Changes in place names offer insight into regions’ culture, politics, and geographical characteristics. This paper proposes an automatic approach to retrieve time-sequenced maps that show place name changes on many maps from across history. The proposed approach utilizes gazetteers (i.e., indexes for geographic names) to retrieve a place’s coordinates and name variants and searches for text labels from maps matching those coordinates and names. To search for multiple-word place names, the approach constructs minimum spanning trees from an edge cost function to link text labels into phrases. We present two experiments: one to evaluate the effectiveness of the minimum spanning tree approach at linking multiple word place names, and the other to evaluate the maps retrieved by the query approach. The resulting maps give rich visual insight into how place names change over time and could facilitate scholarly investigation of geographic name changes at a large scale.

1 INTRODUCTION

Geographic place names carry a great deal of historical, linguistic, cultural, and environmental significance, and observing changes in place names can illuminate the phenomena behind the changes a place underwent in history. For example, place name changes in Spain revealed aspects of past climate change [5], and place name changes in South Africa reflected changes in cultural identities after the end of Apartheid [7]. Existing methods for analyzing place name changes often rely on manual inspection of historical maps [5] or comparisons of names that appear in indexes of place names (so-called gazetteers) of different times [7]. Manually inspecting and analyzing place name changes on a large number of maps requires intensive time. Furthermore, the time range of changes that can be observed is limited by the time range of the gazetteers/maps that are referenced (e.g., gazetteers ranging from 1995 to 2000 were compared in Guyot and Seethal [7]).

While gazetteers have historically been paper documents, recent efforts have been made to compile digital gazetteers that make extensive information about place names available online. To facilitate analysis of place name changes, one study highlights the need for digital gazetteers to store temporal indicators of when each place name was used [2]. To obtain such temporal indicators, we leverage large libraries of scanned, dated, and georeferenced historical maps. We propose an approach to automatically link each place name to the dates of historical maps that use that place name, through what we call time-sequenced map queries. These queries aim to, given a place name, retrieve dated, scanned maps that contain that place’s name variants (i.e., alternate names that refer to the same place). The dates of the retrieved maps can be compiled as temporal indicators of when each name variant of the place is used. Furthermore, these maps can be cropped around the name variant they use for the place and plotted over time, giving a zoomed-in view of how names for the queried place change over time on many maps. To retrieve time-sequenced maps, the proposed approach takes two main steps: 1) retrieve a list of name variants of that place and the place’s geographic location from digital gazetteers, and 2) search for historical maps that include those name variants at the place’s location. Thus, the final output of each query is a list of historical maps that include the place, each matched with the name variant found on the map and the year the map was published.

For the first step of this query approach, the list of name variants is needed because places can use different names on different maps (both due to historical name changes and maps being in different languages). The geographic location of the place is also needed because some place names are shared by many places (e.g., “Springfield” refers to many different U.S. towns). Fortunately, this information can be retrieved from digital gazetteers. However, existing gazetteers often lack extensive temporal data for each place name. The proposed approach aims to obtain more extensive temporal context via published years of historical maps.

Regarding the second step of searching textual content on maps to find name variants of the place appearing at the place’s location, we investigate existing studies that convert text labels on maps into analytic-ready data. These methods include a crowdsourcing approach (e.g., GB19002) and machine learning tool (e.g., mapKurator [4]). We employ the results from one of the existing studies, the mapKurator, an end-to-end pipeline for processing scanned map images and outputting recognized text labels from maps. When mapKurator processes georeferenced maps, the recognized text labels are georeferenced based on the geocoordinates of their bounding polygons. To retrieve time-sequenced maps, we leverage the outputs of machine learning tools such as mapKurator to search for text labels that appear near the location of the queried place, and match one of the name variants for that place.

However, challenges exist in using the outputs of machine learning tools. First, the detected text bounding polygons and recognized text labels are usually separated by word, not entire place name phrases. This causes a further challenge of linking which text labels are parts of multiple-word place names (e.g., connecting the recognized text labels “Saint” and “Paul” into one name, “Saint Paul”). Figure 1 illustrates factors that make determining which text labels on maps are parts of multiple-word phrases difficult. These factors include: 1) the density of text labels on maps, 2) the inconsistency

---

1 e.g., the David Rumsey Historical Map Collection, https://www.davidrumsey.com/

2 https://maps.nls.uk/projects/gb1900
with which multiple-word place names are laid out, and 3) the presence of unrelated text labels appearing between text labels that are in multiple-word phrases.

