JOURNAL OF SPATIAL INFORMATION SCIENCE
Number 3 (2011), pp. 65–101
doi:10.5311/JOSIS.2011.3.55

RESEARCH ARTICLE

GeoCAM: A geovisual analytics workspace to contextualize and interpret statements about movement

Anuj Jaiswal[1,3], Scott Pezanowski[3,4], Prasenjit Mitra[1,2,3], Xiao Zhang[2,3], Sen Xu[3,4], Ian Turton[3,4], Alexander Klippel[3,4], and Alan M. MacEachren[3,4,1]

[1]College of Information Sciences and Technology, The Pennsylvania State University, PA, USA
[2]Dept. of Computer Science and Engineering, The Pennsylvania State University, PA, USA
[3]GeoVISTA Center, The Pennsylvania State University, PA, USA
[4]Department of Geography, The Pennsylvania State University, PA, USA

Received: April 28, 2011; returned: June 14, 2011; revised: September 15, 2011; accepted: September 28, 2011.

Abstract: This article focuses on integrating computational and visual methods in a system that supports analysts to identify, extract, map, and relate linguistic accounts of movement. We address two objectives: (1) build the conceptual, theoretical, and empirical framework needed to represent and interpret human-generated directions; and (2) design and implement a geovisual analytics workspace for direction document analysis. We have built a set of geo-enabled, computational methods to identify documents containing movement statements, and a visual analytics environment that uses natural language processing methods iteratively with geographic database support to extract, interpret, and map geographic movement references in context. Additionally, analysts can provide feedback to improve computational results. To demonstrate the value of this integrative approach, we have realized a proof-of-concept implementation focusing on identifying and processing documents that contain human-generated route directions. Using our visual analytic interface, an analyst can explore the results, provide feedback to improve those results, pose queries against a database of route directions, and interactively represent the route on a map.

Keywords: geovisual analytics, text analysis, text-to-sketch, geographic information retrieval, spatial cognition, machine learning, movement, geospatial databases

1 Introduction

Research in geographic information retrieval (GIR) has been quite successful in developing computational methods to identify, extract, and geocode place names found in text documents [31, 33, 40]. While place name disambiguation is an important and sometimes hard task (e.g., there were 1530 instances of the name "Columbia" in the geographic names information system, GNIS [47], when retrieved on September 12th, 2011), place name extraction is just a small part of the challenge of making effective use of geographic information encoded in documents. We present a geovisual analytics approach to identifying, extracting, mapping, and querying human-generated route directions. Our results have potential direct applicability within wayfinding systems that can generate maps from verbal directions. The approach is also relevant as a method to build a large corpus of human-generated route directions, which can serve as input to research on human conceptualization of movement in space [49]. The approach developed is generalizable to a wider range of movement statements beyond directions, e.g., topological-cognitive assessments of geographic scale movement patterns [23], conceptualization of turn directions in travel documents [25], and so forth.

Extracting and mapping human-generated route directions computationally is a hard problem because these directions can include many ambiguous terms, inconsistent abbreviations, and imprecise references to places (e.g., "from the north"). Further, the typical geocoding challenge of locating named entities is harder for streets than named places since they can be even less unique (e.g., "Main Street"). Human analysts are often able to interpret even vague and imprecise geographic references by inferring the correct context and drawing upon external information such as maps and images. To achieve successful interpretation and mapping of human-generated route directions computationally, the challenges to be met can be grouped broadly into the following classes (illustrated in Figure 1):

1. *Document classification*. A first step here is the automatic classification of documents into those that contain directions and those that do not. A related problem is the automatic classification of documents based on the mode of transportation used and the type of spatial movement pattern described by the directions (for example, regional/local movement patterns).

2. *Movement entities*. Given documents with directions, the components of movement (origins, destinations, and the route actions, i.e., the movement patterns required to be taken to go from the origins to the destinations) must be identified, extracted, and associated. Since documents potentially contain multiple start/end points and corresponding actions, the association of the respective start point to actions and to end points is necessary in order to extract consistent movement patterns.

3. *Entity georeferencing*. To map extracted route components (or other statements about movement), the key entities making up the route must be extracted and georeferenced. To this end, individual place names and other entities (e.g., buildings, streets, city names) must be extracted, disambiguated, and their spatial coordinates derived.

4. *Query support*. Given a large volume of directions (or other movement statements), methods are needed to enable the end-user to search for relevant documents and visualize the directions extracted from these documents on maps.

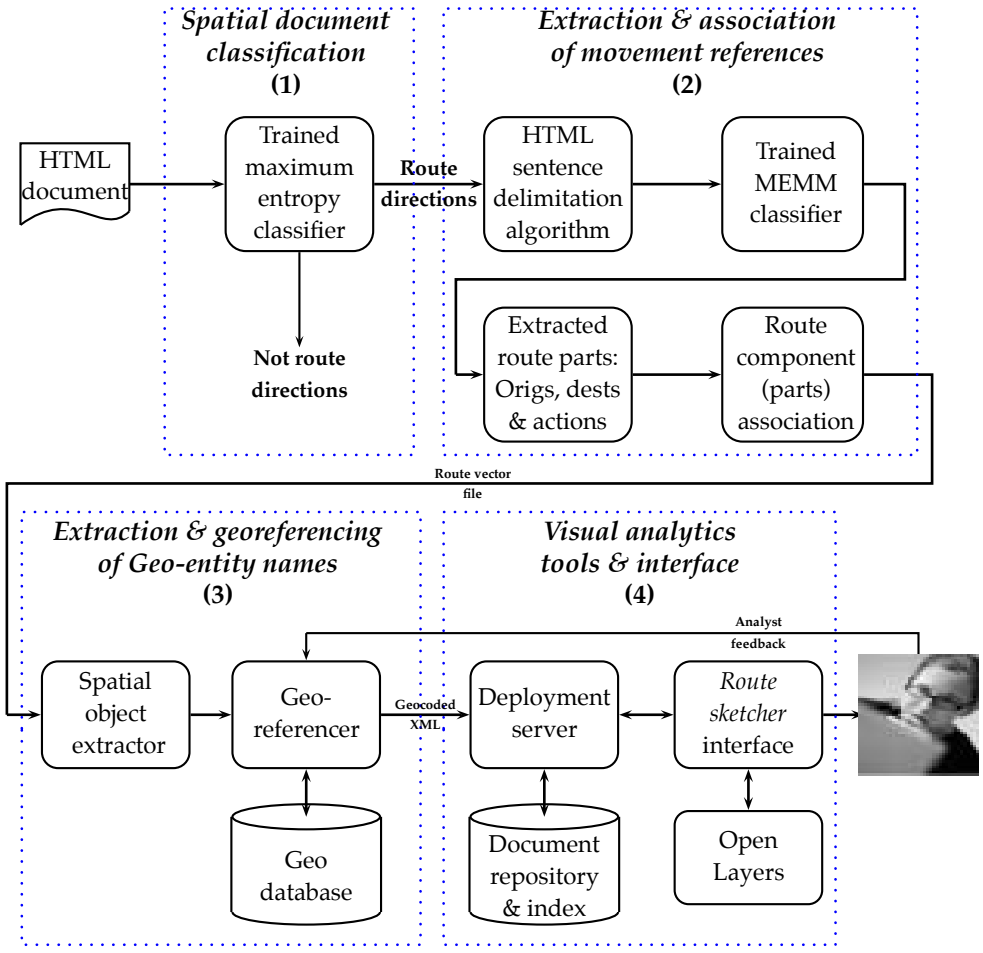

Figure 1: Set of sequential challenges (shown in dotted boxes) that must be solved to build a geographic information retrieval and decision-making system to contextualize and interpret geographic movement references. This figure also illustrates the architecture of the GeoCAM system.

A visual analytics approach to address such challenges is to utilize computational data processing methods along with effective visual interfaces that are designed to support human reasoning and decision making [24]. We adopt this general approach here. We developed an iterative, human-in-the-loop system that does the bulk of the work computationally and relies on the analyst to assist the system in case of errors. We addressed each sub-challenge while developing a geovisual analytics environment for identification and examination of documents containing directions.

In the research presented here, we focus narrowly on human-generated written directions as our target category of statements about movements. However, the specific problem addressed here is a subset of the larger challenge to leverage the wealth of geographic

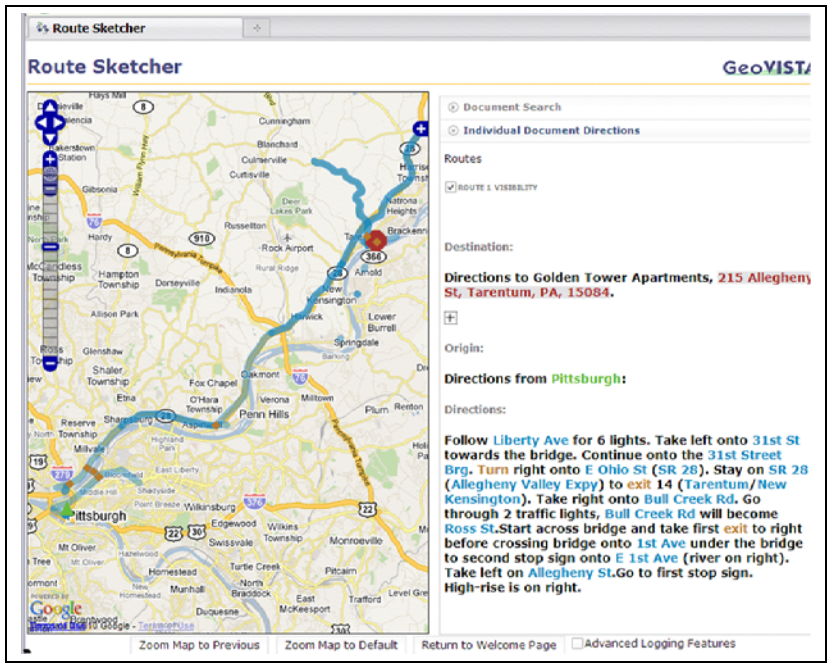

Figure 2: Example *route sketcher* depiction of directions. The interface displays the destination, origin, and directions extracted from a document submitted by the user and highlights key entities using color coding.

information that is currently inaccessible because it appears within unstructured or semi-structured documents rather than in neat tables or geographic databases. Examples of untapped sources of geographic information range from news stories are about events that are tied to locations, through crisis situation reports that address location-based events and actions, to scientific documents in fields such as ecology or urban planning that focus on characteristics of, or processes involving specific locations. Key parts of the approach presented can be generalized to the kinds of movement statements that will be found in such sources.

1.1 Geovisual analytics workspace

This article describes the GeoCAM (GEOvisual analytics workspace to Contextualize and interpret statements About Movement) system that demonstrates the effectiveness of utilizing a geovisual analytics approach for extracting descriptions of directions from text and displaying them on a map. Visual analytics tools integrated with computational algorithms provide an efficient framework for analyzing large document collections containing statements about movement. The workspace presented in this paper includes capabilities to iteratively run the computational algorithms discussed below; an integrated spatial database; a Lucene-based document index; a server to handle database-client communi-

cation; and a user interface module, *route sketcher*[1], that currently accepts geocoded places (such as origins, destinations, and route segments) from the route parsing and geocoding software and displays them on a web map. Once an individual route is loaded and displayed in the *route sketcher* user interface (UI), the user can mouse-over on routes, location names, and other landmarks found in directions (such as, schools, towns, or hotels) in one panel. The corresponding geocoded route features on the map are then highlighted in the adjacent panel (Figure 2). If a user is interested in a more detailed view of an area on the map, Google Street View is available through a click on a map feature. Using the ability to highlight listed directions and visually inspect the cartographic display of the features, users can see the directions unfold on the map.

The analyst can use *route sketcher* to assess the computational output and provide feedback to the system about it (e.g., the ability to allow the analyst to tell the system to restrict its solution to a particular US state, after identifying that there is confusion between places in different states in the initial output). In addition to processing and display of single directions, *route sketcher* supports querying of direction repositories. Consider the scenario where an analyst has a large set of direction documents available to her. She wants to search the documents for all directions that include mention of schools, to support research on how landmarks are used in direction statements. She can do a query on "school" to generate a map of all directions in the database that mention a school and then drill down to any particular directions. Or, she could query for a specific named school if she was trying to determine what business destinations might be inconvenienced by a school homecoming parade that closes particular streets.

