t)$ must be for the local context to be trusted over the document context.

I define $P(a|t)$, the administrative level component, to be the fraction of representative points for a location $\hat{l}$ out of the number of representative points for all candidate locations $l \in t$,

$$\frac{||R_{\hat{l}}||}{\sum_{l' \in t} ||R_{l'}||}$$

where $||R_l||$ is the number of representative points of l. This boosts states and countries since higher probability is assigned to locations with more points (and cities have just one point).

Taken together, the above definitions yield the TRAWL resolver, which selects the optimal candidate location $\hat{l}$ according to

$$\hat{l} = \arg\max_l P(a_l|t)(\lambda_t P(l|t, c_t) + (1-\lambda_t)P_{DL}(l|t, d)).$$

Figure 3.6 gives a diagram of the components of the TRAWL resolver, the most complex resolver presented thus far. When a corpus with toponyms already identified is used, the need for the named entity recognizer component is eliminated.

The runtime of the evaluation phase of the TRAWL resolver is $O(||T|| \cdot maxamb(T))$, where $||T||$ is the number of toponyms in the corpus and $maxamb(T)$

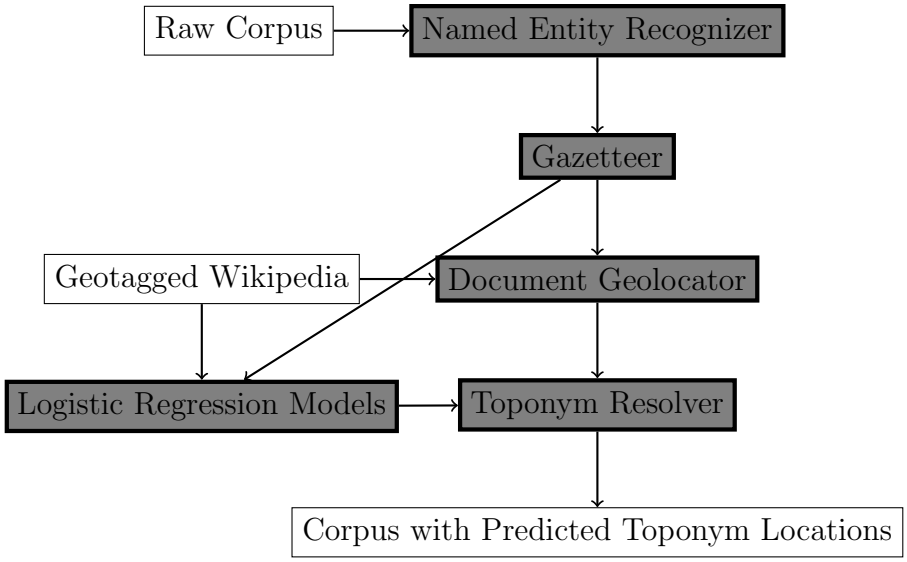


Figure 3.6: Components of the TRAWL resolver.

is the maximum ambiguity of any toponym, since it uses a classifier as in WISTR for each toponym and a similarity metric between each candidate location and the language model associated with its surrounding grid cell, assuming these language models have been trained ahead of time. The worst-case running time of the training phase is $O(||T_{TR}|| \cdot maxamb(T_{TR}))$, where $||T_{TR}||$ is the number of toponyms in the training corpus and $maxamb(T_{TR})$ is the maximum ambiguity of any toponym in the training corpus.

3.2 Combining Resolvers

SPIDER begins with uniform weights for each candidate location of each toponym. WISTR and TRAWL both output distributions over these locations based on outside knowledge sources, and can be used as more in-

formed initializations of SPIDER than the uniform ones. I call these combinations WISTR+SPIDER and TRAWL+SPIDER. I scale each toponym's distribution as output by WISTR or TRAWL by the number of candidate locations for that toponym, since the total weight for each toponym in SPIDER is the number of candidate locations, not 1.

3.3 Backoff

WISTR fails to predict when encountering a toponym it has not seen in the training data, and TRIPDL fails when a toponym only has locations in cells with no probability mass. TRAWL fails when both of these are true. In these cases, I select the candidate location geographically closest to the most likely cell according to TRIPDL's $P(c|d)$ distribution.

3.4 Document Size

For resolvers whose runtime is dependent on the size of documents in a greater-than-linear way, such as MinDist and SPIDER, and for resolvers which take into account the surrounding document when determining how to resolve a toponym, such as TRIPDL and TRAWL, it can often be beneficial to divide documents into smaller subdocuments.

MinDist and SPIDER both have a runtime proportional to the square of the number of toponyms per document and are thereby intractable for large documents such as the books found in CWar. Therefore, I divide each such

document into smaller subdocuments of at most twenty sentences. The predictions of these resolvers also depend on which toponyms are included within a document, so their results typically change when document size is altered. How large each document should be is a question that must be empirically answered based on both tractability and accuracy concerns.

Both TRIPDL and TRAWL (and WISTR, when using TRIPDL as a backoff), take into account the predicted location of each document when predicting locations for toponyms within that document. Thus, the results of these resolvers also depend on how documents are cut up (if at all). Just as with MinDist and SPIDER, keeping large documents such as entire books whole is likely suboptimal, but in this case large documents are only likely to degrade performance, not runtime. These resolvers are linear in the number of toponyms per document, so the size of documents does not affect runtime in a meaningful way.

London	30037	Poland	4213
United States	19539	Boston	3953
Australia	17018	Richmond	3791
Canada	9130	Dublin	3658
New York	8370	Moscow	3569
Washington	7935	Victoria	3567
Paris	7332	Florida	3493
Scotland	7255	Ireland	3405
England	7155	Berlin	3397
Toronto	7017	Pittsburgh	3338
Manchester	6653	Virginia	3177
Hong Kong	6396	India	3164
Chicago	6261	Norway	3136
Sydney	6105	France	3112
Tokyo	5545	Ottowa	3037
New Zealand	5265	United Kingdom	3015
Singapore	5034	Rome	2998
Melbourne	4877	Seattle	2900
Philadelphia	4875	Montreal	2899
San Francisco	4576	Los Angeles	2804
Jerusalem	4572	Japan	2754
Birmingham	4365	Pennsylvania	2748
New Jersey	4316	Miami	2625
Cambridge	4288	Calgary	2562
Portland	4213	San Diego	2490

Table 3.1: The 50 toponyms with the most training instances extracted from GeoWiki by WISTR.

Chapter 4

Evaluation

Some of the material in this chapter also appears in Speriosu and Baldridge (2013).

A consistent set of evaluation metrics is essential for the fair and quantitative comparison of different methods in a task such as toponym resolution.

Leidner (2008) uses precision and recall, two metrics commonly used in many natural language processing and information retrieval tasks, to evaluate toponym resolvers. One advantage of such an evaluation scheme versus just a single number such as accuracy or average error distance is that it captures both how many toponyms were resolved correctly of those that were attempted to be resolved as well as how many toponyms received no resolution or were not even identified by the named entity recognizer.

A major disadvantage of using precision and recall is that some decision must be made between what is a correct resolution and what is an incorrect resolution. Clearly, if the distance between a toponym's true location and the location a resolver assigns it is zero, this is a match. The use of different gazetteers in the annotated corpus and by the resolver itself can result in small discrepancies in the coordinates of locations that are really the same

location. If this distance is only tenth of a kilometer, one can likely still call the resolver's prediction a match. At greater distances, whether a match has been made becomes less clear. At some point a cutoff must be defined, where distances greater than the cutoff constitute incorrect predictions.

For example, the nation of Israel is represented by the point (31.76213, 34.79446) in GeoNames, (31.5, 34.75) in TR-CoNLL, and (31.783333, 35.216667) in GeoWiki. While appearing to be very near each other, these points form a triangle whose sides are of lengths 29.45 km, 54.26 km, and 39.98 km. Clearly, deciding on a cutoff distance under which two coordinates are considered the same location is impossible.[1]

An evaluation metric that avoids this problem involves measuring the distance between the correct and predicted location for each toponym and computing various statistics on this set of distances, such as the median and mean over all error distances as computed by e.g. Eisenstein et al. (2010, 2011), Wing and Baldridge (2011), and Roller et al. (2012). A resolver that attains a low mean but not particularly low median assigns few toponyms correct locations but makes few egregious errors. A resolver with a low median but high mean gets most toponyms reasonably correct but makes some errors with very large distances. In this thesis, I use both mean and median error as evaluation metrics.

[1]The fact that Israel is a country of over 22,000 square kilometers rather than a single point complicates matters further, but the use of multiple representative points and the distance metric defined in §2.1 largely assuage this.

I also compute a "best match" accuracy, where I measure the fraction of toponyms whose predicted location is as close as possible to the correct location, given the choices in the gazetteer. This accuracy measure avoids the problems of both gazetteer mismatch and choosing a cutoff threshold.

In addition to this accuracy metric, I also use the fraction of toponyms resolved to within 161 kilometers (about 100 miles) of correct, in order to give an idea of how many of the toponyms are within a large metropolitan area of correct. This allows for some comparability to work such as that of Cheng et al. (2010) and Roller et al. (2012) (though their task is document geolocation), who use the same metric.

Some toponym resolution methods may not be able to make a prediction for every toponym. For example, there may be insufficient training material for the WISTR resolver to choose a location for each toponym. In these cases, I have chosen to back off to a resolver with greater coverage (e.g. TRIPDL) so that all comparisons for a given dataset are on exactly the same set of toponyms. This eliminates the need for using precision and recall mentioned above. Using a named entity recognizer can also result in false positives and false negatives (which can be measured with precision and recall), but I am primarily giving results using gold toponym identifications since the main task of this work is toponym resolution rather than named entity recognition. I also give results on TR-CoNLL using a named entity recognizer in order to provide an idea of the performance of the system if it is given nothing but raw text.

4.1 Results

For convenience, Table 4.1 lists the names of all resolvers discussed so far along with brief descriptions of each.

Table 4.2 gives results of various resolvers on TRC-DEV. Mean error in kilometers, median error in kilometers, accuracy (best match), accuracy (percent within 161 km of correct), precision, recall, and F-score are the performance measures shown. The first four measures reflect performance when gold toponyms are used, while the last three reflect performance when NER is used. Table 4.3 shows the same for CWAR-DEV, with the exception that precision, recall, and F-score are omitted due to the absence of annotated states and countries in that corpus.

While the baselines RANDOM and POPULATION are given for comparison, I consider neither to be a true toponym resolver. POPULATION ignores the problem of toponym ambiguity altogether, effectively only knowing about one location for each toponym. RANDOM represents minimum performance: its accuracy should be taken as the minimum accuracy for any approach using a gazetteer (rather than 0%) and its error values the maximum error (rather than infinity or half the circumference of the earth, which is about 20,000 km).

For both datasets, MINDIST achieves errors and accuracies much better than RANDOM, validating the intuition behind the minimum distance resolvers that authors tend to discuss places near each other more often than not. SPIDER accounts for another jump in performance, validating the idea

that some locations are more prominent in a given corpus despite being farther away from other locations mentioned in the same document than they could be.

The question of which resolver performs best overall depends on the corpus and on the performance metric. POPULATION achieves the lowest mean error on TRC-DEV, but TRAWL+SPIDER is only two kilometers behind. All of the resolvers involving WISTR or TRAWL achieve higher accuracy on TRC-DEV than POPULATION, though POPULATION is barely in the lead in terms of accuracy within 161 km. WISTR (whether using LISTR as a backoff or not) achieves the highest performance when toponyms identified with NER are used, besting the POPULATION baseline by about 2% in both precision and recall. It is clear that while POPULATION does well in the average case on this dataset, similar or better levels of performance can be achieved in the absence of population data.

The performance of resolvers patterns differently on the CWAR-DEV corpus, as shown in Table 4.3. All of the resolvers with SPIDER as a component achieve very low mean errors. Even MINDIST used alone (an essentially unsupervised approach) performs better in terms of both mean error and accuracy than POPULATION. The overall geographic scope of CWAR, a collection of documents about the American Civil War, is much smaller than that of TR-CONLL, a collection of newspaper articles referring to international events. This makes the task of toponym resolution easier overall, particularly when measured in error distances, and especially for resolvers that

primarily seek tightly clustered sets of locations as solutions like MinDist and SPIDER. However, SPIDER clearly benefits from initialization with the output of WISTR and TRAWL. The WISTR+SPIDER approach achieves a best match accuracy of 87.2%, beating Population's accuracy by almost 30%. The CWar dataset stands as a clear case of where simple baselines like Population are poor resolvers. The modern population data provided by the Population resolver is largely inappropriate for the 19th century corpus. Where a corpus is situated in space and time is a crucial consideration when constructing a toponym resolver or choosing which one to apply.

Initializing SPIDER with TRAWL's output yields a mean error of only 94 km on CWar-dev. In general, TRAWL often achieves a lower mean error than WISTR alone even when its accuracy is not as strong. This is caused by its preference for multipoint locations over single-point locations, all else held equal. Multipoint locations, simply as a result of their larger size, are more likely to be closer to any given point than single point locations. Thus, TRAWL tends to yield low mean errors even when a greater percentage of its predictions are not the best match compared to WISTR.

The LISTR resolver performs relatively poorly on both corpora. Mixing its training instances with those of WISTR in either a mix or a backoff does not hurt performance much, though it does not tend to help either. Taking a closer look at the kind of training material extracted by each can help in the understanding of this phenomenon.

4.2 WISTR and LISTR features

In order to better understand WISTR's strong performance and LISTR's poor performance, it is useful to examine some of the most common unigram features extracted from GeoWiki by these methods. Table 4.4 shows the top 10 most frequent words, in order starting with the most common, found in the local context of the top 5 most commonly extracted locations named *Springfield*. Stop words are omitted. The second column indicates how many instances of the given Springfield were extracted.

It is easy to see that each of these five corresponds to a city named *Springfield* in the American states of Massachusetts, Missouri, Illinois, Ohio, and Oregon, since the surrounding state name is always either the first or second most commonly co-occurring word. These five Springfields are the exact five most populous Springfields worldwide. Many of the common features such as *city* are not good disambiguators between cities named *Springfield*, though they are likely to help disambiguate between cities named *Springfield* and non-city locations with the same name. Other words like *river*, *college*, and *medical* are likely to be good disambiguators between Springfields strongly associated with such concepts and those that are not. Finally, while the most commonly extracted Springfield accounts for almost half of all 1627 instances extracted for 10 different Springfields, the second and third most common Springfield occur almost equally often, as do the fourth and fifth. Such a distribution allows contextual evidence to sway the WISTR resolver rather than it almost always predicting the most likely Springfield a priori.

Table 4.5 presents the data for the top 5 Springfields extracted by LISTR. In contrast with Table 4.4, many Springfields with much smaller populations appear. For example, Springfield, Scotland has fewer than 1000 people, while Springfield, Oregon, the least populous city from Table 4.4, has a population of about 60,000. The distribution over Springfields is also much less uniform, with Springfield, Illinois accounting for about 84% of all instances extracted by LISTR. Springfield, Massachusetts, the most populous Springfield in the world and most commonly extracted Springfield by WISTR, does not even appear. Odd distributions such as this are common in the training material extracted by LISTR, and largely explain its poor performance.

The reasons for LISTR's frequent inability to extract quality training material involve the behavior of Wikipedia authors and editors. If an author recognizes that the string *Springfield* in some article's text is referring to a place, she might add it as a hyperlink to the Wikipedia article by that name, which is at `http://en.wikipedia.org/wiki/Springfield`. However, this is actually a disambiguation page, with links to many places, people, and other entities named *Springfield*. The page itself does not contain a geotag, as it does not correspond to any particular location. Thus, this link is not gathered by LISTR.

In order for a link to be picked up by LISTR, one of two things must occur: either the toponym must refer unambiguously (or nearly unambiguously) to a single location, or the destination of the link must be explicitly set to a particular location by that name, such as `http://en.wikipedia.org/wiki/`

`Springfield,_Illinois`. While there are Wikipedia guidelines regarding the correct way to format articles, including the formation of links, these guidelines change over time, some authors may be unaware of them or forget to use them, and a lack of sufficient time and effort maintaining Wikipedia content causes general inconsistency and incompleteness. Thus, while the manually constructed and maintained link structure of Wikipedia can be useful in supplementing toponym resolution training material (as seen by the performance of the WISTR+LISTR approaches), using it as the only source of training material yields poor results.

4.3 Visualization

Motivation

Visualization is a powerful tool for the exploration of many kinds of data, and data with geographic connections can be displayed visually in an extremely natural way. Once the explicit references to locations in a corpus have been automatically resolved, an effective visualization can show not only the geographic trends of that corpus but also allow the user to quickly refer back to the portions of the text relevant to a location or region for further analysis. As Hill (2006) states, "the ability to visualize the placement of items retrieved from a collection as they are distributed across a geographic landscape instantly conveys information about their relevance to the region of interest that is hidden otherwise." When a corpus contains errors of toponym resolution, visualization can serve as an aid in error analysis.

Google Earth

Static visualization Google Earth is a program used for viewing the earth at various levels of granularity including many annotations with geographic relevance.[2] Developers can generate files in the Keyhole Markup Language (KML) format, which can be loaded by Google Earth and used to display additional annotations and/or graphics.

One visualization I created relevant to toponym resolution that can be encoded into a KML file plots points for each location resolved to, with bars drawn above each such location whose height is proportional to the log of the number of times that location is referred to in a corpus (according to gold labels or a resolver's predictions). Pins are also displayed on the map which, upon being clicked, give the toponym mention and some surrounding context from the corpus (similar to the TextGIS visualization from Leidner (2008)). Figure 4.1 shows a screenshot of such a KML file loaded in Google Earth, zoomed in on the western areas of the United States and Mexico, and showing a mention of San Francisco, California from John Beadle's 1880 book *Western Wilds, and the Men who Redeem Them.* A similar visualization is used in Speriosu et al. (2010) to display region-topic models.

