Social Media Canvassing Using Twitter and Web GIS to Aid in Solving Crime

by

Neil Emrys Stone

A Thesis Presented to the
Faculty of the USC Graduate School
University of Southern California
In Partial Fulfillment of the
Requirements for the Degree
Master of Science
(Geographic Information Science and Technology)

December 2017

To my parents, John David Stone and Barbara Anne Aszman Stone

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank the Faculty and Staff of The Spatial Sciences Institute of The University of Southern California Dornsife College of Letters, Arts and Sciences. Specifically, but not limited to, Dr. Karen Kemp, whose sharp and insightful guidance improved the quality of this work, and my other instructors, Dr. Su Jin Lee, Dr. Wei Yang, Dr. Jennifer Swift, and Dr. Yao-Yi Chiang. The knowledge they shared with me, I consider a lifetime treasure. I am also grateful to Mrs. Robyn MacNab who, I like to joke, "got me into this mess", and Mr. Kendrick Watson whose administrative talents saw me through.

All this would not have been possible without my family. My wife Ashley Stone worked tirelessly to provide time for me to focus, and stability for our daughters Cora and Isla, who by the way, provided hugs whenever I was feeling unmotivated. Ashley's parents, Bill and Cynthia Smith, gave over the second floor of their home so that I could leave my job and focus on school. My own parents, John Stone and Barbara Anne Aszman Stone, provided generous support whenever it what needed. These people are shining examples of humanity and I hope to pay back their innumerable acts of generosity by living up to their shared example.

List of Abbreviations

API	Application Programming Interface
BSON	Binary JavaScript Object Notation
CAPS	Chicago's Alternative Policing Strategy
CSS	Cascading Style Sheets
GIS	Geographic information system
HTML	Hypertext Markup Language
JSON	JavaScript Object Notation
KML	Keyhole Markup Language
NoSQL	Not Only SQL
POI	Point of Interest
RDMS	Relational Database Management System
SODA	Socrata Open Data API
SSI	Spatial Sciences Institute
USC	University of Southern California
VPS	Virtual Private Server
XML	Extensible Markup Language

Abstract

This thesis details the creation of an open-source Web GIS application that can quickly match crime incidents and location-tagged social media posts that share location in space and time. The objective was in support of an over-arching goal of providing a publicly available investigative tool that could be used by citizens and members of law enforcement, augment existing social media strategies in law enforcement, and aid in accelerating case clearances. Detailed herein, are the methods for, and the open-source and free-to-use technologies used to, create the application. While the application works as expected, it was discovered that the study area did not produce the amount of location-tagged social media information needed for a consistently high rate of spatiotemporal co-location of crime incidents with social media posts. Analysis of the data collected for this work confirmed the reasons for the lack of co-locations and revealed directions for future work, which are also discussed.

Chapter 1 Introduction

In the first half of 2017, a homicide was committed every 15 hours in the City of Chicago. This statistic is harrowing and in opposition to the 15-year downward trend in violent crime nationwide. Nonetheless, of the 160 Chicago homicides, from January 1, 2017 to April 1, 2017, suspects have been charged in only 38 cases (Thompson 2017). A 24 percent homicide clearance rate implies room for improvement and the need to consider new methods of criminal investigation. The project described in this document is an attempt to improve social media canvassing in law enforcement. Social media canvassing is a recent addition to a broader investigative canvassing strategy.

By accessing social media, investigators can exploit a wealth of information that can be used to locate perpetrators, develop probable cause for warrants, and identify witnesses, before a physical investigative canvass is initiated (Giacalone 2017). To help achieve the goals of social media canvassing, this thesis developed a web-based application that maps crime incidents and social media posts that share common location and time. If spatiotemporal co-locations are found between a homicide or other crime incident and georeferenced Twitter posts, it may be possible to extract useful information about the perpetrators of, witnesses to, or the circumstances surrounding the crime.

The objective of this thesis was the development of an open-source Web GIS application that maps crime incidents and social media posts that are spatiotemporally co-located with the crime incidents. The mapping of crime incidents and the co-located social media posts are based on user-specified criteria: crime incidents are mapped according to user inputs, for crime type and a window of one or many days, while social media posts are filtered with a user definition of spatiotemporal co-location in minutes and meters. This objective is situated within the broader

goal of encouraging the use of location-based social media canvassing to provide information relevant to crime investigation. The next section describes the motivation behind the application.

1.1. Motivation

The intent behind the creation of this Web GIS application is to foster public awareness, promote community involvement, and provide law enforcement with a location-based system for initiating a social media canvass. Both public awareness and community involvement are beneficial in preventing crime (Chicago 1998) and the social media canvass is an important part of a broader investigative canvassing strategy (Giacalone 2015). Further discussion of each of these aspects is covered in the following three subsections.

1.1.1. Public Awareness

Crime mapping applications available online provide anyone with an internet connection access to information about the types of crime committed in specific neighborhoods. While this information can be used by law enforcement to more appropriately allocate personnel and resources, the public can use it to prevent further crime through greater awareness. Prevention measures could include avoiding certain areas at certain times or taking steps to make oneself less vulnerable. By mapping various crime types, users of the Web GIS application developed for this thesis can develop an understanding of where various crimes are occurring in the city of Chicago.

1.1.2. Community Involvement

Community involvement is a key component of Chicago's Alternative Policing Strategy (CAPS) (Chicago 1998). CAPS was developed in the early 1990s as a response to the high rate of violent crime in the City of Chicago. It is the city's version of community policing, a term

broadly used to describe greater engagement between police and citizens. In CAPS, community involvement takes the form of "beat community meetings" where individuals and citizen groups can meet with their beat officers to prioritize local problems and come up with ways to address them.

The application developed for this thesis takes citizen involvement a step further. By making this application available on the web, members of the public can participate in social media canvassing by seeking out spatiotemporally co-located tweets, reviewing the content, and possibly reporting anything they deem relevant. It is possible that members of the public will find meaningful details in the social media content that go unnoticed by police investigators. This opportunity for citizens to aid law enforcement fits nicely within the CAPS community policing strategy.

1.1.3. An Investigative Tool

A 2014 LexisNexis survey of police departments found that four out of five investigating officers used social media in investigations. Despite this, the same survey found that only 50 percent of officers in command positions think that social media is useful in criminal investigations. Similarly, 52 percent of the law enforcement agencies surveyed had no standard procedures in place for the use of social media in investigations (LexisNexis 2014).

This last motivation behind this thesis's application is not to change the perception of the value of social media in criminal investigation. Nor is it to provide a set of social media investigative procedures. Instead, the motivation is to provide investigators with an inexpensive, location-based augmentation to whatever social media strategy is already in place. Even if support for social media canvassing is spotty, an investigator lacking specialized proprietary software (like ArcGIS) and equipment, could benefit from a web application able to run on any

device with a browser connected to the internet. The application built for this thesis was started only a small monthly cost for a cloud server. It could be easily maintained by existing IT personnel with intermediate knowledge of web technologies. An inexpensive application exploiting the location data found in public social media posts that promotes citizen engagement and accelerates case clearances would be a boon to an investigator looking to develop a strategy for social media investigation.

1.2. Application Overview

This application was developed to be useful to both members of the law enforcement community and the public. It is a proof of concept that uses the City of Chicago as a study area and maps crime locations and spatiotemporally co-located Twitter posts. The following subsections describe the reasons for choosing Chicago as a study area, what the application does, and how the application works.

1.2.1. Study Area

Chicago was chosen as the study area for this application for two reasons: because Chicago is one of a few municipalities that release crime data in an easily queried format, and because the city has recently been in the news regarding a spike in homicides and gun related crime in 2016. Even though violent crime is trending down and has been for many years, violent crime rates in Chicago, in the first few months of 2017, are like those of 2016. Furthermore, the homicide clearance rate in 2016 was about 22 percent and is following a similar trajectory in 2017 (Thompson 2017). This clearance rate is far lower than the nationwide clearance rate of 66 percent (Madhani 2016). This makes Chicago is a suitable candidate for this project. Figure 1 is a map of the study area and the bounding box used for data collection.

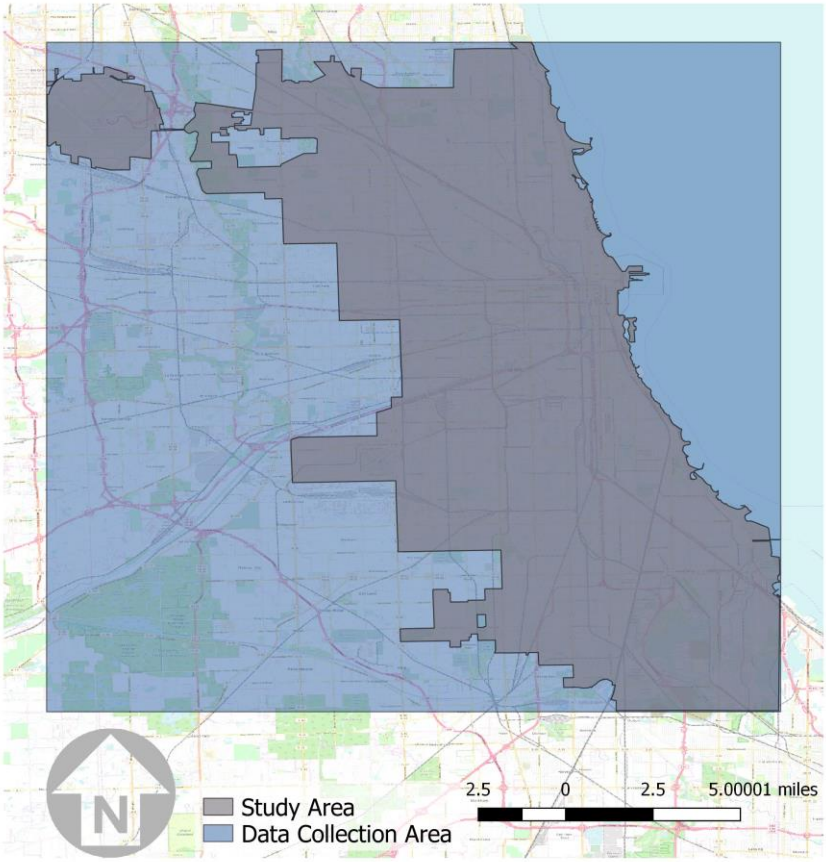


Figure 1: Map of the study area.

1.2.2. Application Functionality

The application maps incidents of crime based on user inputs for crime type, and a date range. When the user clicks a specific incident, the application displays the locations of Twitter posts that are spatiotemporally co-located with the incident based on the incident's coordinates and user inputs for time and distance. Information about the incident is displayed in a window while the content of any spatiotemporally co-located Twitter posts is presented in a scrollable list.

1.2.3. Spatiotemporal Co-location Results

Later in this document it is noted that once the application was working, it became apparent that spatiotemporal co-location between crimes and location-tagged Twitter posts was rare during the two months of data collection. To confirm this observation and help determine if geographic coordinates in social media metadata can be used in social media canvassing to find information useful to crime investigations, analysis of the data was conducted to answer the following questions:

- How many crime incidents had spatiotemporal co-locations with Twitter posts?
- What roles do space and time play in the number of spatiotemporal co-locations?
- If spatiotemporal co-locations are found, do they contain valuable information?

This study concludes with an exploration of these questions using the crime and tweet data collected for this project.

1.3. Structure of this Document

The objective of this thesis was to create a Web GIS application that uses location and timestamps to spatiotemporally co-locate social media posts with crime incidents with the idea that the content of such posts may contain information relevant to the crime. Chapter 2 examines related work and similar crime mapping applications that already exist. Chapters 3 and 4 recount the application requirements, data sources, and the development of the application. Chapter 5 outlines the methods for assessing the application, details the assessment results and draws conclusions about the efficacy of the application. Finally, Chapter 6 explores opportunities for improvement and directions for future work.

Chapter 2 Background

Online crime maps are usually mashups consisting of a basemap overlaid by a law enforcement data source containing geolocated crime incidents. The basemap could be from an open-source or free-to-use API such as OpenStreetMap or Google Maps, or a licensed service like Esri's ArcGIS Online. The service, as provided to the public, is intended to be informative. On the law enforcement side, crime mapping is used to allocate resources, predict incidents of crime, and investigate crime. The web application described here is built for similar purposes and is similarly informative, but also attempts to discover information of investigative utility through the addition of a social media component. The social media data comes from publicly available Twitter posts that have geographic coordinates contained in their metadata. Twitter was chosen as a source for social media data because it is described as one of the "Big Six" social media sites recommended for social media canvassing (Giacalone 2017) and is already the second most used form of social media by law enforcement (Burrows 2015).

Since this project combines crime mapping, Twitter data, and a cloud-based NoSQL database, the following review of related work is divided accordingly. Section 2.1 touches briefly on the history of crime mapping and ends with a description of some current publicly available online crime mapping applications. Section 2.2 examines the relationships between social media, crime investigation, and crime prevention. Section 2.3 covers some of the ways Twitter data have been used in online mapping applications. Section 2.4 discusses NoSQL databases generally and MongoDB specifically as a means for storing spatiotemporal data.

2.1. Web Applications for Crime Mapping

Over the past 100 years, advances in mapping crime have mirrored those in cartography, computerized mapping, and geographic information systems. Before computers, crimes were

mapped simply by marking incidents with pins on a paper map. This process was time consuming and provided for only simple pattern analysis. Other drawbacks included the large amount of physical space required to display a functional crime map (especially one of a large urban area), and the fact that the only way to preserve a wall-mounted pin and paper map, was to photograph it (Harries 1999).

According to Keith Harries, computerized mapping began to chip away at these limitations. The first computerized crime maps from the 1960s used printouts as a means of display. While the data used to reproduce a map were more archivable, the process of initial map creation was still costly in terms of labor for coding map coordinates and crime data onto punched cards. In the late 1980s and 1990s, desktop computers became cheap and ubiquitous. Advancements in display and storage technologies (beyond punch cards and printouts) allowed computerized crime mapping with geographic information systems to become practical and widely used in law enforcement.

While the availability of cheap fast computers made the use of geographic information systems in policework practical, the widespread adoption of computerized crime mapping is further explained by a "crisis in confidence" in policing methods during the 1970s and 1980s (Weisburd and Lum 2005). Traditional policing methods, such as increased preventative patrols and improving rapid response time, were shown to be ineffective at both reducing crime and increasing the likelihood of an arrest on-scene. Weisburd and Lum note that the need for new tools for policing became apparent.

Today, Geographic Information Systems provide many advantages over pin and paper crime maps. Crime mapping is now less time consuming and error prone and it no longer requires a large monetary investment to produce crime maps. Best of all, as Weisburd and Lum

note, GIS allow other geographic data to be compared with crime incident locations in the study area. For example, it is now easy to view population and demographic information along with crime incident locations. Combining disparate data sets has become easy, making previously unseen patterns visible to law enforcement and to the public.

With many sources of crime data and many options for embedding basemaps (Google Maps, Open Street Maps, ArcGIS Online) anyone with a computer connected to the internet can view or even create their own crime maps. It is possible to build a Web GIS application for crime mapping entirely out of open-source software and publicly available crime data. In 2014, Zhou et. al created an open-source Web GIS to for the detection of crime hotspots. It proved to be both a cost-effective replacement for commercial crime mapping solutions and a good decision support system for small law enforcement agencies. A similar open-source strategy is used in this thesis to create a Web GIS crime mapping application that could be easily adopted by agencies using existing hardware and software, and could support location-based social media canvassing.

Crime mapping is no longer exclusive to law enforcement agencies. Reliable high-speed internet and the widespread availability of data, make cloud based geographic information systems possible (Web GIS). Users can now choose from a variety of crime mapping applications that use online data stores to display crime incidents (CrimeMapping.com), choropleth maps, or heat maps (maps.NYC.gov/crime). These applications typically display crime data directly from law enforcement agencies via either a public feed or a direct upload from the agencies. Web scraping, a process where data is programmatically extracted from web pages, is utilized to a lesser extent (Paulson and LeBeau 2013).

CrimeMapping.com is built on Esri's ArcGIS platform. The web map uses crime data sourced from participating law enforcement across North America with the goal of crime reduction through an informed citizenry. The user interface allows the user to filter crimes by type, location, and date. Users may also generate various reports and charts detailing crime in their area of interest (TriTech Software Systems 2016).

Another crime mapping application, similar in functionality to CrimeMapping,com, is SpotCrime. SpotCrime offers the same filtering and reporting options with some notable differences. The user interface is built around the Google Maps JavaScript API basemap, which is free to use so long as the service SpotCrime is providing is also freely available[1]. Even though it is possible to view crimes in multiple jurisdictions across the United States, crime data for a specific area must be selected from a drop-down list of participating jurisdictions. With CrimeMapping.com, crime data for different areas is available simply by panning the map to a new area.

NYC Crime Map is an example of a web GIS crime map that is city specific. The New York City government built the application on the Google Maps for Business platform. Users can map the locations of felony crimes filtered by date. The NYC Crime Map also offers the ability to quickly view crime rates per 1000 citizens by precinct. The stated goal of NYC Crime Map is enabling citizens to make informed decisions about their personal safety (New York DoITT 2013; Reilly 2013; Reilly 2014).

