APPROVAL SHEET

Title of Dissertation: A MULTILAYER SEMANTIC AUGMENTATION MODEL FOR GEOGRAPHIC INFORMATION APPLICATIONS
Name of Candidate: Manesh Ramachandran Pillai
Doctor of Philosophy, 2022

Dissertation and Abstract Approved: (*Signature of Supervising Professor)
George Karabatis
Professor
Information Systems

Date Approved: _____

NOTE: *The Approval Sheet with the original signature must accompany the thesis or dissertation. No terminal punctuation is to be used.

ABSTRACT

Title of Document: **A MULTILAYER SEMANTIC AUGMENTATION MODEL FOR GEOGRAPHIC INFORMATION APPLICATIONS**

Manesh Ramachandran Pillai, Doctor of Philosophy, 2022

Directed By: Dr. George Karabatis, Professor
Information Systems Department, UMBC

Current mainstream geographic information systems and maps in general depend on the user's ability to derive meaning from the multi-layered nature of those maps. A complete answer to the question "what is relevant to my location of interest" should consider the user's context and include records from multiple Geographical Information System (GIS) layers. The problem this research attempts to solve is the inability of organizations to query geospatial data in a way that conforms with the multi-layered nature of GIS data as well as the topological relationships that can exist between the GIS layers containing those geospatial objects.

This dissertation extends the theory behind graphs (or networks) of objects with research into multilayer and semantic link networks to create a formal mathematical model of geographic objects and take advantage of relationships that exist among nodes within a single layer and across multiple layers. This includes the creation of a

geospatial ontology that mathematically represents the relationships between different classes of geographic objects. The algorithms in this dissertation take traditional GIS queries and expand them using semantic reasoning and topological rules to include additional geographic objects that are relevant to the user, introducing the concept of a multi-layer Semantically Linked Network (mSLN).

This research elevates traditional GIS operations into a common mathematical model to simplify the needs of an organization. This mathematical model has been proven to be correct and a framework based on the model has been designed to be easily implemented by organizations that utilize GIS systems.

A prototype system, SAM-GIS has been developed and an empirical evaluation of this framework has been conducted using several real-life case studies relevant to local communities based on data from an actual local government population, along with a performance evaluation of the entire system. Results show that SAM-GIS provides expanded GIS search results with increased accuracy, precision and recall over those of traditional GIS systems.

A MULTILAYER SEMANTIC AUGMENTATION MODEL FOR GEOGRAPHIC INFORMATION APPLICATIONS

By

Manesh Ramachandran Pillai

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, Baltimore County, in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in
Information Systems
2022

Acknowledgements

Nobody accomplishes anything on their own and this work is no different. This took longer than any of us expected, and I am grateful for everyone's patience and support.

I would first like to thank my parents for giving me the opportunity to achieve my academic dreams. They will never fully understand the technical details of this work, but they knew that this was important to me.

I am blessed to have been born into a family that respects education and who have supported me in this. Thank you to my brother and the rest of my extended family.

My advisor and mentor, Dr. George Karabatis, has been invaluable in guiding me for the past 8 years. When I imagine where I was intellectually when I started and where I am now, the difference is noticeable and staggering. The relationship between Ph.D. student and mentor is critical and I cannot imagine doing this with anyone else.

Thank you to my committee: Dr. Zhiyuan Chen, Dr. Arrya Gangopadhyay, Dr. Vandana Janeja, Dr. James Foulds, and Dr. Tim Finin. Your feedback gave me new perspectives on my work and made the final work stronger.

I have had the pleasure of teaching about 400 students at UMBC and it has been a great experience. They have been supportive of my own schoolwork the entire time and the process of teaching undergraduates has improved my ability to discuss technical topics.

Finally, I would like to thank my former colleagues at Howard County and my current colleagues at Montgomery County who have supported my studies.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

The development of geographic information systems emerged because of the need for organizations to manage data and information that contained location characteristics. For example, a database can hold information about addresses but a search for nearby addresses requires the database to be aware of the geographic properties (coordinates, boundaries) of the objects and to provide the ability to make geographic calculations (for example, distance) from those properties.

This chapter outlines the problems of current geographic information systems and their retrieval mechanisms and outlines the solution to this problem that this research proposes.

1.1 Research Problem

Current mainstream geographic information systems and maps in general depend on the user's ability to derive meaning from the multi-layered nature of those maps. That is, a reader of a map (whether it is a paper map or an interactive map) views a series of geographic objects on that map (roads, houses, rivers, etc.) with labels for those objects and a legend defining the styles (such as colors or line thickness) associated with those objects to create meaningful insights from those maps.

Figure 1 is a screen capture of a paper map representing the geologic properties of the soils contained within Howard County, Maryland. Someone reading this map would associate the color-coded shapes with various locations in the region and look up the associated geologic formation in the legend to connect the shape on the map with the label associated with that shape. For example, areas with the "Baltimore

Complex" geological formation are identified with a green color on the map. Based on the legend a reader of the map can also determine that this geologic formation is based on extrusive igneous rocks. If a user wanted to identify all locations in Howard County with a soil based on extrusive igneous rocks, they can identify the two geologic formations with that rock type ("James Run Formation" and "Baltimore Complex"), their colors from the legend ("Purple" and "Green") and then pinpoint the relevant locations on the map with those colors.

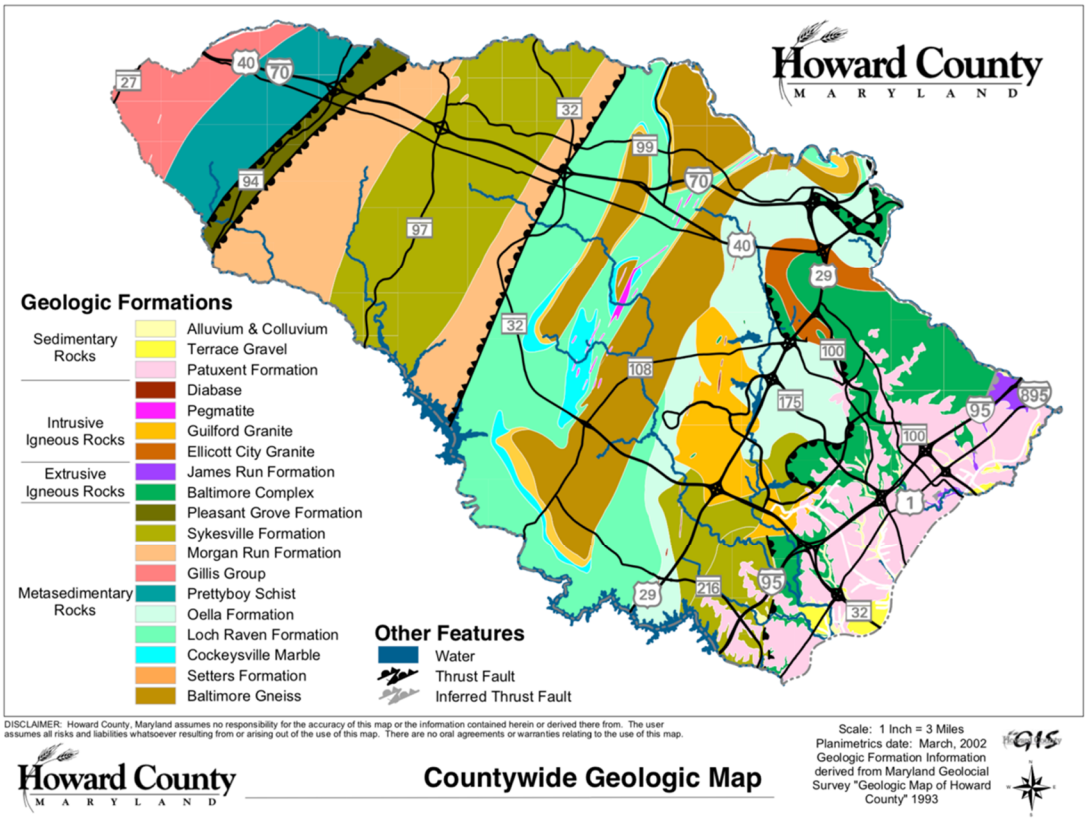


Figure 1: An example of a paper map derived from a GIS.

Interactive maps take the concept of reading a paper map and expand it to allow the user to toggle the visibility of the different classes of geographic objects as well as the ability to "zoom in" to offer more detail. Users can zoom into a location either by using

the map's built-in zoom and pan tools or by using a dedicated feature search (typically an address) that zooms into that address.

End-users often request information that is in these systems, but it is not easily retrievable, and when organizations want to help end-users identify relevant geographic objects, developers or professionals must explicitly build these tools.

A geographic information system (GIS) is an information system that stores and processes the geographic (or location) information associated with an organization. The goal of any GIS is to use location information (stored in a database) to supply answers to user queries.

For example, a typical question presented to a GIS could be, "what are the closest parks to my house?" This query involves the user's location (the house) and a table (or GIS layer) containing a list of parks. The distance between the user's location and each of the parks is calculated, and it is used to create a *relevance score* for each park: the more relevant the park to the user (the closer it is), the higher the score. Parks with high relevance scores are returned to the user.

However, a complete answer to the question "what is relevant to my location of interest" should consider the user's context (why they are searching for relevant geographic objects) and include records from multiple GIS layers, such as topography, streams, etc. For example:

- A citizen interested in new construction activity in their community could type in an address and a GIS should return nearby plats, site plans, permits, and capital projects.
- A prospective home buyer might be interested in nearby schools and parks.

- A police commander might be interested in local crime statistics and compare it to road access.

These are examples of end-users who do not have GIS expertise, yet they are the main recipients of this query result. When a query is submitted to a GIS it should produce records that are relevant to the user's context based on different layers in a database. However, in current systems these queries are usually pre-programmed.

Additionally, as enterprises have transitioned to storing their data in dedicated enterprise database management systems (DBMS) they now look towards taking the data stored in those databases and turning them into actionable insights. This reflects the transition of enterprises from managing data and information to creating knowledge. One of the foundations of this transition is a data warehouse that reaches out to a diverse set of data sources (databases, web services, etc.) and aggregates or merges the data from those databases in a way that facilitates querying.

An information system that can query data from multiple databases, understands the geographic properties of that data and the topological relationships between different types of geographic objects should facilitate the types of searches.

To summarize, the problem this research attempts to solve is the inability of organizations to query geospatial data in a way that respects the multi-layered nature of GIS data as well as the topological relationships that can exist between the GIS layers holding those geospatial objects.

1.2 Practical Statement of the Problem

This research proposes a framework that solves the problems summarized in the earlier section. Specifically, it tries to answer the following questions:

- **What is the best way for organizations (such as local governments) that heavily use GIS systems to access relevant geospatial data flexibly without the need for ad-hoc querying and custom development?** Creating a framework that models the relationships between geographic objects can serve as the foundation for future analytical work regarding these datasets (such as data mining or analysis applications). This could also save organizations a considerable amount of time by minimizing the total number of applications that need to be created.
- **Can we create a system that allows an end-user to query a GIS database and to have that database enhance answers to that query with additional information that is relevant to that query without the need for custom applications to be written by GIS professionals and developers?** Can we design interactive GIS systems (including maps) in a more generalized fashion as opposed to creating individual applications written to meet specific GIS query needs?

1.3 Research Problems and Methodological Approach

This dissertation extends the theory behind graphs (or networks) of objects with research into multilayer networks and semantic link networks to create a model of geographic objects and define a practical implementation of this system that can be integrated into existing GIS environments.

Geographic objects exist as entities with natural topological relationships between them, notably distance relationships and this makes the existing research into object

graphs (or networks) a natural fit to model these types of relationships. Two extensions to graph networks that solidify this fit are detailed below.

First, research into multilayer networks that structurally combine graph nodes into groups is a natural fit for the layered nature of geospatial data [1, 2]. Multilayer networks group objects in a graph in a logical way that respects distinct types (or classes) of objects and the unique relationships between those classes. For example, while individual parks, police stations, and houses are all types of geographic objects that are related to each other by distance, organizing these relationships under the framework of a multilayer network acknowledges that these objects can be organized into distinct groups such that there are relationships between objects (such as a closeness relationship) and relationships between types of objects (such as a relationship between "parks" and "schools").

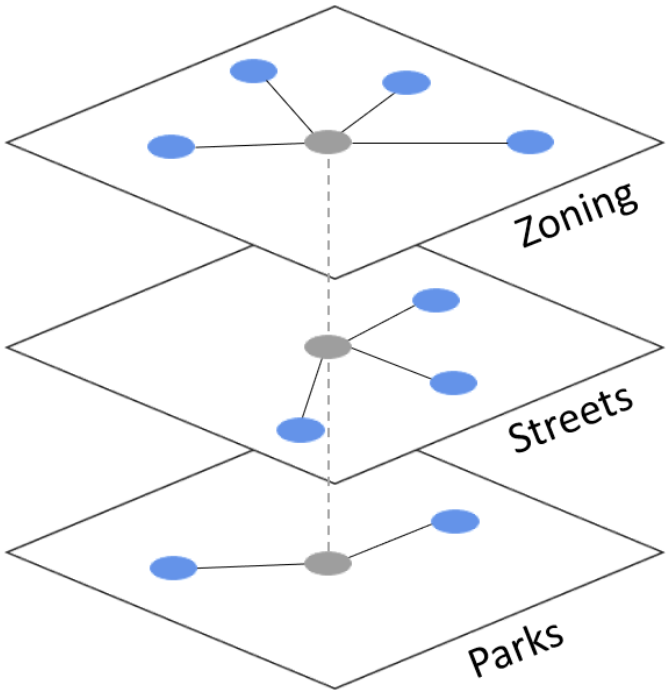


Figure 2: An example of a multilayer network grouping nodes into zones, streets and parks.