Dynamic visualization The style of visualization from above gives a broad, static overview of the corpus at hand. Other, more dynamic visualizations that

[2] `http://www.google.com/earth/index.html`

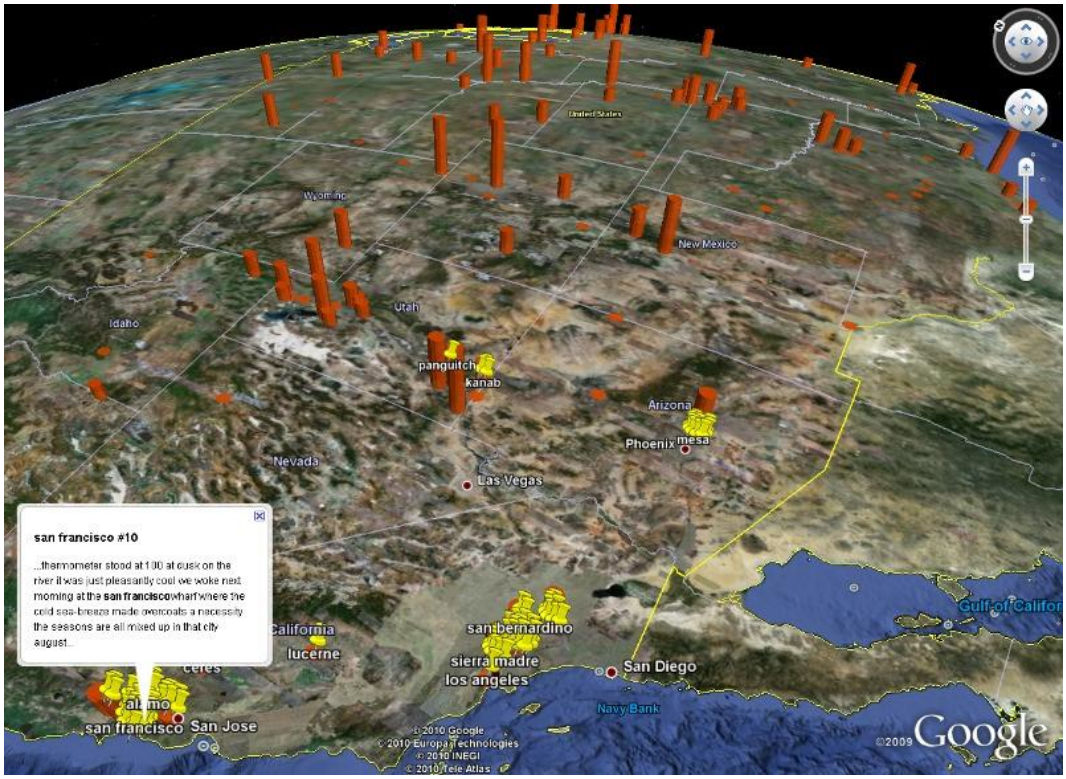


Figure 4.1: Example visualization in KML, showing predicted locations mentioned in John Beadle's 1880 book *Western Wilds, and the Men who Redeem Them*. The height of each three-dimensional bar is proportional to the log of the number of times that location is predicted. The user can click on pins that pop up dialog boxes containing the context of each toponym.

can convey the flow of location mentions in text are possible. When a user clicks a *play* button, Google Earth will jump from one location (or group of locations) to the next, giving a geographic tour of the text. A tour that shows the user which locations are mentioned as time passes, as opposed to as the text itself passes, requires a simultaneous spatial and temporal grounding and is beyond the scope of this thesis. Torget et al. (2012) present visualizations that allow the user to scan across both space and real time, but their corpus of dated newspaper articles from only Texas eliminates the need for both toponym resolution and temporal grounding.

The dynamic visualization shown in Figure 4.2 was created for the 1892 book *Abraham Lincoln: The True Story of a Great Life* from the CWAR corpus using the WISTR resolver. The visualization moves through the toponyms in the book in the order they appear. This can give the user a sense of which regions are more or less prominent in different parts of the book. When clicked, the pin associated with a location gives an excerpt containing the corresponding toponym, allowing the user to investigate further e.g. when an unexpected location arises. The user can choose how many toponyms to show at one time. For the purposes of demonstration in a static medium, Figure 4.2 shows all predicted locations for the first third (above) and last third (below) of the book. There is a clear trend moving from a discussion of locations throughout the eastern United States early in the book towards a discussion more concentrated only in the northeast towards the end of the book. Though there may be individual errors in toponym resolution, such a trend is unlikely

Figure 4.2: Example dynamic visualization in KML, showing a concentration of predicted locations in the northern United States towards the end of an 1892 book about Abraham Lincoln. The above picture covers locations mentioned in the first third of the book, while the bottom picture covers the last third.

to arise by chance, and the user can get a sense of the shift in geographic focus in the book. Finer grained trends are visible in the interactive version.

Processing and Unfolding

Tufte (2001) examines the characteristics that make a graphical display of quantitative information an effective tool. These include showing a large amount of data (more than can be expressed adequately in a table), a minimization of "ink" used for purposes other than showing the data itself, and an honest representation of the data that effectively conveys *what* is being shown while leaving the task of *how* to interpret the data up to the viewer. In some ways, the above visualizations created in Google Earth fail to meet some of these goals. There is a large amount of extra "ink" in Google Earth as it is designed to be a very general purpose geographic visualization tool with large amounts of data that might get in the way of a careful examination of a particular corpus, such as links to Wikipedia articles and geotagged images.[3] The three-dimensional bars in the static visualization (and the three-dimensional nature of Google Earth altogether) would likely be considered "chart junk" by Tufte, or fancy graphical features that hinder rather than aid in the comprehension of the data. The visualization I present in this section strives for simplicity, minimizing the amount of graphics needed to effectively convey

[3]In some cases, such information might be valuable towards a complete understanding of a corpus, but it should not be assumed ahead of time that overlaying vast quantities of many different kinds of data at once is the best default for the geographic viewing of one particular collection of data.

geographic information automatically extracted from corpora.

Processing is a Java library that can be used to create complex graphics with relatively little code.[4] The Unfolding library extends Processing with many functions used for downloading, drawing, and performing common operations on geographic maps.[5]

Figure 4.3 shows a visualization created with Processing and Unfolding that serves a similar purpose to that of Figure 4.1. Rather than draw bars with height, a simple number is placed on each circle representing mentions of a location. When the user clicks on such a circle, the text area to the right gives various information regarding those location mentions: the toponym (type) itself, each toponym token with surrounding context, and a document ID for each such context. Each checkbox on the left is associated with a document in the corpus being visualized, and the user can select any subset of the documents to view (including all or none with the convenience buttons in the upper left). The corpus shown is TRC-DEV resolved with the TRAWL resolver.

Figure 4.4 shows the Processing visualization of CWAR-DEV resolved with TRAWL running on a display made up of six high resolution (1080p) screens at the Texas Advanced Computing Center. Displays such as this and even larger allow vast amounts of data to be displayed simultaneously with a reduction in clutter and overlap as compared to viewing on a single screen, and

[4] `http://processing.org`
[5] `http://unfoldingmaps.org`

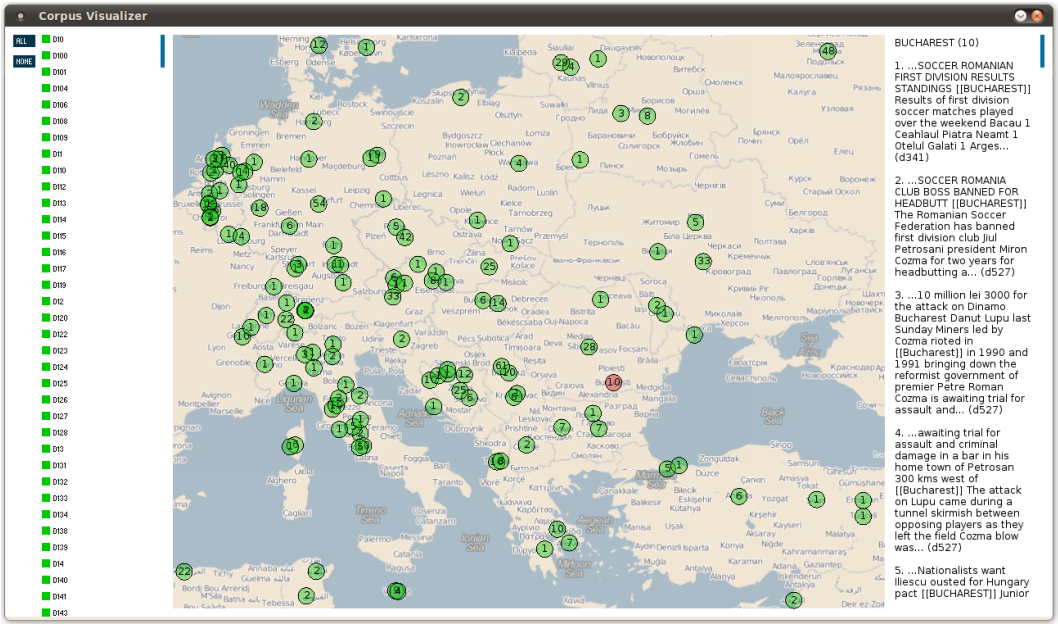


Figure 4.3: Example visualization created with Processing and Unfolding. The user can select all, none, or any subset of the loaded corpus using the checkboxes on the left of the interface. Each circle on the map represents a predicted location, and the number shown in each is the number of times that location is predicted in the active selection of the corpus. Upon clicking a circle, a list of contexts appears on the right, including document names for straightforward reference back to the written corpus.

Figure 4.4: The Processing visualization of the CWar-dev corpus resolved with TRAWL running on a display of six high resolution screens at the Texas Advanced Computing Center.

suggest the usefulness of geographic visualizations of corpora whose toponyms have been resolved by an automated system in the exploration of amounts of data greater than can be easily digested without such tools (here about 58 million words).

Figure 4.5 shows a variant of the Processing visualization where the circles representing each predicted location have a radius proportional to the log of the number of times that location is predicted in the corpus displayed. In

Figure 4.5: A variant of the Processing visualization where more frequently predicted locations are indicated by larger circles.

this case, WISTR's predictions on TRC-DEV over Asia are shown. Locations common in this international news corpus such as China, Beijing, Japan, India, and Pakistan are seen as relatively large circles. This adds some information to the visualization without the excessive "chart junk" of the Google Earth 3D bars visualization.

4.4 Error Analysis

As discussed above, the visualizations I present are useful for data exploration when given correctly resolved toponyms and for error analysis when given incorrectly resolved toponyms. This section gives some examples

of where visualizations can help identify predicted locations that are clearly incorrect as well as cases where different resolvers disagree in their predictions that warrant further investigation of particular parts of the text.

Figure 4.6 shows a type of error the MinDist and SPIDER resolvers typically make, that of resolving a toponym like *Japan* nearer to other candidate locations in a document than its correct location, where local context and prior world knowledge overwhelmingly indicate it should be. SPIDER has placed fully 28 instances of *Japan* from the TRC-dev corpus in Pennsylvania, where a small village by that name exists. Many of the instances are from sports articles, where results of matches between athletes representing various countries, many of them in Europe but a few in Asia and the Americas, are reported. Without any information other than a mapping from toponyms to locations from the gazetteer, SPIDER chooses the small American village because it is closer geographically than the Asian nation to Europe. TRAWL does not make this error, as it is able to take into account the prior probability of each Japan given the GeoWiki training data, the number of representative points of each Japan (one versus many), and at least in theory, contextual clues such as Japanese names if it has seen them near *Japan* in the training corpus.

Figure 4.7 shows quite clearly why SPIDER achieves a lower mean error on the CWar-dev corpus than TRAWL, which has a better accuracy. The CWar corpus primarily deals with the eastern coast of the United States, so nearly all of the toponyms contained in it have at least one can-

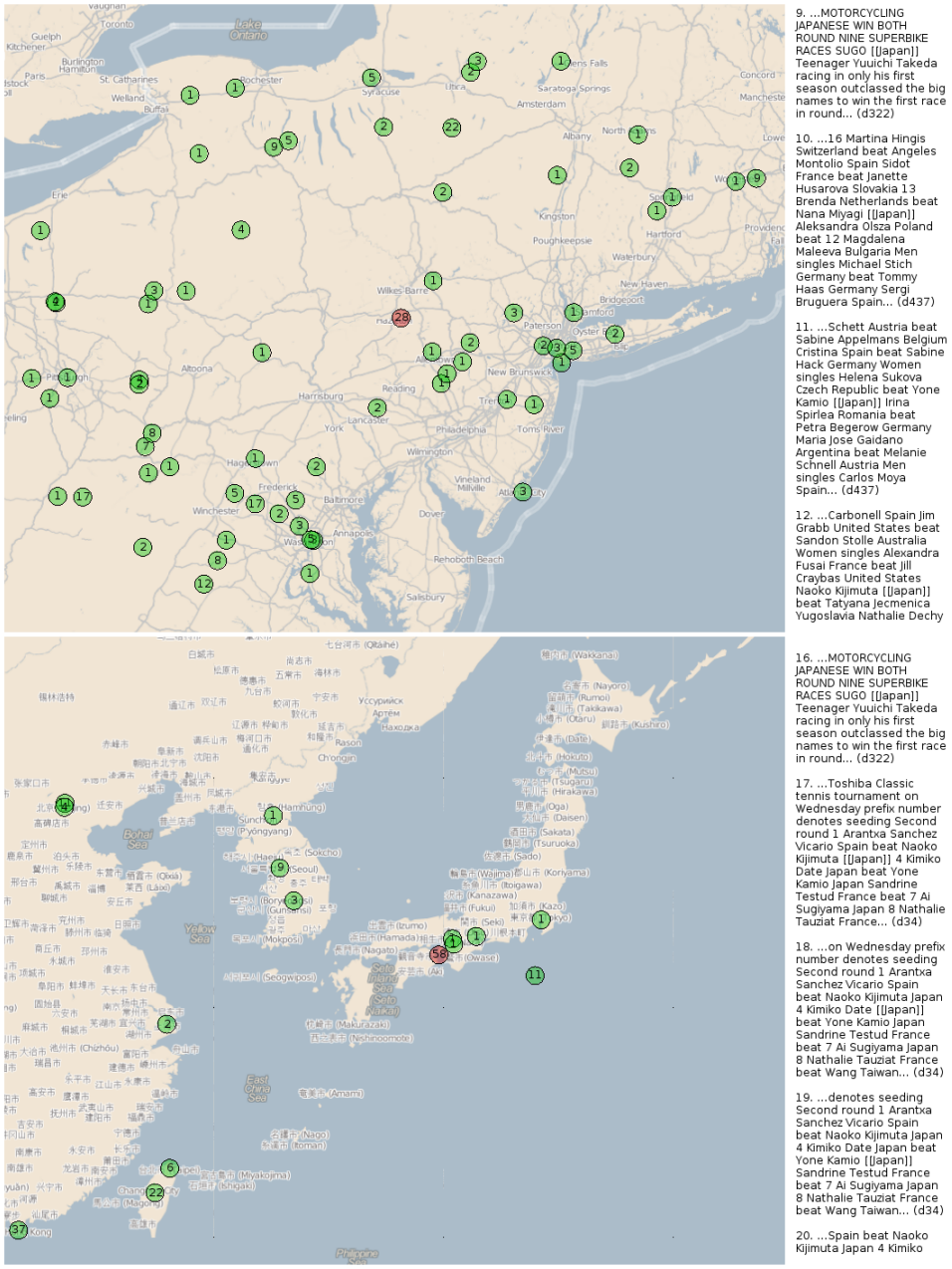


Figure 4.6: Example showing SPIDER's placement of *Japan* in Pennsylvania 28 times (above) in the TRC-DEV corpus due to its frequent mentioning alongside European countries in sports articles, most of which are closer to North America than the easternmost parts of Asia, where TRAWL correctly places *Japan* (below).

didate location in that region. Even for toponyms like *England* that likely refer to Old World locations, locations in the Americas are often named after such places. As Crane (2004) puts it, "Americans used the same placenames over and over again." The minimum distance algorithms take the opportunity to assume that almost every mentioned location must be in the eastern US, resulting in the dense clump of predicted locations there with almost no predicted locations anywhere else. Thus, any error SPIDER makes is unlikely to be more than the size of this clump at its widest, resulting in a low mean error. However, SPIDER is not very discriminative when it comes to choosing a particular York or Philadelphia or Springfield in this area, relying on the distribution of candidate locations of other toponyms near these toponyms in documents rather than any prior knowledge or contextual clues. On the other hand, TRAWL does take advantage of such information and correctly predicts locations on a fine-grained level more often than SPIDER. As a tradeoff, TRAWL occasionally makes wildly incorrect predictions due to idiosyncrasies and gaps in its training data, e.g. it erroneously places a few tokens of *Philadelphia* and *Sacramento* in the CWAR-DEV corpus in South America. Errors like these increase the mean error distance much more than errors within the United States.

Figure 4.8 demonstrates a disagreement between SPIDER and TRAWL on where to place some tokens of *Washington* in the TRC-DEV corpus, with SPIDER tending to disambiguate more to Washington state than TRAWL, which more often resolves to Washington, D.C. Such a difference might prompt

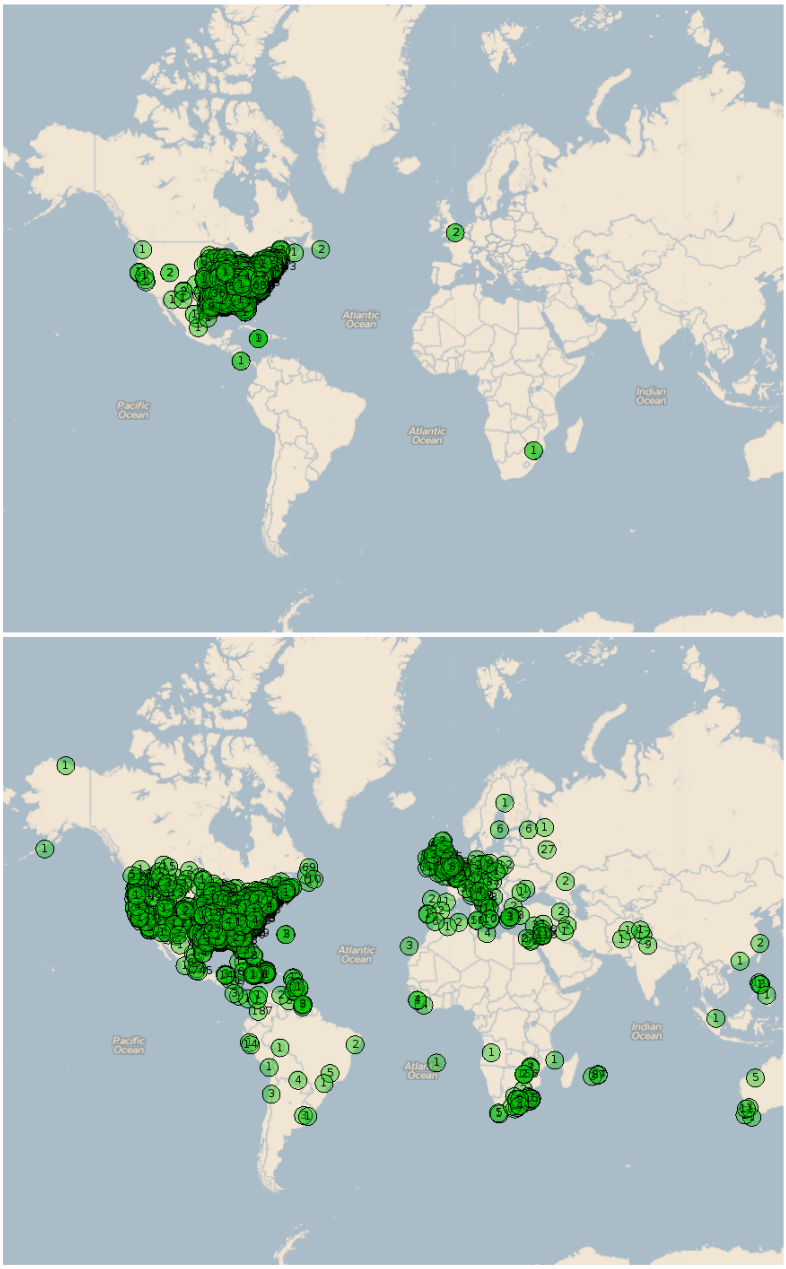


Figure 4.7: Example demonstrating the tendency of SPIDER to clump all predicted locations in the same region (above), in this case for the CWAR-DEV corpus. The version resolved with TRAWL (below) has a much more uniform distribution over the globe, despite the corpus' inherent focus in the eastern United States.