[1] For more information on the terms and limitations of the Google Maps JavaScript API, see: https://developers.google.com/maps/pricing-and-plans/#details.

2.2. Social Media Canvassing in Law Enforcement

In the past ten years, the use of social media in crime prediction, prevention, and investigation has grown. A 2014 LexisNexis survey shows that 82 percent of law enforcement professionals use social media as an investigative tool with 25 percent using social media daily (up from 16 percent from 2012), and 72 percent believe that the use of social media accelerates case closures (up six percent from 2012) (LexisNexis 2014). Joseph Giacalone, a retired detective, extolled the virtues of "Social Media Canvasses" that include the identification of potential witnesses (Giacalone 2017). Members of the public show similar enthusiasm. In a 2014 Accenture Consulting survey of citizens from eight countries including the United States, 88 percent of respondents believed that digital technologies including social media can be used in crime investigations to "ask the public for information on specific cases" (Blackwood and Radvanyi 2014, n.p.).

Despite the apparent widespread use of social media in law enforcement documented in the 2014 LexisNexis survey (4 of 5 officers claimed to use social media in investigations), Detective Giacalone, in a recent communication via Facebook Messenger, lamented that it was "mind-boggling" that many police departments do not have social media canvassing strategies in place. The 2014 LexisNexis survey also indicates that only half of the respondents in command positions support the use of social media as an investigative tool. This has led to a lack of support in terms of training and equipment for investigators to conduct proper social media canvasses.

2.3. Twitter Location and Web GIS

Twitter is a micro-blogging website where users can post quick, 140-character updates called "tweets". When a user tweets, a variety of metadata is associated with the post including a

timestamp, and often some type of location data. In 2016, the service had 320 million monthly active users who produced 500 million tweets a day. Twitter's large user base, the packaging of spatial data with tweets, and the ease of access to publicly available tweets through Twitter's Streaming API, have made it a popular data source for many Web GIS applications.

One web GIS application called World Seer uses Twitter as a source of geotagged photographs and allows users to view the photos by clicking photo locations on a map. This application uses the Twitter Streaming API to collect geotagged tweets with photographs. The tweets and photographs are stored in databases accessible to the application (Yanai 2012).

Another Web GIS application called Senseplace2 maps tweets based on keyword and date queries to improve situational awareness in crisis management. Like World Seer, Senseplace2 uses the Twitter Streaming API for data collection. The Twitter data is stored in a PostGresSQL database. The application also searches for place names in the tweet content, and uses the GeoNames database of place names to geotag tweets that lack specific geographic coordinates. The purpose of Senseplace2 is to provide users with a means to visually filter Twitter data based on location and time to aid in foraging for information relevant to crisis events (MacEachren 2012).

Since the percentage of tweets with metadata containing explicit geographic coordinates is low, the examples mentioned above look to methods for inferring the location of tweets. Location inference is not used in this thesis but is mentioned here as a possibility for future enhancement to the application. Like the method used in the Senseplace2 application, Zhang and Gelernter have developed software that takes place names from tweet content and matches them to entries in the GeoNames place names database so that the place names can be geocoded. This

is accompanied by an algorithm the selects the appropriate GeoNames entry based on other pieces of tweet metadata such as time zone and language (Zhang and Gelernter 2014).

Along the same lines, Li and Sun (2014) described how they extract points of interest (POI) from the tweets' content, match them to entries in a Foursquare database, and then perform linguistic analysis on the content to determine if users are on their way to the POI, at the POI, or leaving the POI. Based on these results, the user's distance from the POI can be estimated.

Other methods of location inference work without the use an external database of place names. Chandra and Muhaya (2011), using only tweet and tweet reply content, demonstrated a method that can estimate user location within 100 miles of the actual user location 22 percent of the time. A 2013 study by David Jurgens demonstrates a method to infer Twitter user locations in a large social network with a small number of initially known user locations. While these methods can make location inferences for a sizable percentage of the posts, the city-level accuracy makes the results of the methods described above unsuitable for this project.

2.4. NoSQL for Big Spatial Data Storage

The increase in the amount of semi-structured data available to researchers has created challenges in data storage using relational database management systems (RDMS). In the RDMS model, the data schema (i.e. fields, tables, and relationships) is somewhat fixed in the design stage. This is fine if the database is to be tailored to a specific data source, but fixed schemas become problematic when considering multiple, disparate data sources with varied formats. Changing the structure of an RDMS database after it has been in use can result in errors and data loss if the changes are not executed with care.

Extended periods of data collection further complicate the use of RDMS in that the formats of individual data sources may change over time. Much of the data available on the web

comes in Extensible Markup Language (XML) or JavaScript Object Notation (JSON) formats where data are arbitrarily defined and often contain nested data points. These data must be reformatted and normalized to fit a tabular data structure. Operations for extracting, transforming, and loading data into an RDMS can require a lot of skill, be time consuming, and be prone to error (Tear 2014).

The problems mentioned above have led to the increased popularity of Not Only SQL (NoSQL) database systems for storing large volumes of semi-structured data. NoSQL systems are not full GIS suites, but there have been efforts to incorporate NoSQL storage capabilities into common RDMS-based GIS software through plug-ins (Peirce 2012; Kaliogirou and Boehm 2017). Others have written Python scripts that can process Esri shapefiles and store them in a NoSQL database (Zhang, Song, and Liu 2014).

MongoDB is a popular NoSQL document database system that supports the storing of semi-structured and nested data in Binary JSON format (BSON) and geospatial data (GeoJSON). MongoDB also supports two-dimensional geospatial indexing (2dsphere), and geospatial queries (near, geoNear) (de Souza Batista et.al. 2014). These features make MongoDB a good means of storing data from the web, especially data from social media networks where location is a component.

Even though geospatial support in MongoDB is limited to the topological functions of intersect and within (de Souza Batista et al. 2014), MongoDB's spatial data retrieval ability has been shown to be extremely efficient when compared to like functionality in ArcGIS (Duan and Chen 2015). While ArcGIS is a fully-formed geographic information system with rich spatial analysis capabilities, the efficiency demonstrated by MongoDB makes it suitable for data storage in support of location-based services on the web.

2.5. Application Considerations

The design of the open-source crime mapping application described in this document drew from existing crime mapping applications such as CrimeMapper.com and the NYC Crime Map. The application objective of spatiotemporally co-locating social media posts supports the overall goal of augmenting existing social media canvassing procedures to increase case clearance rates. Furthermore, its open-source nature and platform independence allows for easy implementation for departments that may lack resources for additional of social media canvassing.

Twitter data is used in this application for two reasons. The first reason is that Twitter is one of the "Big Six" social media sites recommended as starting points in social media canvassing strategy. Secondly, Twitter data is widely used in geospatial research, so the means of collection and storage are well-documented.

The NoSQL datastore, MongoDB v.3.4[2] was selected for this project because it too is well-documented and it provides the very functionality needed to achieve the application objective. Namely, its geospatial indexing scheme (2dshpere[3]) and support for a topological within-query ($near[4]) make the objective of spatiotemporal co-location of tweets with crime incidents possible. MongoDB's demonstrated efficiency in returning query results from large datasets make it desirable for use with web applications like the application described in this thesis. It is possible to store JSON or GeoJSON objects in a column (one object per cell) inside of a relational database like PostgreSQL. This was not done because doing so would add RDBMS functionality that wouldn't be used in the application and could have added overhead

[2] For documentation concerning MongoDB v.3.4, see: https://docs.mongodb.com/manual/
[3] https://docs.mongodb.com/manual/core/2dsphere/
[4] https://docs.mongodb.com/manual/reference/operator/query/near/

either to the reads from, or writes to the database, depending on the JSON datatype being used. Simply put, MongoDB is designed for storing JSON while relational database systems have added in support for JSON.

The last reason MongoDB was chosen over a RDBMS was its horizontal scaling ability. This doesn't come into play during the data collection period of this thesis, but could if the application where used for an extended period and required continuous data collection. In this case, the MongoDB could be easily distributed over many servers as storage needs increased. Horizontal scaling is possible with RDBMS but is complicated by the existence of complex joins and referential integrity rules that are not needed in a NoSQL database like MongoDB, nor are they needed for this thesis's application to function. Entities can be clearly represented in single objects rather than a series of tables joined by keys. MongoDB was built to handle JSON data, it supports the geospatial indexes and queries needed for this thesis's application to function, and is easily scalable if storage needs increase. The following chapter outlines this application's design and functionality essentials.

Chapter 3 Application Requirements

Application objectives and user needs were reviewed to make decisions on the application's design and backend functionality. The following sections state the goals of the application, user requirements, and functional requirements.

3.1. Application Objective

The application objective is to map crime incidents and then map tweets that are spatiotemporally co-located with the crime incidents. The mapping of crime incident locations is based on a user-specified window of time. In the application, a window of time can be one or many days and is defined by the user with date pickers for the window's start and end dates. Crimes occurring within the user-defined dates are displayed on the map. Information associated with the mapped crimes is displayed in an information window when a crime incident location is clicked by a user. Next, the application maps locations of tweets based on three user-defined criteria: a crime incident location, a radial distance from the incident location, and a window of time surrounding the incident. Tweets that fall within the bounds of the criteria are displayed on the map when a user clicks a homicide incident. By clicking on a tweet on the map, users gain access to its content and associated metadata in a scrollable list which can be analyzed for information relevant to the co-located homicide.

3.2. User Requirements

To meet the goals of this application, the user must be able to map homicide incidents based on a start date and an end date. To define the boundaries of spatiotemporal co-location, the user must also be able to specify a time and distance from a crime incident. Inputs for these

criteria are provided in the application as text boxes. For date inputs, a date picker is associated with the textbox.

The specification of spatiotemporal co-location must be flexible based on the nature of the crime and the needs of the investigator or public user. For example, an investigator may be looking for eye-witnesses to the crime itself requiring a spatiotemporal co-location definition with an elevated level of specificity. Alternatively, an investigator may be looking for witnesses who heard events leading up to, during, and after the crime took place so the definition of spatiotemporal co-location could be broadened to include a large area and a wide window of time. Therefore, spatiotemporal co-location is defined by user inputs. As mentioned above, these inputs include a window of time surrounding the homicide (in minutes) and a radial distance from the homicide (in meters).

3.3. Functional Requirements

In considering the user requirements, there are some operations that the application must handle in the backend. Beginning with the user's first interaction, selecting a window of time to map homicides, the application must take the user input and create a query to retrieve homicide data. This is accomplished with some JavaScript that takes the start and end dates, entered by the user, and fills placeholder variables in a query to the Socrata Open Data API used by the City of Chicago for sharing crime data. When the data is returned, another JavaScript function creates markers from the data and places them on the map.

With the homicides mapped, the user defines the parameters for spatiotemporal co-location for the tweets. These inputs must be turned into another query that retrieves data from the GeoJSON file containing the collected Tweets. Again, JavaScript is used to complete this interaction from forming the query to mapping the query result.

3.4. Application Design

Open-source and free-to-use software was used for this application with the web as its platform. Ease-of-use considerations were made so that the user could understand the purpose of the application without a great deal of instruction. The following subsections justify the choices made in the software, the platform, and the user experience design.

3.4.1. Software

As noted above, open-source and free-to-use software and technologies were used to create this application. Atom.io was the integrated development environment used to create the web page, PyCharm was used to write the Python scripts used for data collection and MongoDB was used to store the data. For data assessment, PyCharm was used to extract data from the MongoDB database and QGIS was used to create maps from the data.

While proprietary software, such as ArcGIS Online, may have everything needed to build this application pre-baked, proprietary software can be costly both in terms of money and in terms of freedom. When seen in this light, software licensing fees are detrimental to the repeatability of research as only those with access to licenses, either through their own bank account or through the institutions they are members of, can take part. The open-source components of this project will allow for greater repeatability because of the inherent liberation of open-source software. Anyone with an internet connection could repeat and improve upon this research without concerning themselves with licensing restrictions.

3.4.2. Platform

Since this web application is a proof of concept, it is designed primarily for use on desktop or laptop computers with a web browser. Being browser-based, it can also run on any device with a web browser installed such as smartphones and tablets. Some modifications might

be necessary to make the application easier to use with smaller displays. Although not an objective of this project, this could be accomplished by adding separate cascading style sheets (CSS) for smaller screens and some code allowing the application to recognize what type of device the application is running on.

HTML, CSS, and JavaScript are easy to use and one can focus on what issues need to be solved regarding the application itself rather than issues that come up because of cross-platform inconsistencies. If the application turns out to be useful, it will be easier to port to specific mobile platforms like Android or iOS because the workings of the application have already been mapped out.

3.4.3. UX Design

While the application goals are reached with backend operations querying and then displaying the data on the map, the user experience must be considered. Forms for entering query criteria are intuitive, but also collapsible to provide a better view of the map. The JavaScript libraries JQuery and JQuery-UI are used to ease input (date pickers) and make the forms containing the controls collapsible.

There also must be an area to display data associated with homicide incidents as they are selected by the user. JavaScript is used to display data associated with a marker in an info window off to one side of the map. This decision was made to avoid the default info balloons cluttering up the map and covering other homicide or tweet locations. Chapter 4 describes the development of the application based on the requirements outlined in this chapter.

Chapter 4 Application Development

The application is composed of five parts: a webpage to display the application; the Google Maps API to retrieve a basemap; the Socrata Open Data API (SODA)[5] to access the crime data hosted on the Chicago Data Portal at cityofchicago.org; a virtual private server (VPS) responsible for tweet collection and storage; and, the Twitter Streaming API that provides access to the tweets. A diagram of the application's structure and the tools used to create the application are shown in Figure 2 and Table 1 respectively.

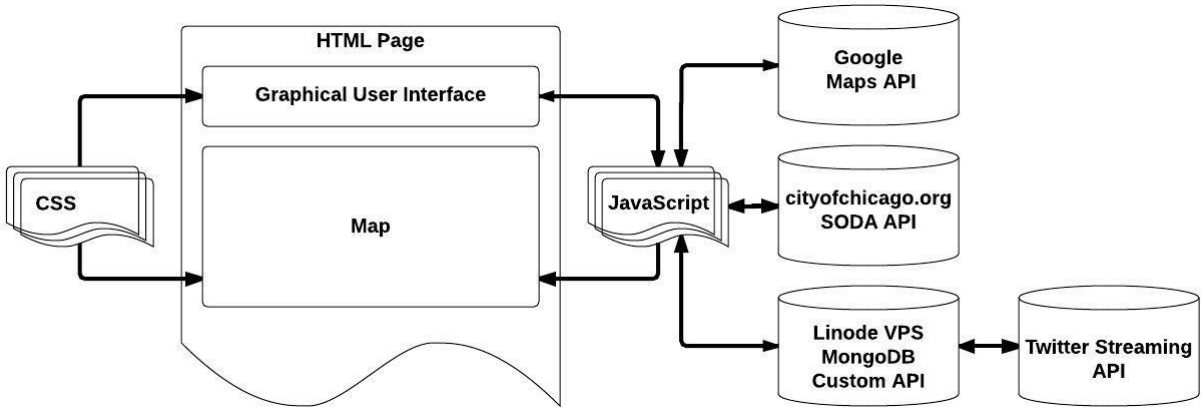


Figure 2: Application structure

[5] For documentation on SODA, see: https://dev.socrata.com/foundry/data.cityofchicago.org/6zsd-86xi.

Table 1: Technologies used in the creation of the application.

Application Area	Technology	Functionality Description
Crime Data	SODA	Used to access the Chicago crime database at the Chicago Data Portal.
Twitter Data	MongoDB	Stores the tweets as GeoJSON documents.
	Node.js	Enables the use of JavaScript on the server side.
	Mongoose.js	Opens a connection to the MongoDB (requires Node.js)
	Hapi.js	Used to create the RESTful API that allows the application to access tweets from the MongoDB database.
Webpage	HTML	Defines the content of the web application page.
	CSS	Defines the layout and style of elements on the web application page.
	JavaScript	Defines the Google Map, responds to user inputs, and build RESTful queries to the two APIs used in the application.
	Google Maps API	Allows the embedding of Google Maps with JavaScript.
	Jquery/JQueryUI	Simplified the selecting of form elements to retrieve values and creating date pickers.

The following sections describe each component in detail, starting with the Chicago Data Portal and the MongoDB database of Twitter posts, and ending with the webpage.

4.1. The Chicago Data Portal

The application gets crime incidents and their related data from The City of Chicago Data Portal. The information is available as a downloadable table called “Crimes – 2001 to present”. The data is also made available, for free, via a RESTful API called SODA. A piece of JavaScript code created for the application builds a RESTful query URL to SODA, and manages the response. The response object from SODA is in JSON format. JSON is a data format that is easy for humans to read and write, and easy for computers to parse. A normal JSON document consists of a JavaScript array containing a list of objects. Information on the JSON format is

available online[6]. In the case of this application, is an incident of crime represented by a series of key : value pairs. The application uses JavaScript to iterate over this list of crime incidents to display them as markers on the map.

4.2. The Database of Twitter Posts

The second database was created specifically for the application. It is not possible to query Twitter's full history via Twitter's free APIs. Therefore, it was necessary to first collect the data and then make it available to the application. The first of the following subsections describes the methods for data collection and storage. The second subsection describes how the twitter data was made available to the application.