Figure 1: Challenges of linking together multiple-word place names on historical maps.

A simple approach to searching for a multiple-word place name is to search through all possible groups of labels that combine to have as many words as that name. Such an approach falls short for two reasons. The first is that many of the words in multiple-word place names are common in text labels extracted from maps (e.g., words like “north” and “city” appear many in different places on maps). This creates a risk of incorrectly recognizing a multiple-word phrase where the labels are not actually related on the map (e.g., recognizing the labels “North” and “Dakota” as the name of a U.S. state when in reality the label “North” is referring to a direction on a compass). The second issue is that, for a multiple-word phrase containing \( k \) words and a map containing \( n \) text labels, there are \( O(n^k) \) groups of labels that need to be searched in the worst case. To avoid falsely recognizing unrelated labels on maps as being part of the same phrase and ensure that the runtime of the queries is feasible, we propose a tree-structured method for first determining which text labels are most likely to be part of the same phrase.

2 METHODOLOGY

This section presents the approach to enable queries for time-sequenced historical maps, which show how a queried place changes names over time. We use a running example of how the proposed method works for a query of “Abu Dhabi”. Figure 2 shows an overview of the approach. There are three inputs of the proposed approach: 1) a gazetteer, which is used to match the queried place name to a geographic location and a list of name variants of that place, 2) a collection of recognized text labels from scanned maps (e.g., the output of the mapKurator system) which are searched to find accounts of the place’s names being used at the place’s location, and 3) a name of a place to query, which can optionally include clarification of the type of geographic entity (e.g., city or lake), or country the place is in. The system’s output is a collection of dated historical maps, each matched with the name variant for the place that appears on the map, which can be used for temporal analysis of the place’s name variants.

2.1 Obtaining Geographic Coordinates and List of Name Variants from Gazetteer

Given a query for the geographic place, we first query a gazetteer to extract the name variants and the geographic coordinates of the place. An example query of “Abu Dhabi” gets matched with the coordinates of the city (e.g., in WGS84), as well as the city’s name variants, including “Abu Debi”, “Abu Zabi”, and other variant spellings and names used in different languages. If multiple entries in the gazetteer match the queried place name, one can specify additional information about the country or type of place they are searching for. For example, specifying “Paris, France” instead of “Paris, Texas” or “Mississippi, the state” instead of “Mississippi, the river”. In the absence of such disambiguating information, we choose to default to the gazetteer entry that contains the most name variants. This step provides where names for the queried place will appear on historical maps and what names to search for that have been used to refer to the place.

Text labels can have different orientations relative to the place they name (e.g., the label “Abu Dhabi” could appear above or below a dot on the map representing the city). This causes a discrepancy between the location of text labels on maps and the location of the places they refer to, and this discrepancy is different on different maps. To provide tolerance for this discrepancy, we initially pad a 1-degree-by-1-degree box around the centroid of coordinates retrieved by the gazetteer. However, the variance in where text labels appear relative to the places they name depends on the place. For example, labels of larger places such as Canada tend vary more widely in location than labels for smaller places, such as Minneapolis. To scale the tolerance with the size of the text labels and the variance in orientations in which labels for that place appear, we update the padding continually to include the bounds of text labels that match the during the query.

2.2 Linking of Multiple Word Place Name Phrases

To search multiple-word place names, the inputted text labels, which each generally contain a single word, must first be linked into the potential phrases they compose on maps. The linkages of text labels can be represented as a graph, where each node represents a text label, and each edge represents whether the text labels are likely to appear next to each other in a phrase. To search for a phrase containing \( k \) words in this graph, we can search for nodes containing the first word of the phrase, and then search along the edges of matching nodes to find the subsequent words in the phrase. If we can ensure that the number of edges in this graph is \( O(n) \), where \( n \) is the number of text labels, the runtime of searching for the phrase in the graph is \( O(n^2) \), as opposed to \( O(n^k) \) if all possible groups of \( k \) labels are searched. To construct such a graph, we utilize Prim’s algorithm [10] to construct a minimum spanning tree (MST) of the text labels. MSTs are useful for this purpose because they include exactly \( n - 1 \) edges, which are chosen such that they minimize an edge cost function, which represents how unlikely each pair of text labels is to appear next to each other in a phrase. Therefore, an MST constructed with a suitable edge cost function can make the search runtime \( O(n) \) by only searching through the groups of text labels that are most likely to be in the same phrase. MSTs have also been applied to the related problem of linking lines of text in scene images [9].