one to go back to the source and decide which is correct. However, for at least a few of the mentions of *Washington*, no decisive correct answer is obvious even given the entire document:

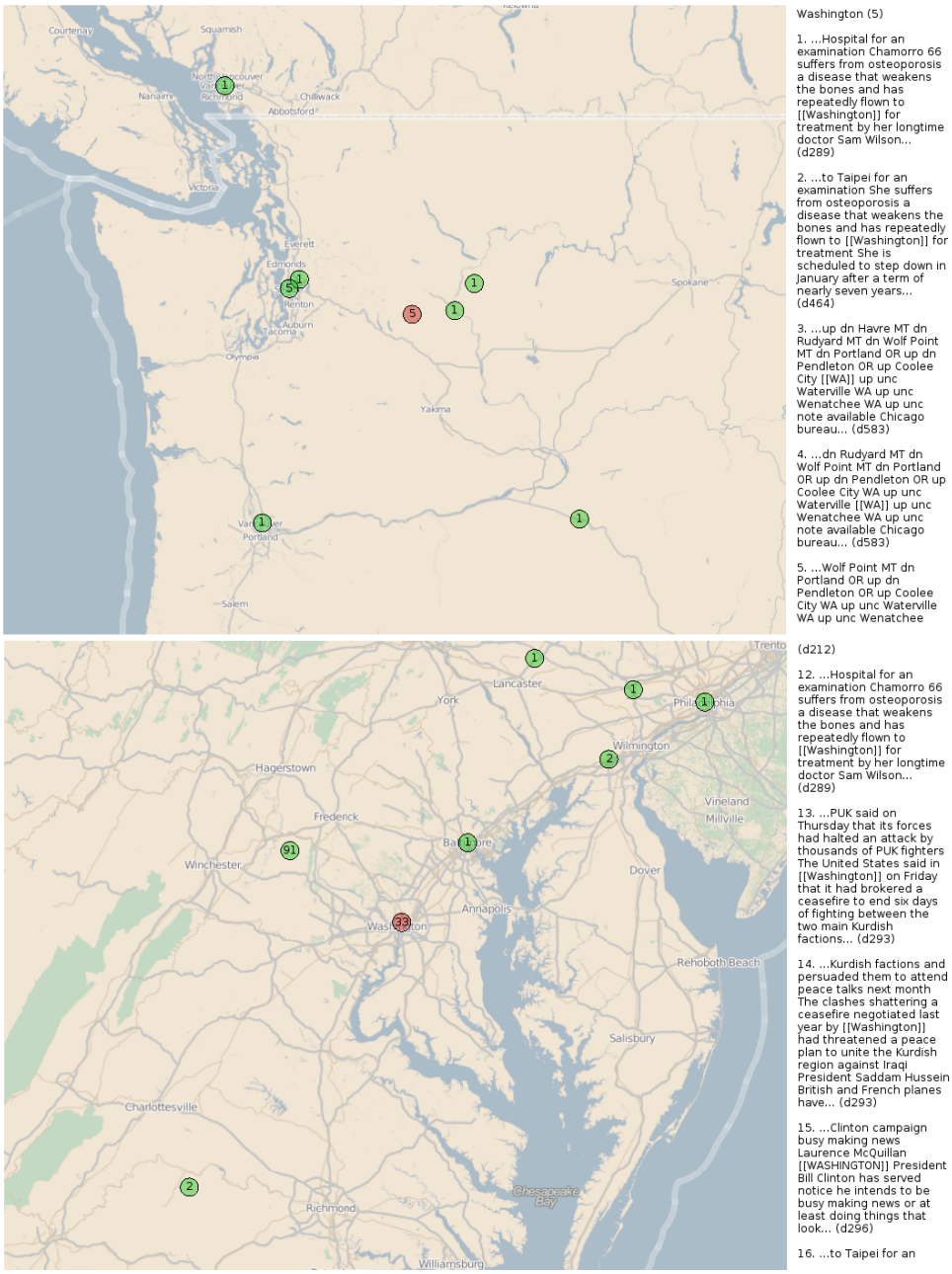
> Nicaraguan President to go to **U.S.** for Medical Care
> **MANAGUA**, **Nicaragua** 1996-08-23
> Nicaraguan president Violeta Chamorro was due to fly to the **United States** on Saturday for a medical check-up to determine if surgery was needed on the lower part of her spinal column, the government said on Friday.
> Chamorro has complained of lower back pain since her trip to **Taiwan** in May, when the pain forced her to go to Taipei University Hospital for an examination.
> Chamorro, 66, suffers from osteoporosis, a disease that weakens the bones, and has repeatedly flown to **Washington** for treatment by her longtime doctor, Sam Wilson.

The TRC-DEV annotation for this token of *Washington* is Washington, D.C. Given the international nature of the article, this choice of a national capital makes perhaps more sense than the U.S. state. World knowledge might indicate that referring to a flight to a state is less likely than the more specific reference to a city. Investigating the source of the article might further shift one's guess towards one Washington or another, e.g. if it is printed in the *Washington Post*, a newspaper from the capital. However, none of these potential clues is particularly well modeled by either SPIDER or TRAWL, or most other resolvers in the literature. As with many problems in natural language processing, there are some cases that seem to require a combined

understanding of language and world knowledge that not even every human possesses.

Table 4.6 shows the ten toponyms that caused the greatest total error distances from TRC-DEV with gold toponyms when resolved by TRAWL. *Washington*, the toponym contributing the most total error, is a typical example of a toponym that is difficult to resolve, as there are two very prominent locations within the United States with the name. Choosing one when the other is correct results in an error of over 4000 km. This occurs, for example, when TRAWL chooses Washington state in the phrase *Israel's ambassador to* **Washington**, where more knowledge about the status of Washington, D.C. as the political center of the United States (e.g. in the form of more or better contextual training instances) could overturn the administrative level component's preference for states.

An instance of *California* in a baseball-related news article is incorrectly predicted to be the town California, Pennsylvania. The context is: *...New York starter Jimmy Key left the game in the first inning after Seattle shortstop Alex Rodriguez lined a shot off his left elbow. The Yankees have lost 12 of their last 19 games and their lead in the AL East over Baltimore fell to five games. At* **California***, Tim Wakefield pitched a six-hitter for his third complete game of the season and Mo Vaughn and Troy O'Leary hit solo home runs in the second inning as the surging Boston Red Sox won their third straight 4-1 over the California Angels. Boston has won seven of eight and is 20-6...* The presence of many east coast cues—both toponym and otherwise—makes it



Washington (5)

1. ...Hospital for an examination Chamorro 66 suffers from osteoporosis a disease that weakens the bones and has repeatedly flown to [[Washington]] for treatment by her longtime doctor Sam Wilson... (d289)

2. ...to Taipei for an examination She suffers from osteoporosis a disease that weakens the bones and has repeatedly flown to [[Washington]] for treatment She is scheduled to step down in January after a term of nearly seven years... (d464)

3. ...up dn Havre MT dn Rudyard MT dn Wolf Point MT dn Portland OR up dn Pendleton OR up Coolee City [[WA]] up unc Waterville WA up unc Wenatchee WA up unc note available Chicago bureau... (d583)

4. ...dn Rudyard MT dn Wolf Point MT dn Portland OR up dn Pendleton OR up Coolee City WA up unc Waterville [[WA]] up unc Wenatchee WA up unc note available Chicago bureau... (d583)

5. ...Wolf Point MT dn Portland OR up dn Pendleton OR up Coolee City WA up unc Waterville WA up unc Wenatchee

(d212)

12. ...Hospital for an examination Chamorro 66 suffers from osteoporosis a disease that weakens the bones and has repeatedly flown to [[Washington]] for treatment by her longtime doctor Sam Wilson... (d289)

13. ...PUK said on Thursday that its forces had halted an attack by thousands of PUK fighters The United States said in [[Washington]] on Friday that it had brokered a ceasefire to end six days of fighting between the two main Kurdish factions... (d293)

14. ...Kurdish factions and persuaded them to attend peace talks next month The clashes shattering a ceasefire negotiated last year by [[Washington]] had threatened a peace plan to unite the Kurdish region against Iraqi President Saddam Hussein British and French planes have... (d293)

15. ...Clinton campaign busy making news Laurence McQuillan [[WASHINGTON]] President Bill Clinton has served notice he intends to be busy making news or at least doing things that look... (d296)

16. ...to Taipei for an

Figure 4.8: Example showing a disagreement between SPIDER (above) and TRAWL (below) as to the placement of some tokens of *Washington* in TRC-DEV. In some cases, such as the topmost excerpt, the full document context is not enough for even a human to make a decisive judgment.

unsurprising that the resolver would predict California, Pennsylvania despite the administrative level component's heavier weighting of the state.

The average errors for the toponyms *Australia* and *Russia* are fairly small and stem from differences in how countries are represented across different gazetteers, not true incorrect predictions.

Table 4.7 shows the toponyms with the greatest errors from CWar-dev with gold toponyms when resolved by WISTR+SPIDER. *Rome* is sometimes predicted as cities in Italy and other parts of Europe rather than Rome, Georgia, though it correctly selects the city in Georgia more often than not due to SPIDER's preference for tightly clumped sets of locations. *Mexico*, however, frequently gets incorrectly selected as a city in Maryland near many other locations in the corpus when TRAWL's administrative level component is not present. Many other of the toponyms contributing to the total error such as *Jackson* and *Lexington* are simply the result of many American towns sharing the same names and a lack of clear disambiguating context.

4.5 The Need for a More Flexible Approach

WISTR presents a strong way to resolve toponyms given only their immediate textual context. It does not consider more than one toponym token at a time, and it has no mechanism for preventing the prediction of locations very far from each other in a single corpus or document. WISTR also is likely to choose the most prominent candidate for a given toponym (e.g. the United States capital city for *Washington*), as that is usually the candidate most often

mentioned in Wikipedia. Though evidence from local context can change this prediction, WISTR leans toward the most prominent candidate in ambiguous cases. Thus, WISTR excels at the TR-CoNLL corpus of international news, where the most commonly discussed locations are very prominent ones that may be very far apart geographically.

SPIDER behaves in almost precisely the opposite way: it completely ignores textual context, and its predictions depend very much on the set of candidate locations in a document. It seeks to squish the set of predicted locations into a small geographic space. Thus, it does relatively poorly on TR-CoNLL and very well on CWar, which mostly spans a small geographic area and may refer to locations that are not currently prominent or never were.

An ideal toponym resolution system would be flexible enough to perform well on either of these extremes. Initializing SPIDER with the distributions output by WISTR or TRAWL is a step in this direction and proves fruitful in some cases investigated so far, but is more like the successive application of two extreme approaches than a truly flexible approach. The next chapter will explore such an approach, framing toponym resolution as the traveling purchaser problem. This approach makes the two factors of individual token accuracy and collective geographic span explicit, without a priori emphasizing one over the other for a given corpus or even document. I explore two main ways of solving this transformation of the toponym resolution problem, one of which is a genetic algorithm that allows its solutions to evolve over time, adapting to the properties of a particular text or set of texts.

Name	Description
ORACLE	Makes the best possible prediction given the system gazetteer
RANDOM	Selects a candidate location uniformly at random
POPULATION	Selects the candidate with the greatest population
MINDIST	Jointly resolves toponyms in a document based strictly on spatial minimality
SPIDER_{10}	Performs MINDIST for 10 iterations, reweighting locations based on frequency of selection
TRIPDL	Selects the candidate that is maximally likely given the predicted document location
WISTR	Selects the candidate that is maximally likely given its local context, using training data based on Wikipedia article geotags
WISTR+SPIDER_{10}	Initializes SPIDER_{10} with WISTR's output rather than uniformly
TRAWL	Selects the candidate that is maximally likely given its administrative level, local context, and the predicted document location
TRAWL+SPIDER_{10}	Initializes SPIDER_{10} with TRAWL's output rather than uniformly
LISTR	Selects the candidate that is maximally likely given its local context, using training data based on Wikipedia links
WISTR+LISTR	Selects the candidate that is maximally likely given its local context, using training data based on both Wikipedia article geotags and links
$\text{WISTR+LISTR}_{Back.}$	Selects the candidate that is maximally likely given its local context, using training data based on Wikipedia article geotags as well as links when the geotag approach found no instances of a toponym

Table 4.1: Names and descriptions of resolvers discussed so far.

Resolver	Mean	Med.	A	A_{161}	P	R	F
ORACLE	58	28.1	100.0	94.5	85.1	64.2	73.2
RANDOM	3827	1536.5	34.9	39.2	25.5	19.2	21.9
POPULATION	**153**	**30.7**	83.0	**91.3**	74.1	55.9	63.7
MINDIST	1750	40.5	62.6	70.3	49.7	37.5	42.8
$SPIDER_{10}$	1548	37.5	64.3	73.0	52.4	39.5	45.1
TRIPDL	1351	39.1	62.5	72.7	52.8	39.8	45.4
WISTR	215	**30.7**	84.2	90.6	**76.4**	**57.6**	**65.7**
WISTR+$SPIDER_{10}$	237	**30.7**	84.2	90.4	76.1	57.4	65.4
TRAWL	177	**30.7**	83.9	91.0	75.7	57.1	65.1
TRAWL+$SPIDER_{10}$	155	**30.7**	84.1	91.2	75.5	57.0	65.0
LISTR	2049	59.0	60.0	63.7	51.8	39.1	44.6
WISTR+LISTR	217	**30.7**	**84.3**	90.4	76.2	57.4	65.5
WISTR+$LISTR_{Back.}$	207	**30.7**	84.2	90.6	**76.4**	**57.6**	**65.7**

Table 4.2: Results on TRC-DEV for various resolvers. Mean error in kilometers, median error in kilometers, accuracy, and accuracy within 161 km of correct are given for experiments with gold toponyms. Precision, recall, and F-score reflect experiments using NER.

Resolver	Mean	Med.	A	A_{161}
ORACLE	0	0.0	100.0	100.0
RANDOM	2283	967.6	13.1	14.6
POPULATION	1833	**0.0**	59.9	61.1
MINDIST	237	**0.0**	64.9	71.0
$SPIDER_{10}$	200	**0.0**	65.7	72.0
TRIPDL	787	**0.0**	53.6	61.0
WISTR	968	**0.0**	67.4	70.5
WISTR+$SPIDER_{10}$	141	**0.0**	**87.2**	87.9
TRAWL	918	**0.0**	69.4	71.4
TRAWL+$SPIDER_{10}$	**94**	**0.0**	82.5	**90.0**
LISTR	1679	608.6	30.5	32.0
WISTR+LISTR	1030	**0.0**	50.4	54.3
WISTR+$LISTR_{Back.}$	969	**0.0**	67.3	70.4

Table 4.3: Results on CWAR-DEV. Precision, recall, and F-score are omitted due to the absence of annotated states and countries in the CWAR corpus.

Top 10 most frequent words co-occurring with *Springfield*	N
massachusetts, city, connecticut, river, center, college, neighborhood, street, park, basketball	765
school, missouri, elementary, university, city, center, catholic, high, middle, state	281
illinois, city, state, retrieved, 2007, february, lincoln, school, university, high	267
ohio, school, high, clark, located, clark, city, county, public, schools	92
oregon, eugene, city, school, high, area, willamette, united, community, medical	75
⋮	⋮
Total for all 10 Springfields found	1627

Table 4.4: Top 10 most frequent words co-occurring with the top 5 most commonly extracted Springfields from GEOWIKI using WISTR.

Top 10 most frequent words co-occurring with *Springfield*	N
illinois, state, lincoln, united, states, abraham, city, born, chicago, capital	1856
oregon, eugene, area, school, high, lane, county, united, station, states	291
essex, st, england, chelmsford, village, village, massachusetts, church, pynchon, great	27
fife, scotland, stratheden, parish, cochrane, hospital, cupar, church, village, united, minister	18
south, australia, adelaide, hill, park, mitcham, suburbs, hayward, edward, college, city	17
⋮	⋮
Total for all 7 Springfields found	2215

Table 4.5: Top 10 most frequent words co-occurring with the top 5 most commonly extracted Springfields from GeoWiki using LISTR.

Toponym	N	Mean	Total
Washington	25	3229	80717
Gaza	12	5936	71234
California	8	5475	43797
Montana	3	11635	34905
WA	3	11221	33662
NZ	2	14068	28136
Australia	88	280	24600
Russia	72	260	18712
OR	2	9242	18484
Sydney	12	1422	17067

Table 4.6: Toponyms with the greatest total error distances in kilometers from TRC-dev with gold toponyms resolved by TRAWL. N is the number of instances, and the mean error for each toponym type is also given.

Toponym	N	Mean	Total
Mexico	1398	2963	4142102
Jackson	2485	1210	3007541
Monterey	353	2392	844221
Haymarket	41	15663	642170
McMinnville	145	3307	479446
Alexandria	1434	314	450863
Eastport	184	2109	388000
Lexington	796	442	351684
Winton	21	15881	333499
Clinton	170	1401	238241

Table 4.7: Toponyms with the greatest total error distances in kilometers from CWar-dev with gold toponyms resolved by TRAWL+SPIDER. N is the number of instances, and the mean error for each toponym type is also given.

Chapter 5

Toponym Resolution as the Traveling Purchaser Problem

5.1 The Traveling Purchaser Problem

The traveling salesman problem (TSP) is a classic computational problem where a traveling salesman must visit a set of cities in such an order that minimizes the total cost of the trip. The problem is frequently given as finding the path through a graph such that all nodes are visited and the total cost of the path is minimized given edge weights. TSP is perhaps the most famous example of an NP-complete problem, and no algorithm is known for finding the optimal solution in the general case that runs in polynomial time.

The traveling purchaser problem (TPP) is a generalization of TSP, and involves a purchaser with a list of goods to purchase, knowledge of different markets that sell various goods at different prices, and knowledge of travel costs between any pair of markets. The purchaser must minimize the total cost of purchasing all necessary goods and traveling among markets. A variety of problems have been modeled and solved as instances of TPP, including the scheduling of multiple jobs on a machine (Ong, 1982), the purchasing of materials for manufacturing (Pearn and Chien, 1998), and the design of

computer networks (Ravi and Salman, 1999).

Formally, let $G = (V, E)$ be a complete directed graph with markets $V = \{v_o, ..., v_m\}$ and edges $E = \{(v_0, v_1), ..., (v_m, v_{m-1})\}$. Let $P = \{p_1, ..., p_n\}$ be a set of goods. Each good $p_k \in P$ can be purchased from a set of markets $V_k \subset V$. Let the travel cost from v_i to v_j be c_{ij}, where $(v_i, v_j) \in E$. Purchase costs are defined as s_{ki} where $p_k \in P$ and $v_i \in V_k$. A visit to a market v_i always involves the purchase of at least one good p_k.[1] A tour is an ordered path through the markets $V_t \in V$ that includes purchases of all $p_k \in P$. A correct solution of TPP minimizes the objective function

$$f_{TPP} = \sum_{v_i, v_j \in V_t}(c_{ij}) + \sum_{p_k \in P, v_i \in V_t}(s_{ki}).$$

TPP reduces to TSP when each good is available at only one market, and is thus also NP-complete. There is a large body of research that has proposed various methods, both exact and approximate, for solving TSP and TPP (Ramesh, 1981; Golden et al., 1981; Ong, 1982; Voß, 1996; Singh and van Oudheusden, 1997; Pearn and Chien, 1998; Ravi and Salman, 1999; Boctor et al., 2003; Laporte et al., 2003; Teeninga and Volgenant, 2004; Riera-Ledesma and Salazar-González, 2005; Bontoux and Feillet, 2008).