Data collection and data storage both happen on a Linode virtual private server (VPS) purchased for the purposes of this project. Linode is a cloud hosting company, like Amazon AWS or Rackspace, but with an eye toward economy. For example, the VPS purchased for this project costs five dollars per month rather than around 205 dollars per month for an Amazon AWS account. The idea behind this application is simple and requires few of the bells and whistles included with Amazon's service. This makes a barebones (yet customizable) VPS ideal for this project.

It was decided to run the collection script on the VPS for two reasons. First, it was more stable than a home internet connection available to the author and prevented interruptions in the running of the collection script. Second, since the application requires the results of data collection to be stored in an online document database, it was easier to run the collection script on the same machine (VPS) that is responsible for storing and serving the data. Twitter data collection began on May 3, 2017 and ended on July 2, 2017 using the TwitterAPI.

[6] For a description of JSON see: http://www.json.org/.

TwitterAPI is a Python library, available from GitHub, for interacting with Twitter's Search and Streaming APIs. With this library installed on the VPS it was possible to write a Python script that connects to Twitter's streaming API, filter the response with a bounding box surrounding Chicago, and log each Twitter post to the console. The script written to collect tweets for this thesis is available in Appendix A.

It should be noted that the Twitter Streaming API only allows free access to a one percent sample of Twitter's total traffic. However, if a filter constrains the result to something less than the one percent cap, all Twitter posts that fit within the filter's criteria will be returned (andypiper 2014).

Since each post in the Twitter API response contains a lot of extraneous metadata, the Python script was also used to select the portions that would be used by the application. These include user_ID and screen name, geographic coordinates, post content, a timestamp, and any URLs that were included in the post. To save space in the database, only the information needed by the application was extracted from tweets. For each Twitter post, the Python script takes these pieces of information and creates a GeoJSON[7] document and pushes the document to a MongoDB document database (See Figure 3).

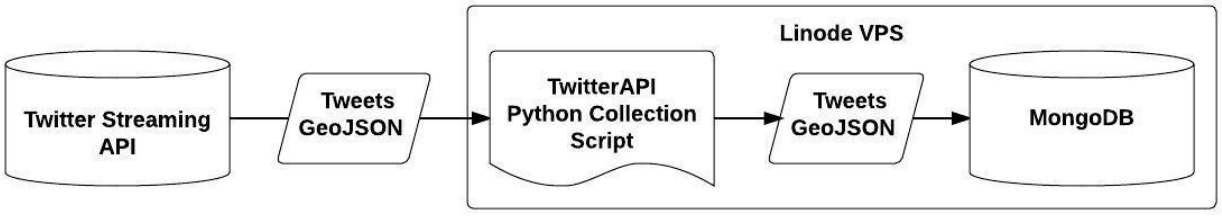


Figure 3: Data collection from the Twitter Streaming API and storage on the Linode VPS.

[7] GeoJSON is a data format based on JSON that supports the storage of a variety of vector types. For GeoJSON standards, see: https://tools.ietf.org/html/rfc7946.

Twitter posts collected for this application are stored in MongoDB database. MongoDB stores documents in JSON format making it ideal for storing the GeoJSON files resulting from the Python collection script. Another excellent feature of MongoDB is the ability to create spatial indexes (if coordinate fields are available). MongoDB's 2dshpere spatial indexing scheme was used to make spatial queries of the twitter data possible. The index was created once during data collection. New tweets were indexed as they were added to the database and there was no observed slow-down or stalling of the data collection script.

With the data stored and spatially indexed, the data then needs to be available to the application as a RESTful API response, the same way as the crime incident data comes from the SODA API. The API needs to accept the criteria for spatiotemporal co-location (the location of a crime incident, a window of time surrounding the incident, and a radial distance from the incident), make a spatial query to the document database, and return the results from the query. To accomplish this, a JavaScript framework called hapi.js was used to create an API endpoint. The endpoint accepts variables for the definition of spatiotemporal co-location, supplies these to a spatial query of the document database, and returns the results to the application in JSON format. Code for this custom API can be seen in Appendix B of this document. The application is then able to iterate through the objects in the response (each object is a tweet) and display them on the map.

4.3. The Web Page

The frontend of the web page is made up of an embedded Google Map, a form for user input of query variables, a window for displaying information about crimes and co-located tweets, and a legend describing map symbology. The backend consists of JavaScript responsible for map display, processing user input, and constructing queries of the external databases. Figure

4 shows the full application window and the following subsections describe each part of the frontend and backend in greater detail.

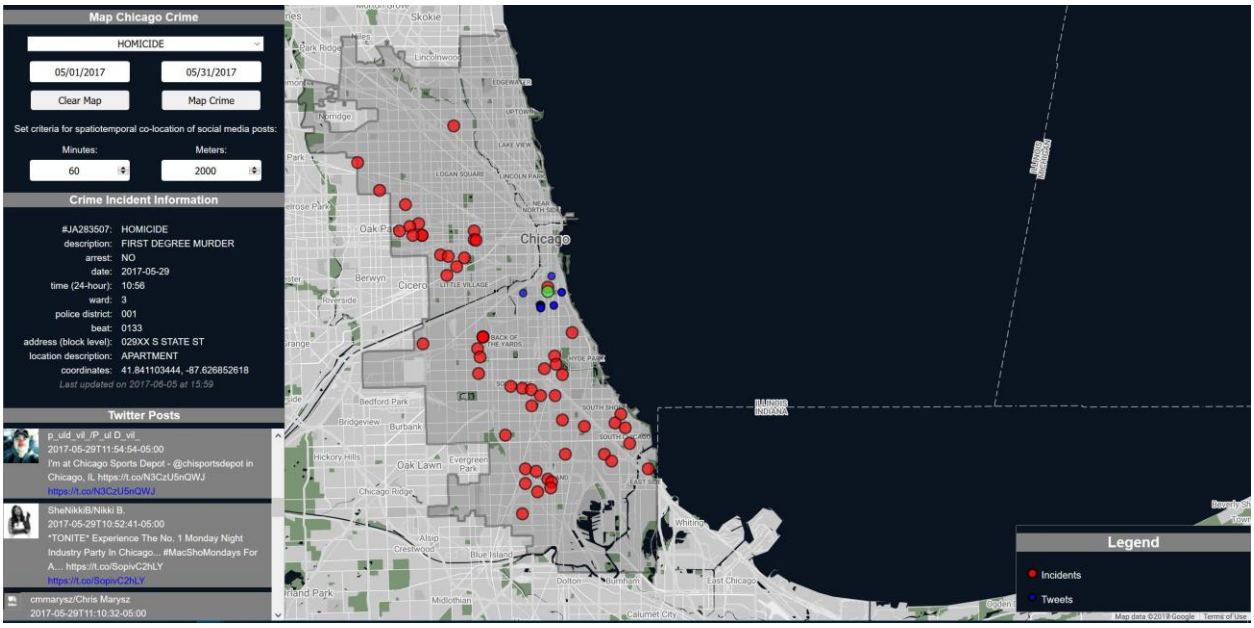


Figure 4: Full application window.

4.3.1. The Basemap

The most prominent feature of the web page is an embedded basemap created using the Google Maps API for JavaScript. Embedding a Google map involves creating an HTML container, usually a element. The container is given an “id” (in this case simply “map”) so that it can be accessible to a piece of backend JavaScript. This script is responsible for retrieving map data from Google and displaying the data inside the “map” container. Rules for displaying the container, i.e. height, width, and page position, can be described either in-line within the HTML document, or, as in the case of this application, an external CSS file referenced in the HTML document. Styling the map data itself was accomplished by adding style definitions for each element of the Google map to the JavaScript responsible for map display (Google 2017).

With the basemap in place, a polygon outlining the City of Chicago was added for user reference as Chicago's boundary is not apparent from a normal Google map. The boundary polygon was created from a KML file downloaded from The City of Chicago Data Portal and was last updated in 2011. While it is possible to overlay a polygon onto a Google Map directly from a referenced KML file, it was discovered that doing so had an undesirable effect on initial map loading and map resizing. Therefore, the decision was made to include the polygon in the map definition in the JavaScript responsible for map display. The map definition used in this project can be seen in Appendix C.

The process for including the boundary in the Google map definition required converting the multi-part KML polygon to a series of JavaScript arrays that could be used in defining a Google Maps polygon. The two formats are quite different. The shape of KML polygons are defined by comma separated coordinates. Coordinate pairs (or triads if a z-value is included) are separated by spaces. These sets are then enclosed in KML `<coordinates>` tags. In a Google map, polygons can be described with arrays of key : value pairs.

The area of the City of Chicago is represented by a multi-part polygon consisting of five parts. For each part, the contents between the `<coordinates>` tags were copied to text and treated as a .CSV file in Microsoft Excel, where each coordinate pair had its own row and each coordinate had its own column. A formula was written to create the appropriate key : value pairs and JavaScript array definitions in the empty cells adjacent to the coordinates. The results for each part were then copied and pasted into the code used to define the Google Maps polygon in the application. Finally, a visual inspection was made to ensure that the resulting Google map definition aligned with the original KML with no errors. The result is shown in Figure 5.

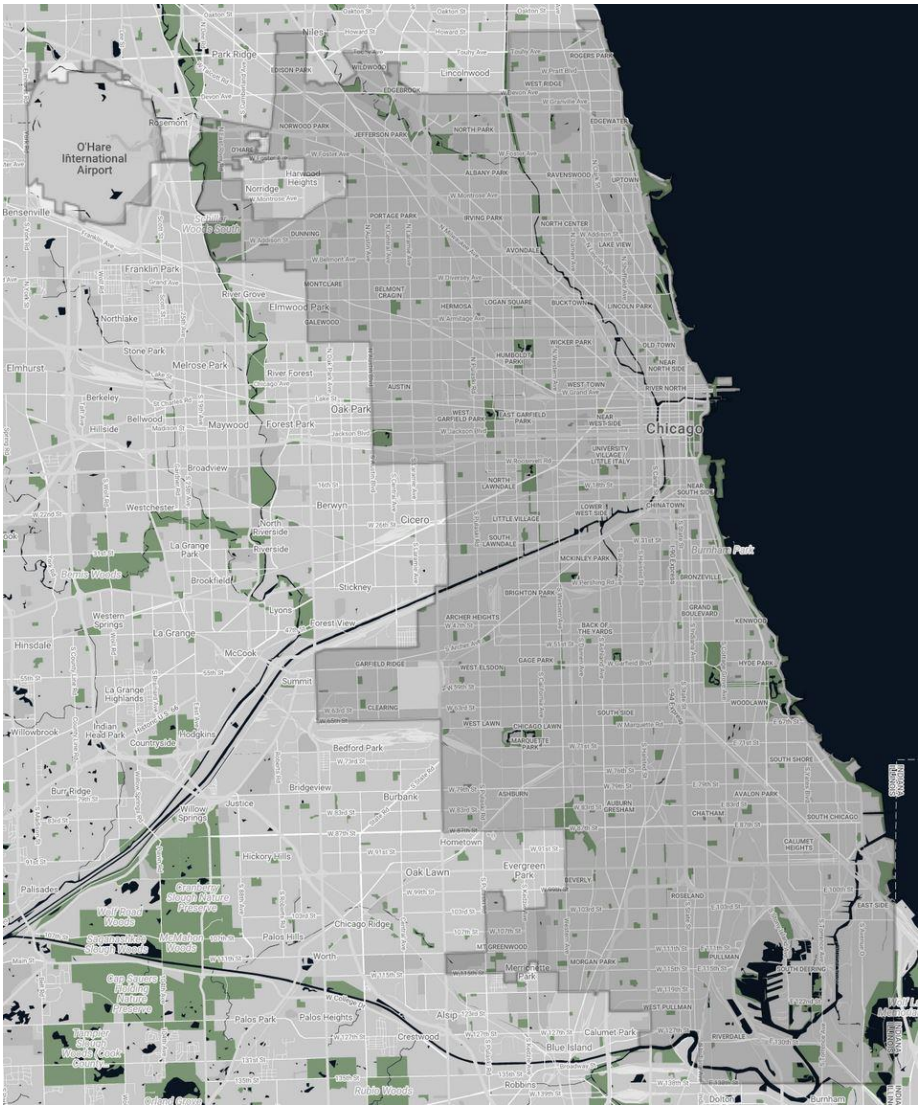


Figure 5: Chicago boundary polygon in Google Maps.

While the use of a formula in Excel prevented human error, there are still opportunities for error. For large numbers of simple polygons or more complex multi-part polygons, it would be more efficient to write a script for doing this type of conversion in batches. For this project, it was easier to save the coordinates to a .CSV file, do the conversion with an Excel formula, and paste the results into code.

4.3.2. The Query Form

At the top-left of the window is a form that allows users to enter variables, shown in Figure 6. The first line of the form has three variables, a type of crime (e.g. assault, homicide…etc.), a start date, and an end date. Crime type is selected via a drop-down menu, and dates are selected via a date picker to ensure proper formatting. The drop-down menu is populated with a query to the SODA API that is executed when the page loads. The query selects distinct crime types from the database and, once these are returned, sorts them alphabetically and injects the list into the HTML defining the drop-down.

Figure 6: Application query form.

Once the variables for crime type and a window of time are set, the user then presses "Go". Crime data fitting the user criteria is shown on the map as a series of points. With the crimes mapped, the user may then set the variables in the second line of the query form. The two variables set the definition of spatiotemporal co-location for the social media posts. The first variable sets the time in minutes before and after a crime is committed. The second variable sets a distance in meters and defines a buffer radius around a crime incident. The defaults for spatiotemporal co-location are set to 10 minutes and 150 meters. The form also has a button for clearing the map.

4.3.3. The Information Window

When a user clicks a crime incident on the map, information about the incident is displayed, in table form, in a box on the left side of the application window. Items in the table include: case number and crime type, an incident description, whether there was an arrest, incident date, time of day, ward, police district, block-level address, a description of the location, and geographic coordinates. This information comes from the City of Chicago Data Portal (see Figure 7).

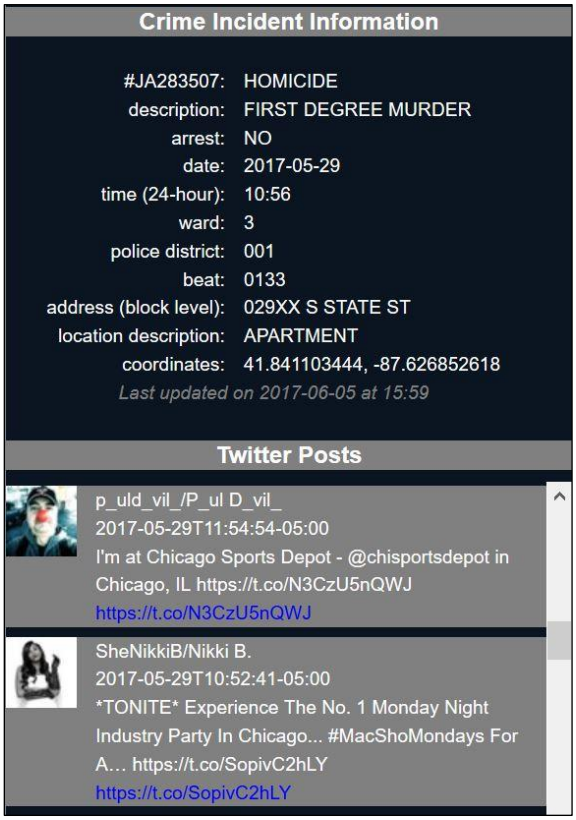


Figure 7: Application information box with crime data table and scrollable list of tweets.

Below the table of incident information, but in the same box, spatiotemporally co-located Twitter posts are listed if any are found for the specified criteria. For each Twitter post, the

user's profile image, user ID, screen name, time stamp, content, and any included URLs are displayed.

4.3.4. The Webpage Backend

The design of the webpage backend is standard and, like many websites, it consists of HTML, CSS, and Javascript files: HTML for what content is to be displayed; CSS to describe how that content is displayed; and, some JavaScript (with some JQuery) that responds to user interactions. JavaScript is also used to display the Google Map basemap, build RESTful queries from user inputs to the query form, and finally, to create markers from the returned data and display their locations on the Google Maps basemap.

Chapter 5 Application Assessment

The objective of creating a crime mapping application that matches crime incident locations with spatiotemporally co-located tweets has been met. Crime incident locations are mapped and clicking these locations updates the information window as expected. If there are spatiotemporally co-located Twitter posts, given the time and distance query variables, their locations are mapped and their content is provided in a scrollable list in the Twitter post window. This chapter reports on the performance of the application and its unexpected results.

5.1. Application Output

Unfortunately, assessment of the application's ability to achieve this project's goals of accelerating case clearances is difficult. It turns out that most crimes that occurred during the collection period have no meaningful spatiotemporal co-location with location-tagged Twitter posts. For the purposes of this thesis, meaningful spatiotemporal co-location is defined as within one average Chicago block (150 meters) and within 10 minutes of the incident. Another issue is that none of the few spatiotemporally co-located posts contain content related to their co-located incidents. While it was expected that there would be few Twitter posts containing relevant information, it was assumed that there would be some. This turned out not to be the case.