Second, research into semantic link networks (SLNs) models the potential transitive relationships between objects [3]. An SLN acknowledges that if a relationship between objects A and B exists and a relationship between objects B and C exists, then a relationship can be inferred between objects A and C. For example, as shown in Figure 3, Park A and Park B might not normally be considered similar or relevant to each other (if the distance between them is great), but if they share the same zoning there exists an informal link between the two parks. A system that acknowledges a transitive relationship between two objects that might not be obvious based on a direct measure (such as distance) would help end-users identify latent relationships between those objects.

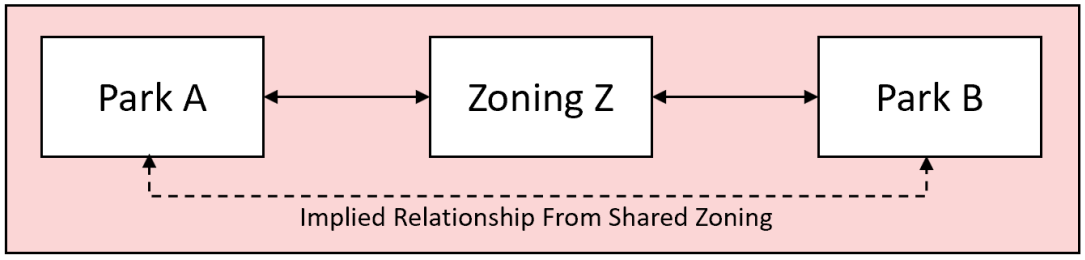


Figure 3: An example of how an SLN derives a relationship between two parks that might not be originally obvious.

1.4 Contributions of this Dissertation

The aim of this research is to create a model of geospatial data that can be used to query objects from multiple databases or services and that respects the special nature of geographic objects and the relationships between them. Such a model can serve as a foundation for services that can query enterprise systems to provide intelligent insights.

This research includes the following components.

A mathematical model of geographic relevance. This model will formalize common GIS concepts (such as layers and topology) mathematically and then extend these definitions to create a complete model that reflects the layered nature of geographic objects as well as semantically inferred links between these objects. This mathematical formulation will be proven through theorems.

An algorithmic approach for enhanced GIS query automation. The algorithms outlined in this dissertation will use the mathematical model to take traditional GIS queries and expand them to include additional geographic objects that are relevant to the user.

An application of multi-layer and SLN graph theory. As described in Sections 2.4 and 2.5, the mathematical theory behind these concepts have been developed in the past ten years but few practical applications have been developed.

The generation of a geospatial ontology. As described in Section 2.3, research into ontologies for geographic applications typically focus on abstract concepts and web-based implementations such as RDL and OWL. This dissertation will outline the creation of a geospatial ontology that mathematically represents the relationships between different classes of geographic objects.

A consolidation of GIS concepts. As described in Section 2.1, the practice of geographic information systems typically involves a series of tasks and workflows that are performed in isolation based on user requests. However, it is possible to abstract these tasks mathematically (using the concepts described above) to simplify the needs of an organization.

The implementation of a prototype system. The mathematical model described above is designed to be easily implemented by organizations that use GIS systems. This system is designed to take advantage of and extend existing GIS infrastructures. A white paper of this implementation is provided as an appendix.

An evaluation of the model and framework. Evaluating the framework using the prototype system includes evaluating the framework using several real-life case studies relevant to local communities in Howard County, MD, USA as well as the evaluation of performance and network complexity for the system based on a typical local government population.

A prototype system, SAM-GIS (Semantic Augmentation Model for GIS), has been created and provides relevant objects to a user based on the context and relevance specified above. This prototype system was developed on a database management system and uses the functionality provided by a relational database.

The dissertation is structured as follows: Chapter 2 outlines the existing research that provides the foundation of this research including the current structure and challenges of geographic information systems, research into geospatial ontologies and semantics, and research into geographic information retrieval. Chapter 3 outlines the theoretical approach of this research including the generation of a geospatial ontology, the development of a model of geographic relevance and the query systems that utilize that model. Chapter 4 discusses the implementation of the prototype system SAM-GIS and its applicability to common scenarios met by GIS users. Chapter 5 includes the experimental evaluation of the framework under this implementation, and Chapter 6 summarizes this work and discusses potential avenues for future research.

Chapter 2: Background and Related Work

This research combines the foundations of geographic information systems and context modeling and aims to use the mathematical principles behind multilayer networks and semantic link networks to create a system of relevance for geographic information retrieval.

In this chapter, we first explore the evolution of geographic information systems, earlier research into context modeling, geospatial semantics and ontologies and their applications and research into geographic information retrieval. We then discuss earlier research and provide mathematical background into multilayer networks. Finally, we explore the development of semantic link networks and their potential applications.

2.1 Geographic Information Systems

"Everything is related to everything else, but near things are more related than distant things" – Tobler's First Law of Geography [4]

We study Geography because almost every activity today has a geographic component. Geographic information systems are the practical IT implementation of the principles of Geography and are the guiding applications of the research described in this dissertation. Additionally, this work is the built on the same technical foundations of traditional GIS systems.

In this section we will define a geographic information system, outline the common structural elements of these systems, and list the typical applications that are enabled by these systems.

2.1.1 Definition of GIS

While the nature and role of **Geographic Information Systems (GIS)** have evolved over the past forty years, for the purposes of this research we will define a GIS as a "general set of hardware and software tools that are used to facilitate the utilization of geographic information to analyze and model data and to solve problems" [5]. What makes a GIS special are the added analytical workflows enabled by the storage and usage of spatial (location) data. Other common terms associated with a GIS are spatial database, spatial analysis, and location analytics.

Any spatial analysis is dependent on an assumption of **spatial dependence**: The statistical recognition that some entity or process is spatially distributed in a non-random manner [5]. Any analysis of geographic objects depends on the idea that there are natural patterns in this type of data and that concepts such as "proximity" matter to the user.

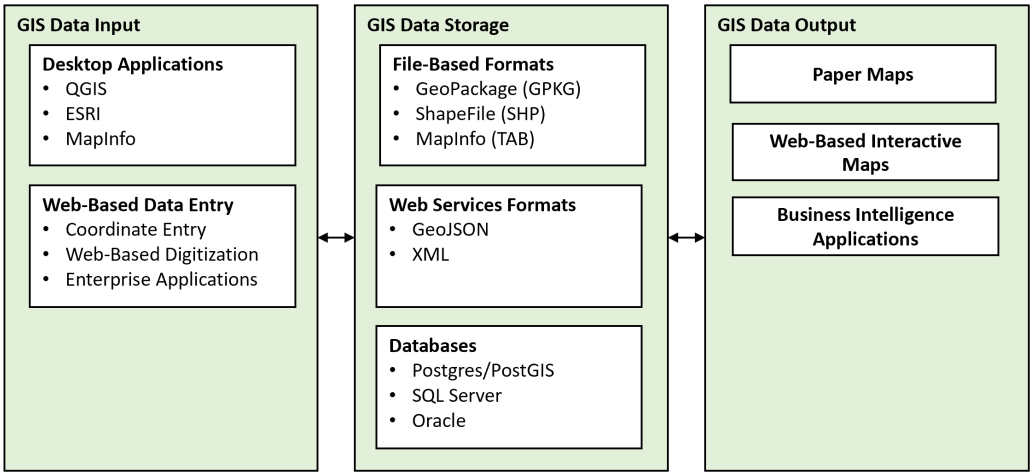


Figure 4: Sample GIS Infrastructure

A complete GIS, like many information systems, consists of components that facilitate the entry of geographic data, the storage of this geographic data and the

analysis or visualization of this geographic data. Examples of these components are shown in Figure 4. These systems can be provided by commercial vendors (ESRI [6], Mapinfo [7], Manifold [8]) and open source projects such as QGIS [8], Geoserver [9], and OpenLayers [10].

Some geographic information systems focus on the Earth as a whole and require integrating the curvature of the Earth into calculations such as distance or area. Examples include systems that model global climate or trade routes. Other geographic information systems focus on local analysis of geographic locations and assume all objects in that area reside on a straight plane. These systems are created by local or municipal governments along with the vendors and contractors that support these governmental information needs.

A **spatial data infrastructure (SDI)** consists of "geospatial data along with agreements on technology standards, institutional arrangements and policies which will enable the discovery and use of geospatial information in a structured manner" [11]. SDIs consist of the entirety of the GIS technologies described in this section as well as the interoperability of data and functions between organizations that use GIS data and technology.

2.1.2 GIS Data Entry

Most data in a geographic information system are entered using a dedicated GIS desktop application (such as QGIS) and stored in either the file system or a database that supports the storage of geographic objects. Organizations with GIS systems employ desktop applications or online interactive maps that provide access to these objects similar to how it was performed for many years using paper maps.

Geographic data "includes the information necessary to create, store and utilize digital representations of the Earth as well as the characteristics associated with specific locations and areas" [5]. Examples include the boundaries of water features as well as the political boundaries dividing countries. Data with a location attached to it is also known as **geospatial data**.

There are several aspects of geographic data that make it unique compared to non-geographic data. First, since location information involves coordinates (a latitude and longitude in most cases) the nature of the data is inherently multi-dimensional. This multi-dimensionality lends itself to analysis using mathematical techniques and algorithms well-suited to multi-dimensional data such as those that utilize matrix operations. This work will primarily use matrix-based operations as a basis for calculations.

Data can be entered into a GIS in multiple ways. The primary method involves using a desktop GIS tool to digitize features. This can involve creating and editing nodes that make up a **feature** (points, lines and polygons), running geoprocessing operations (or geospatial functions) that perform specific set-based operations on the geometries, or by using drafting or architectural tools to enter the data into the system. An example of a GIS desktop data entry interface is shown in Figure 5.

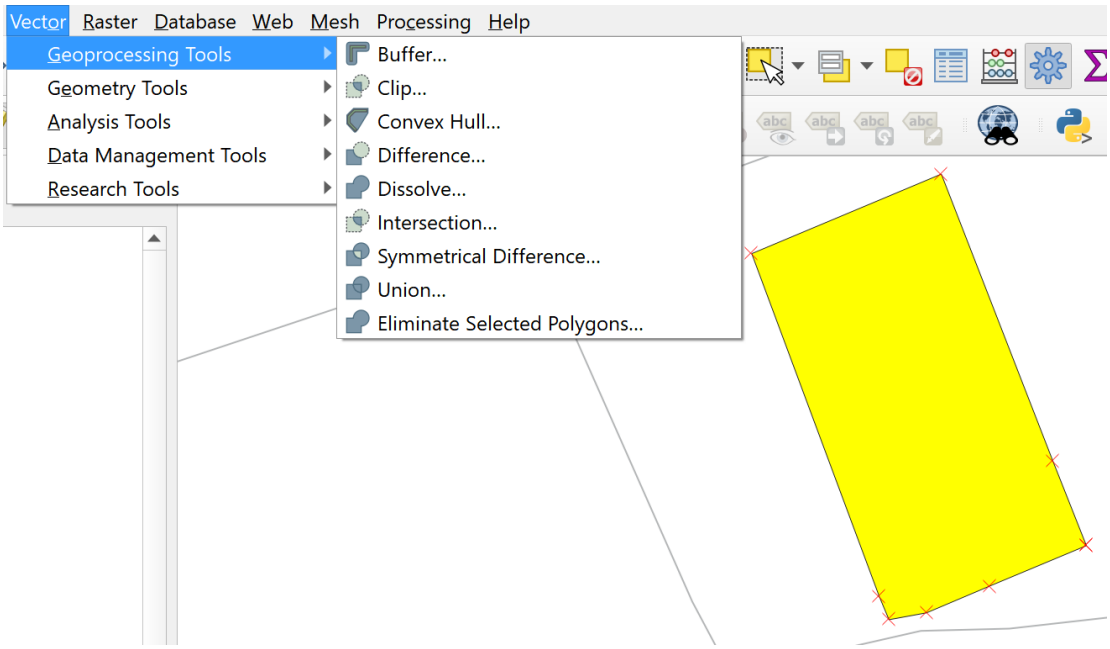


Figure 5: Features in a desktop GIS edited by placing nodes to form a geometry or by performing processing operations on them.

Additionally, as geographic information systems integrate with existing mainstream database-based systems, it has become possible to enter in geometries directly into these databases either by entering in **coordinates** (longitude/latitude) or by **digitizing** (digitally tracing features from a paper copy, scanned image or other reference) features using a web-based interface. This process, as shown in Figure 6, has allowed users to create geospatial objects without directly using GIS software, increasing the prevalence of records in enterprise databases with location and/or geometric attributes.

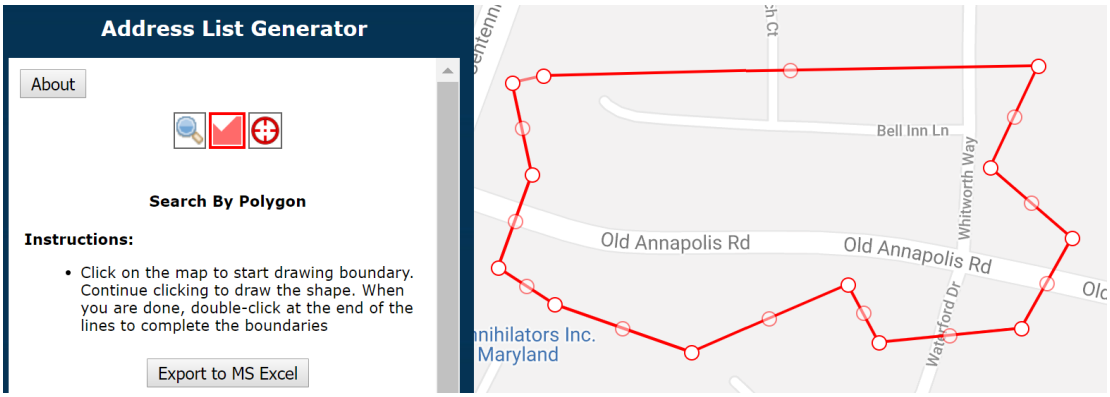


Figure 6: An example of digitizing features through a web interface.