[1] A fruitless visit to a market should always be avoided assuming a complete graph and the usual triangle inequality of Euclidean space.

5.2 Framing the Problem

I provide a way to frame toponym resolution in terms of TPP, with each document being a separate instance of TPP. If this proves successful, then any existing or future solutions to TPP immediately become toponym resolvers. The mapping I use can be summarized as follows:

- A good that must be purchased is a toponym that must be resolved.
- A market selling goods is an area of the earth, here one of the following two options:
  - A uniform grid cell of a specified number of degrees in width (longitude) and height (latitude). Other types of cells, such as the leaves of a k-d tree (Roller et al., 2012) are also possible.
  - A cluster centroid, if one clusters all of the candidate locations for all of the toponyms in a document. I use agglomerative clustering with a specified maximum agglomeration distance.
- Each market sells candidate locations of toponyms.
  - In the grid formulation, each cell is a market that sells the locations in that cell.
  - In the cluster formulation, each centroid sells the locations contained in its cluster.

- The act of purchasing a good from a market is the act of choosing a location to disambiguate a toponym to.
- The cost of traveling from one market to another is based on one of these two options:
  - The physical distance from a market's center/centroid to another's, transformed to fall between 0 and 1.
  - A probability of traveling from one market to another based on the fraction of links from Wikipedia articles for locations in the first article to those in the second. Other knowledge sources, such as human travel data (Kalogerakis et al., 2009), are also possible.
- The cost of purchasing a good from a market (selecting a location for a toponym) may take at least two forms:
  - In the absence of any information other than the corpus and a gazetteer, the cost is simply the distance from the candidate location to the center or centroid of the grid cell or the centroid of the cluster.
  - If other knowledge is available in the form of a distribution $P(l|t,d)$ over candidate locations for a toponym t in document d, a reasonable value for each candidate's cost is $1 - P(l|t,d)$. In experiments presented here, I use the distributions from WISTR.

Solving this form of TPP reflects the intuition of the MinDist and SPIDER resolvers that authors tend to refer to locations near each other on the globe with toponyms near each other in text (in the same document). If a document mentions both *Austin* and *Texas* and there is only one cell or cluster containing both a city called *Austin* and a point belonging to a *Texas* (one of the representative points of the state of Texas), the solution will tend to choose both of these, as would MinDist and SPIDER. On the other hand, if a document mentions both *New York City* and *Paris*, while the distance between New York City and Paris, Texas is less than the distance between New York City and Paris, France, the cost of purchasing the toponym *Paris* from a cell or cluster somewhere in Texas may be much higher than the price of purchasing it somewhere in France, given enough clues from general and contextual knowledge. Thus, it will sometimes be worth the trip to travel further and purchase a toponym at a lower price, reflecting the intuition that authors may discuss places far away from each other when prior prominence of locations or sufficient contextual clues warrant this.

There are several major potential advantages to solving toponym resolution as TPP versus SPIDER. First, some of the known solutions to TPP, many of which have been worked on for considerable time by many researchers, may do a better job of capturing the intuitions behind SPIDER than SPIDER itself. Solutions to TPP seek to minimize an objective function, providing for a clear problem statement. Second, the TPP formulation allows arbitrary amounts of outside knowledge to be included in a modular

way as costs; here, I explore the use of Wikipedia link structure for travel costs and the use of WISTR's output distributions for purchase costs. This allows a resolver based on TPP to draw advantages both from approaches like SPIDER that try to push all of the resolutions of toponyms in a document together jointly (and achieve a low mean error but relatively poor accuracy) and from approaches like the TRAWL resolver that incorporate several sources of knowledge to effectively resolve individual toponyms. Finally, the tours output as solutions to TPP could be used to create a visualization that seeks to visit the various regions a text refers to in a natural way, rather than necessarily visit each predicted location in the strict order they appear in the text.

The ordering of markets in a tour output after solving a TPP instance is not strictly relevant to the resolution of toponyms. Predicted locations correspond to purchased goods, and the order in which markets are visited does not affect these predictions once the set of markets visited and the set of goods purchased at each market are fixed. (Indeed, even the grouping of locations into markets is no longer strictly relevant once all purchases have been made.) However, in practice the ordering of markets typically strongly affects which goods are purchased in the first place. Most approximate algorithms for solving TPP work iteratively, moving from market to market based on both travel costs and the change in total purchase cost associated with visiting a new market. Handling markets one by one in this way greatly simplifies the complexity of the problem. Thus, while not strictly required by the mapping between toponym resolution and TPP, computing an order of the markets

visited is a necessary byproduct of solving TPP instances efficiently.

Market Formulation

An essential component of a TPP instance is the set of markets and each market's set of goods for sale. I experiment with both a grid market formulation and a cluster market formulation. In both cases, the set of all candidate locations in a document is the set of points which must be partitioned into markets, ideally with no two candidate locations of the same toponym in the same market.[2] In the **grid market formulation**, the surface of the earth is simply partitioned into N° by N° square[3] cells, and the candidate locations falling into each cell are the locations of the market corresponding to that cell. The grid market formulation requires that the grid size be chosen up front; it can be tuned according to a performance measure. The center of such a cell can simply be defined as its geometric center or as the centroid of all the candidate locations in it; I use the latter definition here.

In the **cluster market formulation**, the set of candidate locations for all toponyms in a document are clustered. Simple k-means clustering is one means of achieving this, but requires that the number of clusters desired be chosen ahead of time. I choose to use agglomerative clustering instead, where each point begins as a singleton cluster, and clusters are iteratively merged

[2]If this situation does occur, e.g. two or more Springfields are sold from the same market, a sensible solution is to always prefer the Springfield with the cheapest purchase cost.

[3]Due to the curvature of the earth, the grid cells are not quite square. Near the poles, they are much taller than they are wide. Methods of partitioning the earth's surface into regions of equal area are possible.

as long as the distance between their centroids is within some threshold. The set of clusters is final once no two centroids are close enough to agglomerate. The distance threshold must be specified, and can be tuned according to a performance measure.

Travel Cost Functions

An instance of TPP must include a travel function that defines the cost from a source market to a destination market. One way of specifying this function involves imposing a bivariate Gaussian probability density function (PDF) on the earth's surface[4] with a mean at the source market's center or centroid and a given covariance matrix. In this work, I use a symmetric covariance matrix that corresponds with a single standard deviation σ_t, uniform across longitude and latitude, for which I experiment with a few values. The destination market's center or centroid has a probability density according to this distribution. I subtract this density from the maximum density h_{v_i} of this distribution, then divide by h_{v_i} to yield a value between 0 and 1, to obtain the **Gaussian travel cost function** (GTCF):

$$\text{GTCF}(v_i, v_j) = (h_{v_i} - g_{v_i}(v_{j_{lat}}, v_{j_{lon}}, \sigma_t))/h_{v_i}$$

where $v_{j_{lat}}$ and $v_{j_{lon}}$ are the coordinates of the center of v_j and h_{v_i} is the value of g_{v_i} at its center (its maximum value):

$$h_{v_i} = g_{v_i}(v_{i_{lat}}, v_{i_{lon}}, \sigma_t)$$

[4]I assume the earth is flat in order to greatly simplify the mathematics involved.

$g_\mu(x, y, \sigma)$ is the bivariate Gaussian PDF with a mean μ and symmetric standard deviation σ:

$$g_\mu(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\left[\frac{(x-\mu_x)^2+(y-\mu_y)^2}{2\sigma^2}\right]\right)$$

Figure 5.1 gives an example of a TPP instance with markets $v_1 ... v_4$, considering v_1 as the source market. The circle around v_1 represents the standard deviation σ. v_2 is at a distance of $\sigma/2$ (half a standard deviation), v_3 is at a distance of σ, and v_4 is at 2σ. The red points represent the centroids of each market. The GTCF costs to travel from v_1 to each of the other markets are .12, .39, and .86. In this work I set σ to 1610 km (about 1000 mi), which preliminary experimentation has shown to yield reasonable results versus other values for the corpora I use. The setting of this parameter (including more nuanced formulations that do not use just a single standard deviation for all distance measurements), is one of many nontrivial issues involved in setting up TPP instances. In general, a smaller value for σ can be used for corpora with relatively small areas of geographic coverage and a larger value for those with greater geographic coverage, if this is known ahead of time and the use case supports the changing of a parameter on a per-corpus basis. However, the resolver introduced in §5.3 based on ant colony optimization provides flexibility as to the importance of travel costs for each document, lessening the need to tune σ.

It is possible to use external knowledge sources to define a travel cost function. I define the **link travel cost function** (LTCF) in the following

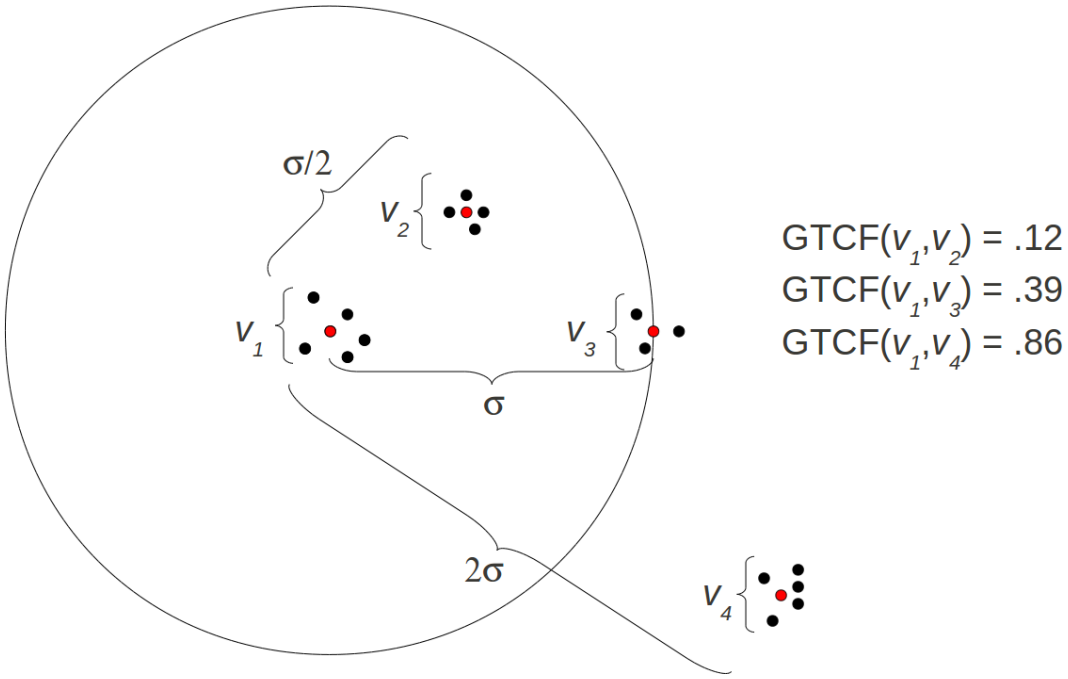


Figure 5.1: Gaussian travel cost function (GTCF) example. The costs to travel from market v_1 to markets at $\sigma/2$, σ, and 2σ away are given.

way. The set of hyperlinks from Wikipedia articles with geographical coordinates to other such geotagged Wikipedia articles form a directed graph. In a document whose toponyms one wishes to resolve, many of the candidate locations of toponyms are likely to have corresponding Wikipedia articles. Given a TPP instance for such a document, the probability $P_{LTCF}(v_i, v_j)$ of traveling from market v_i to market v_j is defined as the number of links from articles whose locations are contained in the source market that point to articles whose locations are in the destination market divided the total number of outgoing links from locations in the source market:

$$P_{LTCF}(v_i, v_j) = \frac{L(v_i, v_j)}{\sum_{v_x \in V} L(v_i, v_x)}$$

where $L(v_i, v_j)$ is the number of links from articles in v_i to articles in v_j. For example, given 10° by 10° grid cell markets and the GeoWiki corpus, the probability of traveling from the market containing London, England to the market containing New York City is about 1%, while the probability of traveling from the market containing Los Angeles, California to the market containing the small town Laramie, Wyoming is only 0.0005%. To change each probability of traveling from one market to another into a cost, I subtract it from 1:

$$\text{LTCF}(v_i, v_j) = 1 - P_{LTCF}(v_i, v_j)$$

Purchase Cost Functions

The final key component of a TPP instance is the cost function that returns the price of a given location l_j at a given market v_i. I define a simple distance-based purchase cost function called the **Gaussian purchase cost function** (GPCF) in much the same vein as the Gaussian travel cost function. For each market, I apply a bivariate Gaussian PDF with a mean at the market's center or centroid and a symmetric covariance matrix such that the standard deviation σ_p is a given value, typically much less than the standard deviation σ_t of the Gaussian travel cost function; I use 161 km. Every candidate location in that market will have a probability density in this distribution, and subtracting this density from 1 yields GPCF:[5]

$$\text{GPCF}(v_i, l_j) = 1 - g_{v_i}(l_{j_{lat}}, l_{j_{lon}}, \sigma_p)$$

This purchase cost function should not be applied when the grid market formulation is used and the center of a market is defined simply as its geometric center, as there is no particular significance to being a location close to the center of a cell with arbitrarily placed boundaries. However, when clustering is used, either in the defining of a grid cell market's (non-geometric) "center" or in the cluster market formulation itself, proximity to a centroid is evidence of being near an area of dense population.

[5]Using h_{v_i} to fix the output of GPCF between 0 and 1, as in GTCF was attempted but adversely affected results.

Any probability distribution over the candidate locations of a toponym token (or type, if the distribution is context insensitive) can be used to define a purchase cost function. In this work, I focus on the distributions output by the WISTR resolver, which compare each toponym token's local context to the contexts of (noisily) labeled training instances by way of a maximum entropy classifier. Again, computing 1 minus a candidate location's score according to the WISTR classifier yields the **WISTR purchase cost function** (WPCF):

$$\text{WPCF}(v_i, l_j) = 1 - P_w(l_j|c_t)$$

where $P_w(l_j|c_t)$ is the probability according to WISTR of location l_j given the local context c_t of the toponym t associated with l_j.

5.3 Solving the Problem

TPP was first proposed by Ramesh (1981). Since then, a variety of solutions have been proposed, both exact and approximate. The exact solutions are intractable for sufficiently large input sizes (where input size includes the number of goods to purchase and the number of markets), as with any NP-complete problem. "Sufficiently large" typically means well fewer than 50 goods and 50 markets. The TRC-DEV corpus contains an average of about seven toponyms per document with an average ambiguity of around 15, making the upper bound on the total number of markets over 100. More geographically inclined documents contain far more than seven toponyms, and the exponential nature of exact solutions to NP-complete problems would make such cases

dominate the runtime. Without dividing the books in the CWar-dev corpus into smaller subdocuments (i.e. maintaining an average number of toponyms per document of almost 700), exact solutions (and likely even approximate solutions) to TPP are certainly intractable, as even the quadratic runtime MinDist and SPIDER resolvers are. Even if the documents are made small enough to mimic the reasonably small toponym frequency of TRC-dev, the average ambiguity of toponyms in CWar-dev is over 30. Thus, exact solutions are not feasible for the datasets used in this thesis. Nevertheless, some of the insights in such solutions are useful in the consideration of more efficient, approximate solutions.

Previous Exact Solutions

Ramesh (1981) proposes a lexicographic search procedure that can be implemented as either an exact or approximate solution. Singh and van Oudheusden (1997) and Laporte et al. (2003) use branch-and-cut methods that construct a binary tree of all possible solutions to a given instance of TPP. The two children of each node represent all (remaining) solutions that contain a particular trip between two markets and all of those that do not. They then find a lower bound on the total cost of each solution in each such subset, and repeat this process until a single tour is found with a lower bound less than or equal to all other lower bounds computed.

Previous Approximate Solutions

Many approximate solutions to TPP involve heuristics. Golden et al. (1981) construct a path while attempting to minimize the cost added at each step. Ong (1982) modifies this strategy and proposes a heuristic based on deleting markets from a complete tour. Pearn and Chien (1998) make further improvements and introduce a heuristic that relies on all goods being available from all markets. Teeninga and Volgenant (2004) improve on subprocedures of this general strategy.

Voß (1996) and Boctor et al. (2003) both utilize tabu search, a method that is less likely to get stuck in local minima when searching the space of solutions than simpler local searches. Ravi and Salman (1999) use a linear programming method that relaxes the constraint that variables must be binary and then rounds the solution to a binary one. Riera-Ledesma and Salazar-González (2005) exchange k markets at a time from paths and achieve better results from previous attempts that only exchanged one market.

Bontoux and Feillet (2008) use ant colony optimization (ACO), a method first introduced by Dorigo et al. (1991, 1996) inspired by the behavior of real ants that cooperate using deposited pheromones to determine the shortest path to a food source. I explore this approach in this work.

ConLAC: a Heuristic Construction Solution

As a first attempt at solving toponym resolution as an instance of TPP, I apply a simple solution template into which a variety of heuristics can be

inserted, and experiment with one particular heuristic. The approach is a construction strategy, fundamentally similar to that of Golden et al. (1981) and those that improve on their work. The strategy of inserting markets into the tour one at a time such that each insertion minimizes the cost increase is used by e.g. Riera-Ledesma and Salazar-González (2005) and Bontoux and Feillet (2008) as local search techniques. The procedure is as follows, given one of the formulations of toponym resolution as TPP outlined in §5.2:

1. Choose an initial market based on a heuristic. Many heuristics are possible, such as choosing the market with the greatest number of goods sold, choosing a market with one or more locations that are unambiguously referred to, choosing a market that will add minimal travel cost to the tour found so far (not applicable when selecting the first market), etc. In this work, I choose the market with the least average purchase cost.

2. Purchase all goods at that market. In general when purchasing goods from a market, re-purchase any goods offered at prices lower than those already paid for those goods and receive a refund of the difference. This eliminates solutions that visit the same markets in the same order as others but do not buy goods at the lowest price seen along the tour.

3. Define the initial tour as starting at the market chosen, with a travel cost of zero.

4. Add markets containing at least one good not yet purchased to the tour according to one of the same heuristics used above.

5. When adding a market to the tour, insert it such that the travel cost increase is minimized.

6. End when all goods have been purchased.

I refer to the version explored in this work as ConLAC: CONstruction via Least Average (purchase) Cost.

An Ant Colony Optimization Solution

An interesting class of solutions to computational problems is that of genetic algorithms, whose parameters and other properties can evolve over multiple iterations in an attempt to adapt to the idiosyncrasies of a particular instance of some problem. Such algorithms have the potential to overcome some of the difficulties encountered by the heuristic construction strategies. Rather than require human intervention in the selection and tuning of solutions, genetic algorithms have the potential to automatically emphasize and de-emphasize various aspects of a problem as their performance dictates according to some objective function.