The result is an application where the user clicks crime incidents looking for spatiotemporal co-locations and little happens until the spatiotemporal co-location variables are broadened so much as to become meaningless. For example, of the 64 homicides that occurred in May, only one homicide had what could be considered a meaningful co-location. Furthermore, if the user had mapped all 64 homicides in the month of May, finding this incident and its co-located Twitter post would involve clicking incidents randomly until discovering the spatiotemporal co-location.

Since it was not possible to confirm by looking at the application's output that it was indeed functioning as intended, a series of tests were made on the Twitter database to confirm the sparseness of spatiotemporally co-located Twitter posts and find out how many exist. To better understand the roles time and space play in spatiotemporal co-location, the data was explored by charting the number of tweets versus the number of crime incidents by hour of the day during the collection period. Then, crime incidents that have spatiotemporal co-locations with tweets were mapped along with location-tagged tweets using QGIS. Finally, the contents spatiotemporally co-located tweets were analyzed and categorized to see if they contained relevant information.

5.2. Assessment of Query Results

Five crime types were analyzed along with the Twitter data. Each of the two months of the collection period was examined individually so that any extreme month-to-month variation could be observed. For simplicity, the periods from 5/3/2017 – 6/2/2017 and 6/3/2017 – 7/2/2017, are referred to as "May" and "June" respectively. Each crime type was totaled and then each record's position was compared to the locations of the records in the Twitter data.

It was found that spatiotemporal co-location of Twitter posts with incidents of crime is rare for a few reasons. Most crimes happen at times when Twitter traffic is low, and for most crime types, the crime incidents are not occurring in areas that generate high volumes of location-tagged Twitter posts. Finally, some crime types, such as kidnapping, are rare, further reducing the opportunity for meaningful spatiotemporal co-location. The following sections describe the process for analyzing the crime and Twitter data and provide detail of the results.

5.2.1. Crime, Location-Tagged Tweets, and Spatiotemporal Co-Location Totals

To begin the analysis, the number of incidents for each of five crime types was totaled. The crime types considered were arson, assault, criminal sexual assault, homicide and kidnapping. These crime types were chosen because they share severity and a lack of case clearances/arrests, and because they differ greatly in their total occurrences per month. The totals for each crime type were generated by importing the JSON responses from the Chicago Data Portal into a MongoDB collection and counted using MongoDB's count query. The resulting totals are available in the Table 2 below.

Table 2: Crime totals

CRIME_TYPE	MAY_TOTAL	JUN_TOTAL
Arson	37	43
Assault	1914	1879
Criminal Sexual Assault	238	228
Homicide	64	84
Kidnapping	18	18

A total of 359,569 location-tagged tweets were collected during the collection period: 174,416 in May and 185,153 in June. Since a bounding box was used to filter the query to the Twitter Streaming API, many of the location-tagged tweets fell outside the boundary of Chicago. These were removed using QGIS 2.18 to select and export to a new layer containing the tweet locations within 150 meters of Chicago's boundary polygon. Any tweet farther than 150 meters from the boundary can be ignored in this analysis as it would not produce a meaningful co-location with Chicago crime incidents. The counts for location-tagged tweets are shown in Table 3.

Table 3: Location-tagged tweet totals for May and June.

MONTH	LOCATION_TAGGED_TWEETS	IN_BOUNDARY	PER_DAY
May	174,416	105,272	3,395.87
June	185,153	109,702	3,656.73
Total	359,569	214,974	3,524.16

The next steps in evaluating the data were to find all the crime incident locations that had meaningful spatiotemporal co-locations, count them, and find their position relative to the rest of the data. Python was used to separate crime incidents with meaningful spatiotemporal co-locations so that they could be counted (See Table 4 and Table 5).

Table 4: Count of incidents with meaningful spatiotemporal co-location in May.

CRIME TYPE	WITH_CO-LOCATIONS	PCT_WITH_CO-LOCATIONS	TOTAL
ARSON	0	0.00%	37
ASSAULT	27	1.41%	1914
CRIMINAL SEXUAL ASSAULT	0	0.00%	238
HOMICIDE	1	1.56%	64
KIDNAPPING	0	0.00%	18

Table 5: Count of incidents with meaningful spatiotemporal co-location in June.

CRIME TYPE	WITH_CO-LOCATIONS	PCT_WITH_CO-LOCATIONS	TOTAL
ARSON	0	0.00%	43
ASSAULT	26	1.38%	1879
CRIMINAL SEXUAL ASSAULT	1	0.44%	228
HOMICIDE	0	0.00%	84
KIDNAPPING	0	0.00%	18

5.2.2. Time of Day and Spatiotemporal Co-Location

Low Twitter traffic at times when crimes are occurring would obviously have a negative effect on the chances of meaningful spatiotemporal co-location with Twitter posts. To observe

this relationship, a mixture of Python scripting, Microsoft Access, and Microsoft Excel was used to export time of day information to .CSV files, perform aggregate queries, and chart the results respectively. The final charts show the number of tweets vs. the number crime incidents for each hour of the day during each month. The processing flowchart is outlined in Figure 8.

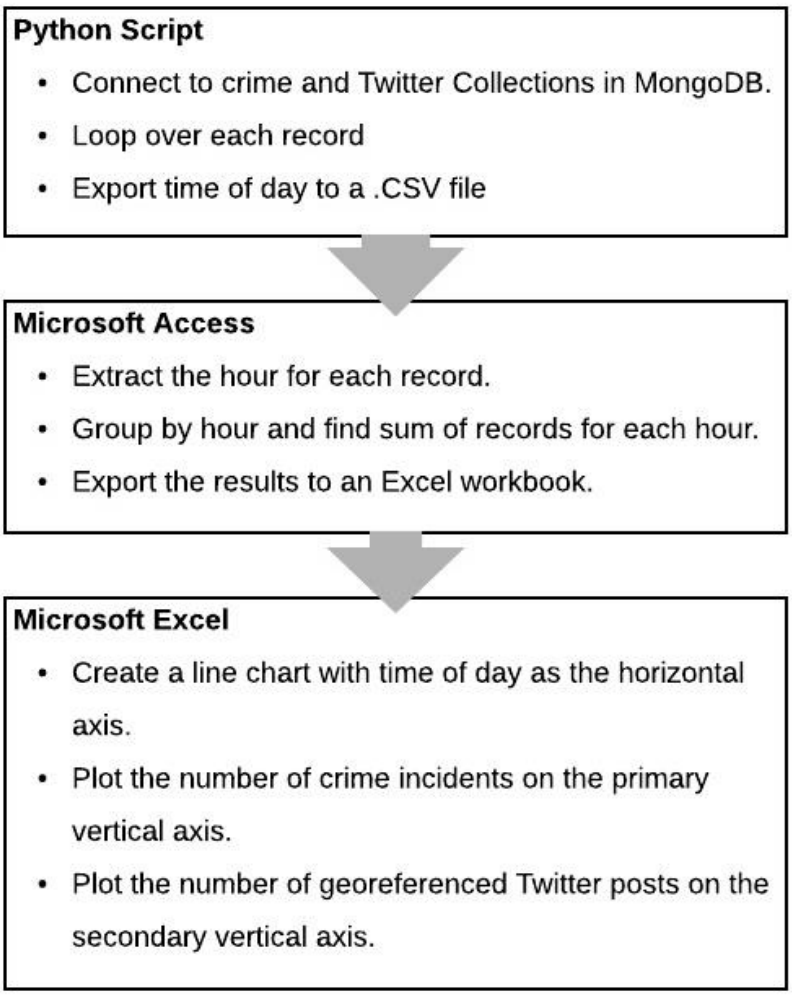


Figure 8: Time of day data processing flowchart.

This process was performed on the Twitter data and five types of crime that included arson, assault, criminal sexual assault, homicide and kidnapping and was repeated for both

months in the collection period. The ten resulting charts are available in Appendix D while the more interesting ones are shown and discussed in this section.

For most of the crime types analyzed, the charts indicate that the number of crime incidents is higher at times of day when the number of tweets is low or middling and that patterns are similar month-to-month. Spikes in criminal activity usually occur when Twitter traffic in Chicago is low but there are some notable exceptions. In the chart for May homicides (Figure 9), the count of homicides peaks at 5 o'clock in the evening at the same time as the peak in Twitter traffic.

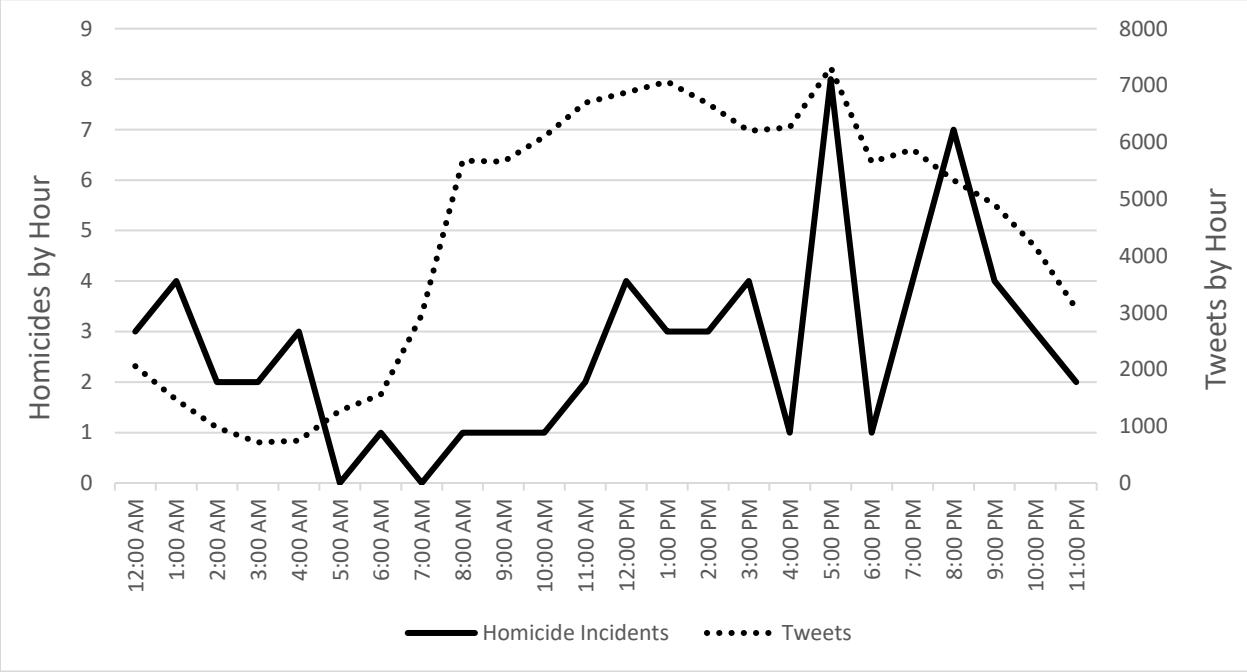


Figure 9: Comparison of homicide incidents and tweets by hour in May.

In another example, the May kidnapping count peaks when Twitter traffic was high at 9 o'clock in the morning (Figure 10).

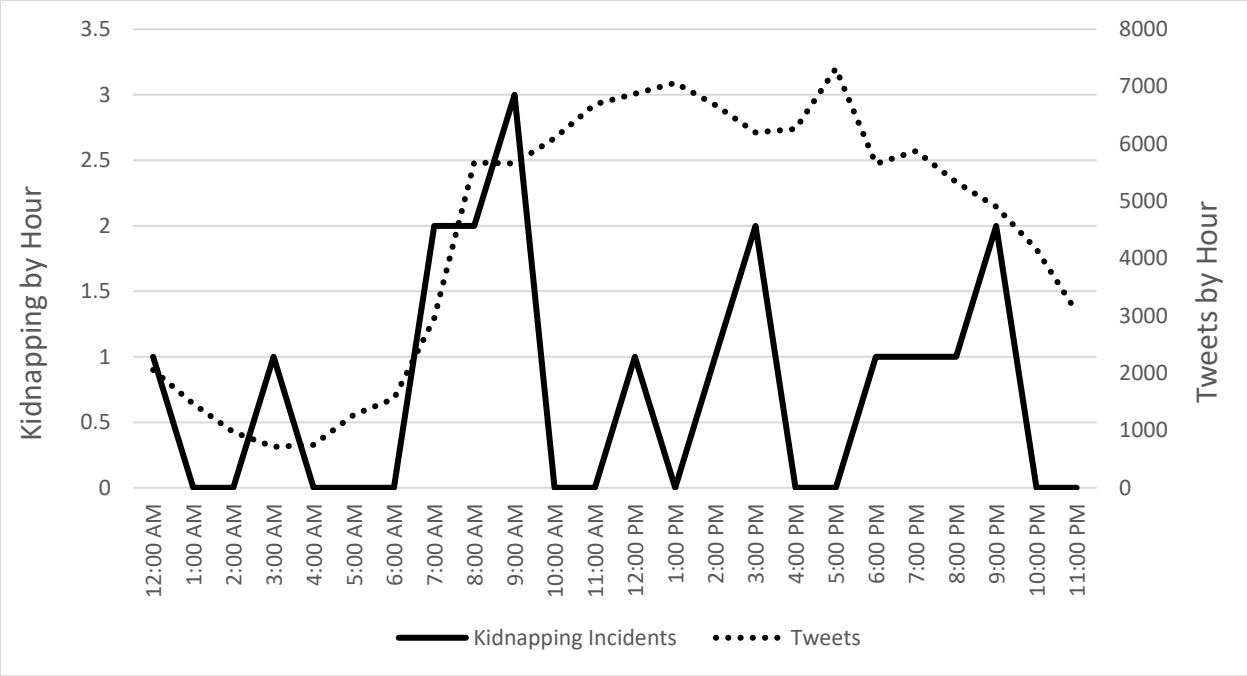


Figure 10: Comparison of Kidnapping Incidents and Tweets by Hour in May.

Peaks in the criminal activity at times when Twitter traffic is high do not necessarily guarantee meaningful spatiotemporal co-location. May homicides generated only one meaningful spatiotemporal co-location and the post occurred at 11:55 at night, when Twitter activity was low. Furthermore, this post turned out to be a traffic reporting Twitter account warning of police activity in the area. It is unlikely that this post would generate potential witnesses or provide any additional information about the crime. In the case of the kidnapping peaks during May and June, no meaningful co-locations between kidnapping incidents and Twitter posts were found. Kidnappings were rare in comparison to homicides, further diminishing the opportunity for meaningful spatiotemporal co-location.

Assault was the only crime type analyzed that consistently showed high rates of occurrence at times when Twitter traffic was also high. Assaults were also much more prolific during both months of the collection period (see Figure 11 and Figure 12). These factors increase

the chances of meaningful spatiotemporal co-location between assault locations and Twitter posts.

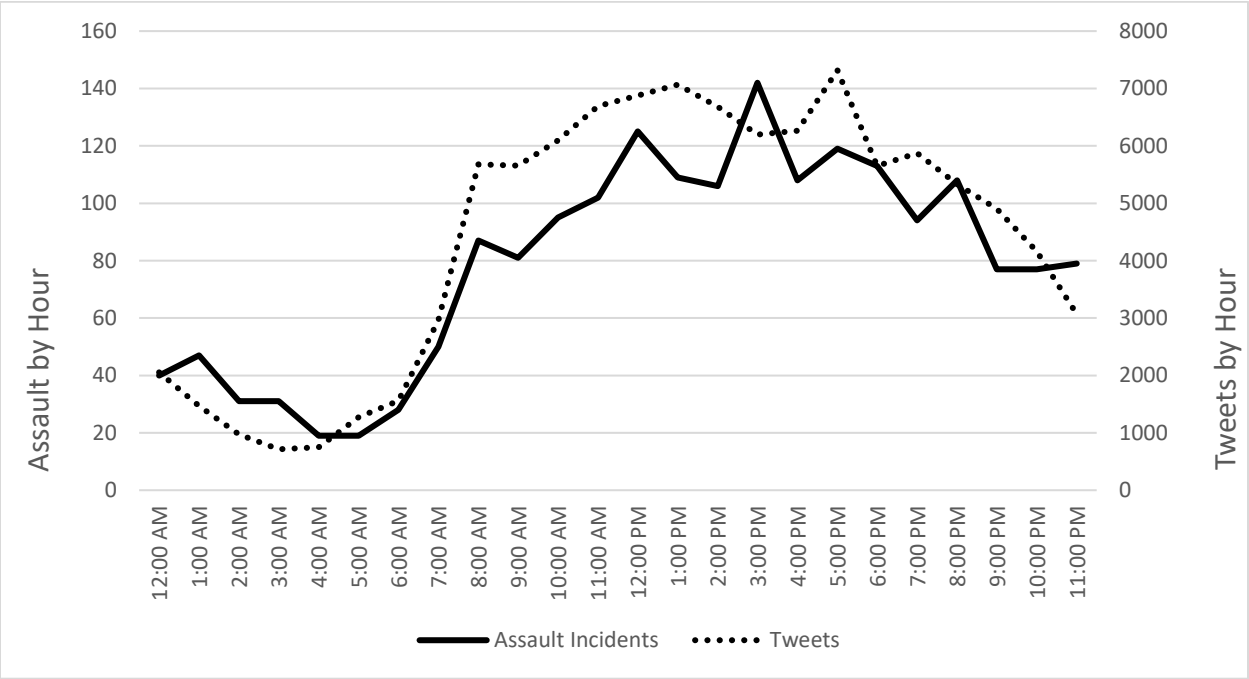


Figure 11: Comparison of assault incidents and tweets by hour in May.