While the GeoCAM system as presented focuses only on direction documents, the design is general in nature and we believe it is extensible to other types of documents containing more general descriptions of movement. Here, we focus on direction documents because they have rich spatial content, while having enough commonalities to make production of an end-to-end system more feasible.

1.2 Overview and contributions

In developing the GeoCAM system, numerous complex challenges were encountered that have been (partially) addressed. Specifically:

1. We show that supervised machine learning-based classifiers can identify and categorize documents as mentioned above with reasonably high accuracy. The results demonstrate that documents can be reliably classified based on mode of transportation when documents use a single mode of transportation throughout. The reliability of classifying documents based on the mode of transportation decreases when multiple modes of transportation have been used in the same document.

2. We have developed tools and algorithms to identify origins and route actions with high accuracy. The identification of destinations remains a challenging problem where we have achieved some success. Detailed experimental results are provided in Section 4.

3. We have developed tools and algorithms to georeference the extracted spatial movement patterns. Specifically, named-entity extraction and regular expressions were

[1] A video detailing the various functionality of the *route sketcher* interface can be found on YouTube, at http://youtu.be/lih_raIph6w.

used to generate a list of locations and landmarks (i.e., place names) from the extracted movement patterns. A novel spatial georeferencing algorithm was developed that utilizes multiple, distinct place names and spatial reasoning techniques to disambiguate and generate the most likely spatial pattern (i.e., movement pattern on a digital map).

In addition, the work provides two contributions related to the geovisual analytics interface:

4. GeoCAM provides an analyst with an interface to process and map new direction documents and/or search for and map existing direction documents. Specifically, a web-interface was designed and developed that allows an end-user to input a document of interest to the system and see the route interpreted by the system on the map.
5. GeoCAM enables an analyst to identify errors in the automatic direction processing and provide feedback to constrain the system to generate improved results.

We believe our system is the first to integrate the use of machine learning-based direction document classification; extraction of direction-related named entities; and the novel visual analytic front-end that allows the end-user to interact with human-generated descriptions of motion and examine the movement on a map. Our visual analytics contribution lies in the integration of computational methods with a visual interface, which allows the analyst to interpret results of the computational methods and to help us as developers of the computational tools refine the tools. While Drymonas et al. [12] present a technique for extracting and visualizing routes from travel documents, their technique focuses on using named-entity recognition to extract place names from text, geocode place names using Google API, and generate a valid path between two disconnected place names using a shortest path algorithm and Google street network data. Their method, thus, does not process movement statements; it estimates possible movement based on a sequence of discrete points. It also relies on input from a well-structured travel guide. It can be argued that such documents (documents describing walking tours in a city, e.g., Amsterdam, from a travel guide) contain landmarks that are comparatively easier to geocode than landmarks in route directions. Lastly, while they generate result maps, the approach does not include an interactive visual interface that supports human-in-the-loop refinement of results.

1.3 Organization

The rest of the paper is organized as follows. Section 2 discusses related work. The system architecture and a broad overview of the GeoCAM system is presented in Section 3. Sections 3.1–3.4 discuss the spatial document classifier, the route component recognition and association, place name extraction and georeferencing, and the *route sketcher* components of the system in detail, respectively. Section 4 outlines some results of our empirical evaluation. Future work is discussed in Section 5. Lastly, we conclude in Section 6.

2 Related work

Our work is informed by an array of recent developments in visual analytics. In particular, we draw from a fairly long history of text visualization and analyses. For a recent

overview, see [43]. Our work relates to recent work by Koch et al. [26] on patent search and by Keim and Oelke [22] on document analysis. However, the focus in the first phase of this research has been primarily on the challenge of extracting directions statements from documents [19, 41], providing visual access to the results; and improving the accuracy of the geo-location with the help of a human analyst. Input to interface design and interaction capabilities in the current *route sketcher* leverages recent work by Tomaszewski [45], and our own past work on complementary systems [32]. Health GeoJunction [32] is a geo-visual analytics-enabled web-based system that allows researchers to quickly locate scientific documents using geography, time, and themes. Similarly, HealthMap [7] is a real-time web application that collects disease outbreak data and allows users to visualize such data using geography, time, and infectious disease agent. Lastly, VisGets [11] are interactive visualizations that allow users to quickly formulate complex queries that combine spatial, temporal, and topical filters.

Substantial research has been performed on georeferencing named spatial entities from text (for an evaluation see [44]). Georeferencing road names and landmarks from descriptions of motion is a somewhat different problem because there are typically more database entries to chose among as a match for any particular road or street name (e.g., "1st. Ave," "Main Street," "Smith Road"). Directions have an abundance of road names in sequence and the sequential information can be used to improve the location name disambiguation. We discuss our method for georeferencing extracted location names in Section 3.3. Jones et al. [21] discuss the various challenges in geographic text retrieval. TEGUS [4] is a system for mining geospatial paths from natural language descriptions using language processing, GIS databases, and graph-based path finding algorithm. Kornai [27] evaluates geographic information retrieval using the Geo-CLEF 2005 task. Hollenstein et al. [18] explore georeferencing vague landmark descriptions such as "Downtown" by harvesting user-generated geocode and tag metadata over a large collection of Flickr images.

Unlike traditional classification models that make independence assumptions, such as naïve Bayes [30] and maximum entropy [39], sequence labeling methods exploit the dependence among the objects. These models assume that the instances to be labeled have sequential dependencies. Such methods include hidden Markov models (HMMs) [13], maximum entropy Markov models (MEMMs) [34], and conditional random fields (CRFs) [29]. HMMs model the joint probability distribution $P(y, x)$ where x represents the observed features of the objects and y represents the classes or labels of x we wish to predict. MEMMs combine the idea of HMMs and maximum entropy (MaxEnt). CRFs directly model the conditional distribution $P(y|x)$.

3 System architecture

The set of advances and capabilities outlined earlier have been integrated into a working prototype. The components of the prototype system and their interrelations are detailed in Figure 1. The overall system can be broadly decomposed into four modules namely, *spatial document classification*, *route component recognition and association*, *geographic entity extraction and georeferencing*, and the *route sketcher workspace interface* which are described in the following sections.

3.1 Spatial document classification

This module consists of several machine learning-based supervised classification algorithms to identify documents that have descriptions of motion in them. A cascade of classifiers built on binary classifiers for the first-level in the taxonomy (Figure 3) has been used. For documents containing route directions, the cascade identifies the mode of transportation. Once a document is determined to contain spatial information, the URL and the actual contents of the web page are then placed in a database for future reference.

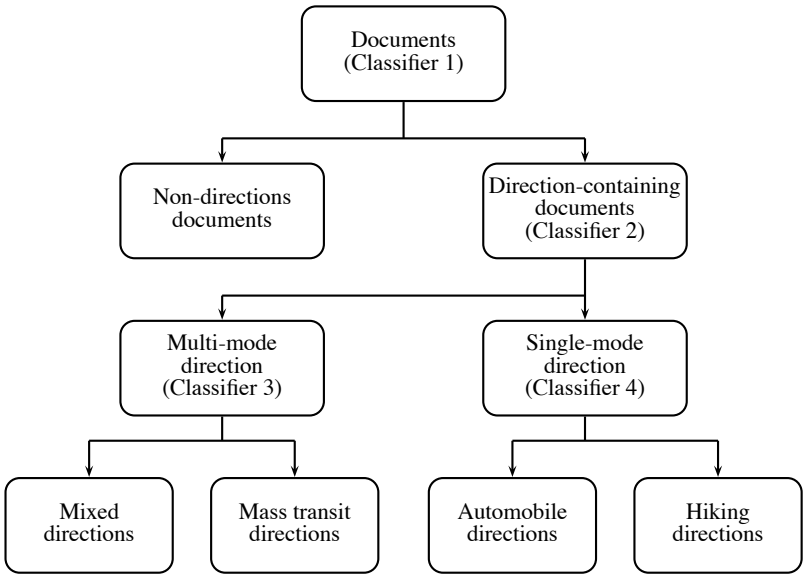

Figure 3: The spatial taxonomy describing the classes of relevant spatial information contained in digital documents.

3.1.1 Document classes

We observed that the Web typically exhibits direction-containing documents that can be classified into the following classes based on the mode of transportation:

1. automobile (or driving) directions,
2. mass-transit (such as subway, airline and bus) directions,
3. hiking/walking directions, and
4. mixed directions (multiple types of mode of transport in a single document)[2].

[2]Note that mass-transit documents typically also contain subway and bus directions within the same document.

Note that mixed directions documents are those documents that contain at least two of the other three types of directions (i.e., automobile, mass-transit, and hiking/walking) documents. In addition to these four classes, our taxonomy contains four other classes, as illustrated in Figure 3, namely:

1. *Documents* (whether they contain directions or not);

2. *Direction-containing documents*: All documents containing directions whether "automobile," "mass-transit," "hiking," or "mixed" direction documents are instances of this class.

3. *Multi-mode direction documents*: This class represents the set of documents that contain descriptions of motion using multiple modes of transportation. "Mixed" and "mass-transit" direction documents are instances of this class. Specifically, mass-transit documents were placed in this class since it was observed that they always contain at least two or more types of mass-transit modes of transport.

4. *Single-mode direction documents*: This class represents the set of documents that contain descriptions of motion using only a single mode of transportation. "Automobile" or "hiking" direction documents are instances of this class.

The relationships between classes, illustrated in Figure 3, were utilized to build multi-stage binary classifiers. All leaf nodes represent a document type while the edges represent the relationship "subClassOf." The features of Figure 3 also illustrate the multi-class spatial document classifier that is built using four binary classifiers.

3.1.2 Basic models and feature sets

To perform document classification, text must be extracted from documents before any processing can take place. Thus, in the first step, text was extracted from the HTML document. Then, term appearance (i.e., appearance of words) in each document was used as the set of features. Maximum entropy [20, 39], naïve Bayes [10, 30], and decision tree [3, 6] classifiers were utilized to categorize documents based on the taxonomy illustrated in Figure 3.

It was expected that the frequency of some words in direction documents would be somewhat different from their frequency in non-route direction documents. Within route direction documents there are differences based on, say, the type of transportation used. For example, mass-transit directions may contain information describing subway stations or bus terminals and the associated train/bus routes that must be taken in order for a user to arrive at a destination from a specific origin. Such documents will not contain phrases associated with road directions such as "take a left" or "continue on Interstate." Thus, documents containing different route direction information should contain a number of different words in addition to common words. Together, these phrases can be used to tag these documents to a spatial ontology using current machine learning algorithms. Furthermore, traditional stop words play an important role in spatial information content. For example, "take," "onto," and "at" are essential in automobile driving instructions. The presence or absence of these word features was utilized in all machine learning techniques that follow.

3.1.3 Multi-stage binary classifier

A four-stage binary classification scheme was implemented as shown in Figure 3. The four-stage binary classification scheme can take as input any document and classify it as "non-direction containing," "mixed," "automobile," "mass-transit," or "hiking." Experimental evaluation of the classification scheme is summarized in Section 4.3.

3.2 Route component recognition and association

This module consists of supervised machine learning algorithms that automatically tag route components, namely origins, destinations, and route actions, at a sentence or phrase level. Our preliminary research on detecting origins, destinations, and route actions has been published [50–52]. In Sections 3.2.1 and 3.2.2, we briefly summarize our previous work. In addition, once a document has been tagged at the sentence-level, the route-component-association module segments the document; uses regular expressions to re-label incorrectly tagged sentences; identifies parts of the document that are associated with each description of motion; and generates a description of motion using associated origins, destination, and route actions. Each associated origin, destination, and set of route actions from directions is stored as a description of motion within the same XML file and added to our database to allow later programs to return to this stage without having to repeat the classification. Section 3.2.3 gives the details.