Bontoux and Feillet (2008) present strong results on known instances of TPP using a genetic algorithm called ant colony optimization (ACO) (Dorigo et al., 1991, 1996). In their approach, virtual ants with differing preferences act as traveling purchasers that leave behind pheromone as they travel, deciding which market to visit next based on a probability distribution that takes many factors into account, including:

- The amount of pheromone already present between the current location and the market
- The ant's attraction to pheromone (called **affinity**), as opposed to a willingness to follow its own new path (called **independence**)
- The ant's emphasis on travel costs (called **laziness**)
- The ant's emphasis on purchase costs (called **avidity**)

Virtual ants are created with random settings for parameters corresponding to the above properties. Each ant chooses which market to visit next from the set of markets it hasn't visited yet based on the following expression:

$$(\tau_{ij})^a \left(\left(\frac{1}{c_{ij}} \right)^s \left(\frac{1}{A_j} \right)^v \right)^d \tag{5.1}$$

where τ_{ij} is the amount of pheromone between markets v_i and v_j, c_{ij} is the travel cost between those markets, and A_j is the economy gained (or lost, if negative) by purchasing all the goods at market v_j not yet purchased and repurchasing any goods sold at v_j already bought for a higher price elsewhere (and receiving a refund for the more expensive version). a, s, v, and d are the four **properties** of an ant: **a**ffinity, lazine**s**s, a**v**idity, and in**d**ependence.

The pheromone left behind by the ants evaporates over time, so that paths more recently traveled are favored. When the total cost an ant has accrued far exceeds the best solution found so far, that ant is killed off. This allows for an evolution of the ant population over time, as those randomly

created with parameters that result in low cost solutions outsurvive those who tend to find higher cost solutions.

Figure 5.2 shows a scenario likely to occur during the TRACO algorithm. Two ants are depicted currently at the market containing Washington, D.C. One has high laziness and low avidity, while the other has low laziness and high avidity. When selecting a market from which to buy the toponym *London*, the former is more likely to choose the London sold in Ontario, since it is closer to the ant's current location. This ant is reluctant (lazy) to travel long distances when it can avoid doing so, but does not value the purchase cost function as much. The other ant does not weigh travel costs highly, but is eager to purchase the London in England for its low purchase cost (due to contextual evidence or simply because London, England is a priori more likely). A similar scenario can be imagined where it is instead affinity and independence that differ, resulting in one ant following a path with high amounts of pheromone and another ant taking the less beaten path.

Taking inspiration from Bontoux and Feillet (2008), my implementation of the ACO approach works as follows:

1. For each document, initialize a fixed number of ants (here 20) with properties drawn from Gaussian distributions with a mean of 1 and a standard deviation determined by experimentation.[6] (In general, larger standard

[6]I use 1 as an initial mean for ant properties, which are all exponents.

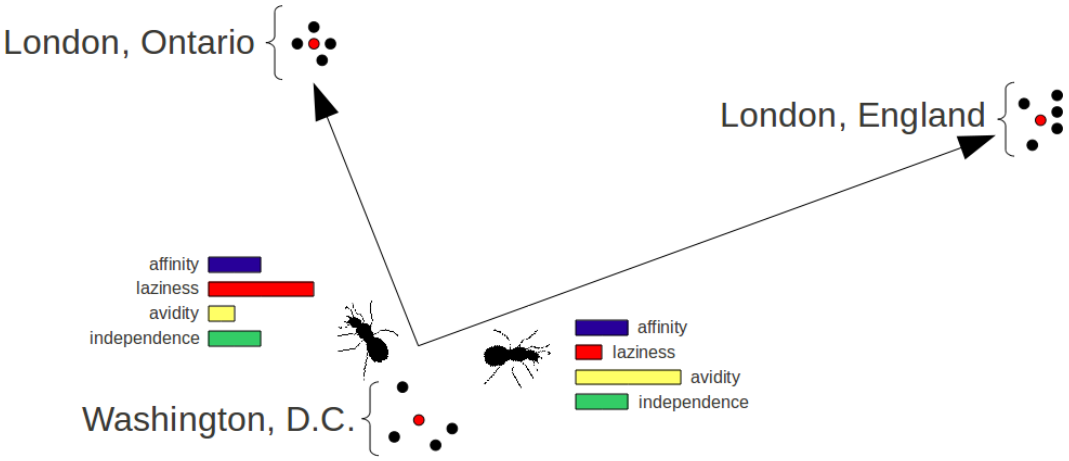


Figure 5.2: Example TRACO scenario in which two ants starting at a market containing Washington, D.C. travel to different Londons due to the preference of each to weigh travel costs or purchase costs more.

deviations make for wilder and/or faster evolution.) I use initial standard deviations of 0.5. Initialize the amount of pheromone between each pair of markets to 1.

2. For each ant, select its next market to visit based on Equation 5.1. Iterate over all ants, moving each one market at a time. After each ant has moved once, evaporate 0.01% of all pheromone.

3. When an ant has purchased all the goods of the TPP instance corresponding to the document, store its tour and the total cost of that tour. Then add 1 pheromone to each market-to-market segment of that complete tour.[7]

[7]I only add pheromone along the paths of successful ants in order not to emphasize poor paths, following Bontoux and Feillet (2008).

4. If an ant's cumulative cost so far exceeds that of the cheapest known complete tour, kill that ant and generate a new one (with an empty tour) whose properties are each drawn from the Gaussian distributions that best fit the properties of the surviving ants. (For example, sample its affinity from $\mathcal{N}_a(\mu_a, \sigma_a^2)$, the Gaussian distribution that best fits the collection of affinity values for all surviving ants, which has mean μ_a and standard deviation σ_a.) This is **local evolution**, and is intended to evolve the ants to adapt to the nuances of the particular document.

5. When a stopping criterion is met (here, the number of tours found is equal to twice the size of the ant population), proceed to the next document.

6. When this process completes on each document, store the properties of the ant population that existed at the end.

7. When this process completes for the whole corpus, fit Gaussian distributions to the properties of the sum of all the ant populations existing at the end of each document's process, and repeat the entire process on the corpus. This is **global evolution**, and is intended to evolve the properties of the ants to adapt to the characteristics of the particular corpus, while still allowing local evolution to distinguish individual documents. Repeat the entire process for a fixed number of iterations. (One could also repeat until ant properties or location predictions stop changing more than some small amount.)

ACO is diverse, as few assumptions are made a priori about whether travel costs or purchase costs should be emphasized for any particular problem. At the same time, the ability to remove ants that are unlikely to find good solutions enables this approach to reduce the search space fairly quickly, making it reasonably efficient. ACO has been shown to find optimal or near-optimal solutions to instances with up to about 350 markets and 200 goods, a scale that suggests the tractability of toponym resolution as TPP on the corpora presented here. These properties combined with the strong results reported by Bontoux and Feillet (2008) make ACO an attractive candidate for an approximate TPP solution.

The version of ACO implemented in this work is referred to as TRACO: Toponym Resolution via Ant Colony Optimization.

5.4 Challenges Involved with TPP

The framing of the problem of toponym resolution as the traveling purchaser problem is not an obvious choice. TPP is a provably difficult problem to solve in and of itself, assuming that one's final goal is to obtain the least expensive tour of markets and purchasing plan. Thus, non-exact solutions must almost always be employed on any TPP instance reflecting some broader task. The choice of non-exact algorithm may depend on the details of the particular TPP instances at hand and what real world problem they represent. Even if TPP were easy to solve exactly, the issues of how to transform the entities in a given problem into markets, goods, and costs, and how to ensure that

minimizing the sum of purchase and travel costs reflects a meaningful solution to the broader problem, are far from trivial.

On the issue of how to represent markets when framing toponym resolution as TPP, there are more questions than simply whether to use a grid or cluster representation. One might imagine a scenario in which every good is purchasable from every market, even though every good is "generated" from only one market. Goods located far from the centroids of markets would likely be expensive at those markets, but that does not preclude an optimal TPP setup from allowing this option. For instance, one might imagine a document almost entirely about Turkey but which makes one passing reference to London. This introduces an empirical question of whether it makes sense to model the selection of London, England as always requiring a trip to a separate market in the United Kingdom area or whether "importing" it to the Turkish market at some relatively high price is worth saving the trip to the British market. For simplicity, I have chosen not to allow such imports in this work.

Issues related to the setting of purchase and travel costs are perhaps the most difficult to solve among TPP-related problems. The most obvious way to define a travel cost is as a simple distance measure in some unit such as kilometers. However, doing so creates the problem of also defining purchase costs (e.g. how difficult it is for any given token of the toponym *London* to be London, England or London, Ontario) also in terms of kilometers. One might imagining probability distributions such as those output by WISTR being multiplied by some constant number of kilometers, e.g. some fraction of the

earth's circumference, in order to be summable with the distance measure of the travel cost function. Choosing this constant would likely require considerable thought and experimentation, as it is unclear a priori how to balance the two cost functions, though they should almost certainly be roughly within the same order of magnitude. Instead of this, I choose to transform distance measures into probabilities via the Gaussian travel cost function and keep the probabilities as they are in the WISTR purchase cost function. Finally, the use of TPP to represent another problem relies on the assumption that the cheapest tours of the TPP instances correspond to the most optimal solutions to the other problem. Unless one is solving a real-world problem that is undoubtedly equivalent to TPP, there is no guarantee of such a correspondence.

The use of a genetic algorithm such as ant colony optimization introduces further complexities. The general components of a genetic algorithm are potential solutions to a problem encoded as strings or vectors (chromosomes), a measure of the fitness of each chromosome with respect to the problem, a way to combine existing chromosomes to create offspring (crossover), and random mutation of new offspring (Mitchell, 1996). Even within the realm of ant colony optimization, there are many different choices that could be made with respect to these components. One might devise additional properties beyond the four I introduced to comprise the chromosomes, or one might omit a property (e.g. independence) and attempt to capture its contribution with another property (e.g. affinity, which is in many ways the inverse of independence). Many genetic algorithms simulate sexual reproduction by mixing the

properties of only two apparently fit chromosomes rather than all surviving chromosomes, as I have done. One might also experiment with mutation operations that are qualitatively or quantitatively different from the Gaussian sampling I have implemented.

In this thesis, I explore only a small fraction of the full set of problems using TPP as a toponym resolution methods introduces. My goal is to show that it is a difficult but not unreasonable way to perform toponym resolution, to provide some preliminary ways to both frame and solve these TPP instances, and to explore some of the noteworthy properties TPP instances and solutions have that my previously introduced resolvers lack.

5.5 Choosing Travel and Purchase Cost Functions

Table 5.1 compares results on a batch of TRACO experiments on TRC-DEV when using the Gaussian travel cost function (GTCF) versus the link travel cost function (LTCF) for various market sizes (both grid and cluster). While there are small variations in results, there is no conclusive winner across the board between the two travel cost functions. Results display a similar pattern for CWAR-DEV. Thus, for the remainder of the experiments I present, I use the geographic distance-based GTCF for its simplicity and reliance on fewer knowledge sources as compared to the Wikipedia link-based LTCF.

Table 5.2 compares the WISTR purchase cost function (WPCF) with the Gaussian purchase cost function (GPCF) using TRACO with GTCF on

Market Size and TCF	Mean	Med.	A	A_{161}
60° GTCF	566	33.7	80.9	86.4
60° LTCF	549	34.0	80.9	86.2
30° GTCF	537	35.8	80.9	86.4
30° LTCF	576	35.8	80.7	86.0
5000 km GTCF	473	32.6	81.9	87.8
5000 km LTCF	501	32.8	81.7	87.7
2500 km GTCF	497	33.7	81.2	87.1
2500 km LTCF	487	33.7	81.1	87.0

Table 5.1: Results when running TRACO on TRC-DEV comparing the Gaussian travel cost function to the link travel cost function at selected market types and sizes.

TRC-DEV. In this case, WISTR's contribution allows WPCF to dominate GPCF, essentially demonstrating the difference between a (noisily) supervised and unsupervised approach. In the remainder of this thesis, I always use WPCF rather than GPCF.

5.6 Tuning Market Type and Size

Figures 5.3 and 5.4 give performance in mean error versus market size for grid cell markets and cluster markets respectively, when CONLAC is run on TRC-DEV. The cost functions used are GTCF and WPCF. In both cases, performance is not particularly sensitive to market size provided it is under about 30° by 30° or 500 km. In each case, there is at least a local minimum with market sizes much larger than the smallest tested. Figure 5.3 shows the best performance with very small markets (0.5° by 0.5°), which coincides with most markets only containing one candidate location. At this market size,

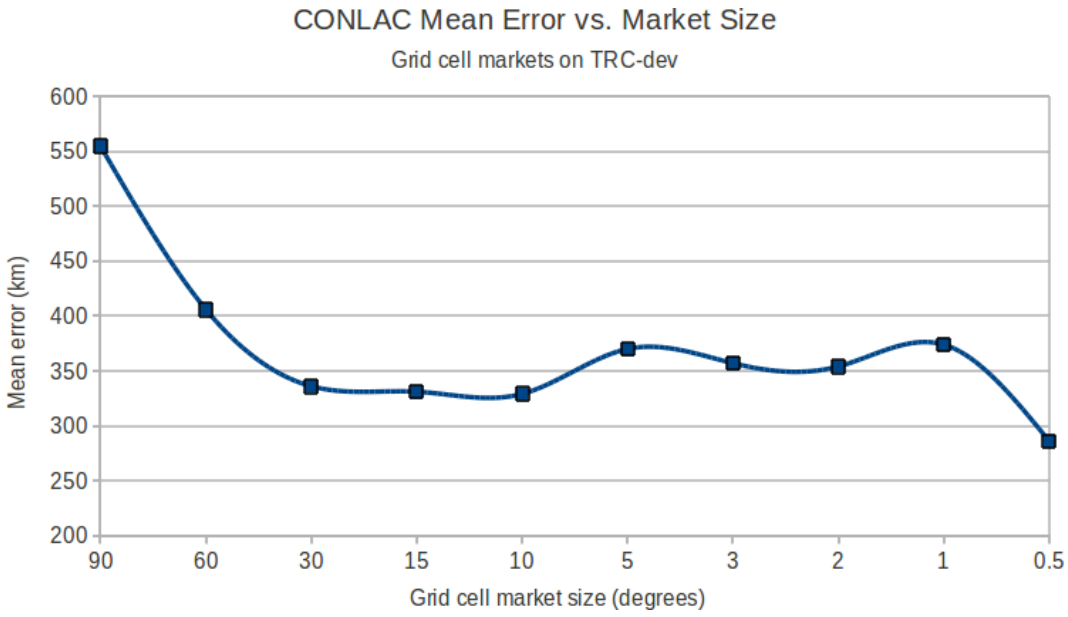


Figure 5.3: Mean error versus market size for grid cell markets with various numbers of degrees on each side of the cell, when ConLAC is run on TRC-dev.

Market Size and PCF	Mean	Med.	A	A_{161}
10000 km WPCF	650	32.8	80.7	87.0
10000 km GPCF	3956	697.1	42.3	45.4
5000 km WPCF	473	32.6	81.9	87.8
5000 km GPCF	3927	745.5	42.6	45.5
2500 km WPCF	497	33.7	81.2	87.1
2500 km GPCF	3634	733.9	42.1	45.3
1000 km WPCF	527	33.7	79.8	85.6
1000 km GPCF	3880	1050.1	39.6	41.8

Table 5.2: Results when running TRACO on TRC-DEV comparing the WISTR purchase cost function to the Gaussian purchase cost function at selected cluster market sizes.

ConLAC behaves very similarly to WISTR itself, almost always selecting the individual candidates with the lowest purchase cost according to WPCF; in other words, the spatial minimality effect caused by multiple candidates being sold in the same market is almost absent. (Recall that the travel cost function is relevant only to the ordering of selected markets in the ConLAC algorithm, so it does not affect performance.) This might appear to negate the potential usefulness of the TPP approach altogether, but the fact that most of the same performance achieved with small 0.5° by 0.5° markets can be achieved with large 30° by 30° markets (on average) is significant for at least two reasons. ConLAC runs at least 10 times faster in the latter case, and the spatial minimality effect of invoking relatively large markets makes up for much of the individual precision lost by averaging individual candidate costs with WPCF.

Figures 5.5 and 5.6 give performance versus market size for grid cell

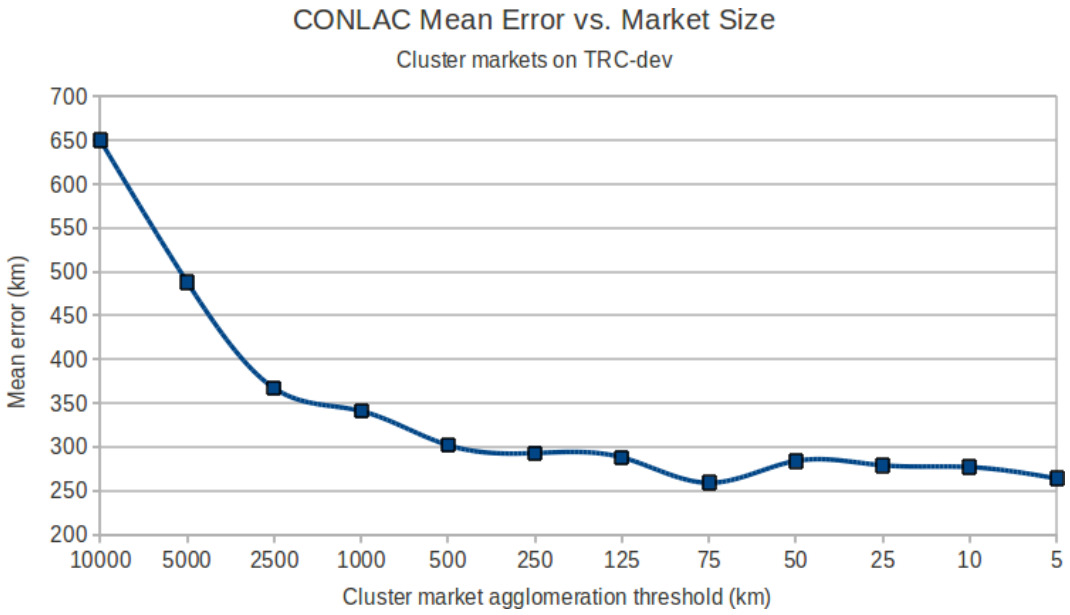


Figure 5.4: Mean error versus market size for cluster markets with various agglomeration thresholds, when ConLAC is run on TRC-dev.

and cluster markets on TRC-dev with the TRACO resolver. As in Figures 5.3 and 5.4, local minima (here likely global as well, especially in Figure 5.6) are found at market sizes well above the smallest tested, at about 60° by 60° and 5000 km. This phenomenon suggests two ideas. The first is that the spatial minimality due to fairly large markets is more important for TRACO than for ConLAC, since the best performance is here *not* found with very small markets, which for TRACO approximates using WISTR directly but still accounting for the travel costs between selected markets (most of which contain only one candidate at such small market sizes). The local minima Figures 5.5 and 5.6 are also much deeper than those in Figures 5.3 and 5.4. The second is that the TR-CoNLL corpus appears to prefer relatively large

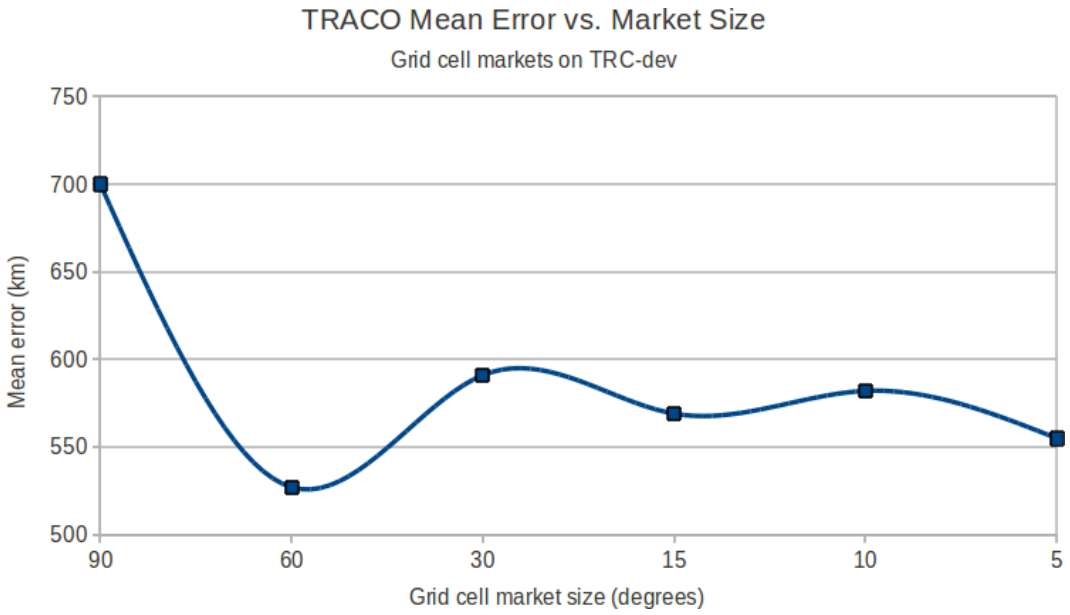


Figure 5.5: Mean error versus market size for grid cell markets with various numbers of degrees on each side of the cell, when TRACO is run on TRC-DEV.