GIS data can also be imported using Extract-Transform-Load (ETL) tools: Either the traditional tools provided by most DBMS packages or special-purpose GIS ETL tools.

2.1.3 GIS Data Storage

Geographic Information Systems have historically mimicked traditional database systems when storing geographic information. For example, the legacy Shapefile format [12] consists of a dBase database containing the tabular information augmented with additional files containing the geometries associated with those tabular records, as shown in Figure 7.

ParksPolygon.dbf	5/16/2019 9:58 PM	DBF File	977 KB
ParksPolygon.prj	5/16/2019 9:58 PM	PRJ File	1 KB
ParksPolygon.shp	5/16/2019 9:58 PM	SHP File	208 KB
ParksPolygon.shx	5/16/2019 9:58 PM	SHX File	1 KB

Figure 7: An example of a Shapefile on a file system. The DBF is a dBase database, and the SHP file contains geometries associated with those database records.

Modern database management systems such as Oracle, SQL Server, and PostgreSQL have augmented their functionality with support for geospatial data associated with the records in their tables, as shown in Figure 8. This has allowed GIS

functionality to be integrated into databases that would not normally hold geospatial data. For example, a traditional customer database might hold information about a customer's address, but with the addition of spatial columns, this information can be associated with the coordinates of the same address.

Results | Spatial results | Messages

	ID	ParkKey	ParkName	Acreage	ParkType	Geometry
1	1	COL	Collins Property	5.0320000000000000	NATURAL	0xC808000001040E000000333333B3E9BF34419A999999D...
2	2	LAS	Lash Property	16.6299999999999990	NATURAL	0xC808000001040D000000CDCCCC4C30BD3441CDCCCCCCF...
3	3	BTB	Bollman Truss Bridge	0.5075000000000000	HISTORIC	0xC80800000104130000006666666E65AC73441666666664...
4	4	SAV	Savage Park	87.4699999999999989	REGIONAL	0xC80800000104570000000000000ABC53441000000000C...
5	5	HUN	Huntington Park	11.0000000000000000	NEIGHBORHOOD	0xC80800000104140000003333333B3F5AD3441000000002...
6	6	GUI	Guilford Park	11.2539999999999996	NEIGHBORHOOD	0xC80800000104100000009A9999197BCA3441666666667...
7	7	ATH	Atholton Park (school)	9.5470000000000006	NEIGHBORHOOD	0xC80800000104160000009A99991991993441000000008...
8	8	RPH	Headquarters	7.9459999999999997	NEIGHBORHOOD	0xC80800000104140000000000000A4C23441666666667...
9	9	HHP	Holiday Hills Park	6.5460000000000003	NEIGHBORHOOD	0xC808000001042D000000333333B3878A3441333333331...
10	10	KWP	Kiwanis Wallas Hall and Park	25.1750000000000007	COMMUNITY	0xC80800000104240000006666666608913441CDCCCCCC8...
11	11	HAW	Hawthorn Park	9.9979999999999993	NEIGHBORHOOD	0xC80800000104180000003333333E18934419A9999999...
12	12	DAY	Dayton Park	12.6859999999999999	NEIGHBORHOOD	0xC80800000104090000003333333B36D173441000000001...
13	13	LIS	Lisbon Park	9.2500000000000000	NEIGHBORHOOD	0xC80800000104080000000000000D8BA3341666666664...

Figure 8: The same dataset represented as a SQL Server database table. The geometry is stored as a binary blob associated with traditional database rows.

One issue with many GIS implementations is that while the problem of storing GIS data in databases has been mostly solved, many of these were not designed for interaction and interoperability. Many GIS implementations are self-contained and collaboration among multiple GIS organizations has historically been problematic [13].

2.1.4 GIS Data Analysis and Visualization

There are multiple ways to use the data stored in a geographic information system to provide meaningful insights for users. Historically, the geographic objects were rendered using paper layout tools to create a map that could be converted to a postscript format such as PDF or printed to create a paper map.

Interactive Maps and Dashboards

Additionally, the development of online interactive maps extended this concept to provide dynamic functionality for their users.

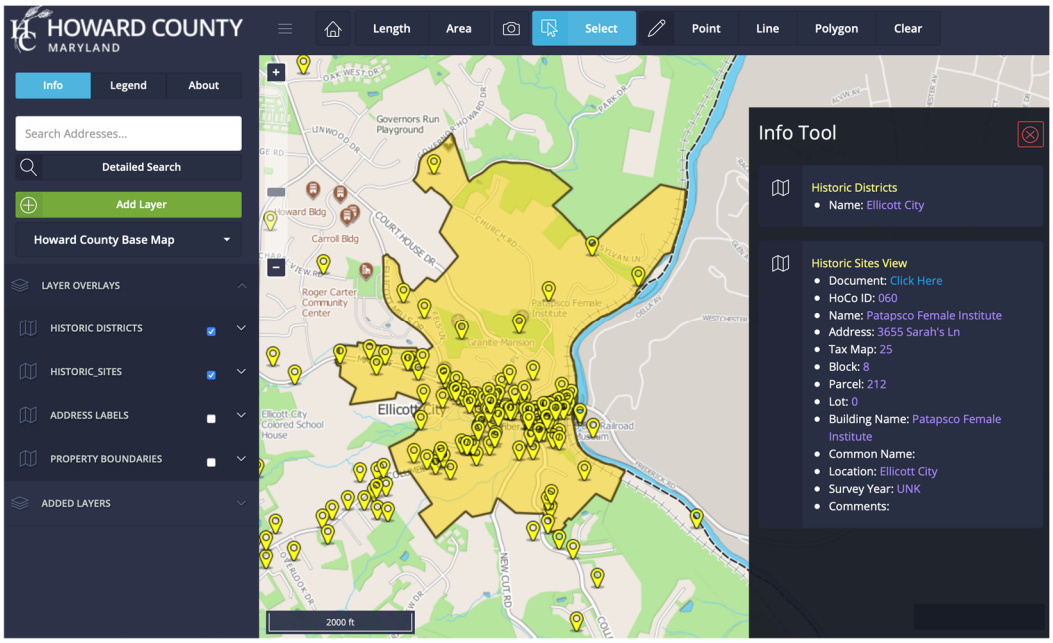


Figure 9: An example of an online GIS Interactive Map

Figure 9 is an example of a typical GIS interactive map. These interactive maps expose a subset of the functionality provided by desktop GIS applications. There are two challenges created when designing an online interactive map. First, since interactive maps typically run inside of a user's browser and the designer of that interactive map does not have control over the performance of the machines running that web browser, interactive maps cannot perform the full set of functions a desktop application can. Second, creating an effective web-based user interface that can work on multiple devices (desktops, tablets, mobile phones) puts restrictions on the interface elements that can easily fit on the screen.

More recently, database technology has transitioned to more interactive dashboards and summaries of data that provide answers to specific questions presented by users. This type of reporting does not necessarily require a map for visualization but can provide summarized information about the data from their geographic properties. For example, a report can aggregate crime statistics based on the zip code of the crime. Many COVID-19 dashboards base their statistics on geographic properties of a government jurisdiction. While this type of reporting does not require a map to be displayed, it does depend on the geographic properties of the required datasets to produce the report.

Abstracting GIS Operations

Experienced GIS users perform a variety of tasks in their day-to-day work. Existing research has tried to abstract these common tasks and analyses into various categories. If common GIS operations can be abstracted into a series of primitives a model of GIS data can be designed to fulfill those operations.

Table 1 shows a summary of common GIS analytical operations [14]. A goal of this dissertation is to abstract many of these operations into a common mathematical model.

Table 1: A classification of common GIS Operations

Search operations	Locational Analysis	Terrain Analysis
Interpolation Spatial Search Thematic Search Reclassification	Buffer Corridor Overlay Thiessen/Voronoi	Slope/Aspect Watershed Drainage/Network Viewshed
Distribution/ Neighborhood	**Spatial Analysis**	**Measurements**
Cost/Diffusion/Spread Pattern/Dispersion Centrality/Connectedness Shape	Multivariate Analysis Pattern/Dispersion Centrality/Connectedness Shape	Measurements

Any abstraction of GIS needs to work independently of the data fed into the GIS and should represent a consistent logical view of geographic information. Our model should be a single comprehensive model of geographic data that combines many of the existing GIS functions as described above.

A location of interest given by a user represents a type of context that would be useful to the user's interactions with the system. This can provide an additional filter when querying a database. In the next section we will explore the concept of context and how it is modeled in computer systems.

2.2 Context Modeling

When humans communicate with each other, the participants do not need to explicitly convey every idea necessary for the conversation to take place, particularly background information [15]. In this section we define context in the abstract, review earlier attempts to model context, and ultimately move towards a definition more suitable for computation.

Ubiquitous Computing

The idea that computers might be embedded in the environment or might learn to work together based on that environment came to the forefront in the early 1990s [16]. **Ubiquitous computing** is the idea that computers can interact with humans in any scenario. When the term was developed, desktop computers were common and the idea of having a computer follow a human wherever they go (like they currently do with mobile phones) was relatively new.

This trend described a move away from the concept of a computer that worked in isolation. If a computer could be embedded into its environment based on various contextual factors (such as the time of day or the location of the device) then it becomes increasingly important to define abstract representations of the data and records stored on that computer.

We first need a definition of a computer model since the development of a model of context is one of the goals of this research. A **model** is a simplified abstraction representing some aspect of reality. **Model Driven Development (MDD)** is an approach to developing software that proposes using machine-readable models at various levels of abstraction as its main artifacts. The purpose of MDD is to take abstract models of natural concepts and convert them into concrete models that can be modeled using mathematics and computation [17].

When designing these models, we need specific tools and mathematical formalisms to create the elements required for MDD. This section introduces definitions of context and proceeds to describe ways to model context in information systems.

2.2.1 Definitions of Context

The Merriam Webster dictionary defines context as "the interrelated conditions in which something exists or occurs" [18]. This definition alone is not enough to provide a mathematical and computational foundation upon which to create applications which use context [19], therefore we examine the literature to look at prior attempts to define context in a way that proves useful to developing information systems.

Context is defined by [20] as "any information that can be used to characterize the situation of an entity. Elements for the description of this context information fall into five categories: individuality, activity, location, time, and relations." Another definition states that context "is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves" [17]. Other definitions on context are based on example and are built around a specific application [21, 22]. Context is an operational term in that something is context if it used in the interpretation of data [23]. Since data is modeled and analyzed for a specific application, we apply context to that application.

From a knowledge perspective, the following are true: [24]

- Context is knowledge, and knowledge is context.
- Context is defined with respect to a focus of attention (knowledge use).
- At any given moment there is external knowledge and contextual knowledge.
- Pieces of contextual knowledge are structured around the focus of attention.
- Contextual knowledge has a granularity that depends on the distance to the focus of attention.

- Context structure evolves dynamically with the focus of attention.
- Context is relative to an observer.

A key aspect of most definitions of context is that an object's context usually involves **contextual elements** such as physical properties like time and temperature or non-physical "human" properties like beliefs and desires [15, 25, 26]. Most definitions of context include the following [22]:

- **Location** – This includes physical location as well as electronic locations (such as IP addresses).
- **Identity** – Applications can be developed to store user information such as their preferences, knowledge and detailed activity logs of physical space movements and electronic artifact manipulations.
- **Time** – This can include time in an absolute sense as well as relative time (the position in a sequence).
- **Environment or Activity** – The artifacts and physical location of the current situation.

Contextual elements can also be defined by the institutions that structure both the ongoing and activity and social relations which the activity is embedded [27].

There have been adaptations or alternatives to the definitions of context in terms of different domains and applications but ultimately any definition of context has proven to be narrow or subjective and a framework to consistently define context in any universal fashion has proven difficult [28].

A **relation** is an aspect of context that refers to a dependency between two objects that emerge from the circumstances these two objects are involved in [20]. Therefore,

context does not only include the intrinsic qualities that an object has but can also include the relationships between multiple objects and the effect of those relationships on those intrinsic properties.

For example, if Person A asks Person B to turn off the light in a room, Person A has not specified which light to turn off, but a shared context between the two people lets Person B know that Person A means the light for the room they are leaving. To come to that conclusion, they needed to know their location in addition to the location of all light switches in the area. With that information they can identify the most likely candidate light switch to turn off and act on that information. This example shows how the computer science definition of context derives from our human understanding of context. Models of context are adapted from our human sense of context-awareness [29].

Context reasoning is the process of taking source data (from databases or sensors) in order to make interpretations about the context of that data using some type of rule-based logical inference [30]. This can be done to provide insight into the data (data mining or statistical inference) or to help resolve inconsistent information when bringing together data from different sources. Traditionally such context reasoning systems have been extensions of existing applications which can be difficult to support or not adaptable to other systems. Therefore, it would help to separate any context reasoning system from the source system. A data warehouse is an example of a tool to facilitate this separation [15].

2.2.2 Computational Applications of Context

If a software system can incorporate implicit information about a user's environment then the software can gain the ability to adjust its behaviors according to the available information in that environment [30].

The development of applications that integrate the user's environment started with the concept of situational computing. **Situational computing** refers to the ability of computing devices to detect, interpret and respond to aspects of a user's local environment based on the sensors installed on the user's device. This functionality can be added to existing applications or can be the centerpiece of new types of applications [31].

While situational computing focused primarily on the transition to mobile computing and therefore the location of the user, the idea of a user's situation expanded to include the different contexts described in Section 2.2.1. The goal of context-aware systems are to move computing devices into the background of human activity and for those devices to provide natural and unencumbered interaction with the user [32].

Context-Awareness

From the perspective of the computing system, **context-awareness** is the capability to provide relevant services and information to users based on their situational conditions [15, 33]. A system is context-aware if it uses context to provide relevant information or services to the user [34].