Based on observations of phrases on maps, labels in multiple-word place names are often spatially close together (although figure 1 shows how unrelated text can be closer to a label than text that is part of the same phrase). Moreover, such labels usually have
similar text height (i.e., font size), are oriented at similar angles, and have the same capitalization. The edge cost function used to construct the minimum spanning tree from the graph computes the cost of connecting two text labels $u$ and $v$ based on four factors. The first is the minimum spatial distance between the bounding boxes of the two text labels on the map image, denoted as $d(u, v)$. The second is the ratio of the height of the label with the taller bounding box to that of the label with the shorter bounding box. This factor denoted as $h(u, v)$, will be minimized at 1 for labels with the same height and, by extension, similar font sizes. The third factor, denoted as $a(u, v)$, is the angle difference between the axes of two bounding boxes, which will be minimized to 0 when the axes of the bounding boxes of two labels are aligned. The final factor, denoted as $c$, represents the difference in capitalization between the labels. The value of $c(u, v)$ is 1 when either the two labels are both all uppercase or neither is all uppercase, or 2 if one is all uppercase and the other is not. The edge cost function is given by:

$$
cost(u, v) = d(u, v) \cdot h(u, v) \cdot (1 + \sin(a(u, v))) \cdot c(u, v)$$

(1)

Constructing an MST from this edge cost function creates linkages that minimize discrepancies in distance, height, angle, and capitalization between linked text labels.

### 2.3 Searching for Text Labels that Overlap Coordinates and Match One Name Variant

Once the query is matched with name variants and coordinates from the gazetteer, we search for labels that match these coordinates and variant names in the recognized text extracted from maps. For each map, we first compare the geocoordinates obtained from the gazetteer with the geocoordinates of text labels on the map. This step filters out labels not appearing near the place being queried. Only labels on the map that overlap the coordinates of the queried place (as retrieved from the gazetteer) are linked into potential phrases in an MST as described in section 2.2. To avoid searching through groups of text labels that are unlikely to form multiple-word phrases, the MST links pairs of labels that are most likely to be next to each other in multiple-word phrases (by creating a tree that minimizes the edge cost function). Then, for each name variant of the place, we search the linkage graph for connected sequences of text labels on the map that match that name variant. We use fuzzy string comparison1 to account for possible errors in the text labels, as well as to recognize minor spelling variations of place names that might not be included in the list of name variants retrieved from the gazetteer. In the case of single-word place names, we search for a single label whose text matches the name variant. For multiple-word place names, we search for the first word of the name variant first, and then search for the rest of the phrase through the edges of matching labels in the linkage graph. For example, if searching for the name variant “Abu Dhabi”, we first search for the word “Abu”, and then for each text label that matches this word, we search for text labels connected to it in the linkage graph that match the text “Dhabi”. This step finds which name variant of the place, if any, is used on the map. If a name variant is found in the MST of text labels, the source map of the labels is added to the query results, along with the name variant found on the label(s) and the date of the map. All resulting maps are then time-sequenced by the published year obtained from the map metadata. To efficiently view the resulting data, the maps can be cropped around the bounding polygons of the text labels that matched the query, giving a zoomed-in view of the queried place as its name changes over time.

### 3 EXPERIMENTS

In this section, we present two experiments and their results. The first evaluates the effectiveness of linking text labels into phrases using MSTs with different edge cost functions, and the second evaluates the effectiveness of the proposed approach for automatically retrieving time-sequenced maps that show a given place and the names it uses across history.

#### 3.1 Linking Multiple Word Phrases with Minimum Spanning Trees

To evaluate the effectiveness of the MST method described in section 2.2 at linking multiple-word place names, we leveraged a competition dataset from ICDAR 2024 containing scanned map images with annotated map text words and phrases [8]. We computed minimum spanning trees of the map text labels using the proposed edge cost function and several simpler edge cost functions and evaluated how many phrases from the annotated data were correctly linked by the MST computed with each edge cost function.