Direction documents contain the following route components [1, 14, 28, 46]: *destination*, which is the place a person travels to, usually expressed as an address or the name of the place; *origin*, which is the place or area a person comes from, usually a city, an orientation (e.g., "From the North"), or a highway name; and *route actions*, which are a set of actions a person should follow in order to reach the destination from the origin. For example, in one document, the destination is the "Delaware Court Healthcare Center." The origin of one of the routes is "from Cleveland." The first few route action sentences given for this origin are "Take I-71 S toward COLUMBUS (117.2 miles)," and "Take the US-36/OH-37 exit." In addition, direction documents also contain information irrelevant to route directions, such as other textual information or advertisements. Such irrelevant textual information contained in a document is assigned to the class called *other* throughout the rest of this paper. A typical direction document usually contains one or a small number of destinations. Each destination is associated with several origins to suit travelers starting from different locations. For each origin-destination pair, a set of route actions are given to instruct travelers to reach the destination from the origin.

Route actions and other information are typically expressed in the form of a complete sentence or a stand-alone phrase. Both complete sentences and stand-alone phrases are referred to as "sentences" throughout this paper. Given the list of all sentences extracted from a document containing driving directions, our system seeks to classify the sentences into one of the four categories: destination, origin, route actions, or other, i.e., to recognize each route component at the sentence level. (For more details, see Section 3.2.1.) Furthermore, two additional techniques were developed for recognition of destinations at phrase level. The first approach attempts to recognize all mentions of the destination names, where candidate destination names are extracted using named-entity recognition. A binary classification scheme can then be used to classify the extracted named entities to be either "destination name" or "not destination name." (For more details, see Section 3.2.2.) The second approach developed extracts destination names by utilizing predefined language

patterns, e.g., "... is located on your right." Such language patterns are frequently used to represent destination addresses or landmarks. The landmarks, once extracted, are marked with confidence levels "high," "medium," and "low" based on the type of predefined language pattern used for extraction. (For more detail, please see Section 3.2.3.) Our system combines both approaches in tagging destination sentences.

3.2.1 Sentence-level route component classification

A list of sentences is first extracted from a document. These sentences are then ordered based on their position of appearance [52]. The machine learning features that were developed are applied to label these sentences as belonging to one of the following classes: destination, origin, route actions, and other (irrelevant) information. Six features were developed as follows:

1. *Bag-of-words features*: Each word/term in the entire set of documents is used as a feature. The presence/absence of a term in a sentence will assign the value 1/0 to the feature of this particular sentence.
2. *Surficial features*: Surficial features capture the "shape" of a sentence, namely: length of a sentence, capitalization of terms, containing digits, etc.
3. *HTML features*: HTML features check if a sentence is in the title, heading, or link of a document.
4. *Language pattern features*: Regular expressions are used to match the sentences against predefined phrases or patterns, such as "directions to ..."
5. *Type noun dictionary feature*: Type noun dictionary features check if a sentence contains a term appearing in our predefined dictionary of type nouns, such as "hotel," "restaurant," etc.
6. *Window features*: Window features capture the characteristics of the surrounding sentences of one particular sentence.

After feature extraction, machine learning models were used to assign to each sentence a class label, such as destination or origin. Furthermore, sentences within a document are observed to have interdependencies. For instance, a destination is often followed by an origin, and an origin is often followed by a set of route actions. Two models that utilize such interdependencies (namely conditional random fields, CRFs [29], and maximum entropy Markov models, MEMMs [34]) were therefore evaluated and compared with two models that do not (naïve Bayes [30] and maximum entropy, MaxEnt [5, 39]). Experimental results show that CRFs and MEMMs outperform the other two. For more details see [50, 52].

3.2.2 Phrase-level destination name extraction and filtering

In addition to extracting route components at a sentence level, the problem of identifying the destination names at a phrase level was further studied (i.e., the names of the destination organizations, companies, and institutes). For example, in the following two sentences, "Levy and Droney is located in the building on the right" and "Driving Directions to Atlantic Country Club," the destination names are "Levy and Droney" and "Atlantic

Country Club," respectively. Such names, if extracted successfully, can be used to obtain the position of the destination by querying geographic databases, and further improve the geocoding quality. The goal is to recognize all mentions of the destination, and different names referring to the same destination, including variations and abbreviations. This problem was solved in two steps: destination name candidates extraction, and destination name filtering. A short-paper on this topic has been published at ACM SIGSPATIAL [51]. We summarize the paper below.

First, OpenCalais and a rule-based extraction method that relies on part-of-speech taggers were used to extract the destination name candidates. Then, a binary classifier was built to classify these entity names to be either "destination" or "non-destination." Various features were explored for the named entity classification task. The 8 feature sets that were explored are defined as follows:

1. *Word shape features*: Shape features were used to capture whether all letters in a word are capitalized or whether the word contains numbers or not.
2. *Language patterns*: The sentence containing the destination candidate is matched against predefined regular expressions, such as "directions to ..."
3. *HTML tag features*: Tag features capture relevance of a destination name candidate by checking whether the destination name is in the page title, heading, or link.
4. *Normalized name frequency feature*: The total number of name appearances is normalized by the length of a document. When this frequency exceeded a predefined threshold, a value of 1 was assigned for this feature; 0 otherwise.
5. *Type nouns features*: A dictionary of 127 type nouns most frequently used in describing destinations was built. If the destination name candidate contained a type noun from this list then a value of 1 was assigned to this feature, 0 otherwise.

In addition to the above features, it was observed that the destination is very often the primary subject of the hosting website. This observation lead to the following important features:

6. *URL registrant matching*: The Whois database [48] is queried to find the domain registrants of the URLs of each document. If the domain registrant name matches the destination name, a value of 1 was assigned to this feature, 0 otherwise.
7. *Phone book search*: When a phone number is detected in the direction document, an online phone book search [36] service is queried to find the registrant names of the phone numbers and is matched against the destination name candidate. A value of 1 was assigned to the feature if the destination name is matched, 0 otherwise.
8. *Normalized frequency in web page titles*: Other web pages on the same site as the direction document were crawled. The number of page titles containing the destination candidate was then computed. This number is then normalized by the total number of pages on the site. If the frequency of matches exceeded a threshold, a value of 1 was assigned to the feature, 0 otherwise.

Three classifiers, naïve Bayes, maximum entropy, and C4.5 decision tree [42], were trained and evaluated for the classification task. The detailed results are shown in Section 4.5.

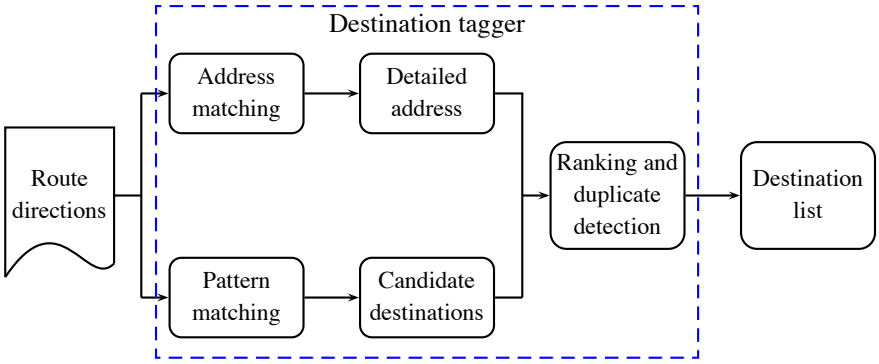

Figure 4: Workflow for *destination tagger* module.

3.2.3 Destination tagging

Experimental evaluation (see Section 4.4) showed that the machine learning-based methods discussed above have difficulty in identifying candidate destination sentences. As discussed previously, incorrect or incomplete destination detection is a significant drawback since when destinations are extracted correctly they work best as anchor points to pin down the spatial range of the route direction. Thus, to improve the precision of the destination detection a specialized regular-expression based component, *destination tagger*, was developed that matches, disambiguates, and ranks potential text references as destinations. A schematic of the *destination tagger* module workflow is shown in Figure 4.

The *destination tagger* module consists of two components namely *address matching* and *pattern matching* for effective extraction of destinations using regular expression-based extraction. Address matching aims to use a list of regular expressions tailored to capture exact addresses written in similar format to the postal address, e.g., "23585 NE Sandy blvd, Portland, OR 97060." A near-comprehensive list of regular expressions was written to capture the postal addresses that are written in various ways. For example, street names may include prefixes like "South," "S," or "S."; suffixes like "street," "st," "st."; the whole address may end with "US," "U.S.A.," five or 9 digit zip codes; or with state abbreviation or full name. A selection of the address matching regular expressions is shown in Table 1.

There are also cases where destinations are not expressed in the form of postal addresses. For example, a pattern such as "... is on your right" in the last sentence of a document signifies the presence of a destination landmark (e.g., "Empire state building") preceding this pattern and should therefore be considered as a candidate destination. It is common usage to include destination information in route-actions to represent arrival information. Parallel to address matching, the pattern matching component was used to handle potential destinations written in particular linguistic patterns. We examined our route-direction corpus and summarized these linguistic patterns (Table 2). However, such language patterns have higher language flexibility (compared to precisely crafted address patterns), and a lower confidence level since some sentences that match these patterns may not be destination landmarks that can be georeferenced (e.g.,"The destination is on the right"; here the phrase "destination" can not be georeferenced as is).

Regular expressions for address matching	Explanation	Sample matches
\d{1,5}((\s—\r—\b)([A-Z] {0,10} [a- z] {0, 15 })) + (\s—\r—\b)?, ?\s +((\s—\r—\b) ([A-Z] [a-z] {1,15})) +(\s—\r—\b)?,?\s +(?i: US_STATE)(\s—\r—,)?\d{5}(- \d{4})?(\s—\r—\b)?(,)?(\s —\n —\r)?(?i:Country_Name)?	Matches address starting with house number. It can take multiple street names, five or 9 digit zipcodes, either followed by country name or not.	20 S Genesee st, Fillmore, NY 14735, US 23585 NE Sandy blvd, Portland, OR 97060 2695 East Henrietta road, Henrietta, NY 14467 256 Cheyenne cir, Round Hill, NV 89448, US 8344 3rd street North, Oakdale, MN
\d{1,5} (\s— \r—\b)?, ?(?i: ORDINAL_GROUP) * ((\s— \r— \b)([A-Z] {0,10}[a-z] {0,15 }))+(\s— \r— \b)?, ?\s +((\s— \r— \b)([A-Z] [a-z] {1, 15}))+(\s— \r— \b)?, ?\s +(?i: US_STATE)(\s— \r—,— \.)? \d{5} (- \d{4}) ?(\s— \r— \b)?(,— \.)?(\s—\n— \r)? (?i:Country_Name)?	Matches ordinal number.	545 Biddle rd. Montoursville, PA 17754 100 E. Brinton avenue Trafford, PA 15085
\d{1,5}((\s—\r)([A-Z]{0,10}[a-z]{0, 15})(\.)?)*(\s—\r—\b)*,?\s*+(?i: APT—BLDG—DEPT—FL—HNGR—LOT—PIER—RM—S(LIP—PC—T(E—OP))—TRLR—UNIT—SUITE)((\s—\t—\.)*)(#?) (\s—\b—\r)?([A-Z]—\d{1,5})(\s—\r—\b—,)*([A-Z][a-z]{1,15})*(\s—\r—\b)*([A-Z][a-z]{1,15})*(\s—\r— \b)?,?(\s—\r—\b)? (?i:US_STATE),?(\s—\r—,—\.) \d{5 }(-\d{ 4})?(\s— \r— \b)?(,—\.)?(\s— \n— \r)? (?i:Country_Name)?	Match apartment addresses	123 Main st., apt 420 Anywhere, PA 12345, US John Doe 555 Blank ave. Apt. 555 Dodge City, PA, 55555 123 Elm St., #123, Dodge city, PA, 55555 236-4565 High Park ave Toronto, ON Canada

Table 1: A selection of regular expressions for address matching in *destination tagger*, with explanation and sample matches.

In order to improve the readability of regular expressions, the *destination tagger* component reads a set of predefined regular expression sets, such as "NUMBER_GROUP" which is defined as the following: (a) harf\sa; (b) half\sa; (c) \d+; (d) \d+\/\d+; (e) one; (f) two; (g) three; (h) (one—a)\sthird; (i); (one—a)\squarter; (j) (one—a)\shalf; etc. Then the regular expression will be constructed when being called. For example, a new regular expression "approximately\s[NUMBER_GROUP]\smiles" can be written with a high inclusive matching pattern and a much better readability. This coding schema allows for creating complex regular expressions on the fly without sacrificing readability of the code. Country_Name, US_STATE, Ordinal_Number, Link_Verb are a few of the predefined regular expression groups that are used in *destination tagger* (see Tables 1 and 2).