In computer science and mathematics the study of context-awareness has been applied to artificial intelligence research [35], information retrieval applications [36], sensor networks [37] , multimedia retrieval, [36] and human-computer interaction [26,

27]. Mathematical formalizations of context-awareness have been developed to provide a basis for these applications [35].

Context-Sensitive Systems

A **context-sensitive system (CSS)** is a computer system that uses context to provide more relevant services or information to support users performing their tasks [38]. There are two types of applications described by these systems: continuous applications which take into account the user's situation and discrete applications which take separate pieces of data about a system and evaluate how they overlap [21].

A focus of context-sensitive systems is to enhance human-to-computer or computer-to-computer interactions to mimic human-to-human interactions [38].

A context-sensitive system needs to deal with the following issues: Which kinds of information to consider contextual, how to represent this information, how to acquire and process it, how to integrate the context usage in the system and how to present it [38].

Challenges in the creation of context-sensitive systems include a lack of conceptual tools and methods to account for context-awareness. These systems are typically driven by the method of data entry (for example, sensor-driven applications) rather than representing an abstraction of the system being modeled. Another challenge includes the ability to distribute, modify and reuse these applications in different organizations [25, 39]. An ideal model of context should be adaptable to any organization and accommodate any user.

Context-Aware Computing

Context-aware computing (CAC) is defined as the application of additional code to existing software that makes that software aware of its context. It takes the information in a user's physical and electronic environment and uses that as a context for the interaction between humans and computers [25, 40]. This definition emphasizes the context that software operates in with a focus on mobile devices which can easily change context (such as location or proximity) [40] and not the context of data independent of the user's interactions with that data. While context-aware computing has been researched for the past 30 years it has become increasingly relevant in the past few years due to the emergence of the "Internet of Things" [41]. Additionally, as mobile devices have proliferated in society and the size of these devices have shrunk it has become increasingly difficult to design user interfaces around these size limitations.

Considering the sensor-based data a mobile device generates in the background, it makes sense to try to use the mobile device's contextual information (for example, its location) to aid the user when interacting with their mobile devices. Similarly, the use of context-aware applications can hide the user from having to adapt to a computer's interface and instead can let the computer intelligently make sense of the user's context (for example, in a smart home situation) to anticipate what that user might want to do [32].

Applications that are context-aware observe user behavior and use this information to determine why a situation is occurring. This is the responsibility of the designer of the application using this information. The designer uses incoming context to determine why a situation is occurring and uses this to encode actions in the application [34].

Such systems generally rely on triggers. These triggers take a user's context and pass it to a program to be processed [21]. For example, a location trigger can send a message to a user when they have entered a restaurant.

Additional features of context-aware applications include [34]:

- **Contextual Sensing** – The ability to detect contextual information and present it to the user, augmenting the user's sensory system.
- **Contextual Adaptation** – The ability to execute or modify a service automatically based on the current context.
- **Contextual Resource Discovery** – Allows context-aware applications to locate and exploit resources and services that are relevant to the user's context.
- **Contextual Augmentation** – The ability to associate digital data with the user's context.

Designing computer systems to be context-aware provides the ability to enhance the interaction between humans and computers and make communication richer. An alternative definition of a context-aware system is one that "uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task" [34].

A problem with developing context-aware systems is the existence of a mismatch between the context-awareness exhibited by humans and attempts for computers to mimic those systems. In the next section we describe attempts to rectify this by creating models that computationally describe context, work reliably, and have the intelligence to execute actions that reflect the user's context [29].

This work includes the evaluation of the relevancy of geographic objects to a user and this definition suits that task. We use the features of context-aware applications described earlier to focus this research.

2.2.3 Models of Context

Providing a good model of context can reduce the complexity of context-aware applications and can improve their maintainability and evolvability. A good model can also reconcile a human understanding of their environment with how a computer understands the same environment. This has led to the development of more formal representations of context in computer systems [42].

This section outlines several proposed approaches to context modeling and provides guidelines to evaluate them.

Widget-Based Context Modeling

Widget-based context modeling involves the extension of the concept of a device driver to system interfaces. It provides a low-level interface for systems or devices (such as sensors) that want to communicate with each other. In this model the rules defining context are hard-coded into the interfaces [25].

Service-Based Context Modeling

Service-based context modeling resembles the client-server infrastructure that the World Wide Web or that an enterprise system would use. This type of modeling utilizes multiple web servers that store context information and exposes web services with protocols such as XML or JSON to communicate messages between these servers [39]. The advantage of these systems is that they are accessible by any computer system, especially those that are web-based, that can access the services. This can help make

these systems platform agnostic: Changing the configuration of an input to the model (sensors, services, and devices) is independent of other inputs to the system.

Another benefit of a service-based model is that it offloads the computational burden of these systems to each sensor. This converts the modeling problem to a primarily infrastructure-based challenge.

The creation of a service-based model of context includes the following. These requirements reflect the web standards used on the Internet today:

- Designing the data formats and protocols used by the infrastructure. These standards are the glue that allows the separate pieces of the infrastructure to interoperate. Ideally, these formats and protocols are as simple as possible.
- The data formats and the protocols need to be rich enough to cover the diverse range of sensors and assorted types of contexts.
- A distinction needs to be made between the data provided by the inputs and the contextualized data communicated through the services. This can be facilitated through the creation and implementation of metadata describing this translation.

Blackboard Model

The **blackboard model** centralizes the data from diverse sources into a central repository. This repository, which resembles a data warehouse, handles all rule-based inference of context and relays the results of this context back to the sources [43].

The blackboard model adopts a data-centric rather than a process-centric point of view. The repository receives input from all sources and stores them centrally. Other devices can access this repository and receive messages based on preset filters [23].

An advantage to the blackboard model is that the central repository provides a standard communication link for each component which is the same for all devices that use the blackboard and avoids the need to provide a peer-to-peer link between all devices on the network. Like the service-based model, when one input fails on the network there is no disruption to the communication between the other inputs and the repository and any other devices that try to access the repository. Since a blackboard is database-driven, the data in the repository can be archived and audited like any other server-based system.

A disadvantage is that there is an overdependency on that central repository, however using modern infrastructure practices the repository can be built with redundancy and reliability.

Model Considerations

When designing a model of context we aim to consider the following factors: [23]

- *Efficiency* – Is deriving context computationally efficient and time efficient? This becomes especially important when designing real-time applications that must evaluate context immediately.
- *Configurability* – Is the model adaptable to different situations? Can it efficiently add new data sources or new factors in its analysis without compromising the overall data model?
- *Robustness* – Can the model handle missing or incomplete data? In a sensor network, can it handle a malfunction with one of its sources? In a GIS, datasets are gradually built and refined and might contain incomplete data until digitization efforts are completed.

- *Simplicity* – Can a model of context be designed so that its mathematical foundations are intuitive? Can such a model be easily programmed as software, for example, as a series of queries and stored procedures associated with a relational database? If it is not, then the practical applications of such a model are limited.

Both the service-based and blackboard models function as middleware. This allows a focus on the model itself rather than the individual inputs and data sources feeding into the model [15]. The blackboard model will be the primary template for this work with extensions that reflect a service-based model.

2.2.4 Ontology-Based Context Modeling

Ontology is a branch of metaphysics dedicated to the study of the nature of beings and things related to those beings. From an information systems perspective ontology is "a formal naming and definition of the types, properties and interrelationships of the entities that fundamentally exist for a particular domain of discourse" [44].

Practically, an ontology defines "a common vocabulary for researchers who need to share information in a given domain. It includes machine-interpretable definitions of basic concepts in the domain and relations among them" [45].

Ontologies allow users to share a common understanding of the structure of information in their domain, enables re-use of domain knowledge, makes domain assumptions explicit, separates domain knowledge from operational knowledge, and helps analyze domain knowledge [45]. Note that these characteristics overlap some of the aims of both service-based context modeling and the blackboard model described

in the earlier section. Therefore, ontologies have been thought of as an effective tool in modeling context.

Ontologies in Information Systems

The users of a computer system will approach the data and applications associated with those systems in unique ways depending on their context. In any data-driven application the differing needs and backgrounds of the users will lead to different viewpoints and assumptions about the same subject-matter. The difference among viewpoints can lead to poor communication between users of these systems, create difficulties in identifying system requirements, limit interactivity and limit interoperability. This difference can also limit the potential for re-use and sharing when developing applications [46]. Section 2.1 discussed how GIS professionals re-create paper and interactive maps because GIS applications derive from the problems those professionals are asked to solve rather than a shared conceptualization of geospatial concepts.

An ontology embodies a shared worldview with respect to a given domain. This world view is conceived as a set of concepts such as entities, attributes and processes, their definitions and inter-relationships. Another definition of ontology refers to it as a formal, explicit specification of a shared conceptualization. A **conceptualization** refers to an abstract model of some phenomenon in the world by having named the relevant concepts of that phenomenon. *Explicit* means that the type of concepts used, and the constraints on their use are explicitly defined. *Formal* refers to the fact that the ontology should be machine readable, which excludes natural language. *Shared* reflects the

notion that an ontology captures consensual knowledge, that is, it is not private to some individual, but accepted by a group [47, 48]:

Properties of Ontologies

Ontologies can be generic or specific. A classification of ontologies from generic to specific include [48, 49]:

- **Generic ontologies -** Generic ontologies describe general concepts, independent of any task or domain (e.g., time, space, etc.).
- **Domain ontologies -** Domain ontologies describe concepts for a specific domain (such as physics or biology).
- **Application ontologies** - They describe concepts which are necessary for a specific application. Application ontologies depend on both the domain and the generic ontologies.

At a high-level, ontologies can be abstracted from existing systems that describe common situations. These ontologies can be shared, reused, and adapted to other systems. On the other hand, low-level ontologies work as a specification of what the system was designed for. In other words, they reflect a specific application [48].

An example of an ontology could involve furniture. Concepts of furniture include "furniture", "chairs", "tables" or "beds". The interrelationships between these concepts could include:

- Chair *is a type of* furniture.
- Chair *is adjacent to* table (for example, in a dining room set).

Classes are the typical focus of most ontologies. **Classes** describe concepts in the domain of interest. We can also have **subclasses** (for example, when the chair is a type

of furniture). We can even define **properties** for our classes (like the type of chair, the materials used, or their size) [45].

An ontology should include:

- A set of classes and properties attached to those classes.
- A **taxonomy** of those classes (defining subclasses).
- Restrictions on the properties of those classes.
- Instances of those classes.

The process of creating an ontology usually involves the following steps [45]:

- Defining the domain of interest and examining previously defined ontologies for that domain.
- Defining all terms in the domain of interest. What concepts are important to the user of a system?
- Defining the classes in the domain. This can be top-down (starting with the definition of the most general concept in the domain) or bottom-up (starting with the definition of the most specific classes in the domain). One class can be represented by multiple terms.
- Defining the relationships between classes. These relationships can be hierarchical or peer-to-peer depending on the domain.
- Defining the properties of classes that are important to the user. These are the characteristics of the objects in the classes that can be derived from their data sources.
- Describing the types of values those properties can contain.
- Populating instances of these classes with data.

This work includes the creation of a geospatial ontology and will attempt to mimic the process outlined above.

Benefits of Ontologies

Ontologies attempt to solve the problem of inconsistent perspectives on information by establishing formal definitions of the conceptualizations described earlier [46]. This includes:

- Providing unambiguous and consistent definitions for terms in a software system. Any systems as part of an application should use those definitions to maintain compatibility between those two systems.
- Providing a shared understanding of concepts through these consistent definitions. While individual users can provide their personalized perspectives on these concepts a common understanding is required for interoperability.
- Establishing a network of relationships between concepts in the system. This work describes a graph-based model of geographic relationships, and this will require a network of relationships to be defined.

Since ontologies represent concepts and their relationships and context are a type of relationship between concepts, we can use ontology to help model context. Ontologies are useful for modeling context because they help to provide common terminologies and rich semantics to enable knowledge sharing and reuse between different systems. Ontologies can also function as an infrastructure piece of a context-aware system.

Evaluating Ontologies

Ontologies can be evaluated using the following criteria:

- **Clarity** – Terms must be defined through necessary and sufficient conditions so that they can be identified unambiguously and communicated effectively.
- **Coherence** – Definitions must be consistent.
- **Ontological Commitment** – Ontologies should make just enough claims about the domain to support the intended knowledge sharing and reuse.
- **Encoding Bias** – Ontologies should be specified at the knowledge level without depending on a particular symbol-level encoding.
- **Extensibility** – Ontologies should offer a conceptual foundation for anticipated and potentially anticipated tasks.

Ontology-Based Models

An **ontology-based context model** represents knowledge, concepts and relationships about a domain and describes specific situations in a domain [42, 46].

When designing computer systems intended to gather information from different sources (like sensors) and communicate meaningfully with each other, each of these different sources must share a common ontology and communication language [50].

Most research into ontology-based context modeling involves the development of a data structure and a language such as OWL (web ontology language) [51, 52].

For the purposes of this research, ontology is the categorization of objects and their relationships to other objects.

2.2.5 Challenges with Context Modeling

The biggest challenge with context modeling is that most of the literature on context focuses on its philosophical and linguistic origins or on user-level modeling languages. When a generalized system is proposed, it involves the creation of general data schemas

(RDF, UML, XML) that conceptually model a system and provide interoperability between systems but does not include a data-driven aspect. Since most context models are fairly complicated (in order to remain flexible) they tend to be harder to manage [24]. Additionally, there have been very few attempts to apply a rigorous mathematical model of context because of this complexity and even those attempts have proven overly complex [53, 54] and therefore impractical for a production environment.

The model of context described in this work will consider these challenges as well as the factors listed earlier when describing models of context (efficiency, configurability, robustness, simplicity).

2.3 Geospatial Modeling

The concepts related to context modeling (such as ontology and semantics) have been extended to geospatial data. These efforts have been developed either as an extension to traditional GIS applications (explored in Section 2.1) or by incorporating factors related to geospatial data to existing context-modeling frameworks (explored in Section 2.2). In this section, we outline attempts to apply and extend concepts related to context modeling to geographic objects.