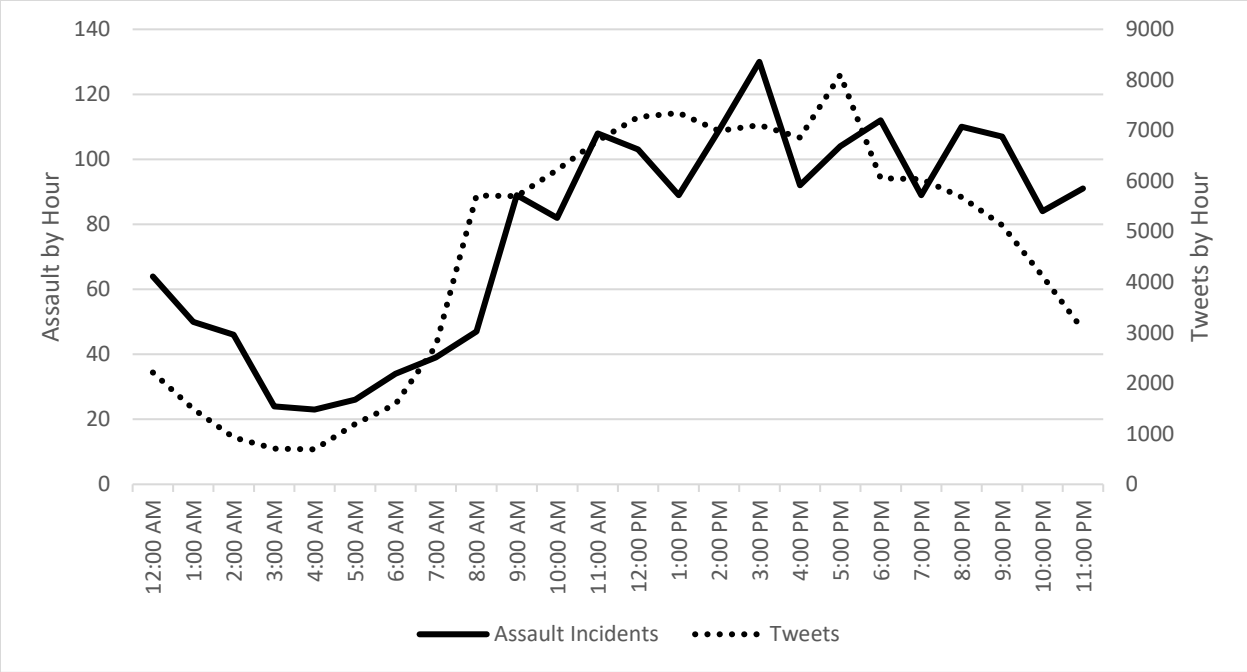


Figure 12: Comparison of assault incidents and tweets by hour in June.

Here, it was observed that only assaults had data worth closer scrutiny. The assault incidents with meaningful co-locations were mapped in QGIS for the months of May and June. Visual inspection of the resulting maps confirmed that most of the assault incidents that had meaningful co-locations were clustered in areas with a high point density of tweets. This was the same for both months of the collection period with two thirds of the assault incidents clustered in an area near Chicago's downtown (see Figure 13 and Figure 14).

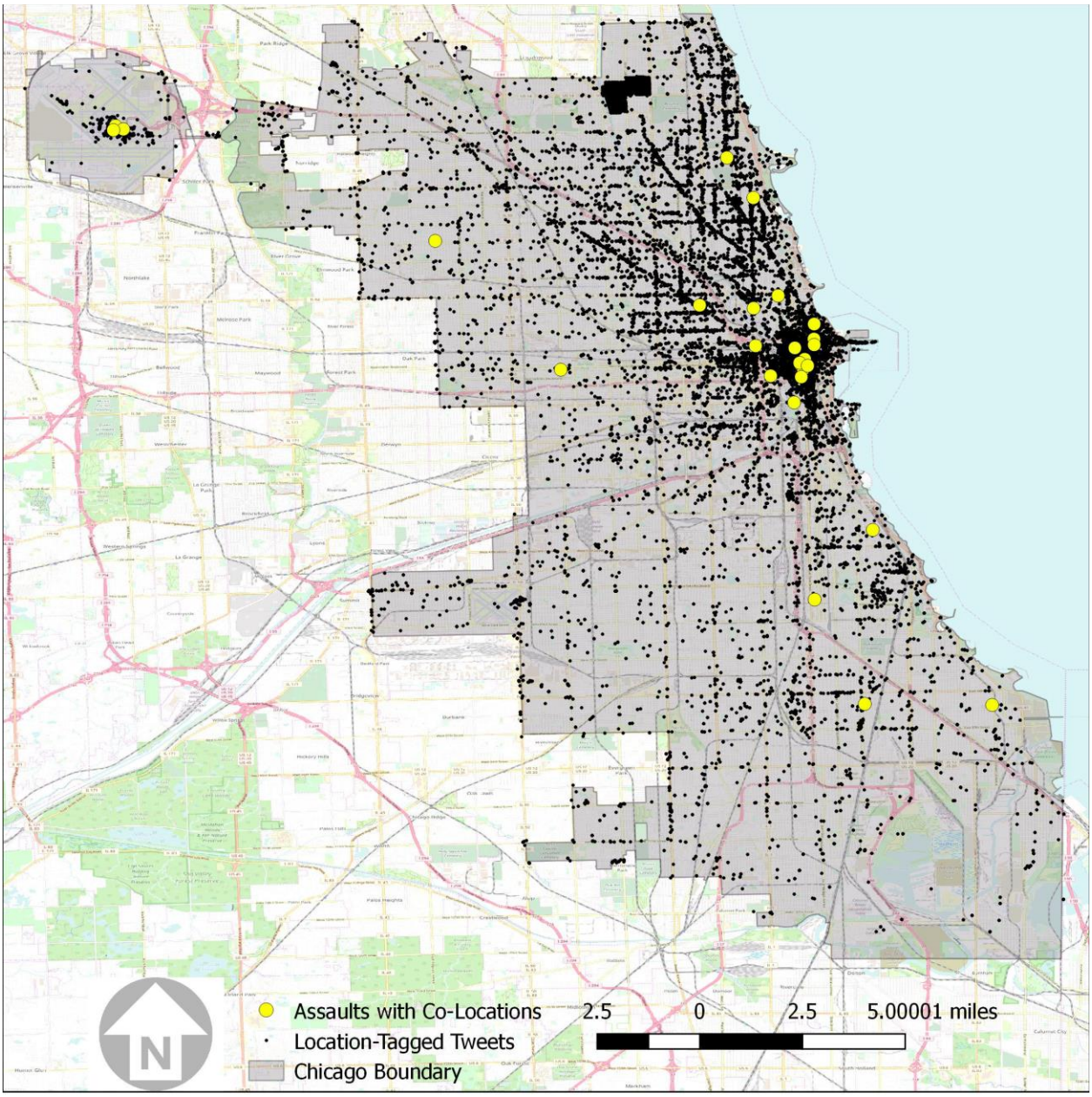


Figure 13: Geographic comparison of co-located assault incidents and all tweets in May.

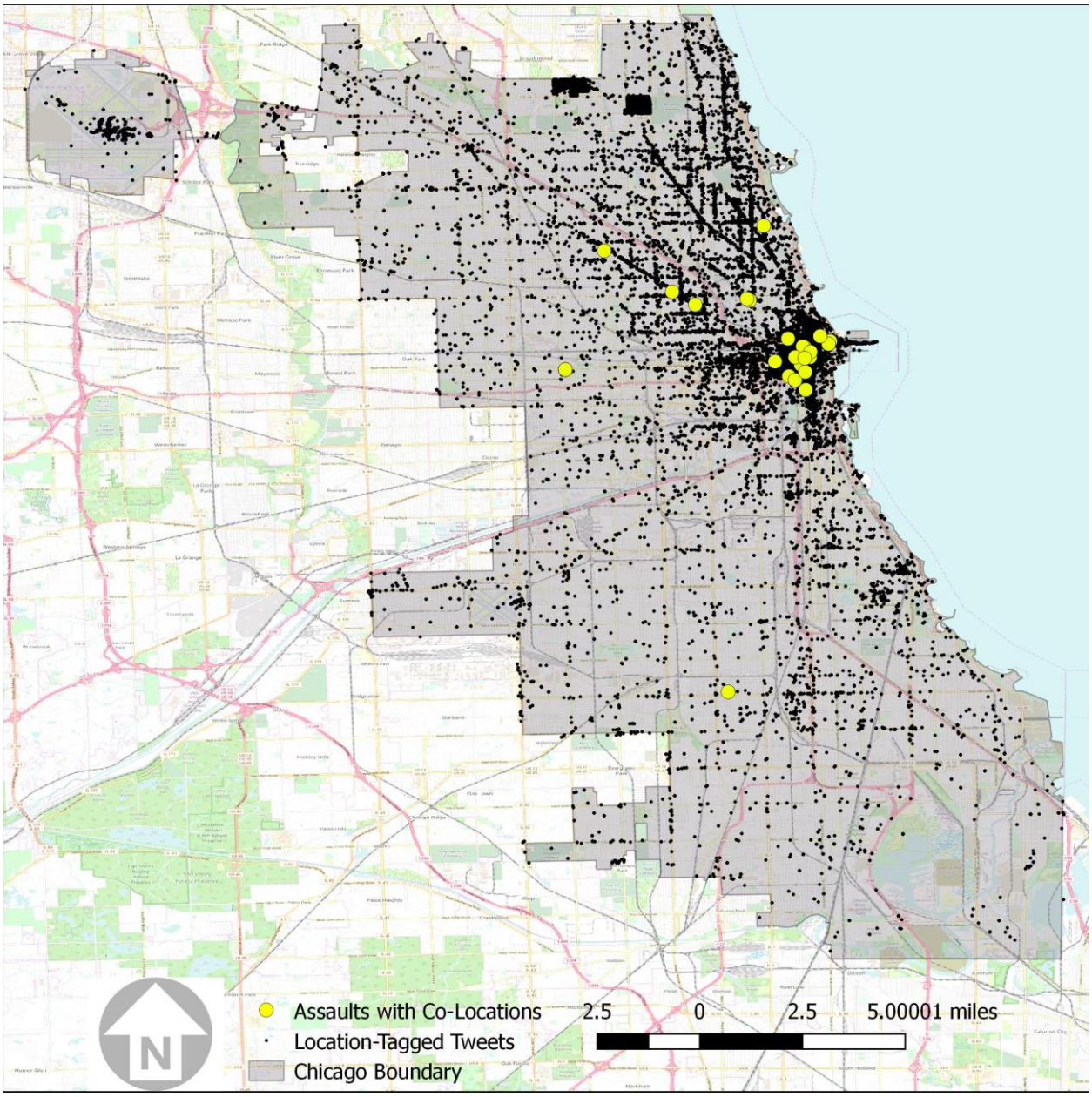


Figure 14: Geographic comparison of co-located assault incidents and all tweets in June.

To see if any of the spatiotemporally co-located tweets had relevant information to their co-located crimes, the last step taken in data analysis was to inspect the content of the spatiotemporally co-located tweets. Comma separated value output from the same Python scripts used find the crime incidents and meaningfully co-located tweets was opened in Microsoft Excel so that each of the co-located tweets could be easily read and categorized by content. Based on

the observed content, four categories were developed: “A” for automatic/advisory, “C” for check-in, “J” for job posting, and “P” for personal posts of arbitrary nature. It was found that the “A” and “J” tweets, which made up nearly 20% of the total, were of little use and efforts to eliminate these in the future should be taken. The remainder, consisted of about 44 percent “C” category, and about 36 percent “P” category (see Figure 15).

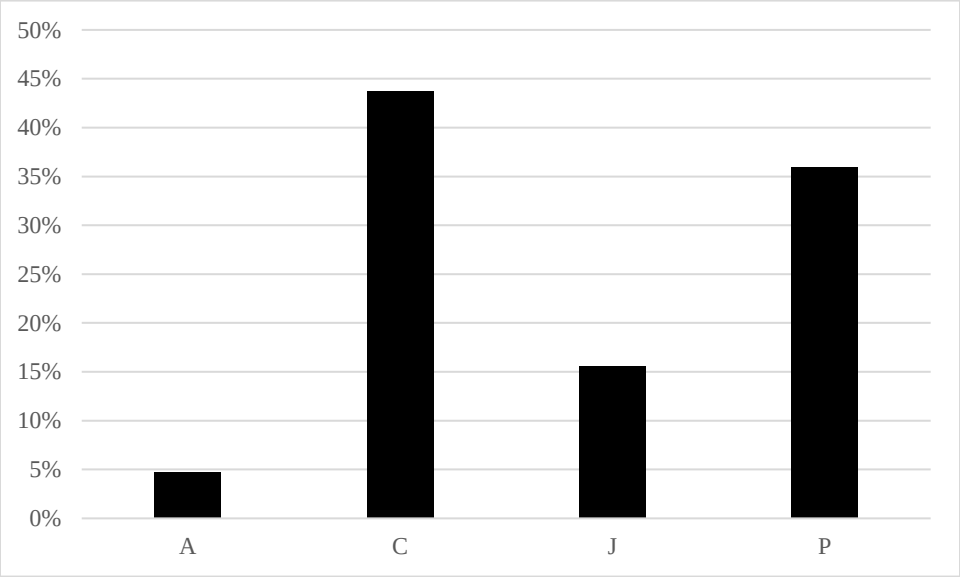


Figure 15: Tweet category percentages during the collection period. A = automatic/advisory, C = check-in, J = job posting, and P = personal.

While categorizing the tweets, care was taken to assess the content of each post to determine whether there was information relevant to the crime. This was highly subjective, but one of the 64 assault-co-located tweets stood out as a possibility. A “P” tweet with the username Tiana Taylor read, “I’m guessing it’s a lil to warm cause y’all wanna let off clips and s**t”. The user might only be commenting on the heat-related irritability of others with some dark humor. Furthermore, the “clip” referred to in the post is assumed to mean a gun clip, but the user might mean something else. Still, it could warrant a follow up.

5.3. Assessment Summary

For most crimes, meaningful spatiotemporal co-location with tweets is exceedingly rare. For the crimes analyzed in the assessment most had none. Only the crime of assault had a quantity of meaningful co-locations worth studying. Even then, the percentage of assault incidents that had spatiotemporal co-locations with tweets was very small. This is because most crimes in Chicago are committed at times when Twitter traffic is low.

Assault was chosen for further analysis because it was the only crime type to show more than a few meaningful spatiotemporal co-locations. Mapping co-located assaults with tweet locations confirmed that spatiotemporal co-locations are far more likely in areas with a high density of location-tagged tweets.

In analyzing the content of the co-located tweets, it was found that some categories of tweet are useless because they are automated and should be ignored during data collection. Other categories could still be useful, but this study found only one co-located tweet, out of the 359,569 tweets collected with *possibly* relevant information in the content.

Chapter 6 Discussion and Future Work

The application turned out to be highly functional in attaining the objective, but the sparseness of location-tagged tweets in most areas of Chicago makes the application of little use in accomplishing the goal of increased case clearances. Furthermore, finding anything of interest using the application is like trying to find a needle in a haystack unless the parameters of spatiotemporal co-location are broadened so much that the results become irrelevant to specific crimes. It comes down to a problem of quantity. The next section points out one way the quantity problem could be addressed. This is followed by a section on user interface enhancements and another on programmatical improvements that could be made in future work.

6.1. Addressing the Quantity Problem

Addressing the quantity problem would mean finding more location-tagged social media posts. Meaningful co-locations between tweets and crime incidents was found to be more likely in downtown Chicago where the density of tweet locations is high. If one could increase the density of social media post locations in areas outside of downtown Chicago, it might be possible to increase the number of meaningful co-locations. Looking to other social media platforms could help but the other widely-used platforms, like Facebook and Instagram, do not provide the same kind or quantity of data as Twitter.

A better way to address the quantity problem is through some of the location inference methods mentioned briefly in Chapter 2. Since check-in (category “C”) tweets make up a sizable percentage of tweet content, location inference methods that compare points of interest mentioned in the content with entries in a POI database, would be an excellent place to start.

6.2. User Interface Enhancements

The application needs a method for finding spatiotemporal co-location based on a user specified location. Finding co-locations for specific crimes can be tedious if only a date range is used to filter them. For crimes that occur frequently, this is especially problematic. Filtering assaults for a date range of one day still results in a crowded map, where specific crimes are hard to find. It would be useful to enable an investigator to enter a location and quickly check for spatiotemporally co-located social media posts.

Another beneficial addition for investigators would be a means of flagging relevant tweets and saving them to a list. The list could be printed out or shared digitally with a team of investigators.