Together, the detailed address extracted from the address matching process and candidate destination from the pattern matching process can produce reinforcing and conflicting results. To solve conflicts, a ranking schema was applied that assigns a confidence level to every pattern and its matched text. Our system assigns different confidence levels of high, medium, and low to candidate destinations extracted. These confidence levels are

Regular Expression	Confidence Level
a) You\swill\ssee\s.*?\son\s(the)?\sDIRECTION_TERM b) Directions\sto\s c) Arrive\sat\s d) is\slocated\sat\s e) \sLINK_VERB\slocated\sat\sthe\sintersection\sof f) \sLINK_VERB\sDISTANCE_PHRASE\sDIRECTION_TERM\sof\s (that—the—this)\sintersection g) \sLINK_VERB\slocated\sDISTANCE_PHRASE\ sDIRECTION_TERM\sof h) \sis\sthe\sbuilding\sin\sfront\sof\syou	Medium
a) End\sat\s b) \sLINK_VERB\son\sthe\sDIRECTION_TERM\sedge\sof\s c) \sLINK_VERB\sright\soff\sof\s d) \sLINK_VERB\sjust\sDIRECTION_TERM\sof	LOW

Table 2: Selected regular expressions for pattern matching in *destination tagger*, with different confidence level.

based on the type of regular expression that extracts the candidate destination (e.g., high confidence is assigned to results extracted from address matching and low and medium confidence to results from pattern matching). Candidate destinations from the address matching and pattern matching extractors are then combined. During this step duplicate candidate destinations are eliminated using exact string matching. Furthermore, approximate string matching [38] is applied to the candidate destinations to detect duplicates. A ranked destination list is finally generated where the confidence level of extraction for a candidate destination is used as the ranking score.

3.3 Geographic entity name extraction and georeferencing

This module consists of algorithms to extract geographic entity names (geo-entity names) mentioned in the descriptions of motion, and to georeference the extracted geo-entity names. The overall architecture of this module is illustrated in Figure 5. It consists of two sub-modules, namely the geographic entity name extractor (Section 3.3.1) and the georeferencer (Section 3.3.2).

3.3.1 Geographic entity name extractor

This sub-module extracts a candidate set of geo-entity names from the text. *Geo-entity names* are textual references to landmark and/or location names (e.g., building names, roads, cities). These may have geographic coordinates that need to be inferred during the georeferencing step. Note that the geo-entity name extractor only extracts candidate text references. To achieve this goal, the XML file generated by the route component recognition and association module is first ingested by this module to generate a list of probable geo-entity names (locations, like city, town names, and/or landmarks, like street names, building names). Geo-entity names can typically be divided into either regional-level or local-level objects. While named-entity recognition strategies can be applied effectively to extract regional geo-entity names, such as city, state, or country names, these strategies are not as

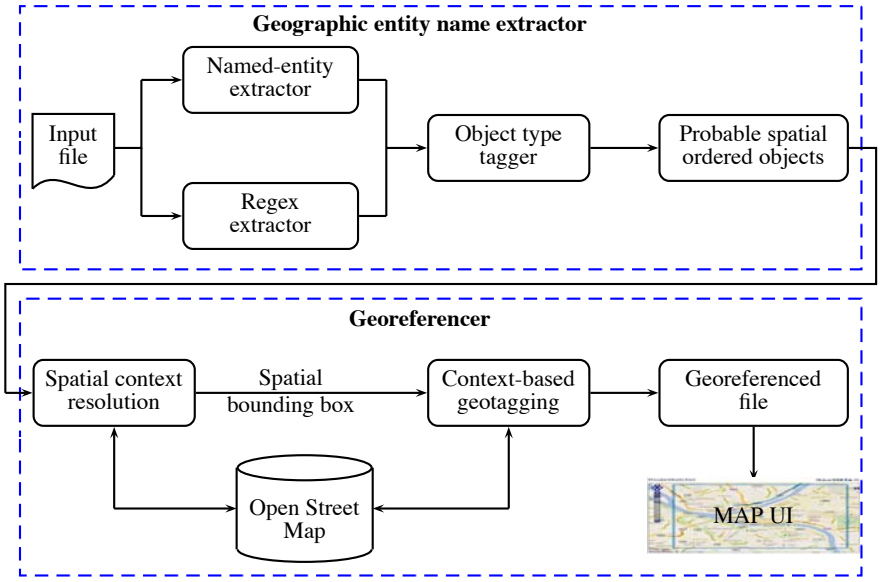

Figure 5: Geographic entity name extraction and georeferencing module architecture.

effective in extracting local-level geo-entity names, such as street names. Furthermore, regular expression-based extraction strategies can be applied for effective extraction of local-level geo-entity names since such objects are typically expressed with a finite number of consistent suffixes (e.g., "street," "st.," "ave.," "blvd."). Since effective georeferencing of the possible movement pattern requires extraction of both local-level and regional-level geo-entity names with high accuracy, the geographic entity name extractor consists of two extraction strategies, as illustrated in Figure 5:

Regular expressions	**Example sentences**	**Extracted geo-entity**
.* turn\s+(?:left|right)?\s(?:onto|to|on |at)\s+**((?:\w*\s)*\s(?i:st|str|street| rd |road|ave|avenue)**.*	turn left onto Main st. turn right on Madison ave turn onto Wisconsin ave	Main st Madison ave Wisconsin ave
.*(?:proceed|continue)\s+(?:along|into| past|on)\s+**((?:\w*\s)*\s(?i:st|str| street|rd|road|ave|avenue)**.*	proceed along College avenue	College avenue
.*exit\s+(left|right)?\s*(at|onto|to)\s+ **((?:\w*\s)*\s(?i:st|str|street|rd| road |ave|avenue)**.*	exit right onto KING rd	KING rd

Table 3: A subset of regular expressions used for extraction of local-level geo-entity names. The capturing groups in bold in RegEx are the candidate geo-entity names extracted.

1. *Named-entity extractor*: ANNIE, the named-entity recognition toolkit in GATE [8], modified with custom-developed rule sets was used to extract regional-level geo-

entity names. This extractor uses part-of-speech tagging and gazetteer-based tagging. The tool is effective in extracting regional-level geo-entity names such as city, state, and country names. In addition, each extracted entity is assigned one of four types: city, state/province, country, or location. The location type is typically assigned to a geo-entity name if the tagging determines that:

(a) The named-entity is a location but the extractor unable to decide whether this entity is a city, state/province, or country; or

(b) The named-entity is a location but is not a city, state, or country name, e.g., "Lake Michigan."

2. *Regular expression (regex) extractor*: A regular-expression-based extractor was also developed that utilizes a customized set of regular expressions developed using language analysis of documents with spatial entities. A subset of regular expression patterns that were utilized to extract local-level geo-entity names from text is shown in Table 3.

S. No.	Type of geo-entity name	Example	Geo-entity name type assigned	Strategy used
1	Regional-level	Pittsburgh	City	NER
2	Regional-level	Pennsylvania	State/Province	NER
3	Regional-level	India, US	Country	NER
4	Regional-level	Lake Michigan, Appalachian mountains	Location	NER
5	Local-level	Main st, Main street	Street	Regex
6	Local-level	1st avenue, Side ave	Avenue	Regex
7	Local-level	Empire state building, Harvard university	Location	Regex

Table 4: Some examples and types of geo-entity names extracted using a combination of named entity recognition (NER) and regular expression extraction (regex).

The *geo-entity name extractor* also captures the position information of each geo-entity name in text. Geo-entity names extracted using both named-entity and regular expression extraction strategies are then accumulated into a list and ordered using this position information. Following this, each geo-entity name extracted using regular expressions is then classified using a type of landmark by using suffix matching. For example, "Main st" or "Main street" are assigned type "street." Table 4 illustrates some examples of extracted entities, the location type assigned, and the type of extraction strategy used. Table 5 illustrates a set of geo-entity names extracted from a document being processed.

3.3.2 Georeferencer

This sub-module utilizes a candidate set of extracted geo-entity names (output of the *geo-entity name extractor* module) and a spatial database to contextualize and georeference the directions. Our spatial reasoning-based georeferencing algorithm utilizes geographic data from Open Street Map [15] for georeferencing. The Open Street Map database consists of

Processing order	Geo-entity name	Type	Geocounts	Order in text
1	Pittsburgh	city	1	1
2	**31 street**	**street**	**7**	**3**
3	Allegheny street	street	64	10
4	Bull creek road	road	83	6
5	E Ohio street	street	84	4
6	E 1st ave	avenue	136	9
7	Allegheny valley expressway	Expy	149	5
8	31st street	street	445	2
9	Ross st	street	710	7
10	1st ave	avenue	3104	8

Table 5: Set of geo-entity names, types, geocounts, and order extracted from a sample document. The second row is an incorrect geo-entity name that was present in the document (31 street instead of 31st street) most likely due to the author's error. Zero count geo-entity names have been dropped.

Table name	Type of geographic features represented	Examples	Num. of records
Point	Point data representing various objects such as airports, buildings, cities, and unnamed points etc.	Mys Shmidta airport, Matei police post, Pittsburgh	14.5M
Line	Roads, boundaries of various objects, census areas	US exclusive economic zone, Alaska, Boce Bay road, Main street, Chatham island	38.4M
Road	Roads, boundaries of various objects, census areas	North slope borough, Nome Census Area, Alaska, Middle road north	3.76M
Polygon	Boundaries of various objects such as countries, resorts, airports, islands, reefs, rivers etc.	Te Awanui island, Lake Marakapia, Honolulu, United States of America, Yukon River	26.8M

Table 6: Open Street Map database tables.

four main tables, which are described in Table 6. Algorithm 1 provides the pseudo code for the spatial reasoning-based georeferencing algorithm.

The ordered list of candidate geo-entity names from text (e.g., "Main street") is the input to the georeferencing module. Note that order in text represents the relative order in which the geo-entity name appears in a particular verbal description. Since each candidate geo-entity name may refer to numerous unique geographic coordinates, a robust strategy for

Feature Name	Number of Unique Segments
Main st/street	17632
1st ave	3104
Interstate 80/I80/I-80	2908

Table 7: Unique geographic references for some common geo-entity names from the Open Street Map database.

Algorithm 1 Georeferencer

Input: L: a list of geo-entity names, T: a list of spatial types
DB: Open Street Map database
Distance: side of bounding box used for context generation
Output: = *G*: a list of georeferenced geo-entity names
$G \leftarrow 0$, List $bbox \leftarrow null$, *Geocounts* $\leftarrow 0$, $i \leftarrow 1$, $L' \leftarrow 0$, $T' \leftarrow 0$
for geo-entity name $l \in L$ with type $t \in T$ **do**
  $Geocounts(i) \leftarrow computeGeocounts(l, t, DB)$; $i \leftarrow i + 1$;
end for
$L' \leftarrow sortPlaceNameList(L, Geocounts)$;{Sort geo-entity names based on geocounts}
$T' \leftarrow sortPlaceNameList(T, Geocounts)$; {Ensure the types match sorted list}
for i in 1 to $size(L') - 2$ **do**
  for j in 1 to $size(L') - 1$ **do**
    for k in 1 to $size(L')$ **do**
      $bbox \leftarrow getListOfBndingBoxes(L'(i), L'(j), L'(k), T'(i), T'(j), T'(k), Distance)$;
      if $size(bbox) > 1$ **then**
        if $doAllBoundingBoxesTouch(bbox)$ **then**
          $bbox \leftarrow bbox(1)$; $break$;
        end if
      else if $size(bbox) = 1$ **then**
        $break$;
      end if
    end for
  end for
end for
if $bbox$ not $null$ **then**
  for $i \in 1$ to $size(L')$ **do**
    $G \leftarrow addSpatialUnitsWithinBoundingBox(bbox, L'(i), T'(i), DB)$;
    $bbox \leftarrow expandBoundingBox(G, bbox, DB)$;
  end for
else
  $G \leftarrow null$;
end if

georeferencing the movement pattern must be developed. For example, there is only one composite reference to Interstate 80 in the real-world and in Open Street Map. However, Interstate 80 is represented by 2908 unique segments in the Open Street Map database to ensure that the coordinates for Interstate 80 are represented with high precision. We refer to geocounts or unique segments as the number of times a geo-entity name appears in a geographic database being searched. For instance, the geocounts for "Interstate 80" is 2908. Each of these unique segments gives the geographic coordinates for a small unique portion of Interstate 80. These 2908 unique segments taken together represent the complete geographic information for Interstate 80, which spans from New York city to San Francisco. Similarly, there are 17632 unique segments that represent "Main st" or "Main street," approximately 6000 unique "Main streets." Table 7 shows geographic references for some common geo-entity names obtained from the Open Street Map database.

