Itinerary Retrieval: Travelers, like Traveling Salesmen, Prefer Efficient Routes*

Marco D. Adelfio Hanan Samet

Center for Automation Research, Institute for Advanced Computer Studies
Department of Computer Science, University of Maryland
College Park, MD 20742 USA
{marco, hjs}@cs.umd.edu

ABSTRACT

Internet users share large quantities of text and multimedia content that becomes easily accessible to others via hyperlinks and search engine results. However, structured datasets generally lack this level of exposure. One example is the travel itinerary, which many Internet users post online in the form of a spreadsheet or web page table, yet the collection of such itineraries remains difficult to search or browse due to insufficient parsing and indexing by search engines. Enabling interaction with user-uploaded itineraries could provide valuable information to trip planners who are researching travel options and to businesses attempting to understand travel patterns. This work examines the challenges of identifying and extracting itineraries from spreadsheets and web page tables to support such applications, with a focus on differentiating between itineraries and other documents with geographic content.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Spatial databases and GIS*

General Terms

Algorithms, Design

Keywords

Travel itineraries, trajectory retrieval, route efficiency

1. INTRODUCTION

For anyone researching travel options for an upcoming vacation in a new part of the world, the advice and experience of previous visitors can be invaluable. Travel guidebooks, travel agencies, and online resources fulfil this role in many cases, however it is often difficult to get a sense of the wide variety of travel options available in a region of interest. Map-based interfaces for browsing uploaded travel itineraries could substantially improve travel research methods, which currently involve searching for travel suggestions using keywords, then visiting each search result to verify that it matches the geographic constraints of the travelers.

Developing such an itinerary browser and search system requires a reliable method for recognizing and extracting travel itineraries from Web-accessible documents. As far as we know, no prior work has looked at the specific problem of itinerary detection and retrieval from tables or text documents (some work on recovering itineraries from GPS logs or other geotagged metadata has been reported, as we discuss in Section 2). Due to the wide variety of itinerary formats in plain text documents, we focus our attention on detecting and extracting itineraries from spreadsheets and tables, which we believe have more regular structure and therefore will be sufficiently recognizable for our needs.

The primary challenge we address in this research is differentiating between itineraries and other geographic tables. While textual clues (i.e., the presence of the word "itinerary" in the title of a worksheet or the caption of an HTML table) can serve as useful indicators, a classification technique based only on text features would identify many false positives and fail to identify many false negatives. Additional criteria, such as whether the table includes a column of dates, may also have a strong correlation with the type of table being processed, but is still far from conclusive evidence that a table is an itinerary.

Our core hypothesis is that spatial analysis is the missing feature for enabling effective itinerary detection. Specifically, for humans, determining whether or not a table contains an itinerary frequently becomes easier when the locations in the table are viewed on a map, with lines connecting consecutive locations, because a variety of real-world constraints on time, money, and fuel encourage human travel that does not include unnecessarily long or inefficient routes. Instead, maps representing true itineraries typically follow spatially *efficient* routes (a concept that we formalize in Section 3.2). To measure the efficiency of an itinerary, our approach makes use of an optimization technique that was originally developed to generate approximate solutions for the traveling salesman problem (TSP). This optimization technique, known as 2-opt, functions by removing two edges from a sequential path through n points and determining whether a shorter overall path can be achieved by substituting edges with swapped endpoints [10, 21]. In Section 3, we restate this optimization in terms of *reasonably ordered* subpaths – subpaths that, when reversed, lead to a longer total path length – and show how the presence or absence of such subpaths is a powerful feature for determining whether a table contains an itinerary.

*This work was supported in part by the NSF under Grants IIS-10-18475, IIS-12-19023, and IIS-13-20791 and by Google Research and NVIDIA.

GIR'14 November 04 2014, Dallas, TX, USA
 978-1-4503-3135-7/14/11 ...$15.00
http://dx.doi.org/10.1145/2675354.2675355.

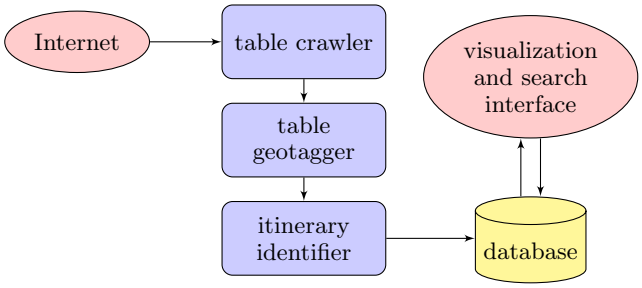


Figure 1: An itinerary processing pipeline.

Figure 2: Three sample Italian vacation itineraries found on the Web.

In addition to harnessing spatial properties of itineraries, a separate challenge is the sparsity of itineraries, as a fraction of documents on the Web, or even as a fraction of geographic tables on the Web. This sparsity means identifying itineraries requires crawling large portions of the Web. Furthermore, reliably extracting table data and assigning geographic interpretations to place names within the tables are prerequisites for accurate identification and extraction of itineraries. For these components, we make use of existing methods that extract tables from the Web and geotag them.

The diagram in Figure 1 shows the processing pipeline for our itinerary extractor. Documents are initially taken from a Web crawl and all tables are extracted to an abstract table format. The table geotagger identifies place references in table rows and assigns geographic interpretations to them. Our primary focus is the next phase, the itinerary identifier, where we classify geographic tables as either itineraries or non-itineraries. An itinerary search system could use the results of this phase to enable browsing and searching over a large database of itineraries, allowing users to visualize and compare itinerary options, like those shown in Figure 2. Targeting tables for itinerary retrieval has the added benefit that metadata for each stop (such as the date of the stop, any activities performed there, and lodging or transportation information), if present, is easy to associate with the stop since it is likely to appear in the same table or spreadsheet row as the stop's location name.