6.3. Programmatical Improvements

The best way to improve this application (aside from adding more social media sources) would be to know which crimes had spatiotemporally co-located social media posts before the user does all the searching. In this project, the tweets are pre-collected and stored, so it is possible to find all the crime incidents with co-locations when the user makes the request for crimes. One of the scripts used to analyze the data for this project goes through all the crimes and finds their co-locations. A similar script could be run in the application and do two important things: separate all incidents with co-locations from those without and store all co-located tweets in memory. Then it would be possible to "turn off" or remove crime incidents without co-locations from the map (or at least symbolize them differently). This would also reduce the number of queries to the Twitter datastore. The script would make the visual information foraging less tedious and the application more efficient.

6.4. Conclusion

Implementation of the above-recommended changes to the interface and programming would produce a valuable tool for both the police and the public, provided that more location-tagged data could be found (through adding data from other social media platforms and/or through location inference). The tool could augment existing social media canvassing procedures in law enforcement, while an interested public could also participate and share their findings. The publicly available, open-source crime mapping application would aid in case clearances and solving heinous crimes.

This application could also be repurposed for a variety of use cases. A similar use case to the one outlined for this thesis would be using co-located tweets for situational awareness during large events where many people are in attendance. In this case, a broader definition of spatiotemporal co-location would be acceptable since phenomena like protests, sporting events, and natural disasters are either spread over a larger area or draw crowds likely to post information about the event to social media.

The datasets used in the application do not need to have anything to do with social media. The two datasets being compared could be anything. With the right data, the application could be repurposed for site selection based on user input criteria or finding points of interest near a user-defined location.

Where this project really shows promise, however, is in its use of MongoDB for data storage and a custom API to for data retrieval. One could add a programmatical step to the API that could be executed between retrieving geospatial data from the MongoDB and sending the results to the application. Different endpoints could be created for a variety of geospatial analyses on data returned from the MongoDB database. The results of this programmatical step

could then be returned to the application via the API. In crime mapping, this method could be used to add functionality like hotspot analysis for crime prediction, or simple logic for managing human assets in the field. In general, this could bring many commonly used spatial analysis tools to the web without the need to download software. If one is interested in using, making improvements to, or repurposing the application, the code is available at: https://github.com/neilemrys/chicago_crimeTweets.

References

andypiper, October 29, 2014 (4:47 a.m.), comment on Luca_filipponi, "Difference between sample and filter streaming API," *TwitterDevelopers* (forum), April 12, 2014, https://twittercommunity.com/t/diffence-between-sample-and-filter-streaming-api/15094.

Blackwood, Deirdre M. and Mark Radvanyi. "Eighty Percent of Citizens Believe More Digital Tools Can Improve Police Services, According to Accenture Survey," Accenture Consulting. Accessed June 20, 2017. https://newsroom.accenture.com/industries/health-public-service/eight-out-of-10-citizens-believe-more-digital-tools-can-improve-police-services-according-to-new-research-by-accenture.htm

Burrows, Tim. 2015. "The Social Media Guide for Law Enforcement," Everbridge Inc. Accessed June 8, 2017. http://go.everbridge.com/Lead-Gen---Social-Media-Guide-for-Law-Enforcement---November-2015---Nixle_Reg---P.html?TRK=MEDIA_PPC_GOO-COMMUNITYENGAGEMENT-LAWENFORCEMENT_BMM-WHITEPAPER&gclid=Cj0KCQjwtpDMBRC4ARIsADhz5O6gUJ4VqhSyTzmRyEbl_1MtybOz7oOdiaZxwuA6IZQZj6jpK4x54-0aAqBsEALw_wcB.

Chandra, Swarup, Latifur Khan, and Fahad Bin Muhaya. 2011. "Estimating Twitter User Location using Social Interactions--A Content Based Approach." Conference Proceedings, *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust, Boston, Massachusetts*, October 9-11.

Chicago Police Department. 1998. *CAPS at 5: A Report on the Progress of Community Policing*. Accessed June 15, 2017. https://home.chicagopolice.org/wp-content/uploads/2014/12/CAPSat5.pdf.

de Souza Baptista, Claudio, Carlos Eduardo Santos Pires, Daniel Farias Baptista Leite, Maxwell Guimaraes de Oliveira, and Odilon Francisco de Lima Junior. 2014. "NoSQL Geographic Databases: An Overview." Chap. 4, In *Geographical Information Systems: Trends and Technologies*, edited by Elaheh Pourabbas. 1st ed., 73-86. Boca Raton: CRC Press.

Duan, Miaoran and Gang Chen. 2015. "Assessment of MongoDB's Spatial Retrieval Performance." Conference Proceedings, 23rd Annual Conference on Geoinformatics, Wuhan, China, June 19-July 31. doi:10.1109/GEOINFORMATICS.2015.7378607.

Giacalone, Joseph L. 2015. "Canvass Crime Scene Safely" *LexisNexis Public Safety Blog*, May 5. Accessed August 4, 2017. http://blogs.lexisnexis.com/public-safety/2015/05/canvas-crime-scene-safely.

Giacalone, Joseph L. 2017. *The Criminal Investigative Function: A Guide for New Investigators* . 3rd ed. Flushing, New York: Looseleaf Law Publications.

Jurgens, David. 2013. "That's What Friends Are For: Inferring Location in Online Social Media Platforms Based on Social Relationships." Conference Proceedings, Seventh

International AAAI Conference on Weblogs and Social Media, Cambridge, Massachusetts, July 8-11.

Harries, Keith. 1999. *Mapping Crime: Principle and Practice*. Washington, D.C: U.S. Department of Justice.

Kalogirou, Vasilios and Jan Boehm. 2017. "Interfacing NoSQL with Open-Source GIS." Paper presented at the Geographical Information Science Research UK Conference, Manchester, UK, April 18-21. Accessed July 24, 2017. http://huckg.is/gisruk2017/GISRUK_2017_paper_41.pdf.

LexisNexis Risk Solutions. 2014. *Survey of Law Enforcement Personnel and Their Use of Social Media.* Accessed June 8, 2017. https://www.lexisnexis.com/risk/downloads/whitepaper/2014-social-media-use-in-law-enforcement.pdf.

Li, Chenliang and Aixin Sun. 2014. "Fine-Grained Location Extraction from Tweets with Temporal Awareness." Conference Proceedings, *37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Gold Coast, QLD, Australia, July 6-11.

MacEachren, A.M., Robinson, A.C., Jaiswal, A., Pezanowski, S., Savelyev, A., Blanford, J. and Mitra, P. 2011. "Geo-Twitter Analytics: Applications in Crisis Management." Conference Proceedings, *25th International Cartographic Conference*, Paris, France, July 3-8.

Madhani, Aamer. 2016. "At End of Bloody Year in Chicago, Too Few Murders Solved." *USA Today*, December 30. Accessed July 8, 2017. https://www.usatoday.com/story/news/2016/12/30/chicago-murders-clearance-rate-2016/96009878/.

Paulson, Derek and James LeBeau. 2012. *Survey and Evaluation of Online Crime Mapping Companies*, (North Charleston, South Carolina: SCRA Applied R&D).

Pierce, Brent. "Integrate Big Data with the ArcGIS System using a Plug-in Data Source." *Esri ArcGIS Blog*, July 16, 2012, https://blogs.esri.com/esri/arcgis/2012/07/16/integrate-big-data-with-the-arcgis-system-using-a-plug-in-data-source.

Reilly, Colin. "NYC Crime Map," *NYCityMap & Beyond* (blog), December 9, 2017, https://nycitymap.wordpress.com/2013/12/09/nyc-crime-map/.

Reilly, Collin. "City of New York and Google Maps Help People Stay Street Smart," *Official Google Cloud Blog*, March 20, 2014, https://cloud.googleblog.com/2014/03/city-of-new-york-and-google-maps-help.html.

SpotCrime. 2017. "SpotCrime Crime Map." Accessed May 20, 2017. https://spotcrime.com/.

Tear, Adrian. 2014. “SQL or NoSQL? Contrasting Approaches to the Storage, Manipulation and Analysis of Spatio-temporal Online Social Network Data.” Conference Proceeding, *14th International Conference of Computational Science and Its Applications*, Guimarães, Portugal, June 30 – July 3.

Thompson, W. H. 2017. "HeyJackass! Illustrating Chicago Values." Accessed July 8, 2017. http://heyjackass.com/home/.

TriTech Software Systems. 2016. “CrimeMapping: Helping You Build a Safer Community.” Accessed May 20, 2017. https://www.crimemapping.com/.

Weisburd, David and Cynthia Lum. 2005. "The Diffusion of Computerized Crime Mapping in Policing: Linking Research and Practice." *Police Practice and Research* 6 (5): 419-434.

Yanai, Keiji. 2012. "World Seer: A Realtime Geo-Tweet Photo Mapping System." Conference Proceedings, *s*, Hong Kong, June 5-8.

Zhang, Wei and Judith Gelernter. 2014. "Geocoding Location Expressions in Twitter Messages: A Preference Learning Method." *Journal of Spatial Information Science* 2014 (9): 37-70.

Zhang, Xiaomin, Wei Song, and Liming Liu. 2014. "An Implementation Approach to Store GIS Spatial Data on NoSQL Database." Conference Proceedings, *22nd International Conference on Geoinformatics*, Kaohsiung, Taiwan, June 25-27. doi:10.1109/GEOINFORMATICS.2014.6950846.

Zhou, Guiyun, Jiayuan Lin, and Xiujun Ma. 2014. "A Web-Based GIS for Crime Mapping and Decision Support." In *Forensic GIS: The Role of Geospatial Technologies for Investigating Crime and Providing Evidence*. 1st ed. Vol. 11, edited by Gregory A. Elmes, Jamison Conley, Jay D. Gatrell, George Roedl, and Ryan R. Jensen, 221-244. Dordrecht: Springer.

Appendix A Python Script for Tweet Collection

```
from TwitterAPI.TwitterAPI import TwitterAPI
from geojson import Feature, Point
import json
import time
from pymongo import MongoClient

while True:
    try:

        CONSUMER_KEY = 'XXXXXXXXXXXXXXX'
        CONSUMER_SECRET = 'XXXXXXXXXXXXXXXX'
        ACCESS_TOKEN_KEY = 'XXXXXXXXXXXXXXXX'
        ACCESS_TOKEN_SECRET = 'XXXXXXXXXXXXXXXX'

        api = TwitterAPI(CONSUMER_KEY,
                         CONSUMER_SECRET,
                         ACCESS_TOKEN_KEY,
                         ACCESS_TOKEN_SECRET)

        r = api.request('statuses/filter', {'locations': "-
87.940033, 41.644102, -87.523993, 42.023067 "})

        client = MongoClient("localhost:27017")
        db = client.twitterAPI

        for item in r:

            if item['coordinates'] is not None:

                lng = item['coordinates']['coordinates'][0]
                lat = item['coordinates']['coordinates'][1]
                tweetPoint = Point((lng, lat))
                placeName = item['place']['full_name']

                name = item['user']['name']
                screeName = item['user']['screen_name']
                image = item['user']['profile_image_url']

                tweetDate = str(item['created_at'])
```

```
                tweetTimeStampMS = float(item['timestamp_ms'])

                content = item['text']
                hash = list((object['text'] for object in
item['entities']['hashtags']))
                urls = list((object['url'] for object in
item['entities']['urls']))

                tweet = Feature(geometry=tweetPoint,
                                properties={"userName": name,
"screeName": screeName, "image": image, "tweetDate": tweetDate,
                                            "tweetTimeStampMS":
tweetTimeStampMS, "text": content, "hashTags": hash,
                                            "urls": urls,
"placeName": placeName})

                db.wCoords.insert_one(
                    {
                        "type": "Feature",
                        "geometry": {"type": "Point",
"coordinates": [lng, lat]},
                        "properties": {
                            "userName": name,
                            "screeName": screeName,
                            "image": image,
                            "tweetDate": tweetDate,
                            "tweetTimeStampMS":
tweetTimeStampMS,
                            "text": content,
                            "hashTags": hash,
                            "urls": urls,
                            "placeName": placeName
                        }
                    }
                )
                print json.dumps(tweet, indent=2)
                print "sent to wCoords"
            else:
                placeName = item['place']['full_name']

                name = item['user']['name']
```

```
                screeName = item['user']['screen_name']
                image = item['user']['profile_image_url']

                tweetDate = str(item['created_at'])
                tweetTimeStampMS = float(item['timestamp_ms'])

                content = item['text']
                hash = list((object['text'] for object in
item['entities']['hashtags']))
                urls = list((object['url'] for object in
item['entities']['urls']))

                tweet = Feature(geometry=None,
                                properties={"userName": name,
"screeName": screeName, "image": image, "tweetDate": tweetDate,
                                            "tweetTimeStampMS":
tweetTimeStampMS, "text": content, "hashTags": hash,
                                            "urls": urls,
"placeName": placeName})
                db.noCoords.insert_one(
                    {
                        "type": "Feature",
                        "geometry": None,
                        "properties": {
                            "userName": name,
                            "screeName": screeName,
                            "image": image,
                            "tweetDate": tweetDate,
                            "tweetTimeStampMS":
tweetTimeStampMS,
                            "text": content,
                            "hashTags": hash,
                            "urls": urls,
                            "placeName": placeName
                        }
                    }
                )
                print json.dumps(tweet, indent=2)
                print "sent to noCoords"
        pass
    except Exception as e:
```

```
print("Something went wrong: "+repr(e))
time.sleep(15)
```

Appendix B hapi.js Custom API

```
var Hapi = require('hapi');
var mongoose = require('mongoose/');

mongoose.Promise = global.Promise;

mongoose.connect('mongodb://localhost/tweets');
var db = mongoose.connection;

var pointSchema = mongoose.Schema({
    geometry: Object,
    type: String,
    properties: Object
},
{
    collection:'wCoords'
});

var Point = mongoose.model('Point', pointSchema)

var server = new Hapi.Server({
    connections: {
        routes: {
```

```
                cors: true
            }
        }
    });
    server.connection({ port: 8080 });

    server.route([
      // Get tweets
      {
        method: 'GET',
        path: '/tweets',
        handler: function(request, reply) {
            var lng = request.query.lng;
            var lat = request.query.lat;
            var dist = request.query.dist;
            var minTime = request.query.minMS
            var maxTime = request.query.maxMS
            var result =
Point.find({$and:[{geometry:{$near:{$geometry:{type:"Point",coor
dinates:[parseFloat(lng),parseFloat(lat)]}, $minDistance:0,
$maxDistance:parseInt(dist)}}},{"properties.tweetTimeStampMS":{$
gt:parseFloat(minTime),$lt:parseFloat(maxTime)}}]});
            result.exec(function(err, points) {
                  reply(points);
```

```
            })
          }
        }
      ]);

      server.start(function(err) {
        console.log('MongoDB is listening to
http://localhost:8080');
      });
```

Chapter 7 Appendix C JavaScript to display a Google Map

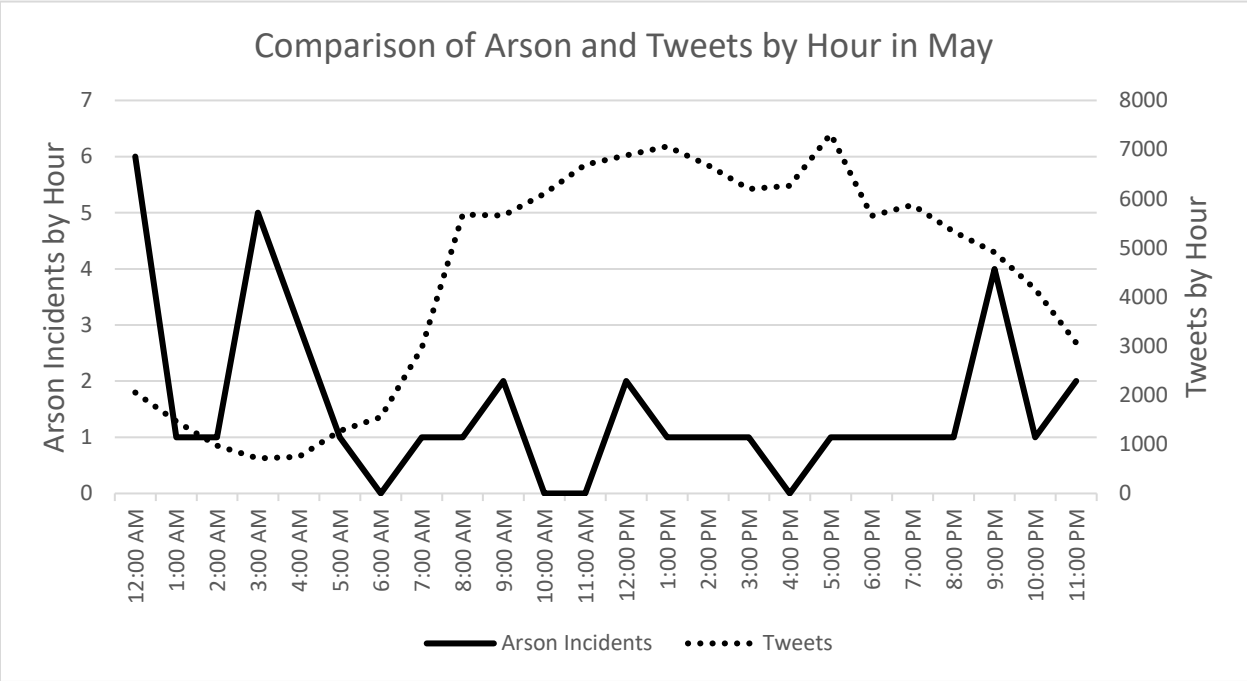
```
var map;
function initMap() {
//MAP STYLE DEFINITION BEGIN
  var styledMapType = new google.maps.StyledMapType(
    [
      {
        "elementType": "geometry",
        "stylers": [{"color": "#c9c9c9"}]
      },
      {
        "elementType": "labels.icon",
        "stylers": [{"visibility": "off"}]
      },
      {
        "elementType": "labels.text.fill",
        "stylers": [{"color": "#616161"}]
      },
      {
        "elementType": "labels.text.stroke",
        "stylers": [{"color": "#f5f5f5"}]
      },
      {
        "featureType": "administrative.land_parcel",
        "elementType": "labels.text.fill",
        "stylers": [{"color": "#bdbdbd"}]},
        {
          "featureType": "poi",
          "elementType": "geometry",
          "stylers": [{"color": "#b8b8b8"}]
        },
        {
          "featureType": "poi",
          "elementType": "labels.text.fill",
          "stylers": [{"color": "#757575"}]
        },
        {
          "featureType": "poi.park",
```