First, the georeferencer obtains the unique geographic references in the geographic database for each geo-entity name. The Open Street Map database is thus first queried to establish the number of occurrences in the real-world (or sub-region specified by the user) matching each of these extracted geo-entity names (entities). Some entities may return a zero count and are dropped from further processing. This may happen due to two reasons: the regular expressions may incorrectly extract entity names resulting in an entity name that is non-existent; or a correctly extracted entity may not appear in the geographic database as it has not been tagged by the Open Street Map community yet (or may not occur within the region specified).

At this point, the system has identified landmark/location names (geo-entity names from text, i.e., output of the geo-entity name extractor) and a corresponding count of the number of geo-entity names in the real-world that match these names. There are several strategies to disambiguate geo-entity names and georeference the movement pattern. The first strategy is to use only the geo-entity names having a unique geographic reference (such as a specific destination, city name etc.) to construct a bounding box (geographic/spatial context). Other non-unique entities can be then disambiguated contingent on falling within this bounding box. However, this strategy, while easy to implement, has inherent disadvantages. First, documents containing geo-entity names that correspond to unique geographic references are infrequent. It was found that only three out of 20 documents contained geo-entity names that had a unique geographic reference. Second, directions typically use the most common road names to represent movement patterns. Such road names typically have a high number of geographic references in geographic databases. For example, approximately 25% of streets and avenues in Open Street Map have at least five or more geographic references. Last, such a strategy ties the system's accuracy of finding the correct spatial context to that of the unique geo-entity name being used to generate the bounding box. In these cases, an incorrectly extracted geo-entity name can generate an incorrect bounding box, which in turn results in an incorrect map or failure to find any solution. For such a strategy to be effective, all geo-entity names must be correctly extracted (100% precision and recall) and be present in that form in the geographic database, which is unreasonable to assume.

Spatial object type	**Number of unique names**	**Number of names with unique geographic reference**	**Number of names with non-unique (5 or greater) geographic references**
Street	11817	4758	7059 (2991)
Road	193515	102376	91139 (31875)
Avenue	8318	3316	5002 (2247)

Table 8: Descriptive table statistics from Open Street Map database showing the number of unique records based on exact string matching and number of unique and non-unique geographic references. Values in brackets represents the number of geo-entity names that have five or more geographic references.

As illustrated in Algorithm 1, the georeferencing module uses three distinct geo-entity names and their geo-counts to locate the smallest bounding box that encompasses these entities. Specifically, three distinct entities with the lowest geo-counts are first selected. The georeferencer then attempts to find a list of bounding boxes (of side 5km) that contains at least one geographic reference to each of these entities. Table 5 illustrates the processing

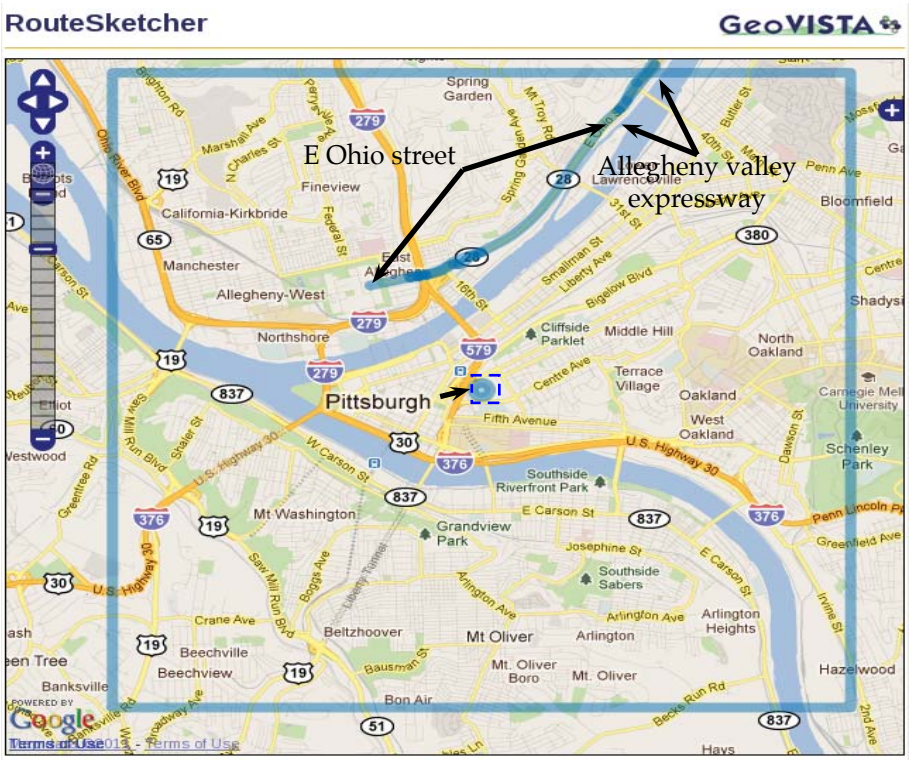

Figure 6: Spatial context (bounding box) generated by the geo-entity name extractor and georeferencing module for example geo-entity names given in Table 5. The spatial context was generated by the three units "Pittsburgh" (blue dot), "E Ohio street" (blue street segments) and "Allegheny valley expressway" (blue street segment). For perspective, the geographic coordinates of the enclosing bounding box containing the three geo-entity names are also shown. The initial enclosing bounding box is generated using the geo-entity name that has the lowest geocount of the three geo-entity names, i.e., "Pittsburgh" in this case.

order for a document being processed by the georeferencer. For example, the algorithm first chooses "Pittsburgh," "31 street," and "Allegheny street" and tries to find a square bounding box of side 5km from the geographic database that includes at least one instance of these three objects. If only one bounding box is found, the spatial context is resolved and the georeferencer utilizes this context for resolving other geo-entity names and generating the movement pattern. If multiple bounding boxes are found (most likely for large cities where there may be similar road names in different boroughs) and they all intersect, the first is selected to represent a starting context. If bounding boxes do not intersect, that iteration is rejected. If no bounding box is found or the iteration is rejected, the georeferencer keeps trying sets of three entities until an acceptable bounding box is found.

Therefore, a second strategy was implemented to generate the geographic context. Rather than using just a single unique geo-entity name and its geographic reference, the strategy utilizes spatial reasoning across multiple entities and their geographic references. In addition, this strategy also minimizes the computational complexity of geographic con-

text generation by utilizing the uncommon geo-entity name. Specifically, those geo-entity names that have low geographic counts are used first in geographic context computation. The spatial-reasoning based georeferencing algorithm utilizes triangulation of three distinct spatial entities in deciding the spatial context. Proximate locations of multiple (three) uncommon geo-entity names are unlikely to occur in multiple locations. The algorithm utilizes three distinct geo-entity names (regional/local-level) and attempts to find a square bounding box of side 5km that contains all these three geo-entity names. The square bounding box of side 5km was specifically chosen as it provided the best trade-off between computational time complexity and the possibility of matching three distinct records. A larger bounding box contains a greater number of records that need to be checked. However, these algorithms can be easily modified to use a larger bounding box for rural movement patterns and a smaller bounding box for urban movement patterns. Furthermore, the classification techniques described in Section 3.1 can be used to determine the class of movement pattern and thus dynamically determine the bounding box size depending on document classification.

This strategy has two advantages over starting with a single, unique geo-entity name. First, an incorrect geo-entity name will result in a failed iteration in determining the spatial context. Thus, this strategy inherently treats even low geo-count entities with some amount of uncertainty. Second, such a strategy is more efficient in finding spatial context since the absence of some entities in a geographic database does not affect georeferencing as there are multiple combinations of entities that will result in the same spatial context (bounding box). Figure 6 illustrates an example spatial context generated by the georeferencing module. Input to the georeferencing module consists of geo-entity names illustrated in Table 5.

Once the spatial context is generated, other location name entities are then found in this context and added to the movement pattern. The spatial context is slowly expanded based on addition of each geographic reference to the movement pattern. For example, once the unique bounding box is found, only entities with geographic references that are within a small distance (i.e., 5km) from the spatial context are found and considered for addition to the movement pattern. Each geographic reference for an entity that is found and added to the movement pattern may also affect the spatial context. For example, adding a very long road (such as a segment of an Interstate of say 10km in length) should affect the spatial context (as the initial spatial context is a box of side 5km). In addition to the geographic reference for a found entity that is added to the movement pattern, the feature ID, primary, and secondary names of the entity are also stored in the movement pattern. The feature ID is the record ID assigned by Open Street Map to a geographic reference in the Open Street Map database. Thus, the spatial context is recomputed after addition of every entity to the movement pattern, allowing it to slowly expand to reflect the growing movement pattern. Once all entities are georeferenced, the movement pattern is generated and written to an XML file using GML (geography markup language) format.

Once the above steps are accomplished, the georeferenced geo-entity names are added to the route document in the main database. The storage of the coordinates of the geo-entity names and the coordinates of the route allows spatial queries to be made on the routes database.

Computational complexity: Finding the spatial context (or equivalently finding a bounding box containing three different geo-entity names) is the most computationally expensive step in georeferencing the movement pattern, i.e., function "getListOfBndingBoxes" in Al-

gorithm 1. If n is the number of geo-entity name records in a GIS database, then the number of bounding boxes can be found using a brute force algorithm in $O(n^3)$. However, the spatial context computation can be made more efficient using PostGIS spatial queries (e.g., contained in/intersection queries) and the GiST [16] index.

3.4 Route sketcher user interface

As introduced earlier, *route sketcher* is a Web 2.0 visual interface for displaying and analyzing computationally processed direction documents. The two main functional components include: processing a direction document on-the-fly to map and analyze that document; and querying a repository of preprocessed documents to find documents of interest (e.g., that include route segments that pass near a feature of interest). In the following subsections, we outline the *route sketcher* implementation and discuss the functionality of the system briefly.

3.4.1 Route sketcher implementation

The *route sketcher* UI client application is built exclusively using HTML and JavaScript technologies (with the exception of the Google Street View component) allowing the UI to be executed in any JavaScript-enabled web browser. The UI communicates with a server-side Java servlet that uses Java API calls to execute spatial document classifier, route component recognition and association, and geo-entity name extraction and georeferencing modules. All communications between the client UI and server side are through AJAX HTTP requests.

The mapping functionality of the *route sketcher* UI is built upon the Open Layers JavaScript mapping client. Open Layers allows integration of our tools with Google Maps for base-layer data. In addition, Open Layers allows for the overlay of geographic data drawn from a web feature service (WFS) request. Geographic features found in the movement pattern are queried from the Open Street Map database through a WFS request based upon the feature ID previously found by the georeferencing module. Once the requested data is returned to the client, it is drawn using scalable vector graphics (SVG) methods in the Open Layers client. Geographic data is drawn in the client using vector format to allow for browser events—such as, mouse-over, mouse-out, and mouse-click events—to be attached to the geographic features for highlighting and linking with the direction terms panel.

The Dojo toolkit was utilized for cross-browser JavaScript support enabling highlighting of tokens, drawing of menus, and pop-ups within the *route sketcher* UI. Lastly, the UI integrates Google Street View, which is a Google JavaScript API. Google Street View's 360 degree street-level images allow the *route sketcher* user to get a street-level view of the surroundings of their route directions.

3.4.2 On-the-fly processing and mapping

Analysts can use the *route sketcher* UI to visualize route directions in text documents. An analyst can either upload an HTML document of interest to the system or point the system to its URL. When a user activates the system to start processing the document, a local copy of the document is first made at the server. Once the HTML document is saved at the

server, the *route sketcher* UI uses AJAX requests to communicate with a Java web service allowing the UI to execute the appropriate modules on the server to classify, extract, and georeference the movement patterns contained within the document. Once processing is complete, the directions are displayed on the map.