Itineraries come in many formats and presentation styles, making it challenging to identify them and distinguish them from other documents that contain listings of place names. Such a decision is necessary because clearly there are many geographic datasets online that include columns containing place names, but which do not intend those place names to be interpreted as a series of stops in an itinerary, such as tables containing demographic datasets or listings of customer addresses. For our purposes, the term *itinerary* describes a list of places which are intended to be visited in the listed order, while a *non-itinerary* is a list of places which does not have this property. We formalize the problem of identifying itineraries as follows:

Definition 1. *Let L be the set of all valid latitude / longitude locations. Then, given an ordered collection $I = l_1, l_2, \ldots, l_n$ of locations $l_i \in L$, the* ***itinerary decision problem*** *(IDP) is to determine whether I represents an itinerary.*

Unfortunately, the problem is difficult to solve accurately, even for humans, so the expected confidence in an algorithm's solutions must be tempered. However, as we show in Section 3, there are reasonably effective means of addressing this problem, even when the only availably indicators are lists of geographic coordinates.

We can re-formulate the problem to include additional context along with each location, given that our source documents are tables and spreadsheets, not simple lists of geographic coordinates.

Definition 2. *Let T be a table containing an ordered set of relations $r_1, r_2, \ldots, r_n$, where each relation r_i has an associated location l_i. The* ***table itinerary decision problem*** *(TIDP) is to determine whether T represents an itinerary.*

Both the location-only IDP and context-inclusive TIDP can be addressed with statistical and machine learning methods, by incorporating several indicators that have a correlation to the outcomes of the decision problems. The following features are included in our implementation.

- Efficiency of stop ordering (applies to IDP and TIDP). In general, travel itineraries are designed with some constraints on the time and effort required to travel between all the stops, which results in nearby stops being visited consecutively. In place listings where spatial relationships are not taken into account, the expected length of an itinerary visiting each place in order will be distributed according to the total travel length required to visit those places in a random order.
- Returning to the start (IDP and TIDP). Itineraries are frequently "round-trips" where the starting and ending locations are the same.
- Ordering columns (TIDP only). Itinerary tables frequently contain an ordering columns such as the date that the corresponding location will be visited, or an ordinal number representing which day within the trip the location will be visited.
- Presence of travel terminology (TIDP only). Some words and phrases are commonly found in itineraries (e.g., the text "at sea" appears often in itineraries for cruise ships) and can serve as indicators of the subject of the document.

The rest of this paper is organized as follows. Section 2 contains analysis of related work. Section 3 provides details of our table processing and itinerary detection methods. In

Date	Location	Delivery #
12/16/04	Oestrich-Winkel, DE	20031
03/17/05	Lavera, FR	20053
03/17/05	Lavera, FR	
04/27/05	Marl, DE	
05/25/05	Beringen, BE	
06/23/05	Schwechat-Mannswörth, A	
09/08/05	Dordrecht, NL	
11/21/06	Litvinov, CZ	
11/10/05	Pasir Gudang, Johor, MY	
11/10/05	Pasir Gudang, Johor, MY	
12/14/05	Antwerpen, BE	
11/16/05	Tehran, IR	
12/19/05	Brüssel, BE	
01/19/06	Torre Boldone (BG), IT	
01/19/06	Torre Boldone (BG), IT	
...	...	

Day	Dest	Activities
1	Vienna	Hotel check-in
2	Vienna	City tour
3	Vienna	Transfer to Budapest
	Budapest	City tour
4	Mohacs	Pecs excursion
	Villany	Wine tasting
5	Vukovar	Yugoslav Civil War tour
	Novi Sad	Walking tour
6	Belgrade	City tour
7	Iron Gates	Full day cruising
8	Vidin	Belogradchik excursion
9	Giurgiu	Palace of Parliament
10	Rousse	Disembarkation
	Plovdiv	Walking tour
11	Erdine	Lunch stop
12	Istanbul	City tour
13	Istanbul	Tour Topkapi Palace
14	Istanbul	Return flight home

Date	ETA	Location	Notes
9/19/07	8:00	Splendora FBC	Depart
	10:11	Nacogdoches, TX	Gas Stop
	12:09	Marshall, TX	Gas Stop & Lunch
	14:51	Texarkana, AR	
	15:22	Hope, AR	Gas Stop
	15:57	Gum Springs, AR	
	16:23	Arkadelphia, AR	Stop
9/20/07	7:30	Arkadelphia, AR	Depart
	7:39	Caddo Valley	Gas
	11:16	Dardanelle, AR	Gas Stop
	13:06	Jasper, AR	Lunch
	14:26	Dogpatch USA	Scenic/Photos
	14:42	Harrison, AR	Gas Stop & Lunch
	16:33	Francis, AR	
	16:49	Eureka Springs, AR	Stop & Gas
9/21/07	9:00	Eureka Springs, AR	Depart
	10:48	Ozark, AR	
	11:17	Van Buren, AR	Gas & Lunch
	12:53	Fort Smith, AR	
	12:55	Entering Oklahoma	
	15:10	Sunset Corner, OK	
	16:04	Entering Arkansas	
...	...	...	...

Figure 3: Portions of tables containing possible itineraries.

Section 4, we describe the experimental evaluation of the itinerary detector. Finally, Section 5 highlights the benefits of our approach and concludes the paper.

2. RELATED WORK

Our work complements other work in information retrieval that seeks to expose geographically rich content and is motivated by our earlier work in browsing spatial data [14, 29]. While we primarily focus on the spatial, rather than temporal, aspects of itineraries, research on document-based spatio-temporal extractors addresses some related tasks. Strötgen et al. [30] described a system for extracting ⟨time, location⟩ pairs from unstructured text documents, to support browsing the documents as trajectories (for example, following the path of explorers as described on their Wikipedia pages). A trajectory browser displays the extracted trajectories on a map and provides relevant text snippets for selected stops. The emphasis of this work is on building accurate spatial and temporal profiles of targeted document, so it does not address ways of identifying which documents contain trajectories. Spatio-temporal extraction systems also exist for a variety of other source documents, such as RSS feeds [22].