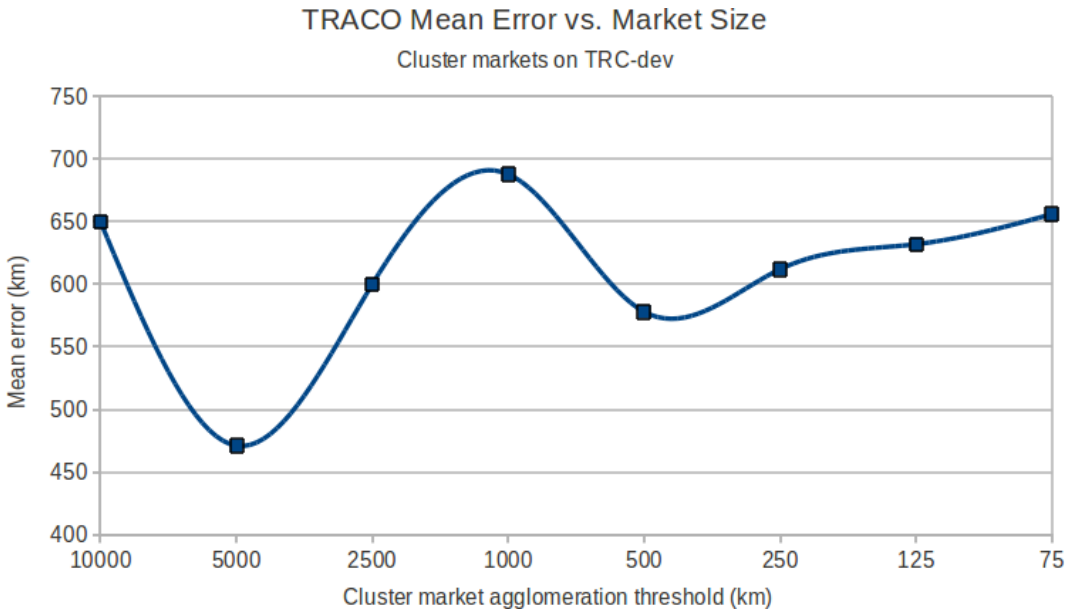


Figure 5.6: Mean error versus market size for cluster markets with various agglomeration thresholds, when TRACO is run on TRC-DEV.

markets, which is intuitive given its international nature. The fact that the TRACO algorithm runs much more quickly with large markets than small markets (of which there are many more) is fortuitous.

Figure 5.7 shows performance versus grid cell market size on CWAR-DEV with CONLAC. As shown by the minimum at the very small market size of 0.5°, the CWAR corpus prefers much smaller markets than the TR-CONLL corpus, reflecting CWAR's characteristic geographic clumping. While spatial minimality has a role to play with CWAR as well as TR-CONLL (indeed, a strong role as the extremely spatially minimalistic SPIDER algorithm shows), it must be invoked on a relatively small geographic scale: grouping locations thousands of kilometers apart is ineffective when almost all locations men-

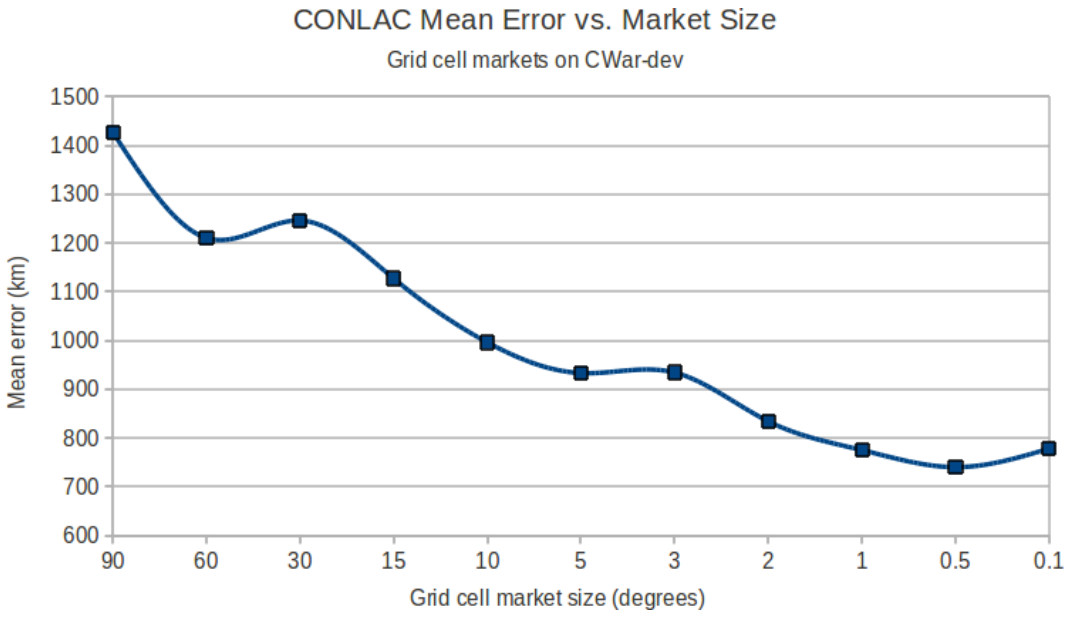


Figure 5.7: Mean error versus market size for grid cell markets with various numbers of degrees on each side of the cell, when ConLAC is run on CWar-dev.

tioned in the corpus are in the eastern United States.

5.7 Results

Table 5.3 gives results of various resolvers on TRC-DEV, repeating results from Chapter 4 for some of the resolvers presented prior to this chapter alongside results for CONLAC and TRACO. Mean error in kilometers, median error in kilometers, accuracy (best match), accuracy (percent within 161 kilometers of correct), precision, recall, and F-score are the performance measures shown. The first four measures reflect performance when gold toponyms are used, while the last three reflect performance when NER is used. Table 5.4 shows the same for CWAR-DEV. Subscript distances in degrees and kilometers indicate the use of grid markets and cluster markets of those sizes, respectively. For TRACO, the number after the colon reflects the number of global iterations run.

Overall, the resolvers introduced in this chapter do not provide quantitative performance benefits beyond the best resolvers from Chapter 3. Comparing CONLAC to TRACO, the simpler heuristic technique tends to perform better. While CONLAC's results are competitive with those of resolvers such as WISTR and TRAWL, TRACO introduces further complexities to the TPP framework not present in CONLAC. These complexities, such as local and global evolution, likely provide too many degrees of freedom for instant success, though studying their behavior is worthwhile. To my knowledge, this work is the first attempt at framing toponym resolution as TPP and solving

Resolver	Mean	Med.	A	A_{161}	P	R	F
Oracle	58	28.1	100.0	94.5	85.1	64.2	73.2
Random	3827	1536.5	34.9	39.2	25.5	19.2	21.9
WISTR	215	**30.7**	**84.2**	90.6	**76.4**	**57.6**	**65.7**
WISTR+SPIDER$_{10}$	237	**30.7**	**84.2**	90.4	76.1	57.4	65.4
TRAWL	177	**30.7**	83.9	91.0	75.7	57.1	65.1
TRAWL+SPIDER$_{10}$	**155**	**30.7**	84.1	**91.2**	75.5	57.0	65.0
ConLAC$_{10°}$	329	33.7	83.1	88.5	74.2	55.9	63.8
ConLAC$_{75km}$	260	31.4	83.0	89.1	75.2	56.7	64.7
TRACO$_{60°:1}$	566	33.7	80.9	86.4	72.7	54.9	62.5
TRACO$_{60°:20}$	502	33.7	81.3	86.8	72.8	54.9	62.6
TRACO$_{5000km:1}$	473	32.6	81.9	87.8	73.7	55.6	63.4
TRACO$_{5000km:20}$	454	31.7	81.9	87.9	73.6	55.5	63.3

Table 5.3: Results on TRC-dev for ConLAC, TRACO, and selected other resolvers. Mean error in kilometers, median error in kilometers, accuracy, and accuracy within 161 kilometers of correct are given for experiments with gold toponyms. Precision, recall, and F-score reflect experiments using NER.

the resulting TPP instances, and I leave refinements to this complex process to future work.

Comparing grid markets with cluster markets, the latter clearly perform better with both ConLAC and TRACO, verifying the intuition that drawing market borders in data-driven, non-arbitrary ways is preferable. The use of global evolution (omitted from the 3° TRACO run on CWar-dev for tractability concerns) does not appear to change results to a large degree.

Though none are clear winners quantitatively, some of the TPP-based resolvers come close to the performance levels of the best previously introduced resolvers. On CWar-dev, ConLAC performs better than WISTR,

Resolver	Mean	Med.	A	A_{161}
ORACLE	0	0.0	100.0	100.0
RANDOM	2283	967.6	13.1	14.6
WISTR	968	**0.0**	67.4	70.5
WISTR+SPIDER$_{10}$	141	**0.0**	**87.2**	87.9
TRAWL	918	**0.0**	69.4	71.4
TRAWL+SPIDER$_{10}$	**94**	**0.0**	82.5	**90.0**
CONLAC$_{0.5^\circ}$	740	**0.0**	67.8	71.0
CONLAC$_{0.1^\circ}$	778	**0.0**	69.4	72.0
TRACO$_{10^\circ:1}$	1148	**0.0**	61.2	64.5
TRACO$_{10^\circ:20}$	1060	0.0	61.9	65.2
TRACO$_{3^\circ:1}$	1044	0.0	58.8	61.8

Table 5.4: Results on CWAR-DEV for CONLAC, TRACO, and selected other resolvers. Mean error in kilometers, median error in kilometers, accuracy, and accuracy within 161 kilometers of correct are given.

demonstrating that there is usefulness in the way such TPP approaches group locations into markets above and beyond the information that WISTR provides. As implemented here, this grouping along with the way TPP instances are solved does not provide as much of a performance boost over using just WISTR as does applying SPIDER to WISTR's output. As is frequently the case in problems of computation, a simple approach given enough data outperforms the prototype versions of more complex approaches with much greater search spaces in terms of design and parameters.

The fact that TRACO demonstrates a further step down in performance (though not to a severe degree when compared to WISTR alone) while being a step up in complexity over CONLAC further supports this observation. In order to beat simple approaches without much room for improvement,

one must explore the far larger space of more complex approaches. Sheer performance figures aside, these more complex approaches often come with interesting properties that can reveal hidden trends in the data as well as provide some direction in the exploration of advanced techniques.

Comparing the TRACO experiments done with no global evolution (only one iteration) to those with global evolution (20 iterations), the general trend is that such evolution improves results. This finding suggests at least two facts. The first is that the objective function, which is the mechanism used to determine when to kill off an ant and generate a new one, is correlated with the correct disambiguation of toponyms. This serves as another general justification of the TPP approach. The second fact is that the evolutionary process itself is justified: the adjustments to the ants' four properties overall lead to lower values of the objective function as well as improved results. As I will show, there is also value in tracking the mean values of each property over the population as evolution progresses.

5.8 Evolutionary Trends

The TRACO algorithm allows for two forms of evolution: local evolution and global evolution. Local evolution occurs on a per-document basis and describes the changing of the properties of the ant population as ants are killed off and created. This allows the ants to adapt based on the characteristics of the particular document. Global evolution occurs on an iterative, corpus-wide basis, and refers to the process of keeping track of the final ant

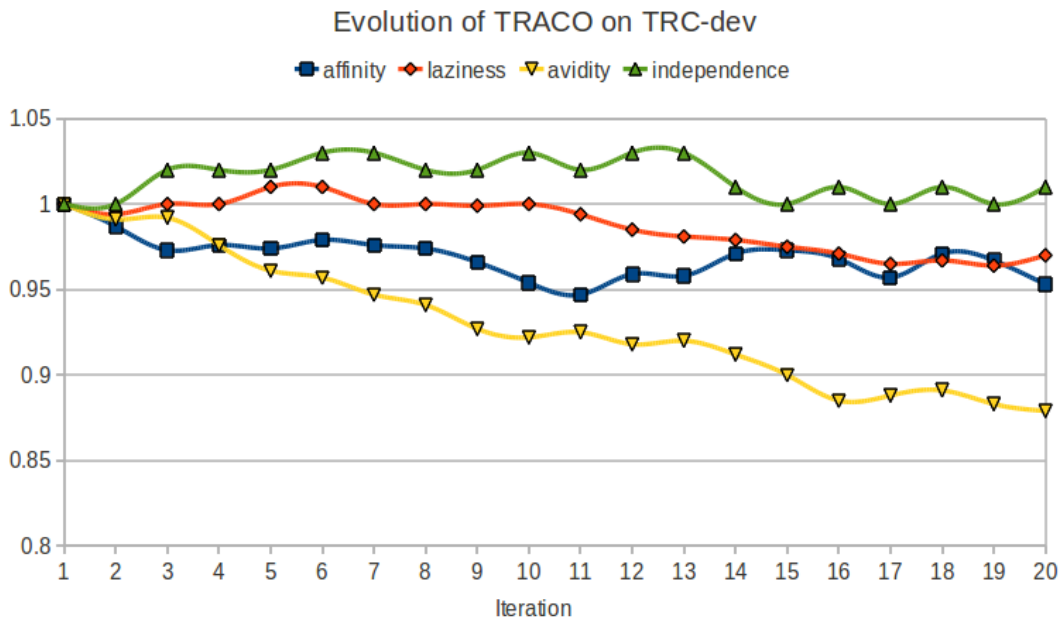


Figure 5.8: Global evolution across 20 iterations of TRACO on TRC-DEV. The y-axis represents the mean of each property at the start of each iteration.

populations of all documents and initializing new ants in a new corpus-wide iteration based on the properties of the final ant populations from the previous iteration. While local evolution allows ants to treat different documents within a corpus differently, it can be instructive to track the global evolution of ants on one corpus and compare it with that of other corpora. Such comparisons reveal which aspects of each corpus are most important to the average ant population of that corpus.

Figure 5.8 shows the global evolution of the four ant properties of one run of TRACO on TRC-DEV, while Figure 5.9 shows the global evolution of a run of TRACO on CWAR-DEV. The means of the four ant properties, affinity, laziness, avidity, and independence, before each of 20 iterations are shown. In

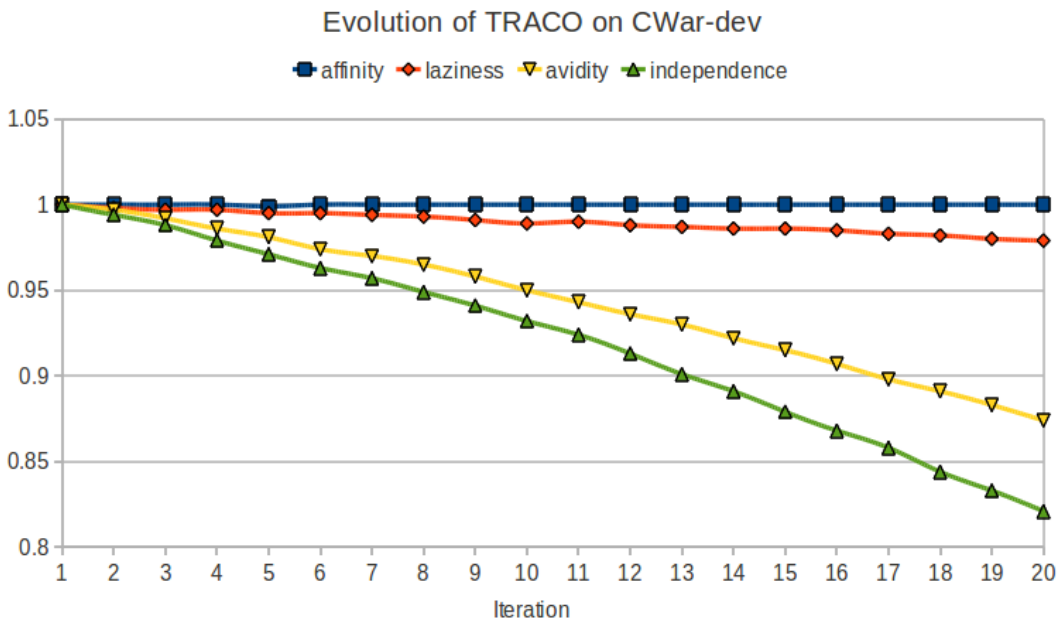


Figure 5.9: Global evolution across 20 iterations of TRACO on CWar-dev. The y-axis represents the mean of each property at the start of each iteration.

each case, grid cell markets of 10° by 10° were used, and all properties were initially drawn from distributions with a mean of 1 and a standard deviation of 0.5. While there is a random component to the global evolution process, neither chart contains curves characteristic of random walks: there are clear trends, especially in the CWar case. Rerunning these experiments yields different but not wildly different evolution patterns. In general, larger corpora yield smoother global evolution patterns, while the heavier weight of each document in a smaller corpus allow differences in individual documents' ant populations to pull the corpus-wide ant population more strongly.

Several trends can be observed from these figures. The first is that affinity (an ant's attraction to pheromone) and independence (an ant's willingness to ignore pheromone and forge its own path) almost never move in the same direction at the same time. In Figure 5.8, the affinity and independence curves are near mirror images of each other, while in Figure 5.9 affinity remains fairly constant as independence decreases on each iteration. This trend reflects the intuition that the overall ant population does not tend to become both more attracted and less attracted to pheromone simultaneously: at most, one changes while the other stays constant. Another trend related to pheromone is that on TR-CoNLL, independence is always greater than affinity, while the opposite is true of CWar. A plausible explanation of this is that the same locations tend to get referred to many times in CWar, while TR-CoNLL is more sporadic.