The *route sketcher* interface (see Figure 2) consists of two views. The directions on the map are shown on the left view. Extracted and highlighted geo-entity names within text directions are shown on the right view. Mapped and highlighted geo-entity names consist of origins, destinations, cities, streets, and general geographic features. In the current version of the tool, the entire road segment is highlighted on the left panel. Using the intersections of the previous and next road segment in the directions, it should be straightforward to clip the entire road segment and show only the part of the road that was part of the direction. We intend to implement this in the future.

The mapped geographic movement patterns are drawn in a graphics layer overlaid upon a Google Maps base map allowing for direct user interaction to help assess matches between the text and the features. Standard linked brushing between map features and text entities is supported. Mousing over geo-entity names in the text view (right) allows the movement pattern to visually take shape (in the left view). For example, in Figure 2 the origin of Pittsburgh, PA, is highlighted (green triangle). Streets, geographic features, destinations, and so forth can be explored. An analyst can zoom into the map to a feature of interest using a control-click on the feature of interest.

An important feature of this research is the integration of computational methods with direct human interaction to address hypotheses about text interpretations. One feature that illustrates this combination is the ability for the analyst to override the system's initial geographic extent choice for the destination and cause the document to be reprocessed. Doing so changes the bounding box applied to filter candidate matches for geographic features. Analyst reporting can be done (within the interface) on the accuracy of the processed document to assist analysts in the future and to help improve the overall system.

3.4.3 Querying of document repository

Our repository of preprocessed documents is stored in a PostgreSQL relational database with the PostGIS extension. A Lucene text-based search of the repository returns a subset of matching document snippets along with each document destination plotted on the map (Figure 7).

The system allows for storage and retrieval of large corpora of directions documents and supports large scale analyses. Using a relational database to query a large set of text documents could result in poor performance. Therefore, the directions documents are indexed using the Lucene API. When the analyst inputs a text query to the system, the system sends such requests to the Lucene index to retrieve the matching documents. Document IDs and text snippets are returned as results, and the system then retrieves the actual document and geographic references from the database. This combination of Lucene indexing and relational databases allows for scalable solutions for large datasets.

Figure 7 shows the results of a search to find all documents that contain the term "library." In this example, the analyst has highlighted one particular set of directions for an Alexandria, VA, library branch. Clicking on the destination or document snippets drills down to the individual georeferenced documents as discussed in the previous section.

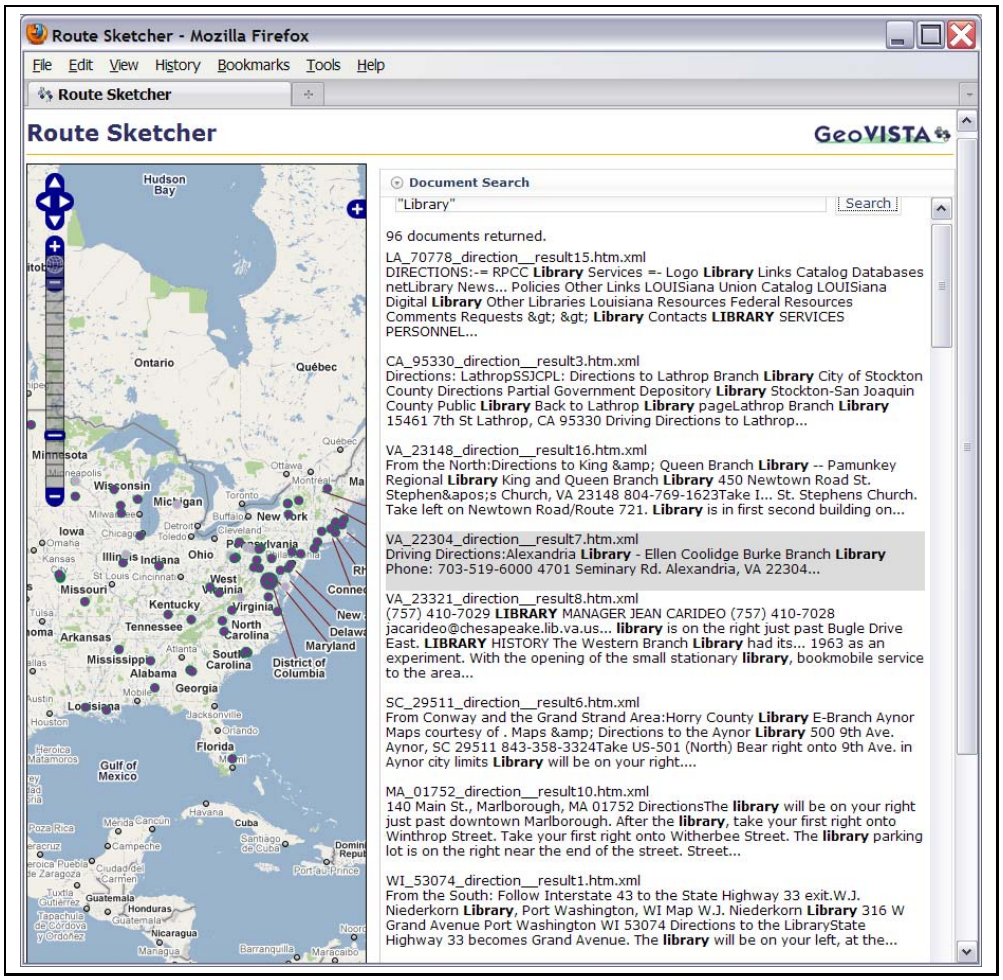

Figure 7: Repository search for the query "Library."

4 Experimental evaluation

In this section, we first introduce metrics for evaluating the various computational methods that we have developed. The analytic part of the system can be evaluated using manually generated "gold standards." In Section 4.2, we discuss our dataset generation and collection. In Sections 4.3–4.5, we provide a systematic evaluation of the text analytic components of the system. Then in Section 4.8 we describe some preliminary full-system evaluation results.

4.1 Evaluation metrics

We define *sensitivity* and *specificity*, which are statistical measures of the performance of a binary classification test. Consider a study evaluating a test that screens people for cancer.

Each person taking the test either has or does not have cancer. The outcome of the test can either be positive (predicts that person has cancer) or negative (predicts the person does not have cancer). Four results are possible:

1. true positive: a person having cancer diagnosed as having cancer,
2. true negative: a healthy person diagnosed as not having cancer,
3. false positive: a healthy person diagnosed as having cancer, and
4. false negative: a person having cancer diagnosed as not having cancer.

We can then define the following measures:

$$Sensitivity = Recall = \frac{\text{Number of true positives}}{\text{Number of true positives} + \text{Number of false negatives}}$$

$$Specificity = \frac{\text{Number of true negatives}}{\text{Number of false positives} + \text{Number of true negatives}}$$

$$Precision = PPV = \frac{\text{Number of true positives}}{\text{Number of false positives} + \text{Number of true positives}}$$

$$Accuracy = \frac{\text{Number of true positives and true negatives}}{All}$$

$$All = \text{Number of true positives} + \text{true negatives} + \text{false positives} + \text{false negatives}$$

$$\text{F1 score} = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

The F1 score is the harmonic mean of the precision and recall and has been widely used as an evaluation metric in information retrieval and machine learning. It captures the accuracy of the classification.

4.2 Directions data set for classification testing

Our experimental dataset contained the following types of documents:

1. automobile or route directions documents,
2. mass-transit directions documents,
3. hiking directions documents,
4. mixed directions documents, and
5. non-direction-containing documents.

A route directions document dataset of over 11,000 web pages was identified using the search results of the Yahoo! search engine. The search engine was queried with a set of carefully selected keywords such as "direction turn mile go," "turn mile follow take exit," etc. These terms are typically present in documents containing route directions. Each set of queries contains 4 to 7 keywords. Manual examination shows 96% of these documents contain route directions. Table 9 illustrates some sample queries and the number of unique documents obtained from the returned results page.

	Search queries	Number of documents
1	direction, turn, go, mile	986
2	direction, turn, left, right, exit	775
3	direction, turn, mile, take, exit	588

Table 9: Sample search queries and number of documents retrieved.

Similarly, the mass-transit and hiking directions document datasets were constructed by using selected keywords such as "subway bus directions train center" and "hiking directions trail walking difficulty" respectively. All documents were then manually inspected and labeled as either "hiking," "mass-transit," or "mixed directions" documents. The mass-transit document dataset has 240 documents and the hiking directions dataset has 271 documents. The mixed directions document set was constructed by choosing those documents that contained both mass-transit and route directions during the manual inspection of the route directions and mass-transit document sets discussed above. The final mixed document set contained 216 documents. As negative examples, i.e., documents that do not contain any movement references, 216 documents were randomly selected from a subset of 10,000 web pages obtained using a random sampling of the Web using the method proposed by Henzinger et al. [17].

4.3 Evaluation of spatial document classification

We evaluated the document classification algorithms as follows. Since the route-direction and non-direction-containing documents are very large in comparison to the other datasets, N documents are randomly chosen from each of the five datasets to ensure that there is no bias towards any specific dataset. Here N is the size of smallest dataset, i.e., size of the mixed direction document set. A traditional stratified k-fold cross validation [9, 37] scheme is used to sample the datasets into k-samples of equal size. For our experiments, we chose k=10. During each training-testing cycle, classification is performed using five classifiers: the maximum entropy [20, 39], decision tree [3, 6], naïve Bayes [10, 30], a stacked naïve Bayes and decision tree, and a stacked naïve Bayes and maximum entropy classifier. The classifiers were implemented using the MALLET [35] machine learning toolkit.

It was observed that the maximum entropy classifier performs best when identifying the class of directions that documents belong to (as shown in Figure 3). The maximum entropy-based classifier achieved 99% accuracy, sensitivity, and specificity for classifiers 1 and 4, 96% accuracy, 95% sensitivity, and 97% specificity for classifier 2 ("multi-mode" and "single-mode" direction classifier). However, for classifier 3 ("mass-transit" and "mixed" direction classifier), a stacked naïve Bayes plus maximum entropy classifier performed the best with 75% accuracy, 78% sensitivity, and 72% specificity. Generally, all classifiers performed extremely well when classifying "direction-containing" and "non-direction-containing" as well as "automobile" and "hiking" documents.

However, classifier 3 did not perform as well classifying "mass-transit" and "mixed" direction documents. The average accuracy rates for these documents were 72%, 69%, and 74% for the maximum entropy, decision tree, and naïve Bayes based classifiers. Stacked classifiers performed slightly better with the stacked naïve Bayes and maximum entropy classifier having the best accuracy rate of 75%. Similarly, the stacked naïve Bayes and decision tree classifier had an accuracy rate of 74%. Both stacked classifiers had higher

sensitivity rates in comparison to the other classifiers. The naïve Bayes technique had the best specificity of 75%. Thus, classification of documents containing spatial information requires a hybrid combination of classifiers in multi-stage binary configuration where classifier 3 uses a stacked naïve Bayes and maximum entropy classifier while the other classifiers use the maximum entropy classifier.

4.4 Evaluation of sentence-level route component recognition

As reported in [50], over 10,000 sentences were labeled from 100 documents manually. In order to compare different models, 10-fold cross validation was applied. For each model, the precision, recall, and F1 score was calculated for all four classes. As expected, the experiment results show that CRFs and MEMMs outperform naïve Bayes and maximum entropy. F1 scores for these models are presented in Table 10.

Classifier	Destination	Origin	Route Action	Other
Naïve Bayes	**0.56155**	0.66109	0.90568	0.81159
MaxEnt	0.51803	0.70324	0.91366	0.81667
CRF	0.46536	0.75446	**0.93298**	**0.83867**
MEMM	0.51373	**0.77559**	0.92778	0.83094

Table 10: F1 scores of four class labels for sentence-level route component classification. Best scores are highlighted in bold.