Systems for inferring itineraries from metadata, rather than documents, have been developed on top of various data sources, including geotagged photo streams [12] and GPS tracks [32]. These efforts have a rather different focus than ours, stemming from the fact that the locations in these efforts are specified numerically (e.g., as latitude / longitude pairs) rather than textually, and are known to be personal itineraries based on the nature of the data source. Yoon et al. [32] use GPS logs to identify stops (called "stay points") and transitions between clusters of stops, which allow them to recommend itineraries based on time constraints and the popularity of the stops. The periodic and granular location information provided by GPS tracks make them a valuable source for itinerary data. However, capturing this data requires that users upload large quantities of GPS tracks to the system or to a public location, so privacy concerns may hinder its availability. Additionally, extracting segments of GPS tracks that are relevant as itineraries requires addressing a separate set of challenges.

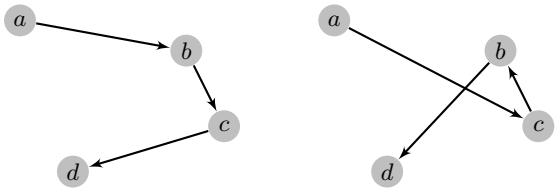
Accurate itinerary retrieval requires accurate geotagging, for which there is a rich body of relevant research [4, 6, 16, 17, 19, 23, 25, 26]. The more specific problem of geotagging data tables has been addressed in some settings, such as for ontology extraction [11], entity discovery in Fusion Tables [24], and general spreadsheet and table geotagging [20]. For itinerary geotagging, we use our probabilistic model for geotagging collections of place names [1, 3, 18], which identifies place categories for geographic table columns, then disambiguates toponyms in the context of that category. The method achieves high accuracy on sample tables and appears to be a good fit for itineraries, which tend to visit places that share similarities of geography, type, and/or prominence.

3. METHODS

3.1 Importing and Geotagging Tables

Our procedure for extracting tables from the Web employs a previously developed algorithm that we developed for segmenting table rows by function [2] in order to separate the data portions from metadata or non-data portions. This builds on the prior methods of the WebTables project [7, 8] and related techniques [15, 31]. As we are using tabular data extraction as a pre-processing phase and it is not the focus of this work, we summarize our procedure here and recommend examining our schema extraction algorithm and the WebTables architecture for more information. In our system, raw spreadsheets and HTML tables are first converted to an abstract textual format consisting of a two-dimensional array of textual cell values. Next, a classifier trained on cell features determines whether the table is likely to be a data table or a table that is used for a different purpose (e.g., for layout in an HTML page or as a calendar or form) that is not useful for information extraction. For data tables, a second classifier identifies the header row and data rows using another set of cell attributes as features. These rows are then passed on to the table geotagger.

From the resulting collection of data tables we must identify those that are geographic tables, from which we will obtain a collection of itineraries. For this, we use a method



(a) path *abcd* (b) path *acbd*

Figure 4: Itineraries generally follow efficient routes. For this example, we expect that an itinerary visiting locations a, b, c, and d is more likely to visit them in the order *abcd* (shown in (a)) than the order *acbd* (shown in (b)). Conversely, tables containing places that are ordered efficiently are more likely to be itineraries than tables containing ineffecient place orderings.

based on Combined Hierarchical Place Categories (CHPC) [3], as mentioned in Section 2. The key step in the method is identifying a common thread that can be used to categorize the interpretations of all toponyms in the set (e.g., "cities in Bavaria with population > 10,000"). For our purposes, this is useful because many place names in itineraries are not fully specified, as itinerary authors expect human readers to use the surrounding places or other context to disambiguate place references. The method achieves high accuracy for geotagging place lists found in tables on the Web.

We modified the CHPC method to use a different "tie-breaker" procedure in situations where multiple interpretations of a toponym exist within a list's assigned category. Instead of selecting the most highly populated interpretation, we select the interpretation that is nearest to the geographic centroid of the other toponyms' interpretations. In cases where multiple toponyms have ambiguous interpretations within a category, we use a greedy approach that iteratively selects interpretations closest to the geographic centroid of all already-selected interpretations.

3.2 Identifying Itineraries

The primary concern of this research is identifying itineraries from among the vast array of geographic tables and spreadsheets. This identification step is necessary because, while the output of the table geotagger is a collection of geographic tables along with interpretations of their place references, the vast majority of these tables are not intended to be itineraries — rather, they are tables that include entities with geographic attributes, not a travel path. Examples of non-itinerary geographic tables are demographic tables, sports team standings, or listings of people that include a column containing each person's hometown. Many itineraries share common characteristics with non-itineraries, but the characteristics, when viewed as a whole, allow us to discern itineraries from non-itineraries in many cases.

Figure 3 shows fragments of several representative tables that were found by our table crawler and determined to include geographic columns. In this example, the tables share similarities in terms of column headings, data types of nearby columns, and place name formatting. In this case, the table on the left is not intended as an itinerary, which becomes more evident when viewing the plotted locations from each table in Figure 5. The fact that the left and right tables share several textual similarities (such as the "Date" and "Location" column headers and the comma-based place name formatting), but only one is an itinerary, suggests that rule-based methods or methods that rely on column header text or data types of nearby columns will have difficulties making the determination. Further, the difference between the mapped visualizations of the tables led us to believe that spatial analysis of the tables was an important component in accurately addressing the TIDP.