Another trend is that the ants' laziness property (unwillingness to travel

long distances) only slightly decreases over time on CWar, while its descent is more rapid on TR-CoNLL. This likely reflects the tightly clustered aspect of CWar, where it is rarely the case that traveling a long distance to purchase a good at a low cost is worth the trip (i.e. there aren't many such goods). Though spatial minimality is also present in TR-CoNLL to some degree, it is more common that a long trip is profitable in that corpus, so the ants' average laziness doesn't tend to be as high as for CWar. This trend stands in contrast to the observation that the avidity property (unwillingness to pay high purchase costs) falls in nearly the same fashion on both corpora, likely reflecting the strong role WISTR's predictions have in the TRACO resolver.

5.9 Corpora from Five Springfields

In order to compare the performance of various resolvers on corpora that are less extreme in their distributions of referenced locations, I compiled five small corpora of five local newspaper articles each from five different cities named *Springfield* in the United States. The articles were selected by taking the top five most recent articles containing at least one instance of the toponym *Springfield* from each newspaper's website just after midnight on June 13, 2013. The text for each document includes the article's headline and body text, with no metadata or other information. The OpenNLP named entity recognizer was used to detect toponyms. Table 5.5 and Table 5.6 gives information and statistics about these corpora and the cities they come from. *Springfield* is a noteworthy toponym because it contains an unusually high

Name	pop	pop_{met}	State	Newspaper and URL
S-MA	153k	700k	Massachusetts	The Republican `www.masslive.com/republican`
S-IL	116k	208k	Illinois	State Journal-Register `www.sj-r.com`
S-MO	160k	437k	Missouri	Springfield News-Leader `www.news-leader.com`
S-OH	61k	138k	Ohio	Springfield News-Sun `www.springfieldnewssun.com`
S-OR	59k	352k	Oregon	Springfield Times `www.springfieldtimes.net`

Table 5.5: Information about the Springfield corpora. The columns shown are the corpus name, Springfield population, surrounding metro area population, state of origin, and name of local Springfield newspaper.

number of candidates with comparable populations and prominences.

Visualizing the output of WISTR and TRACO on these corpora reveals some interesting occurrences. (Recall that TRACO includes the use of WISTR's predictions in its purchase cost function.) Both of these resolvers have a tendency to select Springfield, Massachusetts more often than any other Springfield due to its relative prominence in the GeoWiki training data. This makes performance on the S-MA corpus very good. In a sense, this is the trivial case: every instance of *Springfield* in S-MA refers to what the training data indicates is the default Springfield. Almost any sensible toponym resolver would get most of these cases right.

The predictions on some of the other corpora are more interesting, and highlight TRACO's potential to make correct predictions where WISTR

Name	toks	types	toks_{top}	types_{top}	amb_{avg}	amb_{max}
S-MA	3773	1247	30	10	47.0	101
S-IL	1292	527	7	2	86.7	101
S-MO	1553	636	18	7	60.3	101
S-OH	2588	837	32	9	42.5	101
S-OR	1117	486	14	5	76.3	128

Table 5.6: Numerical data from the Springfield corpora. The columns shown are the corpus name, number of tokens, number of types, number of toponym tokens, number of toponym types, average toponym ambiguity, and maximum toponym ambiguity.

fails. On the S-IL corpus, both WISTR and TRACO erroneously predict three instances of *Springfield* to be in Massachusetts. For two other instances, WISTR predicts one to be in Illinois and one to be in Missouri, while TRACO correctly places both in Illinois (Figure 5.10).

A similar phenomenon occurs with the S-OR corpus: WISTR predicts an instance of Springfield, Oregon to be in Massachusetts, while TRACO gets this example right (Figure 5.11).

Finally, Figure 5.12 shows a case where TRACO's global evolution improves a prediction. Recall that global evolution refers to iteratively re-applying TRACO to the same corpus, each time generating the initial ants for each document from Gaussian distributions over their preferences fit to the surviving ant populations from the previous iteration. While both WISTR and TRACO without global evolution predict an instance of Washington state to be Washington, D.C., TRACO with 10 iterations of global evolution makes the correct prediction.

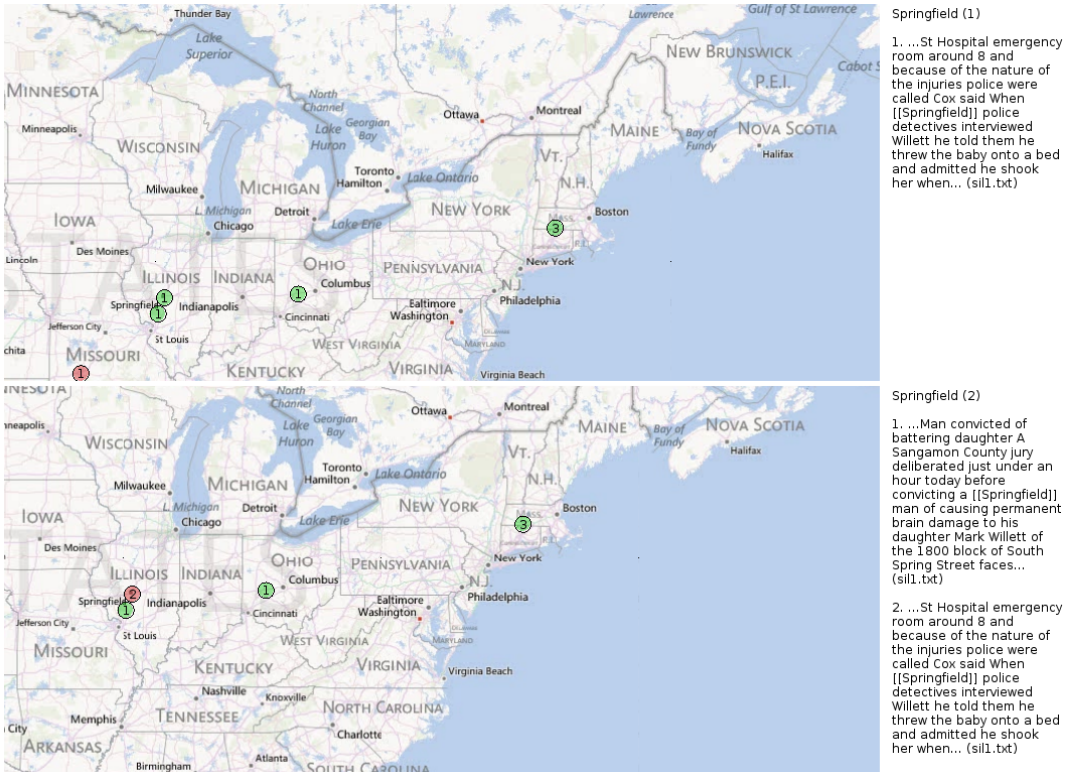
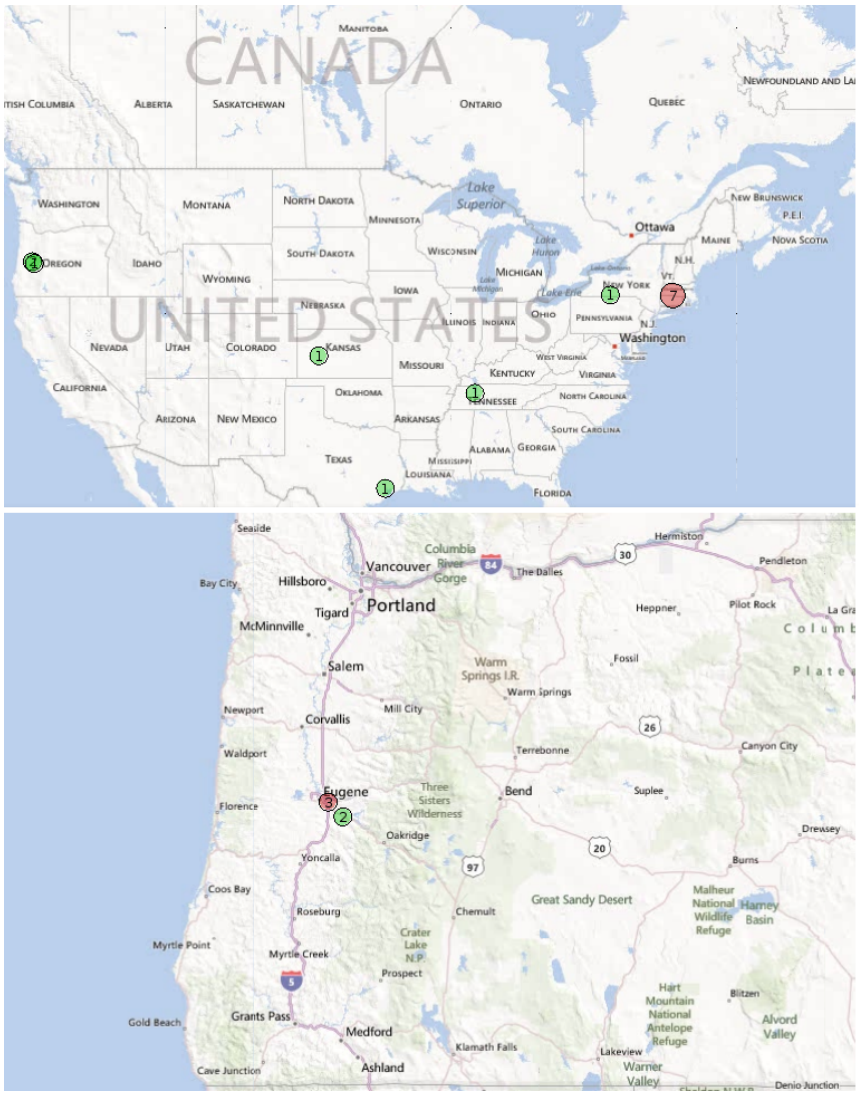


Figure 5.10: On the S-IL corpus, WISTR incorrectly predicts a reference to Springfield, Illinois to be in Missouri (top). TRACO does not make this error (bottom).



passenger with minor injuries was taken to RiverBend hospital in [[Springfield]] and later released Also according to the department four uninvolved adults were standing at the bottom of the embankment when... (sor4.txt)

3. ...Creswell man arrested in [[Springfield]] A Creswell man charged with several sex crimes involving a female minor was found and arrested Wednesday May 15 near... (sor2.txt)

4. ...of a child in a display of explicit conduct online sexual corruption of a child and encouraging child sexual abuse [[Springfield]] police investigators reportedly found that the suspect contacted the unnamed victim using Facebook where he misrepresented his age Roda then... (sor2.txt)

Springfield (3)

1. ...Creswell man arrested in [[Springfield]] A Creswell man charged with several sex crimes involving a female minor was found and arrested Wednesday May 15 near... (sor2.txt)

2. ...of a child in a display of explicit conduct online sexual corruption of a child and encouraging child sexual abuse [[Springfield]] police investigators reportedly found that the suspect contacted the unnamed victim using Facebook where he misrepresented his age Roda then... (sor2.txt)

3. ...lower level of parking structure H under the hospital at about According to PeaceHealth spokesperson Andrea Ash fire crews from [[Springfield]] and Eugene managed to put the fire out in less than 15 minutes and no injuries were reported RiverBend was... (sor5.txt)

Figure 5.11: On the S-OR corpus, WISTR incorrectly predicts a reference to Springfield, Oregon to be in Massachusetts (top). TRACO does not make this error (bottom).

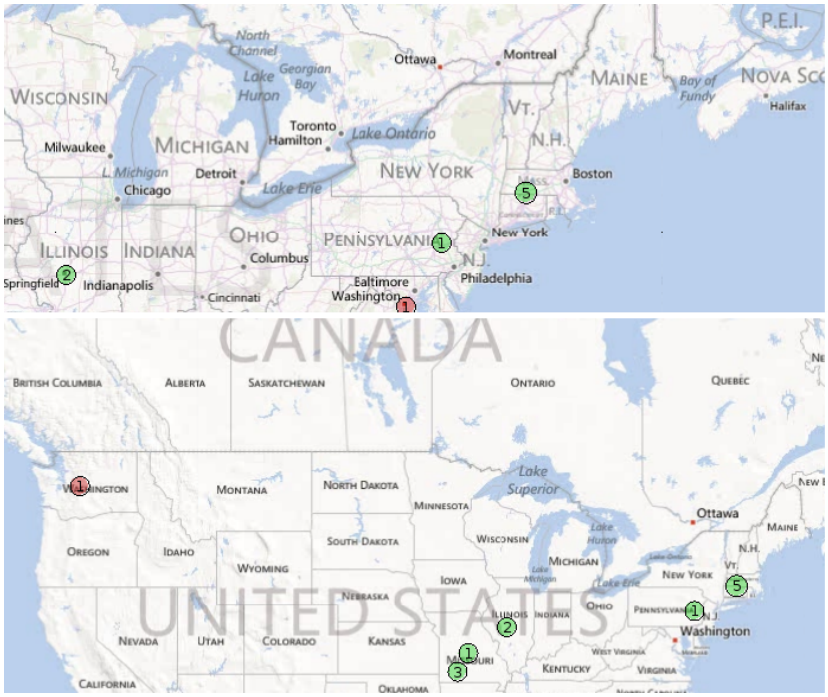


Figure 5.12: On the S-MO corpus, TRACO with no global evolution incorrectly predicts a reference to Washington state to be a reference to Washington, D.C. (top), as does WISTR (not shown). TRACO with 10 iterations of global evolution correctly predicts Washington state (bottom).

These examples show that there are many cases where simply choosing the most prominent candidate for a toponym is doomed to fail, and even more sophisticated context-sensitive resolvers like WISTR can fail if they are overly biased towards a "default" candidate or fail to take into account the geographic distribution of several candidates at once. Resolvers like SPIDER solve the latter problem at the expense of completely ignoring the cues WISTR takes advantage of. TRACO has the potential to balance these two aspects and make correct predictions both in the trivial case and in the case that is the motivation for toponym resolution in the first place: when the correct resolution of a toponym isn't simply the most populous or most discussed candidate.

5.10 Held-out Test Set Results

Tables 5.7 and 5.8 give held-out results on TRC-TEST and CWAR-TEST. These results do not present any major surprises over the development set results. The text-based resolvers WISTR, TRAWL, WISTR+SPIDER, TRAWL+SPIDER, and WISTR+LISTR all outperform the POPULATION baseline in terms of accuracy, which is a strong competitor on TRC-TEST and a weak one on CWAR-TEST. On TRC-TEST, combining the WISTR and WISTR training material in a simple additive manner achieves the best accuracy with gold toponyms and best precision and recall when NER is used, reinforcing the power of machine learning techniques with local context features on corpora made up of international news stories.

On CWAR-TEST, methods employing the SPIDER algorithm continue

Resolver	Mean	Med.	A	A_{161}	P	R	F
ORACLE	107	24.8	100.0	94.3	82.6	59.9	69.4
RANDOM	3891	1523.9	33.0	38.4	26.4	19.2	22.2
POPULATION	**219**	**30.3**	80.5	**90.5**	71.7	52.0	60.2
MINDIST	2251	46.0	54.7	63.7	47.9	34.7	40.2
$SPIDER_{10}$	2175	40.1	56.1	65.3	49.1	35.6	41.3
TRIPDL	1488	37.0	62.1	72.9	51.8	37.5	43.5
WISTR	281	30.5	82.4	89.1	73.9	**53.6**	62.1
WISTR+$SPIDER_{10}$	432	30.7	81.5	87.4	73.2	53.0	61.5
TRAWL	237	30.5	81.4	89.7	72.6	52.6	61.0
TRAWL+$SPIDER_{10}$	300	30.5	81.2	89.1	72.4	52.5	60.8
LISTR	2152	53.1	59.2	64.9	49.1	35.6	41.2
WISTR+LISTR	293	30.7	**82.8**	88.6	**74.0**	**53.6**	**62.2**
WISTR+$LISTR_{Back}$	279	30.5	82.2	88.9	73.7	53.4	61.9
$CONLAC_{10^\circ}$	371	30.7	81.1	87.0	72.4	52.5	60.8
$CONLAC_{75km}$	357	30.7	80.3	87.2	72.4	52.5	60.8
$TRACO_{60^\circ:20}$	497	30.7	80.0	86.3	71.4	51.7	60.0
$TRACO_{5000km:20}$	511	30.7	80.7	86.5	71.2	51.6	59.9

Table 5.7: Results on TRC-TEST. Mean error in kilometers, median error in kilometers, accuracy, and accuracy within 161 kilometers of correct are given for experiments with gold toponyms. Precision, recall, and F-score reflect experiments using NER.

to far outshine those that do not, once again supporting the necessity for spatial minimality on corpora with strong geographic locales, especially when appropriate metadata such as population counts may be unavailable.

The TPP-based resolvers CONLAC and TRACO come close to the performance levels of some of the best other resolvers, and carry with them both interesting studiable properties and a much larger search space of potential ways to set up and solve TPP instances. My work has shown that such

Resolver	Mean	Med.	A	A_{161}
ORACLE	0	0.0	100.0	100.0
RANDOM	2393	1029	11.8	13.4
POPULATION	1749	**0.0**	59.7	62.0
MINDIST	314	**0.0**	57.0	65.5
$SPIDER_{10}$	266	**0.0**	57.5	67.0
TRIPDL	848	**0.0**	51.5	60.2
WISTR	855	**0.0**	69.1	73.3
$WISTR+SPIDER_{10}$	201	**0.0**	**85.9**	87.1
TRAWL	944	**0.0**	67.8	70.8
$TRAWL+SPIDER_{10}$	**148**	**0.0**	78.2	**88.9**
LISTR	1690	597.8	29.0	30.8
WISTR+LISTR	1083	**0.0**	50.9	55.3
$WISTR+LISTR_{Back}$	855	**0.0**	69.1	73.3
$CONLAC_{0.1^\circ}$	711	**0.0**	71.1	74.9
$TRACO_{10^\circ:20}$	966	0.0	63.5	67.8

Table 5.8: Results on CWAR-TEST.

an approach to toponym resolution is viable and should be further explored by future work.

5.11 Future Work

I have made preliminary progress toward the improvement of toponym resolution systems by framing the task as a known computational problem with an explicit objective function. However, there are other ways one might attempt to build better toponym resolvers, aside from exploring the spaces outlined in §5.4. If one's primary goal is only to maximize accuracy—a likely scenario for a commercial application, for example—employing at least a few rules at the beginning of the resolution process is likely to be helpful. For

instance, fixing presumably unambiguous strings like *Roxbury, Vermont* to their correct location before applying methods like those introduced here may improve performance somewhat (especially if syntactic information is available to eliminate spurious cases like *...as opposed to the Boston neighborhood of Roxbury, Vermont has some of my favorite...*), though there is not vast room for improvement in these least difficult of cases. Some incorporation of data such as population is also likely to increase accuracy, if the user is certain of its appropriateness for the corpus at hand.

A more difficult avenue for improvement, but one with much theoretical promise, is that of jointly performing named entity recognition and toponym resolution (or disambiguation of named entities of all kinds). If a document seems to contain many references to locations nearby a town called Calhoun, one might expect that instances of the string *Calhoun* in the same document do indeed refer to that town rather than to a person named Calhoun. (However, caution must be taken to avoid cases where a group of historically important and interconnected individuals eventually became the namesakes for multiple nearby locations, a common scenario in many areas of the world.)