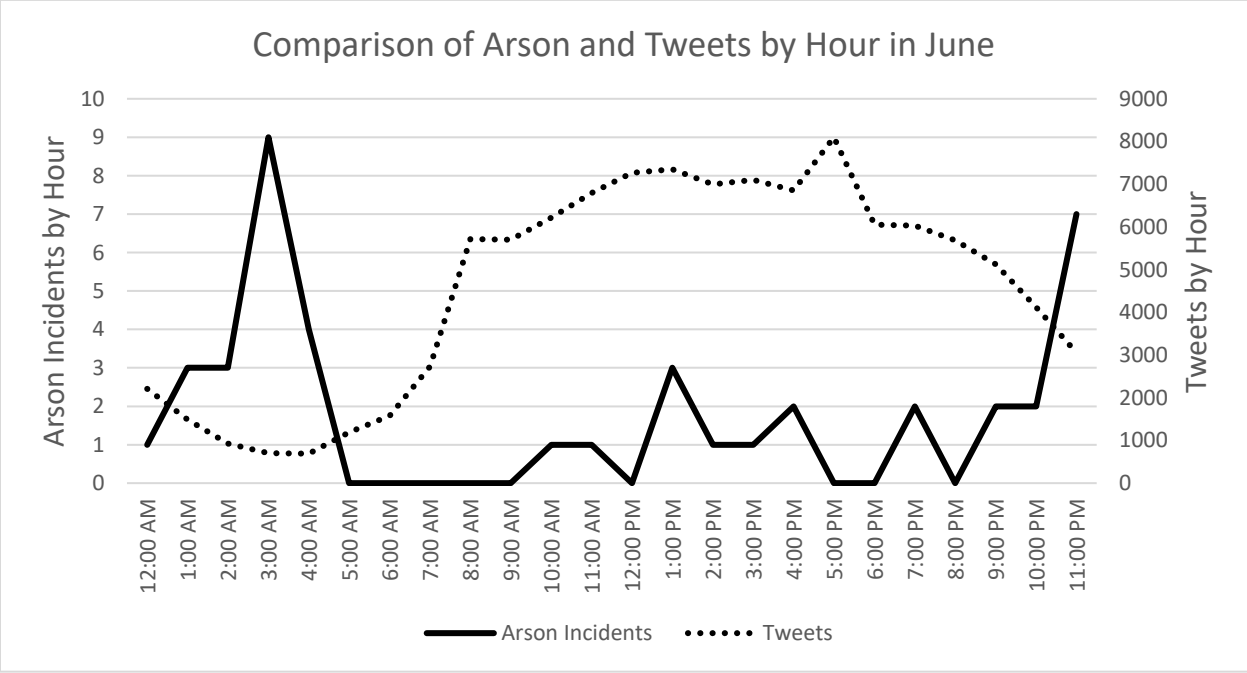
```
  "elementType": "geometry",
  "stylers": [{"color": "#7b9574"}]
},
{
  "featureType": "poi.park",
  "elementType": "labels.text.fill",
  "stylers": [{"color": "#9e9e9e"}]
},
{
  "featureType": "road",
  "elementType": "geometry",
  "stylers": [{"color": "#ffffff"}]
},
{
  "featureType": "road.arterial",
  "elementType": "labels.text.fill",
  "stylers": [{"color": "#757575"}]
},
{
  "featureType": "road.highway",
  "elementType": "geometry",
  "stylers": [{"color": "#dadada"}]
},
{
  "featureType": "road.highway",
  "elementType": "labels.text.fill",
  "stylers": [{"color": "#616161"}]
},
{
  "featureType": "road.local",
  "elementType": "labels.text.fill",
  "stylers": [{"color": "#9e9e9e"}]
},
{
  "featureType": "transit.line",
  "elementType": "geometry",
  "stylers": [{"color": "#e5e5e5"}]
},
{
  "featureType": "transit.station",
  "elementType": "geometry",
```

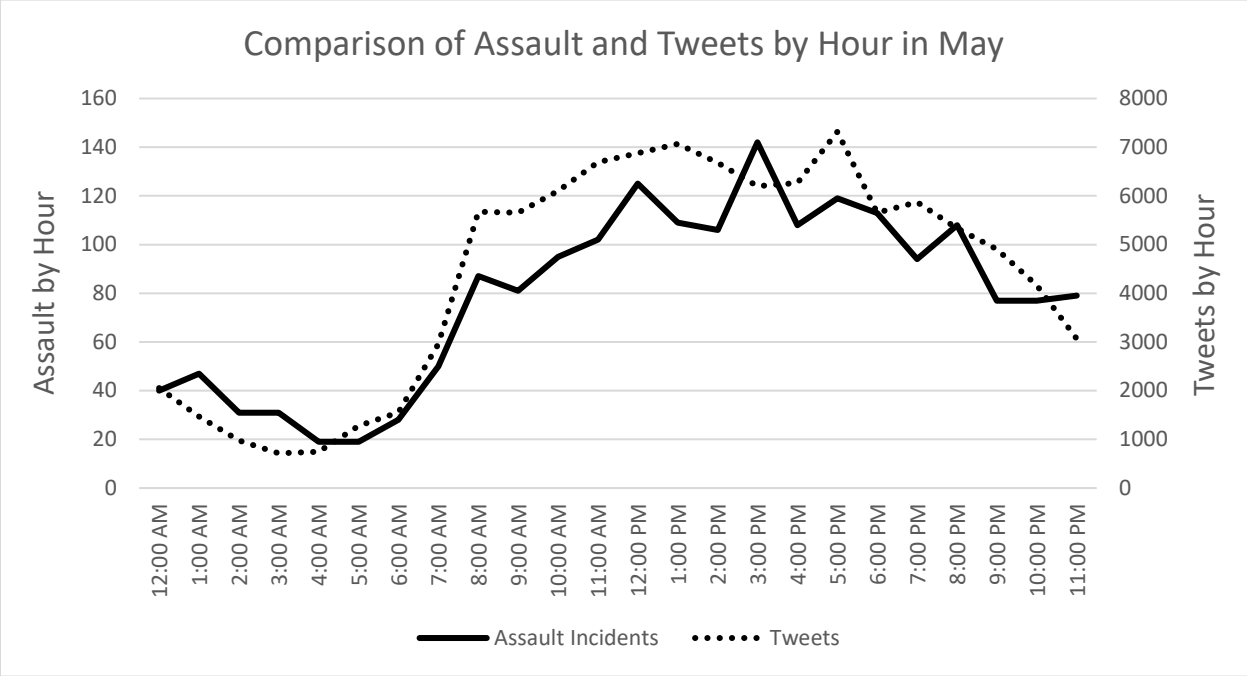
```
          "stylers": [{"color": "#eeeeee"}]
        },
        {
          "featureType": "water",
          "elementType": "geometry",
          "stylers": [{"color": "#0a1521"}]
        },
        {
          "featureType": "water",
          "elementType": "labels.text.fill",
          "stylers": [{"color": "#9e9e9e"}]
        }
      ],
      {name: 'Gray'});
//MAP STYLE END

//MAP DEFINITION BEGIN
      map = new google.maps.Map(document.getElementById('map'),
{
        center: {lat: 41.841832, lng: -87.623177},
        zoom: 11,
        mapTypeControlOptions: {
          mapTypeIds: []
        },
        streetViewControl: true,
        streetViewControlOptions: {
          position: google.maps.ControlPosition.LEFT_BOTTOM
        },
        zoomControl: true,
        zoomControlOptions: {
          position: google.maps.ControlPosition.LEFT_BOTTOM
        }
      });
      map.mapTypes.set('Gray', styledMapType);
      map.setMapTypeId('Gray');
//MAP DEFINITION END


//CHICAGO BOUNDARY POLYGON START
//THIS IS ONLY AN EXAMPLE FOR THIS APPENDIX. THE BOUNDARY
POLYGON FOR CHICAGO WOULD HAVE TAKEN UP FAR TOO MUCH ROOM
```

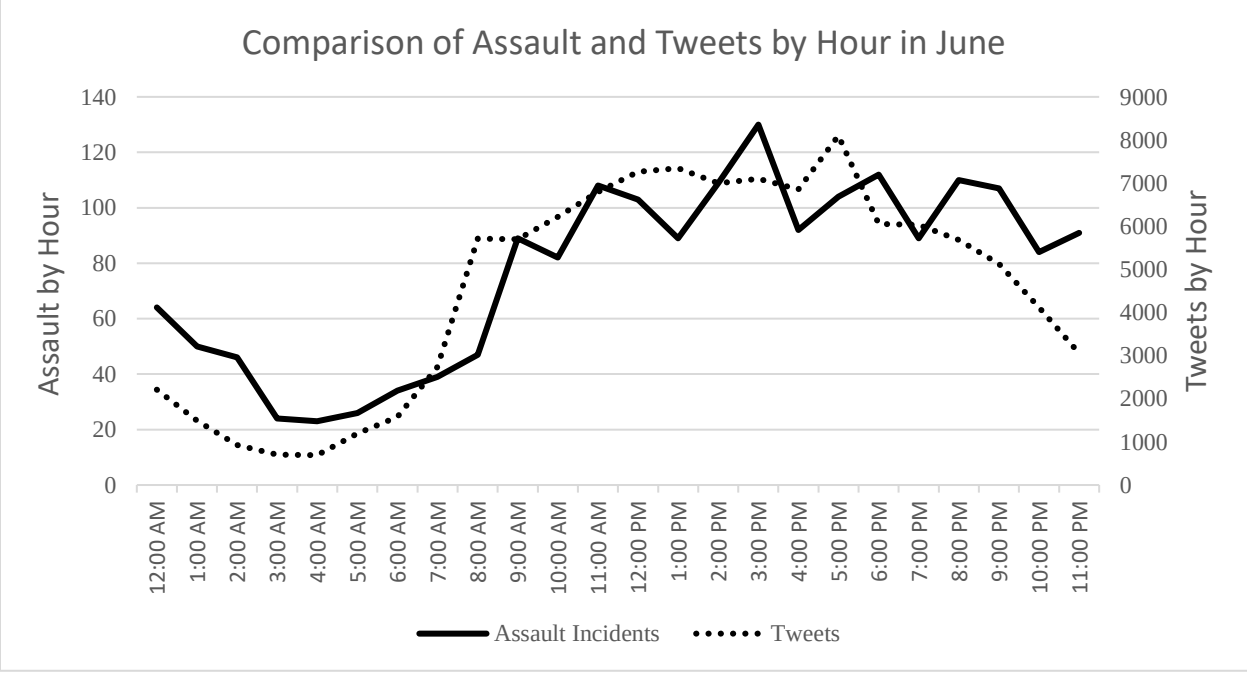
```
    var outer3 = [
      {lat: 42.005424, lng: -87.933485},
      {lat: 42.004888, lng: -87.933611},
      {lat: 42.004882, lng: -87.933613},
      {lat: 42.004930, lng: -87.933977},
      {lat: 42.004970, lng: -87.934273},
      {lat: 42.004983, lng: -87.934373},
      {lat: 42.004922, lng: -87.934372},
      {lat: 42.005424, lng: -87.933485}
    ];
    var chicago = new google.maps.Polygon({
        paths: [outer1, inner1, inner2, outer2, outer3],
        strokeColor: '#000000',
        strokeOpacity: 0.35,
        strokeWeight: 3,
        fillColor: '#000000',
        fillOpacity: 0.15
      });
      chicago.setMap(map);
    //CHICAGO POLYGON END
}
```

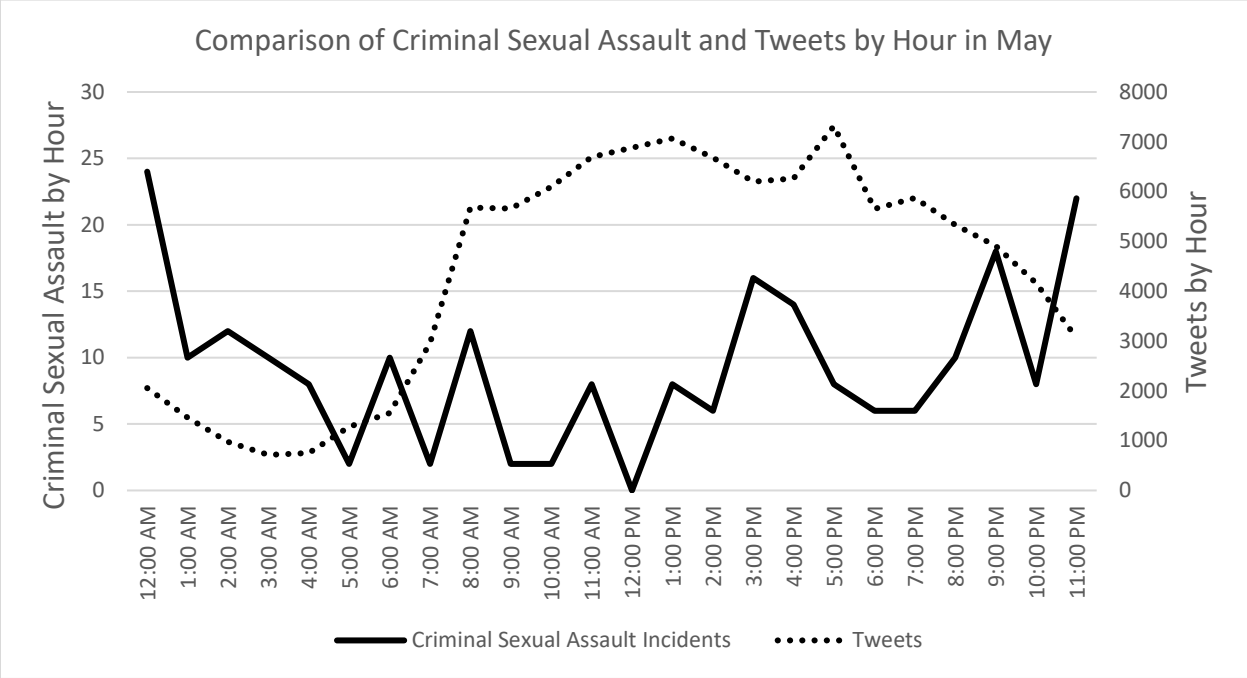
Appendix D Crime Incidents vs. Tweets Charted by Hour