Using these observations, we developed several heuristics to act as indicators for a machine learning classifier. The most useful indicators are based on the observation that itineraries tend to be fairly efficient at visiting stops, in comparison to an ordering of the stops that is not based on their spatial proximity. This is due to the fact that trip planners take costs of transportation and travel time into account. In particular, the tendency to prefer a shorter ordering of stops is measurable by comparing the route length of the original route to that of an alternate route that visits the same stops, but in a different order. Instead of comparing with a globally optimal route, which is intractible to compute for even relatively short itineraries (since the TSP is NP-hard) and does not model true travel itineraries, we use an interchange procedure that underlies the commonly-used 2-opt method for generating approximate solutions for the traveling salesman and other optimization problems [10]. Figure 4a shows an example, where an alternate permutation of the location list could reverse the order of stops b and c. As shown in Figure 4b, this results in a longer total route length than the original, so is less likely (though still possible) to be chosen as part of an itinerary. We call location lists with many pairs of points whose reversal results in longer path lengths *locally efficient*, meaning that the listing could not be made into a shorter route by simply rearranging neighboring stops. Similarly, location lists with many sequences of points whose reversal results in longer path lengths are said to be *generally efficient*.

The edge interchange procedure is the basis for two efficiency measures that we use as features in our itinerary identification algorithm. The first, ϵ_1, measures efficiency at the local level — essentially counting how many consecutive pairs of stops are in the order that results in the shortest path. The second, ϵ_2, measures stop order efficiency over longer sequences of locations — counting subsections of the full stop list that could be more efficiently reconnected to the remainder.

To formalize our concepts of efficiency, we define a preliminary indicator function. For an ordered set of locations $L = l_1 l_2 \dots l_n$ and $d(l_i, l_j)$ = great circle distance between l_i and l_j, let

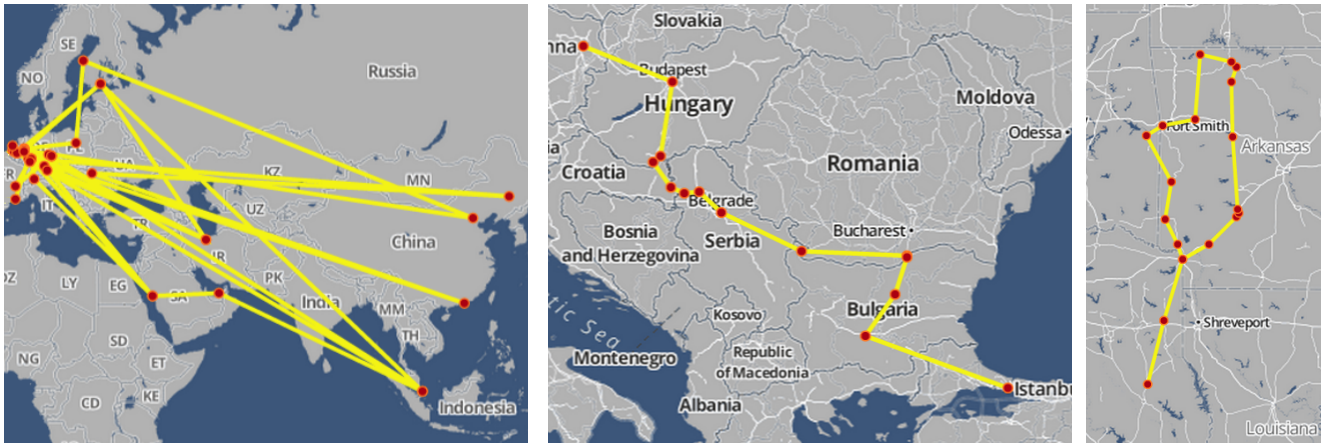
$$\delta_{i,j}(L) = \begin{cases} 1 & \text{if } (d(l_i, l_{i+1}) + d(l_j, l_{j+1})) \leq \\ & \quad (d(l_i, l_j) + d(l_{i+1}, l_{j+1})) \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

The $\delta_{i,j}$ value indicates whether the combined lengths of the edge from l_i to l_{i+1} and the edge from l_j to l_{j+1} is shorter than (or equal to) the combined lengths of edges with swapped endpoints, l_i to l_j and l_{i+1} to l_{j+1}. Equivalently, this indicates whether a permutation of the location list that reverses the order of locations $l_{i+1} \dots l_j$ has a shorter overall path length than the initial permutation. We use this to define two efficiency measures.

- **Local efficiency** is the fraction of consecutive stop pairs whose reversal would lead to a longer total route distance. That is, for locations $L = l_1 l_2 \dots l_n$,

$$\epsilon_1(L) = \frac{1}{n-3} \sum_{i=1}^{n-3} \delta_{i,i+2}(L). \quad (2)$$

- **General efficiency** is the fraction of all unique, non-



(a) (b) (c)

Figure 5: Visualizations of the tables from Figure 3 as itineraries. While the column headers and cell types of the left and right tables are similar, the topology that results from treating each table as an itinerary makes it clear that (a) is unlikely to be an itinerary, while (b) and (c) are both likely to represent itineraries. In fact, the table visualized in (a) came from a listing of shipments for a company that is certainly not intended as an itinerary. The table visualized in (b) contains the schedule for a river cruise through eastern Europe and the table visualized in (c) is the schedule for a motorcycle club's ride through several states in the U.S.A., which are both itineraries.

consecutive edge pairs that would result in a longer total route if their endpoints were swapped. For locations $L = l_1 l_2 \dots l_n$,

$$\epsilon_2(L) = \frac{1}{\binom{n-2}{2}} \sum_{i=1}^{n-3} \sum_{j=i+2}^{n-1} \delta_{i,j}(L). \quad (3)$$

Each efficiency formula counts the number of valid swaps that result in a longer total path length, which is then normalized by the total number of valid swaps. For example, assume we have a table containing five locations ($L = l_1 l_2 \dots l_5$). Then $\epsilon_1(L) = \frac{1}{2}(\delta_{1,3}(L) + \delta_{2,4}(L))$ and $\epsilon_2(L) = \frac{1}{3}(\delta_{1,3}(L) + \delta_{2,4}(L) + \delta_{1,4}(L))$. Using terminology from TSP literature [21], if reversing any sub-sequence of L results in a longer path length, we call L *2-optimal*. This is exactly the property we measure with ϵ_2, so we can say that L is 2-optimal if and only if $\epsilon_2(L) = 1.0$. In general, the goal of the efficiency measures is to quantify the presence of efficient stop ordering. Consequently, we expect itineraries to exhibit high efficiency values (although values less than one are expected, given that many travel itineraries do not follow optimal paths), while non-itineraries will tend to have moderate efficiency values clustered near 0.5.