4.5 Evaluation of phrase-level destination name classification

A short-paper reporting our progress on phrase-level destination name recognition appears in ACM SIGSPATIAL 2011 [51]. In this subsection, we briefly report these results. 98 driving-direction documents were examined. 241 destination names present within these documents were manually labeled. These destination names include the different variations and abbreviations of the same destination name. For example, in one document, the destination name is "the Happy Berry farm." Other mentions of the same destination name in the document include "Happy Berry," "the farm," and "Happy Berry, Inc." All such text references are labeled as destinations. We then ran our algorithms, which resulted in 871 names being extracted that were classified as "not destination." 240 out of these 871 names were randomly selected. This sampling was performed in order to guarantee that the numbers of positive samples and negative samples are balanced, which ensures that the trained model does not suffer from training biases due to the skewed sample sizes. In addition, this ensures that high accuracy of classification due to good prediction of the majority class label is avoided. Three machine learning models were then evaluated namely: naïve Bayes (NB), maximum entropy (MaxEnt), and C4.5 decision tree (C4.5). All experiments used 10-fold cross validation. Table 11 shows the averaged results from 10-fold cross validation for phrase-level destination name classification. MaxEnt and C4.5 decision tree classifiers were found to perform well for this task. Table 11 illustrates phrase-level destination name classification results.

	Destination			Non-destination			Overall
	Precision	Recall	F1	Precision	Recall	F1	accuracy
NB	78.16%	75.61%	76.42%	77.06%	78.60%	77.38%	77.10%
MaxEnt	**85.50%**	74.60%	79.17%	77.61%	**86.32%**	**81.40%**	**80.63%**
C4.5	83.22%	**77.02%**	**79.55%**	**78.46%**	83.73%	80.33%	80.22%

Table 11: Phrase-level destination name classification results. The highest scores are highlighted in bold.

4.6 Evaluation of destination tagger

To evaluate the performance of the *destination tagger* module, a test set was generated that consisted of 190 real addresses selected manually from a spatially-stratified route direction corpus as developed in Xu et al. [49]. The test cases were in original formatting, since they were directly selected from the route direction's original web page, in order to cover the wide flexibility of human-written postal address. Furthermore, the 190 addresses cover 48 states in the continental US and District of Columbia, which ensures covering the address variety from the geographic perspective. Table 12 below shows the evaluation statistics with a selection of examples. The *destination tagger* yielded 88% recall in recognizing addresses. Because the regular expression patterns were designed to capture precise addresses, all recognized addresses are correct with no case of partial extraction.

	Count	Example test addresses
Addresses recognized	168	a) Northtowne plaza, 691 Naamans road, Claymont, DE 19703 b) 2301 C street SW, Auburn, WA 98001-7410 c) 39849 127th st. Columbia,SD 57433 d) 398 Chestnut street, Union, New Jersey 07083 e) For directions on GPS or Mapquest use our old address of 3841 Cobbler mountain road, Delaplane, VA 20144 f) End at Classic Bowl: 8530 Waukegan rd, Morton Grove, IL 60053, US
Addresses not recognized	22	a) 164 Chelsea street, PO Box 96 — South Royalton, VT 05068 b) Freeport center, building Z-15, Clearfield, UT 84016 c) W349 N5293 Lacy's ln, Okauchee, WI 53069 d) 10 Hartford avenue (Rt. 189) Granby, CT 06035

Table 12: Experimental evaluation of accuracy of the *destination tagger* module

As shown in Table 12, *destination tagger* can match addresses in flexible formats, including abbreviations for road name prefixes (e.g., "North," "N.") or suffixes (e.g., "St.," "st," "rd.," "ave."), numbers written in ordinal form or as Arabic numerals, full name or acronym of States ("New Jersey" or "NJ"). Extracting addresses from free-formed text is the key functionality of *destination tagger*. Addresses written in free-form text can be correctly extracted, which improves the input to the georeferencing module. For example, after processing with *destination tagger* the text "Arrive at 44955 Cherry hill road., Canton, MI 48188-1001, on the Left" results in the output: "44955 Cherry hill road., Canton, MI 48188-1001." The unrecognized addresses in Table 12 have unusual characters (e.g., "—" or "()") or lack house numbers or street names (e.g., "Freeport center, building Z-15, Clearfield, UT 84016"). These patterns are not included in the original set of regular expressions. However, with the extendable architecture of *destination tagger*, it is feasible to write new regular

expressions that match the unrecognized pattern and to add them to the existing set to improve the recall.

No.	Geo-entity name 1 (geo-counts)	Geo-entity name 2 (geo-counts)	Geo-entity name 3 (geo-counts)	Bounding box (km^2)	Reference city	Exec. time (ms)
1. (a)	Pittsburgh (1)	31st street bridge (2)	E Ohio st (84)	5	Pittsburgh	941
(b)	Pittsburgh (1)	31st street bridge (2)	E Ohio st (84)	10	Pittsburgh	4336
(c)	31st street Bridge (2)	Allegheny valley expy (149)	31st street (445)	5	Pittsburgh	3804
2. (a)	Phoenix (6)	E Van Buren st (27)	I-17 (168)	5	Phoenix	3310
(b)	Phoenix (6)	E Van Buren st (27)	I-17 (168)	10	Phoenix	4697
(c)	E Van Buren st (27)	I-17 (168)	North 7th st(597)	5	Phoenix	3041
3. (a)	Madison ave bridge (4)	Exterior st (8)	FDR dr (85)	5	NYC	2684
(b)	Madison ave bridge (4)	Exterior st (8)	FDR dr (85)	10	NYC	15482
(c)	E 138th st (7)	Exterior st (8)	FDR dr (85)	5	NYC	555

Table 13: Time required to compute the spatial context for three different movement patterns. Execution time reported is in milliseconds.

4.7 Georeferencing runtimes

Table 13 presents the execution times required to compute the spatial context for three different example movement patterns. All experiments were run on a PostgreSQL 8.4 database server with PostGIS 1.3.6 running on a dual Intel Xeon Server with 48GB RAM and a RAID5 array on four Seagate 7.2K RPM drives. The January 2011 daily dump of the Open Street Map database was downloaded and loaded using osm2pgsql. In addition to the default indexes, the Open Street Map text tags in the tables were indexed using the generalized inverted index (GIN). Other necessary columns were indexed using B-tree indexes. The three movement patterns were route directions in Pittsburgh, Phoenix, and New York City (NYC). Table 13 also illustrates that the georeferencing algorithm was able to compute the same route for a given route-directions input file by using different combinations of geo-entity names. For example, the route for the Pittsburgh example could be computed by using two different sets of geo-entity names. The first set of geo-entity names was "Pittsburgh," "31st street bridge," and "E Ohio st." The second set was "31st street bridge," "Allegheny valley expressway," and "31st street." For the two different sets of geo-entity names, the spatial context generated was different but geographically close. This illustrates the robust capabilities of using such an approach in georeferencing movement patterns. Increasing the bounding box size from $5km^2$ to $10km^2$ increased the execution time for all three examples. This can be attributed to the larger number of records

that must be checked against the larger bounding box. Using extremely large bounding box sizes (e.g., 250km^2) is not recommended since there is a larger likelihood of incorrect spatial context generation as geographically distant geo-entity names (e.g., geo-entity names from different but geographically close cities) can be used in spatial context generation. The execution time for generating the spatial context for New York city was greater due to the higher density of geo-entity names, which was expected.

4.8 Full system subjective evaluation

At this point, we do not have a systematic user-study to report. Metrics for measuring the ease-of-use and utility of our visual interface need to be designed to pursue a thorough evaluation of our front-end UI in the future. We have presented numerous demonstrations of the system for a range of visitors. The coauthors individually tested the full system, which incorporates all computational steps outlined above, to evaluate what works well and what needs improvement. Each individual tested the system using multiple sets of different web pages containing directions. The resulting maps were analyzed by the team. We report brief anecdotal evidence of how our team used the visual analytic system. We believe that the lessons learned are important and hence outline them here.

1. The search feature can be used to quickly search the database and retrieve interesting routes, such as "All routes passing through City X," "Directions to JFK (i.e., New York's John F. Kennedy airport)," etc.

2. In most cases, users could easily tell whether the georeferenced movement pattern displayed on the left panel was in the right general geographic location or not within a few seconds. The system generally does well at finding the correct geographic location for the tested route directions. Currently, errors are usually related to system limits in the ability to detect destinations for some directions (as discussed in the preceding sections) or to matching geo-entity names from text to their geographic references in geographic databases (multiple names for streets and highways cause street segments to be missed). We will continue to improve the georeferencing module of the system. However, when the system goes wrong, at least the analyst is not misled into believing that the geographic reference displayed is correct.

3. Color-coding the landmarks and other locations mentioned on the right pane helped the users identify the origins, destinations, and landmarks in the georeferenced movement pattern displayed more easily. Users often used the highlighting feature on the right-panel to mouse over a location name (geo-entity names) and to identify on the map in the left panel where the location was.

4. There were several cases where the system sketched multiple options for the same route. The analyst seeing the extracted textual description on the right-hand-side of the browser could easily point to the correct choice. A control-click allows the analyst to zoom in to see that sketch. We will add functionality to allow feedback by the analyst to identify the correct option and store that into the database. This will help to improve the system's ability to disambiguate among the multiple sketched movement patterns in the future.

In some cases, errors crept in because the system identified the wrong bounding box due to ambiguous landmarks mentioned in the web page. The georeferencing module in the

system identifies the bounding box and then tries to find the landmarks mentioned in the text within that bounding box. *Route sketcher* provides the analyst with an ability to skim through the route descriptions on the right-panel and to override the system choice of bounding box when needed by inputting a US state bounding box or a user defined bounding box. The system uses this feedback from the analyst to constrain the route sketched to arrive at the correct solution. The system then runs the georeferencing algorithm again using this new information to reprocess the directions and redisplay the new movement pattern on the map.

5 Future work

Future work on the GeoCAM system includes adding more spatial reasoning support to extract missing route segments from street databases, to clip route segments at turns, and to improve the spatial reasoning by defining the spatial bounding box extent in more intelligent ways for limiting spatial queries. One specific role for spatial reasoning methods is to address the challenge of joining street segments that are stored separately in the database (perhaps because some segments are tagged with their street name and others with a highway number). Beyond the problems of interpreting text statements correctly, the quality of spatial data limits the accuracy of the GeoCAM system. For example, Open Street Map [15] records have good coverage over developed countries but the street names do not follow consistent naming conventions (for example, "East Main Street" may also be labeled as "East Main St," "E Main St," "East Main Street," "E Main Street"). In these cases, it is difficult to find the entire street without the document having referred to every variation of the name.

The GeoCAM system currently lacks the functionality of clipping streets when retrieved segments extend beyond the actual route. A solution to this could be building a table that stores every intersection of streets present in Open Street Map [15] and utilizing spatial reasoning techniques to perform turn clipping. We will investigate this approach in future work.

6 Conclusions and outlook

In this paper, we show how to automatically process human-generated route directions and plot them on a map. A visual analytics environment, which combines machine learning and natural language processing methods with a geographic database, supports the extraction, interpretation, and mapping of geographic references in context, and the capture of tacit analyst knowledge to refine the results returned by the system. An end-to-end system was implemented and tested. We have assessed the computational methods for identifying documents containing routes, categorizing those with routes on the basis of mode of travel, and identifying and extracting origins, destinations, and route parts. Our empirical assessments demonstrate acceptable performance for the first two of these and reasonable performance on the third. Next steps in the research are to enhance methods for analyst feedback to the system and add additional spatial reasoning capabilities to cope with the many unique situations encountered with messy, real-world documents. In addition, after such improvements to our system, we will carry out formal usability testing to assess the entire system from a user perspective.

We believe that the same general infrastructure can be used to process directions given in a different language. However, this requires a parser for the language. Sometimes even geo-entity names are written differently in different languages , e.g., Spain/Espagne. Furthermore, spatial concepts such as "through" also vary across languages. Thus, a gazetteer that contains the location names in the target language is also a requirement. Once that is available, the named entity extractor, like GATE, can be used for multiple languages. For the classifiers, one would have to identify language features in the alternative language that can be used to identify destinations, origins, and route directions. Of course, the GUI would need to be altered to use a different language and perhaps alternative alphabet or character system.