#### 3.1.1 Dataset

The data from [8] consists of 2,000 × 2,000 pixel tiles of scanned historical maps from the David Rumsey Historical Map collection, each with annotations of each word appearing on the map, a bounding polygon of where the word appears, and the other labels that the make up the phrase the word is in (if any). We utilized the annotations from 99 of these map tiles (all map

---

1[https://pypi.org/project/fuzzywuzzy/]
image tiles in the dataset that contained at least 10 multiple-word phrases) to evaluate the effectiveness of MSTs with various edge cost functions in correctly linking the phrases that exist in the annotations.

3.1.2 Explicit Features of Text Bounding Boxes. For each of the bounding polygons of the words in the annotated data, we computed a minimum area rectangle as the bounding box of the text label. The edge cost functions tested in this experiment were based on four features of the text labels and their bounding boxes: distance, height ratio, angle difference, and capitalization. Each feature is described in more detail in Section 2.2. This experiment first evaluated edge cost functions based on only one of these factors, then functions that combined the best-performing single feature with another feature, and finally on functions that incorporated three or four of these features.

![MST Constructed with Edge Cost Function on Features](image)

![MST Constructed with the Proposed Edge Cost Function](image)

Figure 3: Visualization of minimum spanning trees constructed from map text labels with two different edge cost functions. Labels with blue bounding boxes are not part of multiple-word phrases. Labels with green bounding boxes are in multi-word phrases that are correctly linked by the MST, and labels with red bounding boxes are in phrases that are not correctly linked by the MST. Differences in the performance of the two edge cost functions are circled.

3.1.3 Results. The proportion of phrases from the annotated data that were correctly linked by the MSTs for each function is shown in Table 3.1.2. A visualization of the constructed MSTs and their effectiveness at linking phrases is shown in figure 3. These results show that the edge cost function given by Equation 1 outperforms simpler edge cost functions in constructing the correct linkages in the text labels. This suggests that all four of the features used in this edge cost function are useful factors in determining whether two labels are part of a larger phrase. Considering that 84.46% of multiple-word phrases were correctly linked by MSTs with this edge cost function, it seems that MSTs can be applied effectively to find a large majority of multiple-word place names on historical maps, but there is still room for improvement on this method.

3.2 Searching Place Name Changes

To demonstrate the effectiveness of the proposed approach, we conducted experimental queries to automatically retrieve time-sequenced maps that show how place names change over time.

3.2.1 Datasets.

- As input for the experiment, we employ the open source API of World Historical Gazetteer (WHG) [6], which provides the digital gazetteer comprised of geographic coordinates, country, and name variants of a queried place.
- Regarding text labels and georeferenced bounding polygons on maps, we leverage the mapKurator system’s processed output of the David Rumsey Historical Map Collection, which consists of over 100 million text labels across 57k historical maps.
- For the input place names to query, we initially used the names of the 50 most populous cities in Russia and the 78 most populous cities in Germany (obtained from 2010 census data). The sample of Russian cities formed an interesting case study, as many cities in the region changed names in the 20th century due to political efforts by the Soviet leaders [3]. To evaluate the effectiveness of the query approach in recognizing multiple-word place names, we also processed queries for the 15 most populous cities in the world that currently use multiple-word place names. These names were obtained from [1]. This dataset of roughly 47,000 cities was filtered for cities whose names contain at least one space character (and therefore, multiple words), and then sorted by population size to retrieve the 15 most populous cities that use multiple-word names.

<table>
<thead>
<tr>
<th>Number of Features in Edge Cost Function</th>
<th>Edge Cost Function Used for Edge Costs</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Features</td>
<td>Sine Angle Difference ( \sin(u,v) )</td>
<td>0.0718</td>
</tr>
<tr>
<td></td>
<td>Height Ratio ( h(u,v) )</td>
<td>0.1290</td>
</tr>
<tr>
<td></td>
<td>Distance ( d(u,v) )</td>
<td>0.8160</td>
</tr>
<tr>
<td>Pairs of Features</td>
<td>Distance and Height Ratio ( d(u,v) + h(u,v) )</td>
<td>0.8238</td>
</tr>
<tr>
<td></td>
<td>Distance and Sine Angle Difference ( d(u,v) + (1 + \sin(u,v)) )</td>
<td>0.8322</td>
</tr>
<tr>
<td>Three or Four Features</td>
<td>Distance, Height Ratio and Sine Angle Difference ( d(u,v) + h(u,v) + (1 + \sin(u,v)) )</td>
<td>0.8388</td>
</tr>
<tr>
<td></td>
<td>Distance, Sine Angle Difference and Capitalization ( d(u,v) + (1 + \sin(u,v)) + c(u,v) )</td>
<td>0.8404</td>
</tr>
<tr>
<td></td>
<td>Distance, Height Ratio, Sine Angle Difference, and Capitalization ( d(u,v) + h(u,v) + (1 + \sin(u,v)) + c(u,v) )</td>
<td>0.8464</td>
</tr>
</tbody>
</table>