In addition to efficiency measures, we also use several other features when deciding whether or not a given table represents an itinerary. These include the following ordering features and text features.

- $f_r(t) = 1$ *iff the primary location column of the table includes the same location in the first and last positions.* We call this a *round trip* table and expect that round trip tables will be more common in itineraries than non-itineraries.
- $f_{od}(t) =$ *# of ordered date/time columns found in the table.* Since itineraries are temporal objects, itineraries in tables commonly include a date/time column.
- $f_{on}(t) =$ *# of ordered numeric columns found in the table.* While ordered numeric columns are a component of some itinerary tables (such as the center table in Figure 3), they are also common in non-itineraries. We expect this feature to have a smaller effect on accuracy than the others.
- $f_a(t) =$ *# of text columns found in the table that are sorted alphabetically.* Unlike the previous two ordering features, we expect that tables containing alphabetically sorted columns are unlikely to be itineraries, since it is rare for a table to be arranged both spatially and alphabetically.
- $\vec{f}_t(t) =$ *a term vector of words commonly found in itineraries.* Currently, we use a list of 40 words and phrases that we found to have the highest difference in their TF/IDF values in itineraries versus non-itineraries. Such terms include "itinerary", "trip", "travel", "airport", "hotel", "cruise", month names, and others.

To account for the loose constraints inherent in manually generated tables, columns are treated as ordered or alphabetic if at least 90% of the values in the column are greater than or equal to the preceding value or at most one value is out of order in columns with fewer than 10 values.

We construct a feature vector $\vec{f}(t)$ for each table t using the features listed above, and then apply a binary classifier to compute $Pr(t \text{ is an itinerary} | \vec{f}(t))$. Given that our collection of features includes a variety of feature types (fractional, binary, integer, and term vector), this is not a clear fit for any one specific machine learning classification model, so we examine three: (i) a Naive Bayes classifier [13], (ii) a decision tree [5, 27], and (iii) a support vector machine [9]. We pre-process each feature based on the expected input format for each specific classification model, giving binary features to the Naive Bayes classifier, raw numeric values to the decision tree, and standardized (mean- and variance-adjusted) values to the support vector machine.

4. EVALUATION

4.1 Dataset

The tables for our evaluation were taken from a two million page Web crawl that targeted Microsoft Excel spreadsheets and HTML pages containing tables. We seeded the crawl with search results for queries of the form "⟨*data term*⟩ ⟨*geo term*⟩ ⟨*chaff*⟩ ⟨*filetype*⟩". Each term was randomly se-

lected from a hand-selected set of values or omitted, as our goal was to use a range of queries to uncover a wide variety of documents. The data term was randomly chosen from a list of terms that are often found in documents containing tables (such as "table", "stats", etc.). The geo term was randomly chosen from a large collection of place names found in the GeoNames gazetteer. The chaff term was a randomly chosen letter, number, or both, used to induce a variety of results for a static combination of the other terms. And the filetype component was set to "filetype:xls" or "filetype:xlsx" to search for Excel spreadsheets, or omitted to search for HTML documents containing tables. Statistics for the full table corpus are shown in Table 1. Our table extraction module removed tables that were not found to be data tables (also known as "true" or "relational" tables), resulting in 662 thousand documents. Since some HTML documents contain multiple tables, and spreadsheets can likely contain multiple worksheets, the actual number of data tables in our corpus was 2.1 million. After running our geotag module to locate toponyms in the tables and assign interpretations to them, we obtain a set of 130 thousand documents containing 235 thousand tables. The geographic tables contain many more rows on average, as there are around 53 cells per column, compared to 28 cells per column in the full dataset.

The evaluation was performed using a corpus of tables that we manually annotated as either itineraries or non-itineraries. For a table to qualify as an itinerary, there must be implied travel along the edges between consecutive pairs of places. This definition results in several tables being called itineraries that would not be considered itineraries for the purposes of a sightseeing trip, but which have the implied-edge property, such as a listing of exits along a section of highway or all the stops made by a regional train. For our purposes, these are all types of itineraries.

In all, we annotated 300 tables as either itineraries or non-itineraries. The first 200 were selected at random from our full dataset, of which only 3 were true itineraries. The next 100 were chosen from tables with a large number of stops ($n \geq 10$) and a high efficiency value ($\epsilon_1 \geq 0.8$) to ensure an adequate number of itineraries were included in the evaluation corpus (the number would otherwise be low due to the sparsity problem mentioned in Section 1). Later in this section, we account for the non-random sampling by scaling measurements based on the relative frequency of similar efficiency values within the full dataset. Of the 300 annotated tables, 60 were classified as itineraries, and 240 were classified as non-itineraries. The itineraries had a mean number of stops of 29 and a median of 22, while non-itineraries had a mean of 27 and a median of 14.

4.2 Itinerary Detection

Our evaluation of itinerary detection involved analyzing (i) the discriminatory power of the efficiency measures, (ii) the overall accuracy of our itinerary detector, and (iii) the contribution of individual features to classification accuracy.