A final reasonable direction for future work lies in the gathering of more data for approaches like WISTR, TRAWL, and the TPP-based resolvers. A semisupervised approach to detecting likely references to particular named locations has the potential to yield far more training instances than something like WISTR alone. Starting with a reasonable model of the features indicative of particular locations like one from WISTR, one could imagine a system that

self-trains on completely unlabeled data by looking for instances of toponyms in environments similar to those it has seen before. With caution against the typical pitfalls of unsupervised and semisupervised learning like excessive bias towards already seen features and the potential for unwarranted drift in the feature space, such an approach could multiply the size of a resolver's training data by orders of magnitude, without ever needing additional annotations done by hand. The engineering of other kinds of features like larger n-grams and features related to part of speech, syntax, or semantics, could aid resolvers in recognizing the environments that toponyms occur in and the environments that references to particular locations occur in.

Chapter 6

Conclusion

The recent explosion of data generation and availability presents the opportunity for previously unimaginable benefits throughout many facets of humanity, if that data can be harnessed and processed in useful ways. One might posit a "holy grail" of automated analysis tools that would take as input arbitrary amounts of text, images, video, and metadata such as time stamps and geographic coordinates, and yield all and only the most useful subset of that data, processed for ease of digestion by a human user, so that evidence-backed actions could be taken (or new understandings of historical events revealed) with a level of confidence never before possible. While natural language processing, computer vision, knowledge representation, and other fields are gaining ground on many fronts with respect to such a goal, the challenges involved continue to prove difficult to solve.

One area where there is both clear applicability to real world issues and the potential for high levels of performance in the present day is that of the automatic detection of geographic aspects of data. In this case, the space into which source data must be mapped is a merciful two dimensions, and it is a concrete space that is constantly being precisely measured from

the surface and from space. In contrast, there is no consensus on what the shape of a semantic space should be, though all researchers working on the problem agree that it is not a two-dimensional surface. The very aspect of automated geotagging that makes it so clearly useful is the same aspect that makes it more accessible than many other automated data processing tasks: it is grounded.

Toponym resolution is a further subset of this problem with further simplifying constraints. Its output only concerns explicit references to places (though its set of potentially relevant inputs is unbounded), and though there is difference in coverage and slight disagreement in coordinates among gazetteers, the set of labels a resolver must assign to each toponym is relatively well defined. Thus, toponym resolution is one of the best entry points into the universe of automated data processing.

There are already many ongoing tasks that would benefit from robust automated toponym resolution as well as the kinds of visualizations presented in this thesis. Historical studies could take advantage of much larger corpora than otherwise possible, extracting trends in the connection between text and geography that might otherwise elude researchers (Guldi, 2009; Scheidel et al., 2012; Derungs and Purves, 2013). Similar studies on modern corpora such as news articles and microblogs have an even greater need for high quality automatic processing given their much greater amounts of data (Teitler et al., 2008; Sankaranarayanan et al., 2009; Gelernter and Mushegian, 2011). The World Bank Mapping for Results project aims to monitor the use of its funds

throughout the world toward socially conscious goals, and could better do so with geographically grounded text related to such efforts [1].

Many have shown that a large portion of the toponyms in many corpora can be resolved correctly through simple means such as the POPULATION baseline and reliance on unambiguous sequences of toponyms like *London, England*. This thesis has focused on the less obvious cases, without which toponym resolution would not be a problem. It has explored the space of resolvers that ignore such cues in favor of more general knowledge sources such as a toponym's textual context and the distribution of locations (including candidate locations) on the earth's surface, two types of features always present in the task. By contrast, population information is costly to gather, may be lacking for less populous places, and can be inappropriate for corpora such as the CWAR corpus. Unambiguous toponym sequences are useful when present, but are often absent. Corpora like the sets of local Springfield newspaper articles present a use case where both of these simplistic approaches are likely to fail often. The POPULATION baseline believes Missouri to be the only U.S. state housing a Springfield (or Massachusetts, if metro area data is used instead), so failure is guaranteed for other Springfields. The authors of these articles assume a local audience, so strings like *Springfield, Oregon* are virtually always avoided. Methods that take advantage of all textual cues in context, while taking into account likely distributions of locations on the earth, are the clear winners in these non-obvious cases.

[1] `http://maps.worldbank.org/`

WISTR represents the extreme of text-based resolvers, ignoring all geographic information other than the set of candidates for a toponym in the gazetteer. On the other hand, SPIDER relies almost completely on one preconception of likely geographic distributions, ignoring all textual cues. As expected, the former excels at the TR-CoNLL corpus of international news while the latter achieves very high performance on the smaller scoped, historical CWar corpus. These extremes demonstrate the need for a flexible approach that can adapt to these and other aspects of the corpus to resolve. TRAWL is one such attempt, giving more weight to its local context component the more training data it has given a toponym. The distributions over candidates output by WISTR and TRAWL can be used to seed SPIDER, approximating the balance between textual and geographic evidence. The TPP approaches are a more principled way to combine knowledge sources, and the genetic approach TRACO is particularly suited to adaptation. TRACO has other valuable properties, such as the output of tours that show the geographic (not necessarily temporal or textual) path a document takes, and ants whose evolved properties suggest some of the latent differences between documents and between corpora.

The focus of this thesis has been on toponym resolution methods that take advantage of the knowledge sources inherent in the problem, namely text and geography, rather than on quick and dirty methods that cover the obvious cases. Different corpora call for different emphases on these two aspects (and various sub-aspects of them), so a broadly applicable approach must be flexible

in its treatment of different kinds of features. I have presented a variety of resolvers, some of which lay the groundwork for this type of flexibility, focusing not on sheer accuracy alone but on the more challenging facets of toponym resolution and on resolvers with something interesting to say about their input corpora. Nevertheless, several of my approaches achieve levels of accuracy high enough for users interested in the geographic story hidden behind the words of their text collections.

Bibliography

B. Adams and G. McKenzie. Inferring thematic places from spatially referenced natural language descriptions. *Crowdsourcing Geographic Knowledge*, pages 201–221, 2013.

Benjamin Adams and Krzysztof Janowicz. On the geo-indicativeness of non-georeferenced text. In *Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media*, pages 375–378, 2012.

Einat Amitay, Nadav Har'El, Ron Sivan, and Aya Soffer. Web-a-where: geotagging web content. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 273–280, 2004.

Fayez F. Boctor, Gilbert Laporte, and Jacques Renaud. Heuristics for the traveling purchaser problem. *Computers and Operations Research*, 30(4): 491 – 504, 2003.

Boris Bontoux and Dominique Feillet. Ant colony optimization for the traveling purchaser problem. *Computers and Operations Research*, 35(2):628–637, 2008.

Tobias Josef Brunner and Ross Stuart Purves. Spatial autocorrelation and

toponym ambiguity. In *Proceedings of the 2nd international workshop on Geographic information retrieval*, pages 25–26, 2008.

Davide Buscaldi and Paulo Rosso. A conceptual density-based approach for the disambiguation of toponyms. *Int. J. Geogr. Inf. Sci.*, 22(3):301–313, 2008.

Zhiyuan Cheng, James Caverlee, and Kyumin Lee. You are where you tweet: A content-based approach to geo-locating twitter users. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pages 759–768, 2010.

N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh, and J.C. Mitchell. Client-side defense against web-based identity theft. In *11th Annual Network and Distributed System Security Symposium (NDSS04)*. San Diego, USA, 2004.

P. Clough. Extracting metadata for spatially-aware information retrieval on the internet. In *Proceedings of the 2005 workshop on Geographic information retrieval*, pages 25–30. ACM, 2005.

Gregory Crane. The Perseus Digital Library, 2000. URL `http://www.perseus.tufts.edu`.

Gregory Crane. Georeferencing in historical collections. *D-Lib Magazine*, 10 (5), 2004.

Curdin Derungs and Ross S Purves. From text to landscape: locating, identifying and mapping the use of landscape features in a swiss alpine corpus.

International Journal of Geographical Information Science, (ahead-of-print): 1–22, 2013.

Duarte Dias, Ivo Anastácio, and Bruno Martins. A language modeling approach for georeferencing textual documents. In *Proceedings of the 2nd Spanish Conference in Information Retrieval*, 2012.

Junyan Ding, Luis Gravano, and Narayanan Shivakumar. Computing geographical scopes of web resources. In *Proceedings of the 26th International Conference on Very Large Data Bases*, pages 545–556, 2000.

Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. Positive feedback as a search strategy. Technical report, Dipartimento di Elettronica Politecnico di Milano, Italy, 1991.

Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics*, 26:29–41, 1996.

Jacob Eisenstein, Brendan O'Connor, Noah A. Smith, and Eric P. Xing. A latent variable model for geographic lexical variation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1277–1287, 2010.

Jacon Eisenstein, Ahmed Ahmed, and Eric P. Xing. Sparse additive generative models of text. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1041–1048, 2011.

Eric Garbin and Inderjeet Mani. Disambiguating toponyms in news. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 363–370, 2005.

J. Gelernter and N. Mushegian. Geo-parsing messages from microtext. *Transactions in GIS*, 15(6):753–773, 2011.

Bruce Golden, Larry Levy, and Roy Dahl. Two generalizations of the traveling salesman problem. *Omega*, 9(4):439 – 441, 1981.

C. Grover, R. Tobin, K. Byrne, M. Woollard, J. Reid, S. Dunn, and J. Ball. Use of the Edinburgh geoparser for georeferencing digitized historical collections. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 368(1925):3875–3889, 2010.

J. Guldi. The spatial turn. *Spatial Humanities: a Project of the Institute for Enabling*, 2009.

M. Haklay and P. Weber. Openstreetmatp: User-generated street maps. *Pervasive Computing, IEEE*, 7(4):12–18, 2008.

Qiang Hao, Rui Cai, Changhu Wang, Rong Xiao, Jiang-Ming Yang, Yanwei Pang, and Lei Zhang. Equip tourists with knowledge mined from travelogues. In *Proceedings of the 19th international conference on World wide web*, pages 401–410, 2010.

J.A. Hartigan and M.A. Wong. Algorithm as 136: A k-means clustering algorithm. *Applied statistics*, pages 100–108, 1979.

B. Hecht, S. Carton, M. Quaderi, J. Schöning, M. Raubal, D. Gergle, and D. Downey. Explanatory semantic relatedness and explicit spatialization for exploratory search. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 415–424. ACM, 2012.

Brent Hecht, Lichan Hong, Bongwon Suh, and Ed H Chi. Tweets from justin bieber's heart: the dynamics of the location field in user profiles. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, pages 237–246. ACM, 2011.

Linda L. Hill. *Georeferencing: The Geographic Associations of Information*. MIT Press, 2006.

J. Hoffart, M. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics, 2011.

L. Hollenstein and R. Purves. Exploring place through user-generated content: Using flickr tags to describe city cores. *Journal of Spatial Information Science*, (1):21–48, 2012.

Suradej Intagorn and Kristina Lerman. A probabilistic approach to mining geospatial knowledge from social annotations. In *Conference on Information and Knowledge Management (CIKM)*, 2012.

Neil Ireson and Fabio Ciravegna. Toponym resolution in social media. In *The Semantic Web ISWC 2010*, volume 6496, pages 370–385. 2010.

Christopher Jones, Ross Purves, Paul Clough, and Hideo Joho. Modelling vague places with knowledge from the web. *International Journal of Geographical Information Science*, 2008.

Evangelos Kalogerakis, Olga Vesselova, James Hays, Alexei Efros, and Aaron Hertzmann. Image sequence geolocation with human travel priors. In *Proceedings of the IEEE 12th International Conference on Computer Vision*, pages 253–260, 2009.

Sheila Kinsella, Vanessa Murdock, and Neil O'Hare. "I'm eating a sandwich in Glasgow": Modeling locations with tweets. In *Proceedings of the 3rd International Workshop on Search and Mining User-generated Contents*, pages 61–68, 2011.

S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective annotation of Wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–466. ACM, 2009.

Susana Ladra, Miguel Luaces, Oscar Pedreira, and Diego Seco. A toponym resolution service following the ogc wps standard. In *Web and Wireless Geographical Information Systems*, volume 5373, pages 75–85. 2008.

G. Laporte, J. Riera-Ledesma, and JJ. Salazar-Gonzáles. A branch-and-cut algorithm for the undirected traveling purchaser problem. *Operations Research*, 162:940–951, 2003.

Jochen L. Leidner. *Toponym resolution in text: Annotation, Evaluation and Applications of Spatial Grounding of Place Names*. Universal Press, Boca Raton, FL, USA, 2008.

Huifeng Li, Rohini K. Srihari, Cheng Niu, and Wei Li. Infoxtract location normalization: a hybrid approach to geographic references in information extraction. In *Proceedings of the HLT-NAACL 2003 workshop on Analysis of geographic references - Volume 1*, pages 39–44, 2003.

Yi Li. Probabilistic toponym resolution and geographic indexing and querying. Master's thesis, The University of Melbourne, Melbourne, Australia, 2007.

Vitor Loureiro, Ivo Anastácio, and Bruno Martins. Learning to resolve geographical and temporal references in text. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 349–352, 2011.

M. Louwerse and N. Benesh. Representing spatial structure through maps and language: Lord of the Rings encodes the spatial structure of Middle Earth. *Cognitive science*, 36(8):1556–1569, 2012.

I. Mani, C. Doran, D. Harris, J. Hitzeman, R. Quimby, J. Richer, B. Wellner,

S. Mardis, and S. Clancy. SpatialML: annotation scheme, resources, and evaluation. *Language Resources and Evaluation*, 44(3):263–280, 2010.

Melanie Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1996.

Hoon Liong Ong. Approximate algorithms for the travelling purchaser problem. *Operations Research Letters*, 1(5):201 – 205, 1982.

Simon Overell. *Geographic Information Retrieval: Classification, Disambiguation and Modelling*. PhD thesis, Imperial College London, 2009.

Simon Overell and Stefan Rüger. Using co-occurrence models for placename disambiguation. *International Journal of Geographical Information Science*, 22:265–287, 2008.

Yanwei Pang, Qiang Hao, Yuan Yuan, Tanji Hu, Rui Cai, and Lei Zhang. Summarizing tourist destinations by mining user-generated travelogues and photos. *Computer Vision and Image Understanding*, 115(3):352 – 363, 2011.

W.L. Pearn and R.C. Chien. Improved solutions for the traveling purchaser problem. *Computers and Operations Research*, 25(11):879 – 885, 1998.

V. Petras. Statistical analysis of geographic and language clues in the marc record. Technical report, The University of California at Berkeley, 2004.

T. Qin, R. Xiao, L. Fang, X. Xie, and L. Zhang. An efficient location extraction algorithm by leveraging web contextual information. In *Proceedings of*

the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, pages 53–60. ACM, 2010.

T. Ramesh. Traveling purchaser problem. *Opsearch*, 18:87–91, 1981.

Erik Rauch, Michael Bukatin, and Kenneth Baker. A confidence-based framework for disambiguating geographic terms. In *Proceedings of the HLT-NAACL 2003 workshop on Analysis of geographic references - Volume 1*, pages 50–54, 2003.

R. Ravi and F. Salman. Approximation algorithms for the traveling purchaser problem and its variants in network design. In *Algorithms - ESA 99*, volume 1643, pages 696–696. 1999.

Jorge Riera-Ledesma and Juan José Salazar-González. A heuristic approach for the travelling purchaser problem. *European Journal of Operational Research*, 162(1):142 – 152, 2005.

Kirk Roberts, Cosmin Adrian Bejan, and Sanda Harabagiu. Toponym disambiguation using events. In *Proceedings of the 23rd International Florida Artificial Intelligence Research Society Conference*, pages 271–276, 2010.

Stephen Roller, Michael Speriosu, Sarat Rallapalli, Benjamin Wing, and Jason Baldridge. Supervised text-based geolocation using language models on an adaptive grid. In *Proceedings of EMNLP 2012*, 2012.

Adam Sadilek, Henry Kautz, and Jeffrey P. Bigham. Finding your friends and following them to where you are. In *Proceedings of the 5th ACM In-*

ternational Conference on Web Search and Data Mining, pages 723–732, 2012.

Jagan Sankaranarayanan, Hanan Samet, Benjamin E. Teitler, Michael D. Lieberman, and Jon Sperling. Twitterstand: news in tweets. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 42–51, 2009.

W. Scheidel, E. Meeks, and J. Weiland. ORBIS: The Stanford geospatial network model of the roman world. 2012.

Pavel Serdyukov, Vanessa Murdock, and Roelof van Zwol. Placing flickr photos on a map. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 484–491, 2009.

Kashi N. Singh and Dirk L. van Oudheusden. A branch and bound algorithm for the traveling purchaser problem. *European Journal of Operational Research*, 97(3):571–579, 1997.

A. Skupin and A. Esperbé. An alternative map of the United States based on an *n*-dimensional model of geographic space. *Journal of Visual Languages & Computing*, 22(4):290–304, 2011.

David Smith and Gregory Crane. Disambiguating geographic names in a historical digital library. In *Proceedings of the 5th European Conference on Research and Advanced Technology for Digital Libraries*, pages 127–136, 2001.

David A. Smith and Gideon S. Mann. Bootstrapping toponym classifiers. In *Proceedings of the HLT-NAACL 2003 workshop on Analysis of geographic references - Volume 1*, pages 45–49, 2003.

Michael Speriosu and Jason Baldridge. Text-driven toponym resolution using indirect supervision. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, 2013.

Michael Speriosu, Travis Brown, Taesun Moon, Jason Baldridge, and Katrin Erk. Connecting language and geography with region-topic models. In *Proceedings of the Workshop on Computational Models of Spatial Language Interpretation at Spatial Cognition 2010*, pages 33–40, 2010.

Kate Starbird and Leysia Palen. Voluntweeters: Self-organizing by digital volunteers in times of crisis. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, pages 1071–1080. ACM, 2011.

A. Teeninga and A. Volgenant. Improved heuristics for the traveling purchaser problem. *Computers and Operations Research*, 31(1):139 – 150, 2004.

B. Teitler, M. Lieberman, D. Panozzo, J. Sankaranarayanan, H. Samet, and J. Sperling. NewsStand: a new view on news. In *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, page 18. ACM, 2008.

Andrew J. Torget, Rada Mihalcea, Jon Christensen, and Geoff McGhee. Mapping texts: combining text-mining and geo-visualization to unlock the re-

search potential of historical newspapers. Technical report, University of North Texas, Stanford University, 2012.

Edward R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press LLC, second edition, 2001.

Raphael Volz, Joachim Kleb, and Wolfgang Mueller. Towards ontology-based disambiguation of geographical identifiers. In *Proceedings of the 16th International Conference on World Wide Web*, 2007.

Stefan Voß. Dynamic tabu search strategies for the traveling purchaser problem. *Annals of Operations Research*, 63:253–275, 1996.

Benjamin Wing and Jason Baldridge. Simple supervised document geolocation with geodesic grids. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 955–964, 2011.

D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics, 1995.

Qingqing Zhang, Peiquan Jin, Sheng Lin, and Lihua Yue. Extracting focused locations for web pages. In *Web-Age Information Management*, volume 7142, pages 76–89. 2012.

Wenbo Zong, Dan Wu, Aixin Sun, Ee-Peng Lim, and Dion Hoe-Lian Goh. On assigning place names to geography related web pages. In *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, pages 354–362, 2005.