Spatial Models of Context

Modeling context in a geospatial domain generally includes information about the locations of the user and the location of the objects in their environment. This can include **geometric coordinates** such as latitude and longitude that represent points or areas in a metric space. **Symbolic coordinates** represent identifiers such as room numbers or labels that represent locations [42].

Even if physical location is not a primary context for an application, a spatial organization of context can still be beneficial. Modeling geographic objects can be defined at distinct levels of abstraction [42, 55]. These include:

- Tier 0 - An ontology representing physical reality based on an assumption of one real world.
- Tier 1 - Observations of reality at a given location that produces some kind of measurement value.
- Tier 2 – Single observations are grouped with individual objects that are defined by uniform properties.
- Tier 3 – Social Reality – All objects and relations that are created by social interactions.
- Tier 4 – Modeled Rules that are used by cognitive agents (both human and software) for deduction.

The nature of geographic objects can blur the types and strength of relationships between those objects. For example, boundaries between objects can be loosely defined (for example, between the end of a river and the ocean it flows into). Geographic objects can be one-dimensional, two-dimensional, or three-dimensional with different topological rules defining the different types of relationships between objects, particularly objects of different dimensions. The same object, such as a river, can be represented as a two-dimensional object (the path of the river) or a three-dimensional object (the area of land covered by water), and the nature of geographic objects can change over time (the width of a river depends on the amount of rainfall in the area). **Bona-fide boundaries** represent those boundaries caused by the physical nature of the

Earth (for example, the location of rivers and mountains) and **fiat boundaries** are artificially created boundaries that are projected onto the Earth arbitrarily, such as political boundaries [56].

Ultimately, determining the nature of the relationship between geographic objects can be accomplished with ontologies, but any model of geography will sometimes depend on arbitrary (or consensus) agreements on the nature of these objects and the relationships between them.

2.3.1 Geospatial Ontologies

When examining the semantic relationships between diverse types of geographic objects, ontologies can be helpful because they provide explicit and formal definitions of thematic entities and their relationships [47, 57]. For example, an ontology explicitly defines the relationships between parks and streets (a park should be adjacent to a street since that street provides transportation to the park).

Geographic systems can be said to already contain semantic information related to the geographic objects they contain in the form of strong metadata and multinational standards [55]. In existing geographic information systems, ontologies can also be formed as a **digital gazetteer** which is a modern version of traditional paper mapping and location lookup systems. Applications of digital gazetteers include geocoders, navigation systems, and geographic information retrieval systems [58]. A downside of gazetteers is that they are traditionally limited to manually compiled lists of toponyms (place names) and depend on a basic retrieval mechanism, so ontologies can potentially expand on the traditional role gazetteers have played in geographic information systems [59, 60]. It is generally believed that ontologies can help enhance

organizations' existing spatial data infrastructures by establishing relationships between the different datasets (or layers) in that SDI [61]. There have also been recommendations for GIS practitioners to explore the semantic operability of their datasets as it relates to datasets from other organizations [55].

Semantic Heterogeneity in Geospatial Datasets

Geospatial datasets that can, on the surface, represent the same objects can differ in their interpretation since some concepts can be interpreted differently depending on local ordinances or different perceptions or interpretations of those objects [62]. For example, while many local jurisdictions differentiate between paved roadways and "dirt" roadways, the proper dataset to place gravel roadways is open to interpretation. A local politician and a state politician might have different perspectives on the geographic landscape [63]. Ontologies and the development of conceptual graphs can be used to resolve these **semantic heterogeneities** [57, 64].

What is the difference between a hill and a mountain? What differentiates a river from a stream? The differences in interpreting these terms stems from both the physical phenomena related to geography (such as plate tectonics) and the different types of human activity that relate to these phenomena (humans adapt to living next to a river differently than they would if they lived near an ocean) [65, 66]. A practical example of semantic heterogeneity is found in the differing spatial reference systems (or projections) used by GIS practitioners in different organizations, however most GIS software can automatically reconcile those differences with minimal effort [55].

Existing Ontological Approaches to Geospatial Data

Ontologies can also be useful when dealing with heterogeneous data (different layers and different types of objects), especially data originating from different organizations with different definitions of common geographic terms [67-69]. The relationships between geographic layers can be implied when these layers are visualized on a paper or interactive map, but these relationships are not fully mathematically represented or formalized [5]. For example, a map can convey to the user the relationship between a street, a river, and the bridge on that street crossing that river. However, databases holding geographic data do not formalize this relationship nor provide a mathematical representation of the strength of the relationship between these classes of objects.

Ontologies tend to be hierarchical in nature, and this is true for geospatial data (rivers and lakes can both be classified as bodies of water), but geospatial ontologies can also be defined by functional relationships between objects [65, 66]. For example, a road can cross a stream at a bridge, but this does not imply a hierarchical relationship between roads, streams, or bridges). Additionally, ontological relationships need to consider different types of geospatial objects (points, lines, and polygons) and the topological relationships that exist between those objects. The interpretation of these relationships can vary by organization and this can create ambiguity when attempting to define the relationships between geospatial datasets [65].

Typically, geospatial ontologies can be used to facilitate the interoperability and integration of diverse geospatial systems and most research into geospatial ontologies focus on web service implementations [61, 70, 71]. For this research, a geospatial

ontology will help establish semantic relationships between different types of geographic objects.

In each domain, multiple ontologies can be developed to represent different interpretations of the same shared dataset. These ontologies can be merged into a central ontology representing the entire domain (**ontology alignment**) [69].

Existing research into geospatial ontologies acknowledges the need for geospatial ontologies to support the semantic search of geospatial objects. In fact, several researchers have outlined or designed systems to programmatically create these ontologies. A generalized framework for geospatial semantic search outlines the following necessary components [57]:

- An ontology defined to represent the thematic knowledge in the repository of geographic datasets; the thematic classes are organized in a taxonomy, and besides subsumption, other semantic relations can be set between thematic classes, which can also be defined through description logic axioms.
- Different algorithms to semi-automatically add new datasets to the repository, and consequently introduce new knowledge to the ontology.
- A set of semantic services to enable external clients to find, translate and integrate thematic information from different datasets in the repository.

Early attempts at describing geospatial ontologies started with the development of mathematical formalisms describing the topological relationships between classes of geospatial objects using set theory [72]. Other attempts extended these set-based definitions by utilizing Boolean logic and matrix operations to describe a prototypical

ontological structure [73-75]. There have also been preliminary attempts at geospatial ontology alignment or harmonization [76].

Recent attempts at creating formal processes for creating geospatial ontologies generally involve the extension of GIS applications through the use of web services [67]. These attempts extend ISO standard web services such as WFS, WMS and CSW [70, 77].

Attempts to use ontologies to directly solve geospatial problems have emerged in the past few years. These applications include assisting in the observation and analysis of raster data [69, 78], emergency management [79] and transportation networks [80].

2.3.2 Geographic Semantics and Similarity

The previous section described the semantics of geospatial objects in terms of the relationships between the classes (or layers) describing those objects. In this section we summarize prior research into semantics of individual geospatial objects as well as the nature of the relationships between those objects.

Geographic Relevance

For geospatial applications, similarity measures play a core role in understanding and handling semantic heterogeneity and in enabling interoperability between services and data repositories. When we talk about semantics, we refer to the inherent meaning of the geographic objects and the relationships between them in addition to a strict definition of their relationships [81].

Geographic relevance can be defined as "the quality of an entity in geographic space or its representation in an information system." This quality is expressed as the relation between an entity or its representation and the actual context of using the

representation [82]. While similarity describes a relationship between two geographic objects, relevance refers to the relationship between any geographic object and the user. Geographic relevance corresponds to a user's intentions and how that relevance relates to a user's environment.

Defining a formal abstraction of the relationships between geographic objects using semantic similarity as a measure of the strength of that relationship can lead to a generalization of geospatial data analysis [83, 84].

Relevance in Navigation

By identifying patterns in user behavior, a system can attempt to predict the future behavior of that user in the form of relevant search results. This requires recording user behavior as a sequence of navigation actions and then using these patterns to produce geographic objects that might continue that pattern [85].

As described earlier, traditional GIS applications rely on a series of hard-coded search mechanisms, notably a keyword search on objects. A weakness of a keyword-search approach is that the choice of keywords (or aliases) provided by the user dictates the nature of results returned to the user, rather than the meanings of those terms [86, 87]. For example, a user might think they are searching for a stream when in a GIS database the object they are searching for is a river because that user either does not know the difference between the two or the organization producing the GIS database has made a distinction between the two concepts (stream or river) that the user is unaware of.

A better navigation system should acknowledge the user's context by attempting to understand their intent. This includes a user's interest in items local to them, such as a

search for restaurants nearby. Modern mobile applications handle this with explicit searches dedicated to using the mobile device's GPS location, but these searches tend to be hard-coded and specific to pre-designated classes of geographic objects (such as a listing of restaurants).

When discussing the similarity of geographic objects, we need to be aware of the different meanings of "similarity" depending on the type of application that performs a similarity search. A well-defined geospatial ontology (as described in the previous section) determines the type and strength of relationships but does not go deeper to examine the relationships between the geospatial objects themselves.

A framework that can define how similarity can be computed may include the following steps [88]:

1. Definition of application area and intended audience.
2. Selection of context and search (query) and target concepts.
3. Transformation of concepts to canonical form.
4. Definition of an alignment matrix for concept descriptors.
5. Application of constructor specific similarity functions.
6. Determination of standardized overall similarity.
7. Interpretation of the resulting similarity values.

When examining geospatial semantics, different organizations and different users have different interpretations of the meaning of geospatial concepts and applications [76]. For example, multiple government departments might have differing definitions of what "open space" means. A local government parks department maintains a geospatial dataset containing park and playground polygons. A state park department

maintains a dataset of state parks, and a local government public works department might contain open space that are not parks of any kind. When a user queries a geographic information system to show all land that cannot be built, it must, at a minimum, query all these datasets (see Figure 10 for an example of an open space query results).

Figure 10: A query for open space might return results from two different datasets: Easements (in brown) and parks (in green).

Additionally, geospatial relationships require additional calculations (such as distance or overlap) that go beyond the simple traversal of a graph (such as when querying an RDF graph) [84, 89]. We do not simply acknowledge the existence of a relationship between two geospatial objects; we also mathematically define and calculate that relationship based on a topological measurement on those objects.

Approaches to identifying semantically-relevant objects in a geographic search include adapting the collaborative filtering method to state that objects relevant to users in a specific geographic area in the past are relevant to current users searching in the same area [87]. That is, we can use past geographic searches to predict the context of users to focus future searches.

Applications of research into geospatial semantics and the use of knowledge graphs include those related to crime analysis, transportation networks and communication networks [90].

A challenge in the definition and interpretation of similarity measures is a lack of formal definition or facility to interpret the results of similarity calculations to provide geographic context. This research aims to provide that formal mathematical definition of similarity and semantics for geographic objects. It also tries to create a method to interpret a user's context (such as their current geographic location if they are using a mobile device) to enhance the results of a semantic search.

Geographic Information Retrieval

Geographic Information Retrieval (GIR) can be defined as the relationship between a user's geographic needs and the spatiotemporal expression of geographic objects in the user's surrounding environment [91, 92]. GIR is considered a specialized branch of information retrieval. It includes all the concepts associated with information retrieval with an emphasis on spatial and geographic indexing and retrieval [81, 93]. Many GIR implementations use pre-existing geographic web services to facilitate their queries [94].

A user's needs can be defined in many ways, such as the nearest object to a location or the boundary containing a location. In the GIS field these are known as **topological** constraints [5]. Examples of topological constraints include:

- *Boundary constraints* – What boundaries are relevant to the location of interest? For example, the school district containing a house location.
- *Nearest neighbor* – Finding the nearest (by distance) geographic object to the location of interest. For example, the park nearest to the user's location. This type of metric can have different representations based on the different ways one can calculate it. For example, if a park can only be accessed by a road network, then we must travel along that network to determine the distance to that park. Also, if we want to prioritize geographic artifacts by distance, how does this translate into a relevance score [92]?
- *Adjacency* – If the location of interest is a line or polygon then we might be interested in adjacent geometries. For example, if commercial zoning is adjacent to residential zoning.

Geographic information retrieval research focuses on two areas. The first is the acknowledgement that an increasing amount of computing happens on mobile devices such as phones and that given their limited screen real estate it is expedient to focus the output of mobile computing systems on geographically relevant results [82]. The second is an emphasis on identifying geographically relevant artifacts from text corpora [95, 96].

Traditional geographic information retrieval examines both the thematic, geographic and temporal aspects of objects [71]. Thematic information includes

attributes associated with a geographic object such as the name of a location and various measurements associated with that location. Geographic information includes the geometries (including coordinates) associated with a location [93]. Temporal information accounts for changes in thematic or geographic information over time, such as changing political boundaries.

These algorithms can also help with querying indirect locations. An **indirect location** refers to a location inferred from a user's query. For example, a query for one park indirectly reveals other amenities to the user [97].

Many GIR queries come in the form of a triplet, <what, relation, where>, where "what" and "where" are geographic objects with a relationship between the two [89, 98]. There are two types of relationships we can define of this type: relationships between classes of objects and relationships between objects of the same class. In GIR we define relationships between classes of objects by setting up an ontology on those classes. For example, we can define the class "address point" to be contained within "school boundary." Within an individual layer, we can define geographic relationships between them. For example, two addresses can have a relationship defined by the relative distance between them.

A query to a geographic information retrieval system generally involves the retrieval of keywords from a document of interest and then applying mathematical operations like those used in a geographic information system. Place names are resolved to geometries with coordinates and stored in a spatial database. Then the spatial similarity between the objects is computed by distance or topology measure [93].