The observed probability density functions of the efficiency measures are shown in Figures 6 and 7. The curves are smoothed using kernel density estimation [28] to reveal the trends (and to avoid uninformative peaks at common fractional values such as 0.5, 0.75, 0.666..., etc.). Figure 6 shows the estimated distribution of ϵ_1 values for itineraries and non-itineraries in our training set. The estimates were calculated by scaling each ϵ observation by the relative frequency of similar efficiency values within the full dataset.

Evaluation of each classifier on the table itinerary decision problem was performed using five-fold cross validation against the annotated data set. For each classifier, we computed the average precision (P), recall (R), and F_1 score. Using T_P as the number of true positives (true itineraries correctly classified as itineraries), F_P as the number of false negatives (non-itinereraries incorrectly classified as itineraries), and F_N as the number of false negatives (true itineraries incorrectly classified as non-itineraries), then $P = T_P/(T_P + F_P)$, $R = T_P/(T_P + F_N)$, and $F_1 = 2PR/(P + R)$. The results are displayed in Figure 8. The decision tree classifier achieves the best F_1 score of 0.73, perhaps due to strong interdependence between the features, which decision trees can exploit. This is followed closely by the SVM with an F_1 score of 0.72. The Naive Bayes classifier achieves the worst F_1 score based on a very low precision score. As an example of the limititations of our method, two tables that were incorrectly classified by all three classifiers are shown in Figure 9. The first is a table of Dewey Decimal class numbers for books that focus on individual U.S. states. Interestingly, although this system for organizing books in a library predates computers or computerized search systems, its choice of ordering leads to a path used by many computer-based geographic indices: a space-filling curve. This efficient path leads to high values of ϵ_1 and ϵ_2, which cause the classifiers to deem the table an itinerary, incorrectly. Similarly, the second table is a listing of coastal Italian regions and various related statistics (only the coastline column is included in the figure). The ordering of regions is clearly influenced by their spatial location, but like the Dewey Decimal table, there is no implied edge between consecutive locations in the table, and it is therefore not an itinerary. The existence of tables such as these, which can be described as spatially-arranged non-itineraries, explains much of the classification error observed in our evaluation. This suggests that other spatial features or non-spatial features may be required to successfully detect and classify them as non-itineraries.

Table 1: Dataset characteristics

Full Dataset	
Documents	2,000,000
containing data tables	662,511
Data tables	2,128,032
Columns	10,142,785
Cells	280,170,694
After removing non-geographic tables	
Documents	130,294
Data Tables	235,433
Columns	1,527,890
Cells	80,432,927

Next, we analyzed the contribution of individual features and combinations of features to the accuracy of the decision tree classifier (for the rest of this section, we use the decision tree classifier, as it was the top performer in classification accuracy). We ran the classification test repeatedly, while holding out individual features, and compared the results of each test to the results when all features were included and tabulated the results in Table 2.

As expected, the classifier performed no better when features were removed, with the biggest change coming when we withheld the both efficiency measures. ϵ_1. The F_1 score in this case fell from 0.73 to 0.44, a drop of 0.29, which we call the marginal contribution of ϵ_1 and ϵ_2 to the F_1 score. This is a substantial difference in the F_1 score and suggests that the efficiency measures are quite discriminative, in ways that the other features are not. Somewhat surprisingly, the

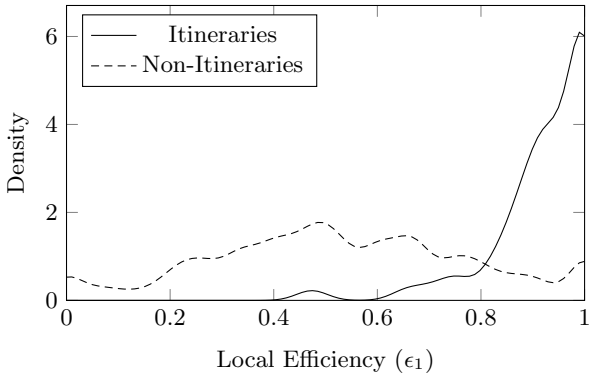


Figure 6: Density of the ϵ_1 measure for itineraries and non-itineraries. As shown, itineraries are much more likely to obtain high ϵ_1 values (> 0.8) than non-itineraries. The vastly different curves suggest that the local efficiency measure is a useful feature for distinguishing between itineraries and non-itineraries.

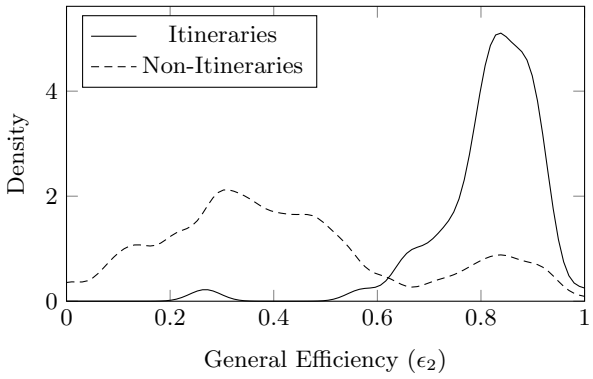


Figure 7: Density of the ϵ_2 measure for itineraries and non-itineraries. Similar to the distributions for local efficiency (ϵ_1) values, itineraries are much more likely to have high ϵ_2 values than non-itineraries.