Table 1: Results of applying minimum spanning trees with various edge cost functions to link together annotated map text labels. Recall is measured as the proportion of phrases from the annotated data that were correctly linked by the MSTs constructed using the edge cost function.
While such name variants are used to refer to these places, they without constructing phrase linkage graphs from map labels, and thus only single-word name variants were recognized. This shows that this automatic approach can be used as a supporting tool for manual inspection to improve the sample size of maps that can be referenced (e.g., [5] investigated seven maps). However, one issue that arose in the multi-word queries was the presence of short, single-word name variants of places (e.g., “la” as a name variant of “Los Angeles” and “Rio” as a name variant of “Rio De Janeiro”). While such name variants are used to refer to these places, they are informal and are used more often in speech than written on maps. Furthermore, these short name variants occur in unrelated map text, which can be incorrectly interpreted as referring to the place. For example, results included matches for “LA” that were part of other names like “La Brea” and matches for “Rio” that were part of names like “Rio Grande”. We plan to mitigate this issue by cross-referencing multiple digital gazetteers to distinguish between name variants that are used informally and name variants that are likely to be used on maps.

Figure 4 shows maps sampled from the 73 time-sequenced map results of a query for “Istanbul”, and maps from the 42 time-sequenced map results of a query for “Sault Ste Marie”. The map from 1892 displays an interesting phenomenon of the name “Istanbul” appearing on maps before being formally adopted in 1930, possibly because that name was used colloquially much earlier [11]. In this way, the time-sequenced map output gives temporal context to place names for scholars investigating the social and political aspects of geographic name changes.

Table 2: Query results for three groups of cities. Queries for 50 most populous Russian cities and German cities were made without constructing phrase linkage graphs from map labels, and thus only single-word name variants were recognized.

<table>
<thead>
<tr>
<th>Query Sample</th>
<th>Avg. # of Maps Retrieved</th>
<th>Avg. # of Maps with Multiple-Word Names</th>
<th>Avg. Timespan of Maps</th>
<th>Number of Name Variants Recognized</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 Most Populous Russian Cities</td>
<td>60.55 maps per query</td>
<td>not attempted</td>
<td>170 years per query</td>
<td>4.25 per query</td>
</tr>
<tr>
<td>78 Most Populous German Cities</td>
<td>141.91 maps per query</td>
<td>not attempted</td>
<td>250 years per query</td>
<td>1.90 per query</td>
</tr>
<tr>
<td>15 Most Populous Global Cities with Multi-Word Names</td>
<td>284.53 maps per query</td>
<td>145.26 maps per query</td>
<td>265 years per query</td>
<td>7.6 per query</td>
</tr>
</tbody>
</table>

4 CONCLUSION AND FUTURE WORK

This paper presented an automatic method for retrieving time-sequenced maps of a place name using gazetteers and recognized text labels from maps. We showed experimental results using the WHG and the mapKurator system’s output from the David Rumsey Map Collection to query for a sample of cities. The major limitation of the proposed approach is the assumption of finding matched place names using geographical proximity and string similarity. We acknowledge that additional methods and rigid studies of place names should be incorporated into the matching process. As the first step in tackling this challenging problem, we believe the proposed approach will set an example of temporal analysis of place name changes over time using large numbers of historical maps. We will improve this method by exploring other methods of linking multiple-word place names. Because constructing MSTs with Prim’s algorithm has $O(n^2)$ runtime, where $n$ is the number of nodes (i.e., text labels) in the graph, we plan to explore faster methods for linking together multiple-word phrases that could improve query runtime. We also plan to utilize this method to compile extensive temporal context for place names stored in digital gazetteers.

ACKNOWLEDGMENTS

We thank David Rumsey for his generous support and for making scanned maps publicly available. All scanned maps used in figures are credited to the David Rumsey Map Collection, David Rumsey Map Center, Stanford Libraries.

REFERENCES

[8] Li Z. Chiang Y.-Y. Jerod W. Lin, Y. 2024. Rumsey Train and Validation Data for ICDAR’24 MapText Competition (1.1) [Data set].