A model for geospatial information retrieval from a traditional spatial data infrastructure (SDI) is a quadruple:

$$\{S, T, Q, R(Q_i, T_j)\}$$

Where S is the collection of all services (WMS and WFS) offered by the infrastructure, T is the collection of all feature types provided by the SDI, Q is the representation of a user's query, and R is a function that determines how relevant a feature T_j is for a query Q_i [99].

References to geographic objects can be linked using a geospatial ontology. For example, any references to cities within the same country can be linked together. Geographic retrieval systems should be able to process these types of queries: Pure textual queries that involve a keyword match between geographic terms (like city names), pure spatial queries that exploit geospatial ontologies (all hotels within a given geographic area), and textual queries with place names (all hotels within a named city) [96].

Ranking Geographic Objects

The above work reflects a desire to query a knowledge base to find all relevant documents for the user but does not include a ranking or a prioritization of the returned results.

In geographic information retrieval, the weighting and ranking mechanisms are based on characteristics of geographic objects such as distance and topology. To estimate the relevance of a geographic object for a given user context, distance and topological relationships need to be converted into a similarity score. The strength of these relationships can be measured and evaluated independently or they can be

aggregated into a single similarity measure [100]. Similarity between geographic objects also needs to include the relative distribution of objects to the user's context. For example, if a user is searching for any restaurants nearby and there are ten restaurants within a city block of the user, no single restaurant would have a strong enough similarity to the user to be significant [55, 92].

The properties of semantic similarity measures in geographic information systems have been explored in [83, 87, 101]. The semantic analysis and querying of geographic objects has been explored in [102-105]. Semantic similarity has been used to measure land cover change [106], in disaster planning and management [107], and in military applications [84].

There are several limitations with current GIR techniques which this research hopes to address: First, most techniques define web-based frameworks (like XML and GML) to define geographic relationships and do not apply a rigorous mathematical framework [108]. They do not have enough constructs to express data semantics [71]. Second, while these types of XML-based querying systems could be used to imply transitive relationships (using OWL and RDL), these transitive geographic relationships could use a more formal mathematical definition. This research aims to use the research on both multilayer networks and semantic link networks to address these two issues.

2.3.3 Spatial Data Mining

Applying traditional data mining techniques to spatial datasets can be a challenge because many spatial phenomenon and patterns are not independent [109, 110]. As mentioned in earlier sections, many aspects of geographic features are dependent on structural properties of the Earth such as plate tectonics and environmental factors.

Cities have historically been built near bodies of water such as rivers and oceans and the ability of rivers to deliver water to those cities is dependent on the topography of that river relative to the mountains that are typically the source of that water.

Spatial autocorrelation refers to the idea that two objects that are close to each other should share similar properties. It is an extension and application of Tobler's Law. Many properties that are plotted on a map, such as population or crime are not randomly distributed throughout a geographic domain. For example, population concentrates around city centers. Spatial autocorrelation is a basis for many spatial data mining techniques such as anomaly detection: If a geographic region does not share the same properties as their neighbors for any given metric, then the spatial autocorrelation is negative, and we can say that this is an anomalous result that deserves investigation.

Spatial data mining techniques derive from both the natural geographic properties of the Earth (of which spatial autocorrelation is one) and existing non-spatial data mining techniques. Most existing clustering algorithms can be used for spatial clustering if two of the fields in the dataset represent the coordinates of the geographic objects. There has been work on creating specialized clustering algorithms such as ST-DBSCAN which separates the criteria for clusters by geographic distance and by time [111].

Spatial heterogeneity refers to the idea that, independent of the phenomenon of spatial autocorrelation, there will be a natural variation in the distribution of objects in a geographic domain. **Co-location** refers to the idea that two different classes of objects can exist within proximity to each other. For example, airports and train

stations tend to be near each other as part of a transportation network. Most research into co-locations typically involve either detecting the existence of heterogeneous objects that exist in the same location [112-114] or detecting anomalous activities that occur in the same location (**cross-outliers**) [115]. Spatial mixture research investigates the diversity and general co-occurrence of different types of objects in the same geographic region [116].

Since many geographic patterns lie within certain geographic restrictions, many techniques have been developed to focus these specialized patterns. For example, clustering techniques have been developed that identify clusters that concentrate near linear features such as roads or rivers [117, 118]. Geospatial extensions to anomaly detection techniques have also been developed [119, 120].

Many geographic features lend themselves to graph-based analysis. This research attempts to extend graph techniques to geospatial applications. Since many GIS databases are very large with many complex features, there have also been attempts to optimize these types of algorithms [121, 122].

2.4 Multilayer Networks

A model of geospatial objects over many layers needs to be based on a data structure that respects the multi-layered nature of geospatial data. Network or graph theory can provide the structure that models these relationships and research into multilayer networks can model the multi-layered nature of geospatial data.

This type of modeling has produced many theories as to the properties and organizational principles of these networks. One theory includes the premise that even

randomly generated graphs have certain observable properties that mimic those in the natural world [123, 124].

Further research into the properties of these networks has revealed that many real-world networks, particularly biological ones, contain network motifs. These motifs represent networks within networks or, more generally, networks of networks [125].

The net result of this research was that networks themselves, if their purpose is to model real-world properties of nature, must become more adaptable to the flexible and dynamic structure of real-world systems [126]. In particular, modeling the properties of a system as one individual network or multiple, yet isolated networks, has proven limited in their ability to model natural systems [127]. Specifically, this practice oversimplifies the natural complexity of these systems which could lead to misleading results [128]. For example, when modeling a transportation network, if every modality of transportation (bus, train, vehicle, etc.) were modeled as a single network, some of its structure would be lost. A bus network can provide an analysis of the time it takes for a passenger to move from one destination to another using a bus, but if a user wants to model a transportation network where the passenger needs to get off of a train and gets on a bus at the same location (the train station), the time required to move from the train to the bus is not modeled if every location is represented by one node in that transportation network.

Therefore, we examine the properties of a multilayer network, or a network of networks where each network is interactive and interdependent with each other. The applications of multilayer networks have been investigated in many research domains

including infrastructure networks [129], social networks [130-132], transportation [133, 134], and biology [135].

A **multilayer network** is a set of networks (or layers) containing objects (depicted as nodes of the networks) connected across multiple dimensions. Example objects are neighborhoods, streets, or parks, and the connections (edges) between them that reflect a similarity (or relevance) between them. In many cases, there exists a transitive relationship between objects. For example, when compared, two properties might not appear to be similar but a common reference to a geographic location (such as a zoning) indirectly links them together. Research into **semantic link networks** (SLNs) explores these transitive relationships.

A **multiplex network** is a subset of multilayer networks that consists of a series of individual networks (layers) and a set of relationships (represented as probabilities) between those layers [136]. A multiplex network imposes a stricter mathematical formalism that will be used in this research. Each **layer** represents a distinct interaction between the nodes on the network [137]. In Figure 11 below, we have a multiplex network with three layers. If M represents the number of layers in a multiplex network, then in this example M = 3. Each layer can be thought of as an individual network (or graph) where a node represents an object (such as a location) and an edge between the nodes represents a relationship between those nodes. In a multilayer network, given N nodes we can represent the complete set of nodes as:

$$\vec{N} = (n_1, \dots, n_N)$$

Given M layers we can represent the complete set of layers in our multiplex as:

$$\vec{M} = (m_1, \dots, m_M)$$

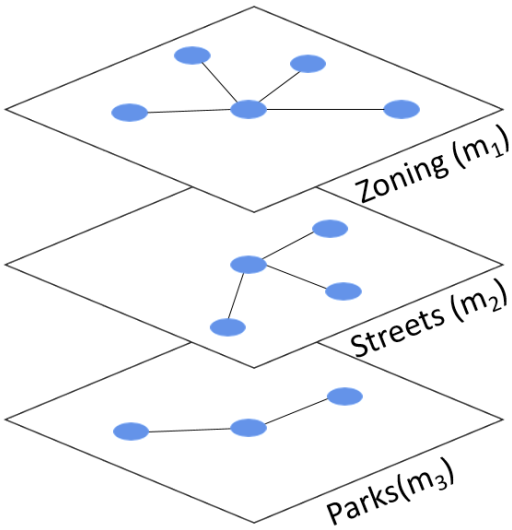


Figure 11: An example of a multilayer network. Each layer m_i is represented as an individual graph.

Representing three different layers $m_1 \dots m_3$ as isolated graphs presents problems when attempting to fully model the connectivity of each node (for example, between zoning and streets). We need to properly model the interconnected nature of these layers. That is, if there exists a relationship between the networks, we need to represent that relationship. Therefore, we look at the **inter-layer correlations** between the layers [133].

Another way of describing a multilayer network is a series of nodes connected to each other with multiple linkages [138]. This definition makes a distinction between a series of interconnected or interdependent networks on one hand [139], and our definition of a multiplex network on the other, which makes the assumption that every node in the network has a counterpart in each layer [128]. By this definition, each node $n \in N$ can be represented as a vector containing elements representing the value in each layer.

$$\vec{n} = [n_1 \dots n_M]$$

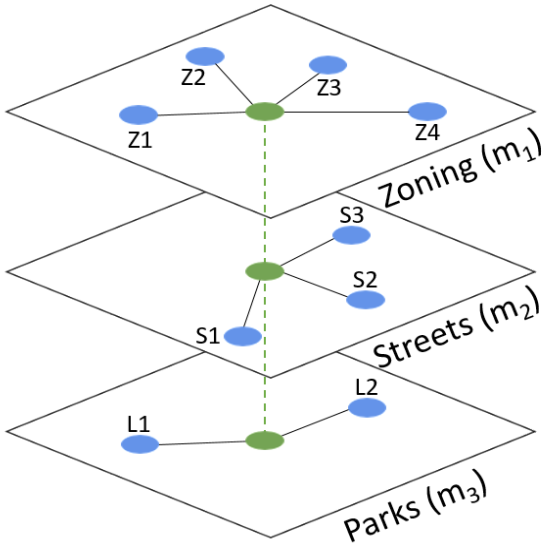


Figure 12: Multilayer network with inter-layer correlations.

In Figure 12 we represent the same multiplex network as before, however, we have highlighted inter-layer correlations as edges (vertical dotted lines) in the network. That is, the step of changing layers as a step in our network is treated with its own dedicated edge and a value (known as a penalty) for traversing from one layer to another [140]. Thinking about the random walk between nodes among the different layers, it is theoretically possible to move from a given node in any layer to any node in a different layer if we use multiple steps. To reduce the complexity of our multiplex construction, a rule where we only allow travel between nodes on the same layer or between layers at the same node is enforced [141].

If we want to travel from one node to another node in a different layer then we must use at least two steps, as shown in Figure 13. One step is necessary to change layers (the dotted vertical edge) and another step is needed to change node (the solid edges on

each layer). We say a minimum of two steps because there might be situations where travel is restricted, and we must navigate the network to reach our destination.

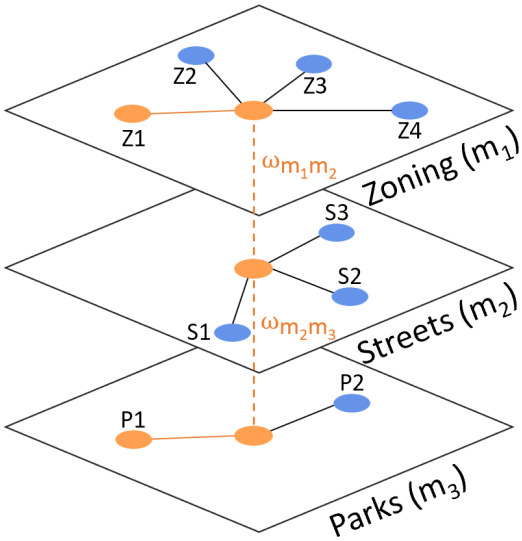


Figure 13: Travel along a multiplex network with restrictions.

In the next sections we investigate the mathematical properties of multiplex networks.

2.4.1 Matrix Representation of Multiplex Networks

Since networks can be represented by matrices [131] this representation can be extended to multiplex networks. In a multiplex network, each layer is a weighted graph where the weights of the edges between nodes represent the strength of the relationship between the two nodes for a given measure. The individual layers have their own adjacency matrices, and this can be supplemented with additional matrices representing the relationships between nodes that belong to different layers. In a simplified case this could be the cost (or penalty) of traversing between layers on the same node (which exists in both layers).

A multiplex network with M layers is a "matrix of matrices" and can be modeled as:

$$\mathcal{A} = \begin{bmatrix} A_1 & \cdots & C_{1M} \\ \vdots & \ddots & \vdots \\ C_{M1} & \cdots & A_M \end{bmatrix}$$

Where $\boldsymbol{A_m}$ is the adjacency matrix for layer m and $\boldsymbol{C_{ab}}$ is the inter-layer correlation matrix between layer a and layer b.

The adjacency matrix $\boldsymbol{A_m}$ represents the relationships between nodes in the individual layer m. Specifically, an individual element in one of these matrices $a_{ij} \in A_m$ represents the value of the relationship (or **relationship score**) between node i and node j at that layer. If we continue with our previous three-layer example in Figure 14 with sample values added, $a_{AD} \in A_{m_1}$ represents the relationship score between nodes A and D at layer m_1 and this value is 0.5.

Filling out our adjacency matrix we calculate:

$$A_{m_1} = \begin{bmatrix} a_{AA} & a_{AB} & a_{AC} & a_{AD} & a_{AE} \\ a_{BA} & a_{BB} & a_{BC} & a_{BD} & a_{BE} \\ a_{CA} & a_{CB} & a_{CC} & a_{CD} & a_{CE} \\ a_{DA} & a_{DB} & a_{DC} & a_{DD} & a_{DE} \\ a_{EA} & a_{EB} & a_{EC} & a_{ED} & a_{EE} \end{bmatrix} = \begin{bmatrix} 0 & 0.3 & 0 & 0.5 & 0.2 \\ 0.3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.9 & 0 \\ 0.5 & 0.9 & 0 & 0 & 0 \\ 0.2 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Note that we declare the relationship score between a node and itself to be $a_{AA} = 0$. If there is no relationship between two nodes, for example between nodes A and C, then the relationship score is defined to be zero. For example, $a_{AC} = 0$.