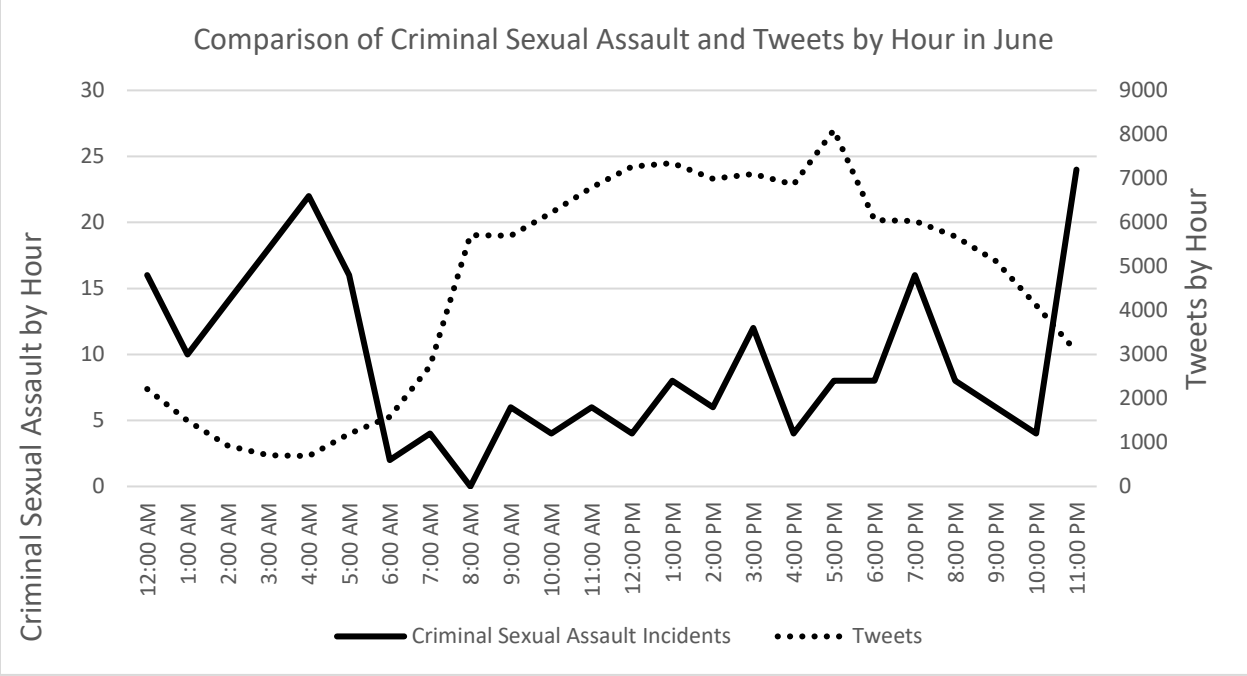


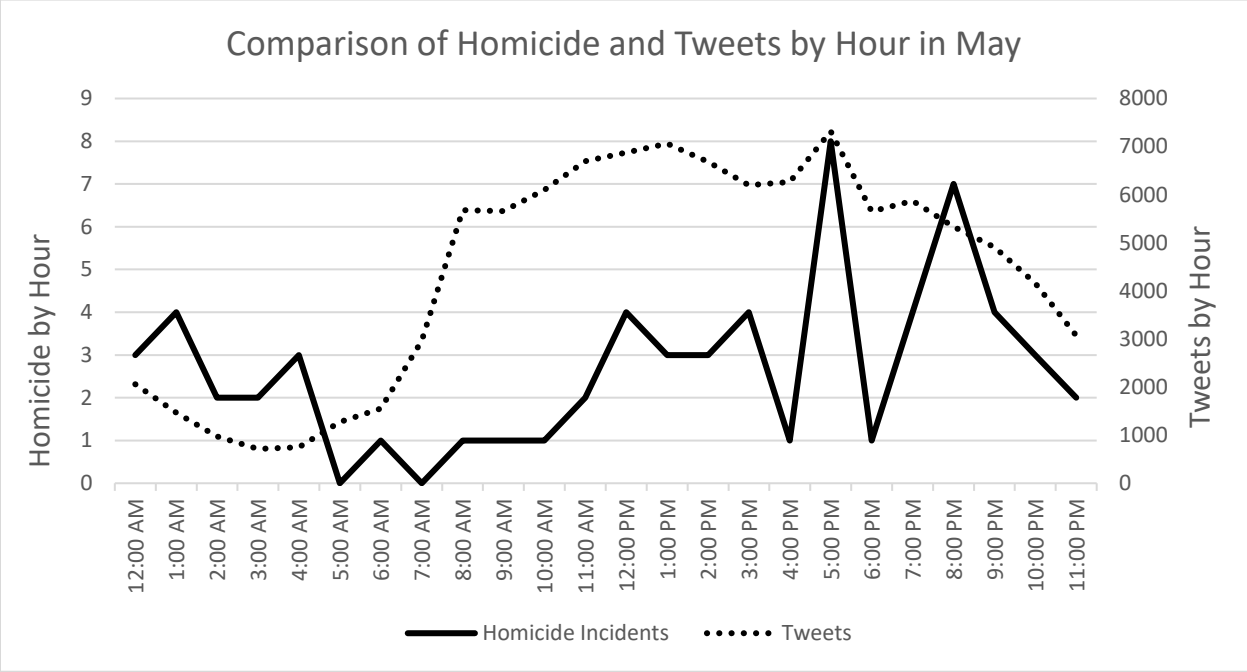


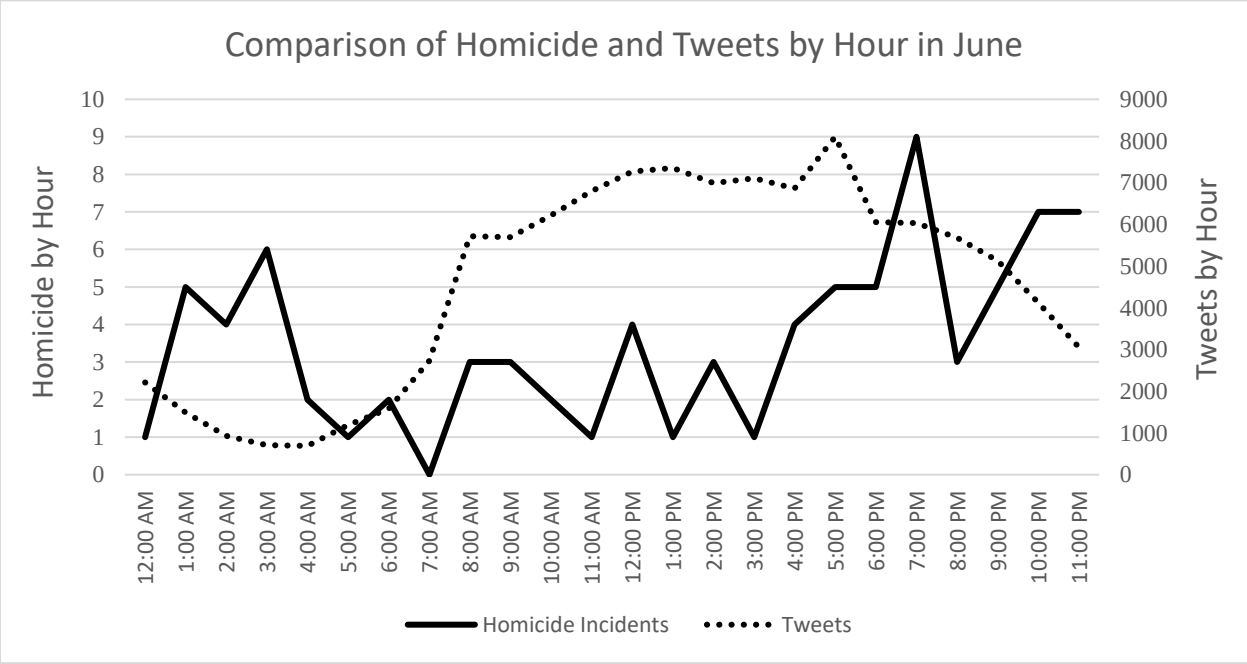

Comparison of Assault and Tweets by Hour in May
Assault by Hour
Tweets by Hour
0
20
40
60
80
100
120
140
160
0
1000
2000
3000
4000
5000
6000
7000
8000
12:00 AM
1:00 AM
2:00 AM
3:00 AM
4:00 AM
5:00 AM
6:00 AM
7:00 AM
8:00 AM
9:00 AM
10:00 AM
11:00 AM
12:00 PM
1:00 PM
2:00 PM
3:00 PM
4:00 PM
5:00 PM
6:00 PM
7:00 PM
8:00 PM
9:00 PM
10:00 PM
11:00 PM
Assault Incidents
Tweets

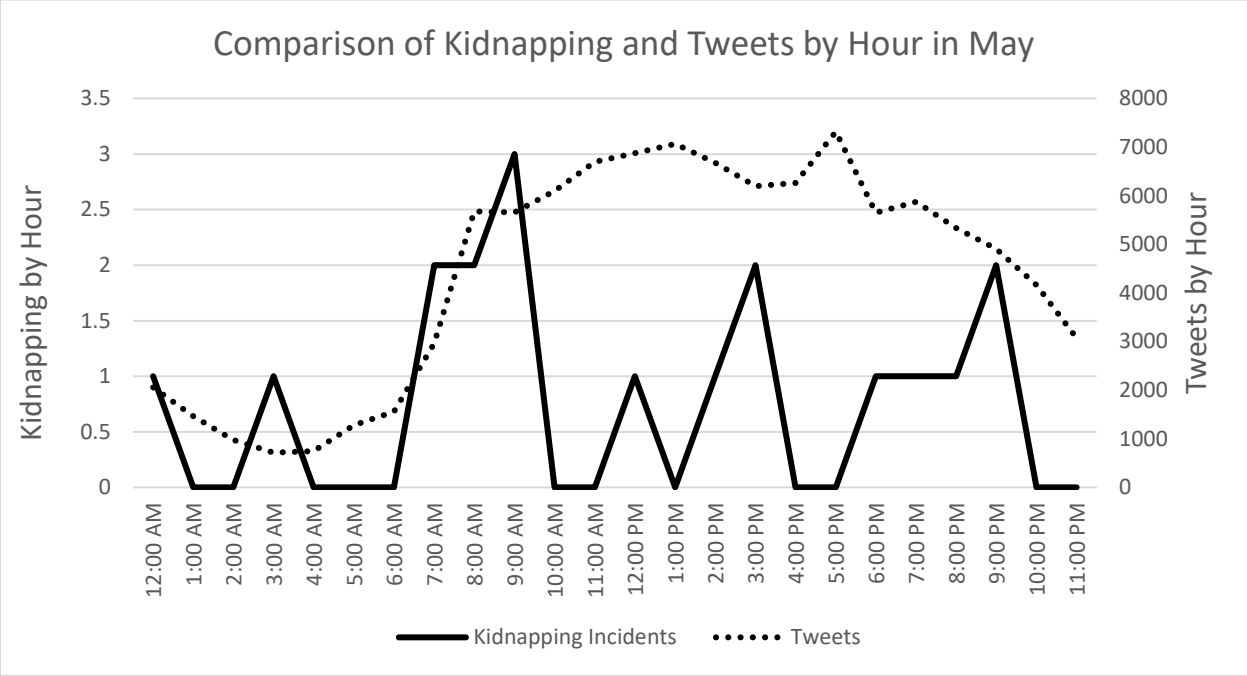

Comparison of Assault and Tweets by Hour in June
Assault by Hour
Tweets by Hour
0
20
40
60
80
100
120
140
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
12:00 AM
1:00 AM
2:00 AM
3:00 AM
4:00 AM
5:00 AM
6:00 AM
7:00 AM
8:00 AM
9:00 AM
10:00 AM
11:00 AM
12:00 PM
1:00 PM
2:00 PM
3:00 PM
4:00 PM
5:00 PM
6:00 PM
7:00 PM
8:00 PM
9:00 PM
10:00 PM
11:00 PM
Assault Incidents
Tweets

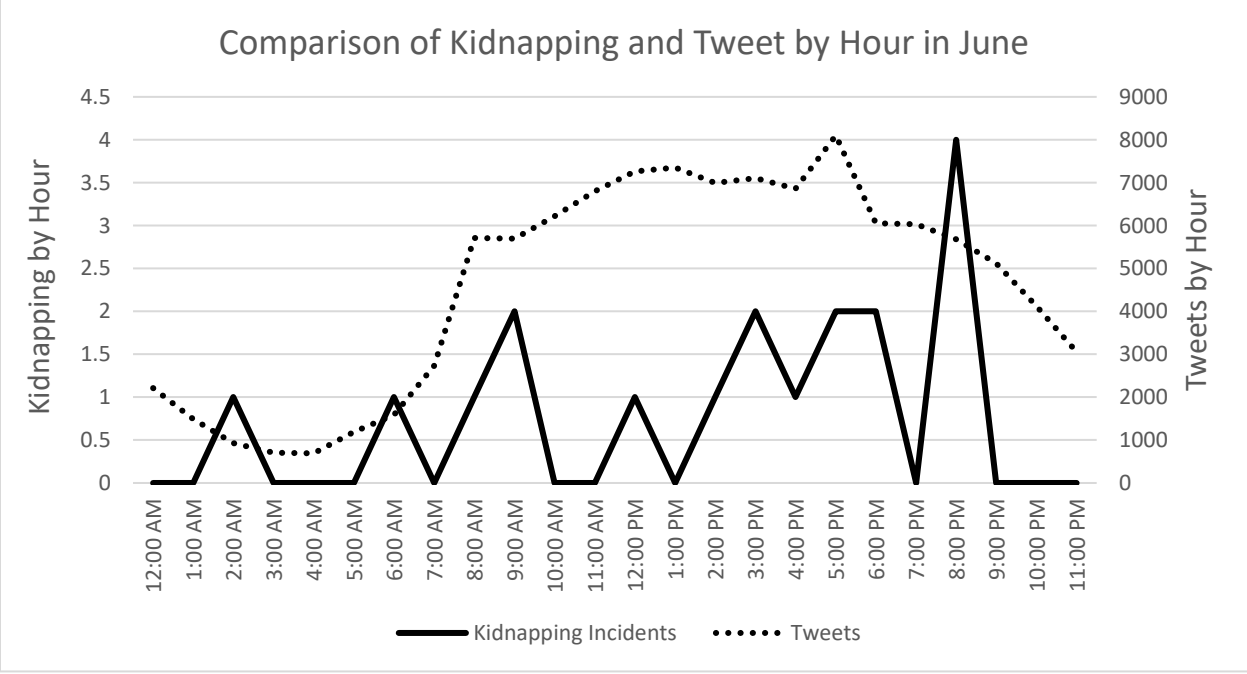

Comparison of Criminal Sexual Assault and Tweets by Hour in May
Criminal Sexual Assault by Hour
Tweets by Hour
30
25
20
15
10
5
0
8000
7000
6000
5000
4000
3000
2000
1000
0
12:00 AM
1:00 AM
2:00 AM
3:00 AM
4:00 AM
5:00 AM
6:00 AM
7:00 AM
8:00 AM
9:00 AM
10:00 AM
11:00 AM
12:00 PM
1:00 PM
2:00 PM
3:00 PM
4:00 PM
5:00 PM
6:00 PM
7:00 PM
8:00 PM
9:00 PM
10:00 PM
11:00 PM
Criminal Sexual Assault Incidents
Tweets


Comparison of Criminal Sexual Assault and Tweets by Hour in June
Criminal Sexual Assault by Hour
Tweets by Hour
30
25
20
15
10
5
0
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
12:00 AM
1:00 AM
2:00 AM
3:00 AM
4:00 AM
5:00 AM
6:00 AM
7:00 AM
8:00 AM
9:00 AM
10:00 AM
11:00 AM
12:00 PM
1:00 PM
2:00 PM
3:00 PM
4:00 PM
5:00 PM
6:00 PM
7:00 PM
8:00 PM
9:00 PM
10:00 PM
11:00 PM
Criminal Sexual Assault Incidents
Tweets


Comparison of Homicide and Tweets by Hour in May
Homicide by Hour
Tweets by Hour
0
1
2
3
4
5
6
7
8
9
0
1000
2000
3000
4000
5000
6000
7000
8000
12:00 AM
1:00 AM
2:00 AM
3:00 AM
4:00 AM
5:00 AM
6:00 AM
7:00 AM
8:00 AM
9:00 AM
10:00 AM
11:00 AM
12:00 PM
1:00 PM
2:00 PM
3:00 PM
4:00 PM
5:00 PM
6:00 PM
7:00 PM
8:00 PM
9:00 PM
10:00 PM
11:00 PM
Homicide Incidents
Tweets


Comparison of Homicide and Tweets by Hour in June
Homicide by Hour
Tweets by Hour
0
1
2
3
4
5
6
7
8
9
10
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
12:00 AM
1:00 AM
2:00 AM
3:00 AM
4:00 AM
5:00 AM
6:00 AM
7:00 AM
8:00 AM
9:00 AM
10:00 AM
11:00 AM
12:00 PM
1:00 PM
2:00 PM
3:00 PM
4:00 PM
5:00 PM
6:00 PM
7:00 PM
8:00 PM
9:00 PM
10:00 PM
11:00 PM
Homicide Incidents
Tweets


Comparison of Kidnapping and Tweets by Hour in May
Kidnapping by Hour
Tweets by Hour
3.5
3
2.5
2
1.5
1
0.5
0
8000
7000
6000
5000
4000
3000
2000
1000
0
12:00 AM
1:00 AM
2:00 AM
3:00 AM
4:00 AM
5:00 AM
6:00 AM
7:00 AM
8:00 AM
9:00 AM
10:00 AM
11:00 AM
12:00 PM
1:00 PM
2:00 PM
3:00 PM
4:00 PM
5:00 PM
6:00 PM
7:00 PM
8:00 PM
9:00 PM
10:00 PM
11:00 PM
Kidnapping Incidents
Tweets


Comparison of Kidnapping and Tweet by Hour in June
Kidnapping by Hour
Tweets by Hour
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
12:00 AM
1:00 AM
2:00 AM
3:00 AM
4:00 AM
5:00 AM
6:00 AM
7:00 AM
8:00 AM
9:00 AM
10:00 AM
11:00 AM
12:00 PM
1:00 PM
2:00 PM
3:00 PM
4:00 PM
5:00 PM
6:00 PM
7:00 PM
8:00 PM
9:00 PM
10:00 PM
11:00 PM
Kidnapping Incidents
Tweets