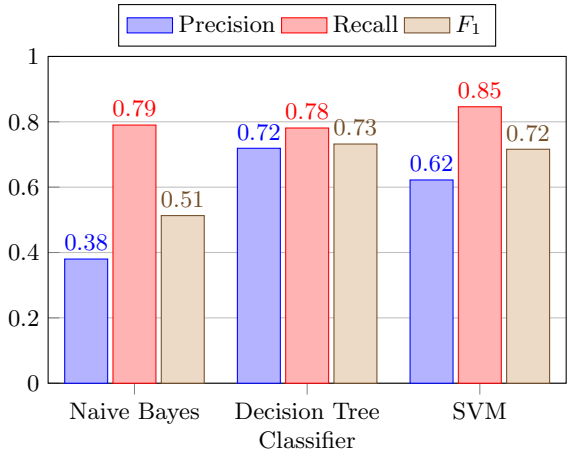


Figure 8: Precision, recall, and F_1 scores for each of the candidate classifiers on the itinerary identification task. The decision tree classifier achieves the highest F_1 score, followed by the SVM and Naive Bayes classifier.

Table 2: Feature evaluation

Feature	F_1 Without Feature	Marginal Contribution to F_1 score
ϵ_1	0.62	+0.11
ϵ_2	0.70	+0.03
ϵ_1 and ϵ_2	0.44	+0.29
f_r	0.71	+0.02
f_{od}	0.69	+0.04
f_{on}	0.72	+0.01
f_a	0.69	+0.04
$\vec{f_t}$	0.72	+0.01
non-efficiency	0.66	+0.07

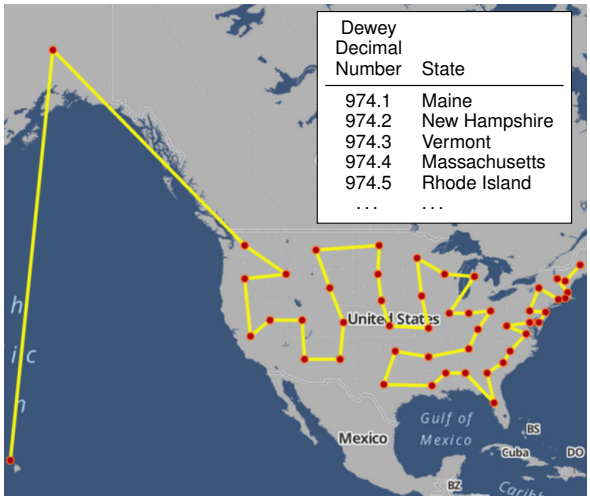
local efficiency score, ϵ_1 causes a much larger drop than the general efficiency score, ϵ_2, when withheld individually. We see two potential explanations for this. First, from a statistical perspective, the nature of the general efficiency measure may be less informative than the local efficiency measure, as the fraction of itineraries with $\epsilon_2 > 0.8$ is relatively lower, while the fraction of non-itineraries with $\epsilon_2 > 0.8$ is relatively higher. Second, from a data analysis perspective, it may be that the nature of itineraries leads to more local efficiency than general or global efficiency. That is, given the scheduling constraints that can shape itineraries, people may be inclined to travel efficiently for short periods, but not aim for a perfectly efficient route from start to finish. Such priorities would explain the disparate impact of these two efficiency features on our classification accuracy.

Other features all contribute to the performance of the classifier, with the ordered date column indicator f_{od} and the alphabetic column indicator f_a both contributing 0.04 to the F_1 score. The least impact is attributable to the ordered numeric column indicator, f_{on}, and the text vector, f_t, whose removal only caused a decrease of 0.01 in the F_1 score. This is somewhat surprising, since ordered numeric columns are much more prevalent in itineraries than non-itineraries. By manual inspection of the annotated table corpus, we see that 44% of true itineraries contain an ordered numeric columns, while they are found in only 15% of non-itineraries. This may be explained by the presence of temporal words in the term vector for $\vec{f_t}$, whose presence may offset the gains otherwise attributable to a numeric column. Still, the small differences in the impact of these features is overshadowed by the impact of the efficiency features.

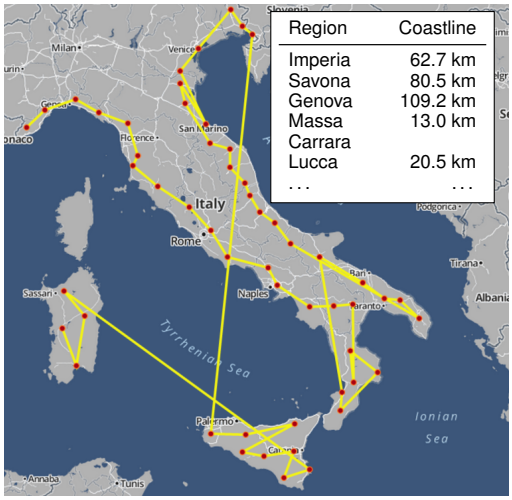
Finally, we looked at the number of itineraries found in our full table corpus. The decision tree model classified 1,206 itineraries out of the 235,433 geographic tables in our corpus, a total of 0.5%. This is consistent with our expectation that itineraries would be rare, but prevalent enough that a more complete crawl of the Web would result in a large quantity of itineraries to allow for map-based browsing.

5. CONCLUSIONS AND FUTURE WORK

We have presented itinerary retrieval as a new area for geographic data extraction and implemented a pipeline of processing methods to evaluate our proposed approach, which uses a machine learning classifier to decide whether a candidate table contains an itinerary. The core of our method involves computing spatial efficiency measures of the locations listed in a table, which match our notions of efficiency and were shown to have a substantial impact on the accuracy of our classifier. As future work, we plan to experiment with additional text features and differentiating between transport schedules and personal travel itineraries. We also intend to build a complete search interface to support applications for travel research.



(a) Dewey Decimal classes for each state in the U.S.A.



(b) Coastal Italian regions and corresponding coastline lengths.

Figure 9: Two examples of mis-classified tables. Both tables include lists of locations that are highly efficient by our definition, causing all three classifiers that we used in our evaluation to label them as itineraries. In (a), the Dewey Decimal system for book topic classification is shown, which orders states along a path that resembles a space filling curve. In (b), a listing of coastal Italian regions presumably follows a path with similarities to some Italian vacations, but is instead an exhaustive list of such regions and related coastal data.