Lastly, a key component of the work has involved creating an ontological characterization of movement [2, 24, 41, 49]. Building on this, we identified a need to account for regional differences and different languages. To this end, we are using tools developed (the spatial document classification module) to create spatially stratified samples of linguistically encoded movement patterns. As a starting point, we sampled documents from the 48 contiguous states in the US and Washington, DC, based on the location of the destination [49]. Moreover, this approach can be easily scaled to account for all (at the moment) English-speaking countries. The purpose of this sampling strategy is twofold: first, to identify linguistic patterns that are region specific; and, second, to use this knowledge to enhance the performance of the automatic extraction and characterization of movement patterns.

Acknowledgments

Research for this paper is based upon work supported by the National Geospatial-Intelligence Agency (NGA) through the NGA University Research Initiative Program/NURI program. The views, opinions, and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the National Geospatial-Intelligence Agency, or the US Government.

References

[1] Allen, G. From knowledge to words to wayfinding: Issues in the production and comprehension of route directions. In *Spatial Information Theory: A Theoretical Basis for GIS*, S. Hirtle and A. Frank, Eds., vol. 1329 of *Lecture Notes in Computer Science*. Springer, Berlin, 1997, pp. 363–372. doi:10.1007/3-540-63623-4_61.

[2] Bateman, J., Hois, J., Ross, R., and Tenbrink, T. A linguistic ontology of space for natural language processing. *Artificial Intelligence 174*, 14 (2010), 1027–1071. doi:10.1016/j.artint.2010.05.008.

[3] Berikov, V., and Litvinenko, A. Methods for statistical data analysis with decision trees. http://www.math.nsc.ru/AP/datamine/eng/decisiontree.htm, 2003. Novosibirsk, Sobolev Institute of Mathematics.

[4] BLAYLOCK, N., SWAIN, B., AND ALLEN, J. Mining geospatial path data from natural language descriptions. *Proc. ACM SIGSPATIAL GIS International Workshop on Querying and Mining Uncertain Spatio-Temporal Data* (2009).

[5] BORTHWICK, A. E. *A maximum entropy approach to named entity recognition*. PhD thesis, New York University, New York, NY, USA, 1999.

[6] BREIMAN, L. *Classification and regression trees*. Chapman & Hall/CRC, 1984.

[7] BROWNSTEIN, J., AND FREIFELD, C. HealthMap: The development of automated real-time Internet surveillance for epidemic intelligence. *Euro Surveillance 12*, 48 (2007), 3322.

[8] CUNNINGHAM, H. GATE: A general architecture for text engineering. *Computers and the Humanities 36*, 2 (2002), 223–254. doi:10.1023/A:1014348124664.

[9] DEVIJVER, P., AND KITTLER, J. *Pattern recognition: A statistical approach*. Prentice Hall, 1982.

[10] DOMINGOS, P., AND PAZZANI, M. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine learning 29*, 2 (1997), 103–130. doi:10.1023/A:1007413511361.

[11] DÖRK, M., CARPENDALE, S., COLLINS, C., AND WILLIAMSON, C. Visgets: Coordinated visualizations for web-based information exploration and discovery. *IEEE Transactions on Visualization and Computer Graphics* (2008), 1205–1212. doi:10.1109/TVCG.2008.175.

[12] DRYMONAS, E., AND PFOSER, D. Geospatial route extraction from texts. In *Proc. ACM SIGSPATIAL International Workshop on Data Mining for Geoinformatics* (2010), ACM, pp. 29–37. doi:10.1145/1869890.1869894.

[13] FREITAG, D., AND MCCALLUM, A. Information extraction with HMMs and shrinkage. In *Proc. AAAI-99 Workshop on Machine Learning for Information Extraction* (1999), pp. 31–36.

[14] GOLLEDGE, R. Human wayfinding and cognitive maps. In *Wayfinding behavior: Cognitive mapping and other spatial processes*, R. G. Golledge, Ed. John Hopkins University Press, Baltimore, London, 1999, pp. 5–45.

[15] HAKLAY, M., AND WEBER, P. Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, 4 (2008), 12–18. doi:10.1109/MPRV.2008.80.

[16] HELLERSTEIN, J., NAUGHTON, J., AND PFEFFER, A. Generalized search trees for database systems. In *Proc. 21st International Conference on Very Large Data Bases (VLDB)* (1998), p. 101.

[17] HENZINGER, M., HEYDON, A., MITZENMACHER, M., AND NAJORK, M. On near-uniform URL sampling. *Computer Networks 33*, 1–6 (2000), 295–308. doi:10.1016/S1389-1286(00)00055-4.

[18] Hollenstein, L., and Purves, R. Exploring place through user-generated content: Using Flickr. *Journal of Spatial Information Science*, 1 (2011), 21–48. doi:10.5311/JOSIS.2010.1.3.

[19] Hornsby, K. S., and Li, N. Conceptual framework for modeling dynamic paths from natural language expressions. *Transactions in GIS 13* (2009), 27–45. doi:10.1111/j.1467-9671.2009.01153.x.

[20] Hosmer, D., and Lemeshow, S. *Applied logistic regression*. Wiley-Interscience, 2000.

[21] Jones, C., and Purves, R. Geographical information retrieval. *International Journal of Geographical Information Science* 22, 3 (2008), 219–228. doi:10.1080/13658810701626343.

[22] Keim, D., and Oelke, D. Literature fingerprinting: A new method for visual literary analysis. In *IEEE Symposium on Visual Analytics Science and Technology (VAST)* (2007), pp. 115–122. doi:10.1109/VAST.2007.4389004.

[23] Klippel, A., and Li, R. The endpoint hypothesis: A topological-cognitive assessment of geographic scale movement patterns. In *Spatial Information Theory*, K. Hornsby, C. Claramunt, M. Denis, and G. Ligozat, Eds., vol. 5756 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, Berlin, 2009, pp. 177–194. doi:10.1007/978-3-642-03832-7_11.

[24] Klippel, A., MacEachren, A., Mitra, P., Turton, I., Jaiswal, A., Soon, K., and Zhang, X. Wayfinding choremes 2.0—Conceptual primitives as a basis for translating natural into formal language. In *Proc. International Workshop on Moving Objects From Natural to Formal Language* (2008).

[25] Klippel, A., and Montello, D. Linguistic and nonlinguistic turn direction concepts. In *Proc. 8th International Conference on Spatial Information Theory (COSIT)* (Berlin, 2007), S. Winter, M. Duckham, L. Kulik, and B. Kuipers, Eds., vol. 4736 of *Lecture Notes in Computer Science*, Springer, pp. 354–372. doi:10.1007/978-3-540-74788-8_22.

[26] Koch, S., Bosch, H., Giereth, M., and Ertl, T. Iterative integration of visual insights during patent search and analysis. In *IEEE Symposium on Visual Analytics Science and Technology (VAST)* (2009), pp. 203–210. doi:10.1109/VAST.2009.5333564.

[27] Kornai, A. Evaluating geographic information retrieval. *Accessing Multilingual Information Repositories* (2006), 928–938. doi:10.1007/11878773_104.

[28] Kuipers, B. Modeling spatial knowledge. *Cognitive Science 2*, 2 (1978), 129–153. doi:10.1207/s15516709cog0202_3.

[29] Lafferty, J., McCallum, A., and Pereira, F. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Internation Conference on Machine Learning* (2001), pp. 282–289.

[30] Lewis, D. Naive (Bayes) at forty: The independence assumption in information retrieval. In *Machine Learning: ECML-98*, C. Nédellec and C. Rouveirol, Eds., vol. 1398 of *Lecture Notes in Computer Science*. Springer, Berlin, 1998, pp. 4–15. doi:10.1007/BFb0026666.

[31] LI, H., SRIHARI, R., NIU, C., AND LI, W. InfoXtract location normalization: A hybrid approach to geographic references in information extraction. In *Proc. HLT-NAACL 2003 Workshop on Analysis of Geographic References* (2003), vol. 1, Association for Computational Linguistics, pp. 39–44. doi:10.3115/1119394.1119400.

[32] MACEACHREN, A. M., STRYKER, M., TURTON, I. J., AND PEZANOWSKI, S. HEALTH GeoJunction. *International Journal of Health Geographics 9*, 23 (2010). doi:10.1186/1476-072X-9-23.

[33] MARTINS, B., MANGUINHAS, H., AND BORBINHA, J. Extracting and exploring the geo-temporal semantics of textual resources. In *Proc. IEEE International Conference on Semantic Computing* (2008), pp. 1–9. doi:10.1109/ICSC.2008.86.

[34] MCCALLUM, A., FREITAG, D., AND PEREIRA, F. Maximum entropy markov models for information extraction and segmentation. In *Proc. 17th International Conference on Machine Learning* (2000), Morgan Kaufmann, pp. 591–598.

[35] MCCALLUM, A. K. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[36] MICROSOFT. Bing phone book search API. http://msdn.microsoft.com/en-us/library/dd251005.aspx, 2011.

[37] MOSTELLER, F. A k-sample slippage test for an extreme population. *The Annals of Mathematical Statistics 19*, 1 (1948), 58–65. doi:10.1007/978-0-387-44956-2_5.

[38] NEEDLEMAN, S., AND WUNSCH, C. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology 48*, 3 (1970), 443–453. doi:10.1016/0022-2836(70)90057-4.

[39] NIGAM, K., LAFFERTY, J., AND MCCALLUM, A. Using maximum entropy for text classification. In *Proc. IJCAI-99 Workshop on Machine Learning for Information Filtering* (1999), vol. 1, pp. 61–67.

[40] PAN, C., AND MITRA, P. Femarepviz: Automatic extraction and geo-temporal visualization of FEMA national situation updates. In *IEEE Symposium on Visual Analytics Science and Technology (VAST)* (2007), pp. 11–18. doi:10.1109/VAST.2007.4388991.

[41] PUSTEJOVSKY, J., MOSZKOWICZ, J., AND VERHAGEN, M. The recognition and interpretation of motion in language. *Computational Linguistics and Intelligent Text Processing* (2010), 236–256. doi:10.1007/978-3-642-12116-6_20.

[42] QUINLAN, J. *C4.5: Programs for machine learning*. Morgan Kaufmann, 1993.

[43] RISCH, J., KAO, A., POTEET, S., AND WU, Y. Text visualization for visual text analytics. *Visual Data Mining* (2008), 154–171. doi:10.1007/978-3-540-71080-6_11.

[44] TOBIN, R., GROVER, C., BYRNE, K., REID, J., AND WALSH, J. Evaluation of georeferencing. In *Proc. 6th Workshop on Geographic Information Retrieval* (2010), pp. 1–8. doi:10.1.1.167.3749.

[45] Tomaszewski, B. Producing geo-historical context from implicit sources: A geovisual analytics approach. *The Cartographic Journal 45*, 3 (2008), 165–181. doi:10.1179/000870408X311369.

[46] Tversky, B., and Lee, P. Pictorial and verbal tools for conveying routes. In *Spatial Information Theory. Cognitive and Computational Foundations of Geographic Information Science*, C. Freksa and D. Mark, Eds., vol. 1661 of *Lecture Notes in Computer Science*. Springer, Berlin, 1999, pp. 752–752. doi:10.1007/3-540-48384-5_4.

[47] US Geological Survey (USGS). Geographic names information system (gnis), 2011. Last access on Sep 2011 at: http://geonames.usgs.gov.

[48] whoisxmlapi.com. Whois API. http://www.whoisxmlapi.com/.

[49] Xu, S., Jaiswal, A., Zhang, X., Klippel, A., Mitra, P., and MacEachren, A. M. From data collection to analysis—exploring regional linguistic variation in route directions by spatially-stratified web sampling. In *Computational Models of Spatial Language Interpretation (CoSLI) Workshop at Spatial Cognition* (2010), R. J. Ross, J. Hois, and J. Kelleher, Eds., pp. 49–52.

[50] Zhang, X., Mitra, P., Klippel, A., and MacEachren, A. Automatic extraction of destinations, origins, and route parts from human generated route directions. *Geographic Information Science* (2010), 279–294. doi:10.1007/978-3-642-15300-6_20.

[51] Zhang, X., Mitra, P., Klippel, A., and MacEachren, A. M. Identifying destinations automatically from human generated route directions. In *ACM SIGSPATIAL* (2011).

[52] Zhang, X., Mitra, P., Xu, S., Jaiswal, A., Klippel, A., and MacEachren, A. Extracting route directions from web pages. In *Proc. International Workshop on Web and Databases* (2009).