Remember that each layer has its own adjacency matrix, so in our example we would similarly construct A_{m_2} and A_{m_3}.

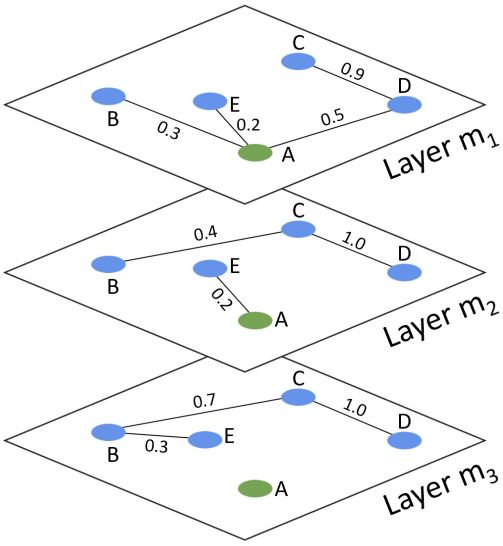


Figure 14: Multiplex example with sample values.

The **inter-layer correlation matrix** represents the strength of the relationship between individual layers. There exists an inter-layer correlation matrix between every layer in the system. Remember that travel between layers is restricted to the same node, therefore this matrix is a scalar matrix where all values are along the diagonal. Given two layers i and j and a distinct and singular relationship score between those two layers $\omega_{i,j}$ the inter-layer correlation matrix can be defined as:

$$C_{m_i,m_j} = \begin{bmatrix} \omega_{i,j} & \cdots & 0 \\ \vdots & \omega_{i,j} & \vdots \\ 0 & \cdots & \omega_{i,j} \end{bmatrix}$$

Each inter-layer correlation matrix consists of N diagonal values $\omega_{i,j}, i \in N$ which represent the relationship score between the same node i between the two layers. If the cost of traversing layers at a node is reflexive then $\omega_{i,j} = \omega_{j,i}$ and $C_{m_i,m_j} = C_{m_j,m_i}$ [128]. All remaining values in C are zeroes since we have established a rule that travel

between layers can only happen at the same node (or location). If there is no relationship between layers i and j, then $\omega_{i,j} = 0$ and all the values in C_{m_i,m_j} are zeroes.

The **supra-adjacency matrix** $\mathcal{A}$ is still an adjacency matrix and structurally this implies that a multiplex network shares similar properties with individual networks [142]. The existence of a value in the matrix represents a relationship between nodes in any layer and in fact treats the same node in multiple layers as separate nodes. For example, if we had N = 5 nodes and M = 3 layers (each with its own 5 x 5 adjacency matrix) the supra-adjacency matrix would be a 15 x 15 matrix. In a sense, this multiplex network is similar in structure to having 15 nodes in one layer.

Based on the example above, if we take the first row of the supra-adjacency matrix, representing travel between node A starting at the first layer, it will look like this:

$$[1 \quad 0.3 \quad 0 \quad 0.5 \quad 0.2 \quad \omega_2 \quad 0 \quad 0 \quad 0 \quad 0 \quad \omega_3 \quad 0 \quad 0 \quad 0 \quad 0]$$

This row has 15 (N x M = 5 x 3) columns. The first five columns are drawn from the adjacency matrix at the first layer. The second five columns represent travel to the second layer (with a ω_2 representing the probability of moving from layer 1 to moving to layer 2) and the last five columns represent travel to the third layer (with a ω_3 representing the probability of moving from layer 2 to layer 3). The zeroes represent the restriction we have placed on the multiplex where we can only move to another layer at the same node. This is why both ω values are in the first column of the group of five.

The construction of the supra-adjacency matrix contrasts with the construction of the aggregate adjacency matrix which is constructed by using an aggregation function on the edge weights between nodes.

2.4.2 Multiplex Network Properties

Newman et al. explored the relative connectivity of an individual node in the network since the connectivity of a node might hold a clue to the relative importance of that node [143]. When looking at a graph, how does one prioritize or value certain nodes, edges, or layers over others [144]? For example, examining a network focusing on an isolated node might not provide as much insight as a node with connections to many other nodes. In this section we examine different measures of the connectedness of individual nodes in a network.

The **communicability** (or correlation) of a node represents the degree to which changing (or removing) that node in the graph can affect other nodes in the rest of the graph [145, 146].

Centrality is the term used in network theory to measure the relative importance of a node, edge, or a subset of the network relative to the entire network [128, 130]. For example, a large capital project might be more relevant to other geographic objects and activities in the area than smaller capital projects. Centrality in this example reflects the relative impact of that capital project compared to others.

One measure of centrality is degree. The **degree** of node $n \in N$ is the number of nodes connected to that node, or:

$$k_n = \sum i \in N \; where \; A_{in} > 0$$

In a multiplex network one can calculate multiple degrees, one for each layer. The degree of node n at layer m is:

$$k_n^m = \sum i \in N \; where \; \mathcal{A}_{m,in} > 0$$

Note that there might be situations where a node's degree at one layer could be different than its degree at another layer and sometimes the degree is zero at certain layers (it is an isolated node at that layer) [147]. In multiplex networks the connectivity of individual nodes is especially important since nodes might be connected in one layer but not another. A **walk** represents a path from one node to another in a network and all walks in a network represent every path from every node to another. Specifically, a walk is a series of nodes and edges along the graph. With a standard network with adjacency matrix A, the walks of k length can be represented by the kth power of A, or A^k. Degree centrality is equivalent to walks of length one (where k = 1).

This can be translated to the multiplex by applying the same kth power to the supra-adjacency matrix: $\mathcal{A}^k$. The set of all walks of all lengths from any node to any node in our supra-adjacency matrix can be represented as a vector of matrices:

$$W = [\mathcal{A}, \mathcal{A}^2, \dots, \mathcal{A}^{MxN}]$$

The set of all walks on our supra-adjacency matrix is limited to MxN elements since our supra-adjacency graph essentially contains MxN nodes (M representations of N nodes, one for each layer).

We additionally distinguish between intra-layer walks (those that only visit nodes and edges belonging to only one layer in the multiplex and inter-layer walks (walks that visit nodes and edges in more than one layer of the multiplex).

The **reachability** of a node is the average distance from that node to every other node in the graph. The **multiplex reachability** of a node i is the average distance L_i from i to any other node of the multiplex [147]. We can use this value as a benchmark to evaluate the connectivity of a node at each layer.

The **shortest path** between two nodes in a multiplex network is a minimum path that starts from the source node in any layer and reaches the destination node in any layer. The **interdependence** of a node i is

$$\lambda_i = \sum_{\substack{j \in N \\ j \neq i}} \frac{\psi_{ij}}{\sigma_{ij}}$$

Where ψ_{ij} is the number of shortest paths between node i and node j which uses edges lying on more than one layer and σ_{ij} is the total number of shortest paths between i and j in the entire multiplex. Interdependence is a measure of the value gained by using a multiplex construction [148].

When comparing two layers we can look at the **interlayer degree correlation** between any two layers in a network. This measures the degree of a node at each layer and compares it to the degree of that node at every other layer. If nodes that have high degree in one layer correspond to the nodes that have high degree in another layer than we say that the new layers are highly correlated. On the other hand, if a node has high degree in one layer but has very few connections in a second layer than those two layers are not highly correlated [129]. In our travel example, we could say that the road layer and airport layer are correlated because airports that serve as air hubs typically serve as road hubs. On the other hand, an air network and a rail network might have low correlation since train stations and airports tend to not be near each other.

2.4.3 Layer Aggregation and Multiplex Entropy

The previous section discussed network properties such as interdependence that compare the paths made available by modeling values in a multiplex construction. By default, our behavior when determining what layers we would want to use in our

multiplex network would be to take every attribute of our graph and make a layer out of them. However, as shown in the methodology section, doing this would create a great deal of additional overhead and computational time to our analysis. As shown in Table 2, every added layer would add 2N - 1 sub-matrices in our supra-adjacency matrix.

Table 2: Each layer added to the multiplex in our example increases the number of sub-matrices by 2N-1

# Layers	# Adjacency Matrices	# Inter-Layer Correlation Matrices	Total
2	2	2	4
3	3	6	9
4	4	12	16
5	5	20	25

Therefore, we need to know when it is appropriate to remove or aggregate layers and/or fields in those layers without removing a significant amount of information [135].

One measure of the information gained from adding additional layers (or information lost from aggregating some of those layers) is entropy. **Information Entropy** is the amount of information about a natural system that a data model does not provide [137]. Naturally, it makes sense that if we model our system with a multilayer network with values from all layers, we will have lower entropy than if we aggregated those attributes into a single value, where the nuanced properties of individual locations could be lost. When balancing the size of a multilayer network

with performance considerations, entropy can be used to help identify layers for our multiplex in the methodology section.

2.4.4 Modeling Multiplex Networks as Tensors

Our primary model of multiplex networks involves creating a supra-adjacency matrix which represents the multiplex as one supra-adjacency matrix. An alternative method is to use a tensor to model a multiplex network [137].

Given a tensor-based multiplex network it is possible to "flatten" it into a supra-adjacency matrix. We do this by slicing off each dimension (a layer adjacency matrix) and placing it in an appropriate position in the supra-adjacency matrix. This results in a bijective mapping between the two layers in a process known as flattening, unfolding or matricization [1]. What is important is that both the tensor representation and the matrix representation store the same information, just in different formats.

This dissertation uses the supra-adjacency matrix representation of a multiplex network rather than the tensor representation for a couple of reasons: First, given our interest in merging the multiplex theory with that of the semantic link network (discussed in Section 2.5) a matrix works better since the SLN involves matrix-based calculations [141, 149]. Second, representing the multiplex network as a matrix is a more intuitive model and matrices are the least removed from the database origins of the data we will be using in this research [150, 151]. Third, this type of representation lends itself to software implementations of our algorithms [150].

2.4.5 Challenges with Multiplex Networks

The study of multiplex networks is relatively new and because of this there are no standard notations or mathematical structures and no consensus on how to mathematically model multiplex structures. This dissertation makes decisions as to which mathematical notion and structure for multiplex networks we will use, most notably the decision to work with the supra-adjacency matrix model as opposed to the tensor model.

Additionally, the literature on multiplex networks focuses on the mathematical foundations of the theory. While there have been attempts to apply multiplex network theory to practical applications such literature is limited.

2.5 Semantic Link Networks

This section defines semantics and uses that definition to describe the semantic link network as a data structure to help describe the relationships between objects. We have previously mentioned that part of the context of an object is the relationship between that object and other objects.

2.5.1 Semantics

Semantics is the meaning of objects (such as property boundaries or police stations) and the relationships between them.

The idea of semantics as a computational concept has been used significantly within the context of the semantic web. The World Wide Web (WWW) can be described a series of distributed resources (web pages, multimedia) linked together via hyperlinks. This is what is colloquially known as Web 1.0. From the World Wide Web came the

development of web services which took the hyperlinked nature of the Internet and applied it to data. The protocols and languages associated with web services (Web 2.0) include RDF, OWL and SPARQL. These languages are designed to be machine readable and to be interpreted by applications to provide meaning to the data contained inside. OWL, for example, models the ontologies discussed in Section 2.2.4 [152].

2.5.2 Semantic Link Networks

A **Semantic Link Network (SLN)** is defined as "a self-organized semantic data model for semantically organizing resources, which can be abstract concepts or specific entities such as texts, images, videos, and audios" [3]. This model is built around a matrix-based mathematical foundation, and this is the one reason we chose to use a supra-adjacency matrix to model a multiplex network rather than the tensor representation.

An SLN has the following properties:

- It reflects various semantic relations between classes and between entities.
- It is a semantics-rich self-organized network. Any object can be related to any other object (although in many cases two objects might not be connected).
- It can derive implied semantic links based on a set of reasoning rules. That is, given a series of existing links we can infer new links based on a pre-defined series of rules.
- The semantics on the network keep evolving with various operations on the network. That is, when new information is added to our network the SLN should adapt and absorb the new information and modify linkages or create new linkages from this information.

A **reasoning rule** is a set of operations (usually mathematical) that can be applied to the links in an SLN to imply new linkages that might not have existed otherwise. There have been attempts to create a set of generic reasoning rules [153, 154]. A reasoning rule might say that if nodes A and B have a relationship and nodes B and C have a relationship then nodes A and C have an implied relationship even if that relationship wasn't originally established. If two connected semantic links exist a third semantic link could be derived from those two links with the proper reasoning rule.

SLNs can be associated with ontologies. Given an ontology that describes a taxonomy of concepts or classes and given data that fits into those classes we can establish relationships between the individual instances of those classes [155].

SLNs have been used in network applications, knowledge management and to enhance search [156].

2.5.3 Mathematical Definition of SLN

Mathematically, an SLN is a directed network consisting of semantic nodes and semantic links. A **semantic node** is a concept (like a road or a park). A **semantic link** reflects a kind of relational knowledge represented as a pointer with a tag describing semantic relations. These semantic links usually reference prepositional logic. Examples of semantic links include "is inside of" and "is similar to." A **semantic relation** resembles the concepts in ontology and can include relationships such as similarity between the two semantic nodes [3, 157].

An **SLN schema** is a triple of $<Nodes, Semantic\ Links, Rules>$, where

- A **node** is an object type (address, park) denoted by $n_i \in N$ and its features are represented using a vector $\vec{V}_{n_i} = [f_1 \dots f_n]$ where f_i is a feature of node n_i.