6. REFERENCES

[1] M. D. Adelfio and H. Samet. GeoWhiz: Toponym resolution using common categories. In *SIGSPATIAL*, pages 542–545, Orlando, FL, Nov. 2013.

[2] M. D. Adelfio and H. Samet. Schema extraction for tabular data on the web. *PVLDB*, 6(6):421–432, 2013.

[3] M. D. Adelfio and H. Samet. Structured toponym resolution using combined hierarchical place categories. In *GIR*, pages 49–56, Orlando, FL, Nov. 2013.

[4] E. Amitay, N. Har'El, R. Sivan, and A. Soffer. Web-a-Where: Geotagging web content. In *SIGIR*, pages 273–280, Sheffield, UK, July 2004.

[5] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and Regression Trees*. Chapman & Hall, New York, 1984.

[6] D. Buscaldi and P. Rosso. Map-based vs. knowledge-based toponym disambiguation. In *GIR*, pages 19–22, Napa Valley, CA, Oct. 2008.

[7] M. J. Cafarella, A. Halevy, D. Z. Wang, E. Wu, and Y. Zhang. Uncovering the relational web. In *WebDB*, Vancouver, Canada, June 2008.

[8] M. J. Cafarella, A. Halevy, D. Z. Wang, E. Wu, and Y. Zhang. WebTables: Exploring the power of tables on the web. In *VLDB*, pages 538–549, Auckland, New Zealand, Aug. 2008.

[9] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[10] G. A. Croes. A method for solving traveling-salesman problems. *Operations Research*, 6(6):791–812, 1958.

[11] I. F. Cruz, V. R. Ganesh, and S. I. Mirrezaei. Semantic extraction of geographic data from web tables for big data integration. In *GIR*, pages 19–26, Orlando, FL, Nov. 2013.

[12] M. De Choudhury, M. Feldman, S. Amer-Yahia, N. Golbandi, R. Lempel, and C. Yu. Automatic construction of travel itineraries using social breadcrumbs. In *HT*, pages 35–44, Toronto, Canada, June 2010.

[13] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, New York, second edition, 2000.

[14] C. Esperança and H. Samet. Experience with SAND/Tcl: A scripting tool for spatial databases. *JVLC*, 13(2):229–255, Apr. 2002.

[15] W. Gatterbauer, P. Bohunsky, M. Herzog, B. Krüpl, and B. Pollak. Towards domain-independent information extraction from web tables. In *WWW*, pages 71–80, Banff, Canada, May 2007.

[16] M. D. Lieberman and H. Samet. Multifaceted toponym recognition for streaming news. In *SIGIR*, pages 843–852, Beijing, China, July 2011.

[17] M. D. Lieberman and H. Samet. Adaptive context features for toponym resolution in streaming news. In *SIGIR*, pages 731–740, Portland, OR, Aug. 2012.

[18] M. D. Lieberman, H. Samet, and J. Sankaranarayanan. Geotagging: Using proximity, sibling, and prominence clues to understand comma groups. In *GIR*, Zurich, Switzerland, Feb. 2010.

[19] M. D. Lieberman, H. Samet, and J. Sankaranarayanan. Geotagging with local lexicons to build indexes for textually-specified spatial data. In *ICDE*, pages 201–212, Long Beach, CA, Mar. 2010.

[20] M. D. Lieberman, H. Samet, J. Sankaranarayanan, and J. Sperling. Spatio-textual spreadsheets: Geotagging via spatial coherence. In *GIS*, pages 524–527, Seattle, WA, Nov. 2009.

[21] S. Lin and B. W. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21(2):498–516, 1973.

[22] B. Martins, H. Manguinhas, and J. Borbinha. Extracting and exploring the geo-temporal semantics of textual resources. In *ICSC*, pages 1–9, Santa Clara, CA, Aug. 2008.

[23] S. E. Overell and S. M. Rüger. Using co-occurrence models for placename disambiguation. *IJGIS*, 22(3):265–287, 2008.

[24] G. Quercini and C. Reynaud. Entity discovery and annotation in tables. In *EDBT*, pages 693–704, Genoa, Italy, Mar. 2013.

[25] G. Quercini, H. Samet, J. Sankaranarayanan, and M. D. Lieberman. Determining the spatial reader scopes of news sources using local lexicons. In *GIS*, pages 43–52, San Jose, CA, Nov. 2010.

[26] E. Rauch, M. Bukatin, and K. Baker. A confidence-based framework for disambiguating geographic terms. In *GEOREF*, pages 50–54, Stroudsburg, PA, May 2003.

[27] L. Rokach and O. Maimon. *Data Mining with Decision Trees: Theory and Applications*. World Scientific, New York, 2008.

[28] M. Rosenblatt. Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27(3):832–837, 1956.

[29] H. Samet, H. Alborzi, F. Brabec, C. Esperança, G. R. Hjaltason, F. Morgan, and E. Tanin. Use of the SAND spatial browser for digital government applications. *CACM*, 46(1):63–66, Jan. 2003.

[30] J. Strötgen, M. Gertz, and P. Popov. Extraction and exploration of spatio-temporal information in documents. In *GIR*, pages 16–23, Zurich, Switzerland, Feb. 2010.

[31] P. Venetis, A. Halevy, J. Madhavan, M. Paşca, W. Shen, F. Wu, G. Miao, and C. Wu. Recovering semantics of tables on the web. *PVLDB*, 4(9):528–538, June 2011.

[32] H. Yoon, Y. Zheng, X. Xie, and W. Woo. Smart itinerary recommendation based on user-generated GPS trajectories. In *UIC*, pages 19–34, Xi'an, China, Oct. 2010.