- A **link** is a node-to-node relation $l_{a,b} \in NxN$. For example, addresses are connected by a distance and zoning polygons are connected by an adjacency measure between those zones. These weights act as a probability that two nodes are related to each other. The set of links $l_{a,b}$ can be combined into an N x N matrix. This matrix is known as the **Semantic Relationship Matrix (SRM)**. The SRM acts as a type of adjacency matrix where the values specifically reflect the values of the semantic links between two nodes.
- A **rule** is a reasoning mechanism on semantic links. It is a mechanism by which we can define implicit relationships between nodes. Denoted by: $n_i \overset{\alpha}{\rightarrow} n_j, n_j \overset{\beta}{\rightarrow} n_k \Longrightarrow n_i \overset{\gamma}{\rightarrow} n_k$ where α, β and γ are weights on semantic links and $\alpha \cdot \beta = \gamma$.

This rule is a mathematical formula to derive a new link from two existing links and to generate a weight on the new link by multiplying the weights of the two existing links. For example, in Figure 15, if α = 0.5 and β = 0.7 then we say that $\gamma = \alpha \cdot \beta$ = 0.5 · 0.7 = 0.35. Since these are probabilities, multiplying the two values will always produce a value lower than the two original probabilities (unless both probabilities are 0 or 1). We define the transitive relationships between nodes that are not apparent from the original matrix by this mechanism.

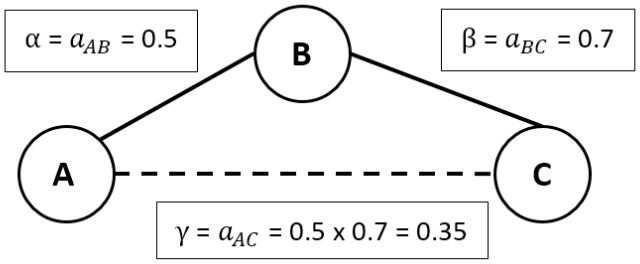


Figure 15: An SLN reasoning action

Mathematically and intuitively the strength of these implicit relationships ($n_i \xrightarrow{\gamma} n_k$) should be weaker than the direct relationships ($n_i \xrightarrow{\alpha} n_j, n_j \xrightarrow{\beta} n_k$) that define them. However, in some situations those implicit relationships are stronger than the direct relationship defined in the original matrix and should be the preferred path between two nodes.

We will take the multiplex construction and apply the rules defined by the SLN schema to try to define implicit connections between nodes within a layer and across layers.

Chapter 3: Theoretical Approach

Querying databases for relevant geographic objects has limitations that were outlined in the earlier chapters. This chapter summarizes the problems that lead to these limitations, mathematically defines a solution to these problems, and describes a framework that models their solution.

3.1 Overview of the Problem

Geographic objects are features that exist in the observable world. Sometimes these are physical objects (such as trees or buildings) or abstract concepts (such as political boundaries or districts). Historically, these geographic objects have been modeled as paper maps with symbols on the map representing these objects. For example, on many maps, dots might represent important landmarks. Digital maps extended this concept by representing the paper maps (and their underlying geographic objects) on a digital screen that can be manipulated (zoom in/out and panned) by a user.

Figure 16: A map showing multiple geographic objects in a park. Blue represents a water body and the seven labeled objects (A through G) represent pavilions within the park.

This dissertation focuses on the digital representation and storage of the geographic objects within a domain of interest and the relationships between these objects with a goal to develop a mathematical model of geographic relevance between these objects. For example, calculating the parks and amenities that are relevant to a resident in a community, or the political boundaries relevant to a proposed major construction project.

3.2 Mathematical Foundations

This section mathematically describes the set of geographic objects in a domain of interest and the relationships between those objects. Specifically, we mathematically define geographic objects in a domain and the grouping of those objects into layers. We then define the potential topological relationships between these geographic objects as well as various distance measures for these objects. Finally, we define potential relationships between layers and derive potential semantic relationships between geographic objects.

3.2.1 Geographic Objects

We define D to be the **domain of interest** for a given endeavor, typically defined as a geographic boundary. For example, all objects in the domain of a major city include all geographic objects (physical and abstract) within that city's borders. A domain could also be defined as an individual park's boundary or defined as the whole Earth.

Definition: A **domain** consists of the data and information relevant to an area of study. Given the set of all geographic objects on planet Earth $\mathcal{G}$ (the largest domain

relevant to geographic inspection) and a domain of interest D, we define $\mathcal{G}_D$ to be a subset of those objects relevant to our domain.

$$\mathcal{G}_D = \{g_1, g_2, \ldots, g_n \mid g_i \in \mathcal{G}\}$$

We can define **subdomains** within $\mathcal{G}_D$ at will to define subsets of objects within our primary domain. This can include grouping objects by similar object types (such as a group of all park benches) or by different geographic **envelopes** (such as all geographic objects in one park or one district).

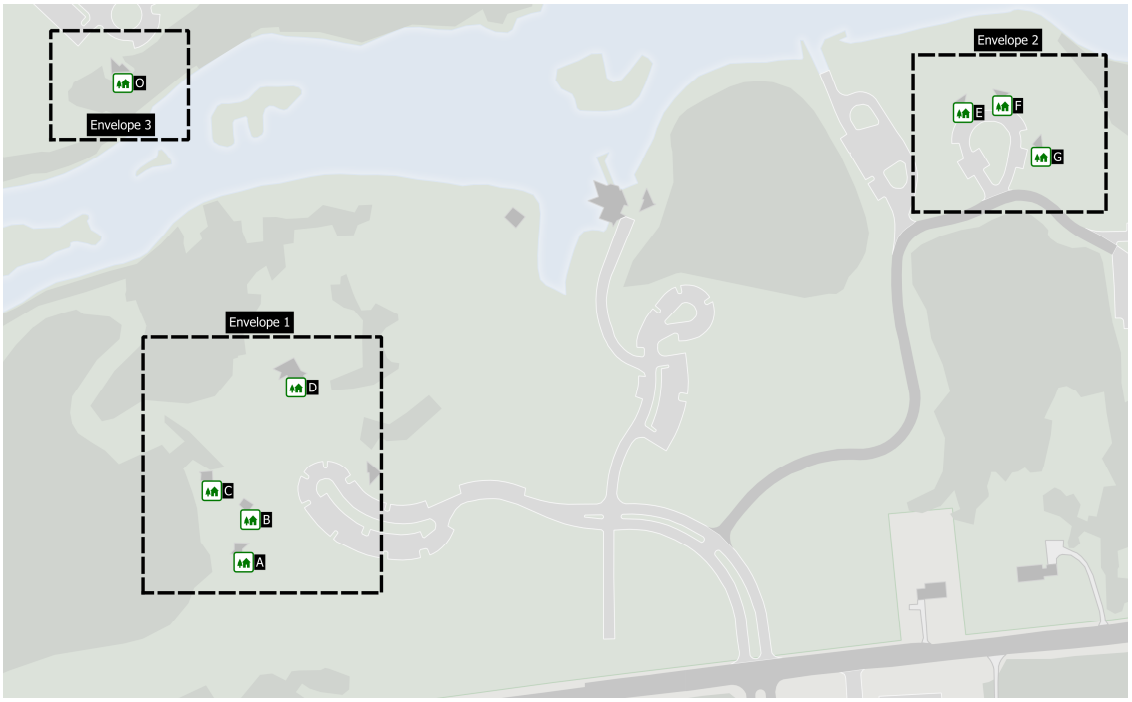


Figure 17: In this park example, we define three envelopes of interest. Each envelope contains one or more geographic objects (pavilions).

Definition: *An **envelope** is a geographically defined area of interest. We define $\mathcal{G}_E$ as the subdomain of all objects within a specified envelope.*

$$\mathcal{G}_E = \{g_1, g_2, \ldots, g_n \mid g_i \in \mathcal{G}_D\}$$

Geographic information systems typically store similar types of objects (such as parks) into database tables or files known as layers. For example, one layer could represent all park pavilions in an area and another layer could contain all lakes.

Definition: *A **layer** is a grouping of similar types of objects in a geographic area. For any given layer of interest L, we define $\mathcal{G}_L$ as the subdomain of all geographic objects within that layer.*

$$\mathcal{G}_L = \{g_1, g_2, \dots, g_n \in \mathcal{G}_D\}$$

In a domain, the grouping of geographic objects into layers is independent of grouping layers into an envelope. For example, the set of all amenities of all types located inside one park (the envelope) is independent of the set of all park benches in a domain (the park bench layer). That is, the set of potential envelopes in a domain and the set of potential layers in a domain are orthogonal.

Proposition: *The combination of the objects in all layers results in the set of all objects in a domain*.

$$\sum \mathcal{G}_L = \mathcal{G}_D$$

An individual geographic object represents a physical object in the environment or an abstract object such as a political boundary. Geographic objects can be spatially represented as a series of component objects and the specific structure of a geographic object depends on the nature of that object. For example, park boundaries can be represented as polygons and park benches within that park can be defined by point features. Database systems that support the storage of spatial objects such as points or polygons standardize the formats for distinct types of objects (points, lines, polygons) and expose functions on those objects.

Geographic objects can be represented as point objects if they can be represented as a set of coordinates. These coordinates can be represented in a spatial database as either a longitude/latitude or as a series of planar coordinates (x and y) that conform to a standardized projection in the European Petroleum Survey Group (EPSG) set of map projections and can generally be treated as Cartesian coordinates for calculations.

Definition: *A **point** is a type of geographic object g_p that can be defined mathematically as a series of coordinates x and y. These coordinates can represent planar coordinates or longitude/latitude:*

$$g_p = (x, y) \in \mathbb{R}^2$$

Figure 18: A line (curve) is a sequence of vectors (line segments) connecting points.

Curves are typically found in a map. In general, such curves are called lines. A **line** in a spatial database is composed of a sequence of vectors (line segments). The vectors connect when the endpoint of one vector shares coordinates with the beginning of the next vector in the sequence.

Definition: *A **line** is a type of geographic object g_l that can be defined mathematically as a series of points g_i, vectors (line segments) connecting those points in a defined sequence $\overrightarrow{g_i g_{i+1}}$, and the summation of those vectors into a series.*

$$g_l = \sum \overrightarrow{g_i g_{i+1}} \in \mathbb{R}^2$$

A polygon is an enclosed series of line segments, where the end of the final line segment in the sequence shares coordinates with the beginning point of the first line segment. A polygon's constituent line segments bifurcate the Earth into two partitions and the "inside" is intuitively defined to be the non-infinite partition created by the line segments. Spatial databases sometimes require line segments to be digitized in a pre-defined direction (clockwise or counterclockwise) to better define the inside and outside region. Some databases automatically detect the inside of a polygon.

Figure 19: A polygon is a series of line segments in sequence (orange arrows) connecting vertices (red dots).

Definition: *A **polygon** is a type of geographic object g_p consisting of a sequence of line segments where the final point in the sequence connects to the initial point in the sequence and is compact (all points lie within some fixed distance of each other).*

We have defined geographic objects and defined sets of geographic objects as layers or envelopes that are subsets of all geographic objects in an organization's domain of interest.

The geographic objects in a domain of interest do not exist independently. At a minimum, these objects exist in proximity to one another. For example, two parks that are close to each other might have a stronger relationship between those far apart. In other situations, objects can exist within other objects. For example, a park bench (represented as a point) whose coordinates are within the polygon boundary of its park implies a relationship between the park bench and the park itself.

We assign a numerical value to describe the strength of relationship between two geographic objects.

Definition: *A **relevance score** is a probability that represents the strength of relationship between two geographic objects or how relevant one geographic object is to another. A relevance score of 0 represents no relevance between two objects and a relevance score of 1 represents total relevance between two objects.*

Relevance scores can be calculated by describing the topological relationships and/or distance relationships between geographic objects and the layers that contain these objects. In this work, the relevance score (either between objects or between layers) is represented by a probability which is a Bayesian probability. The higher the relevance score between two objects, the higher the probability that a user interested in

the first object will also be interested in the second object. In later sections, the final model will represent a Markov chain representing these probabilities for all geographic objects in a context profile.

In the following sections, we describe the mathematical techniques used to calculate the relevance score between any two geographic objects.

3.2.2 Topological Relationships Between Geospatial Objects

In spatial databases, the existence of a relationship between two geographic objects is also known as a **topological relationship**. Examples of topological relationships are shown in Table 3. These topological relationships can be used to define a formula that calculates the strength of a relationship or a similarity between two geographic objects. The formula is chosen based on the types of objects that we need to relate with each other [158].

Table 3: Different topological relationships and their similarity calculations

Type of Relationship	Similarity Definition
Distance	Linear or non-linear distance between two objects
Intersection (Points/Lines)	Existence of intersection or not
Overlap	Percent of overlap compared to total length or area
Adjacency	Existence of adjacency or not

We expand on the definitions in Table 3 with more detail. We define the similarity between two geographic objects as a probability between 0 and 1, where 0 means no similarity and 1 means the two objects have the most similarity. The calculations

detailed below take an intuitive definition of these concepts and applies a mathematical structure to them.

Distance

This is the default similarity calculation used in our framework and is our default calculation for any geographic objects. While distances between two geographic points depend on both the curvature of the Earth and the altitude of the two points of interest, we use the Euclidean distance between two objects.

Definition: *For any two geographic objects g_i and g_j in our domain, each defined as a set of coordinates x and y, we define the **distance** between those two objects.*

$$d(g_i, g_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

Any geographic object's distance to another can be calculated for any type (point, line, polygon). For example, a park can be represented by its boundary (a polygon) and the distance from that polygon to a point of interest can be calculated in multiple ways: Distance from the point to the polygon's centroid or the distance from the point to the nearest edge on the polygon. Distance measures will be explored in more detail in the next section.

Intersection (Points or Lines)

The intersection of a line or a point with a polygon (for example, an address within a specified Congressional district) or the intersection of two lines (for example, two streets) is binary. For example, in Figure 20 we display the intersection of two streets in Ellicott City, Maryland, Court Avenue and Emory Street.

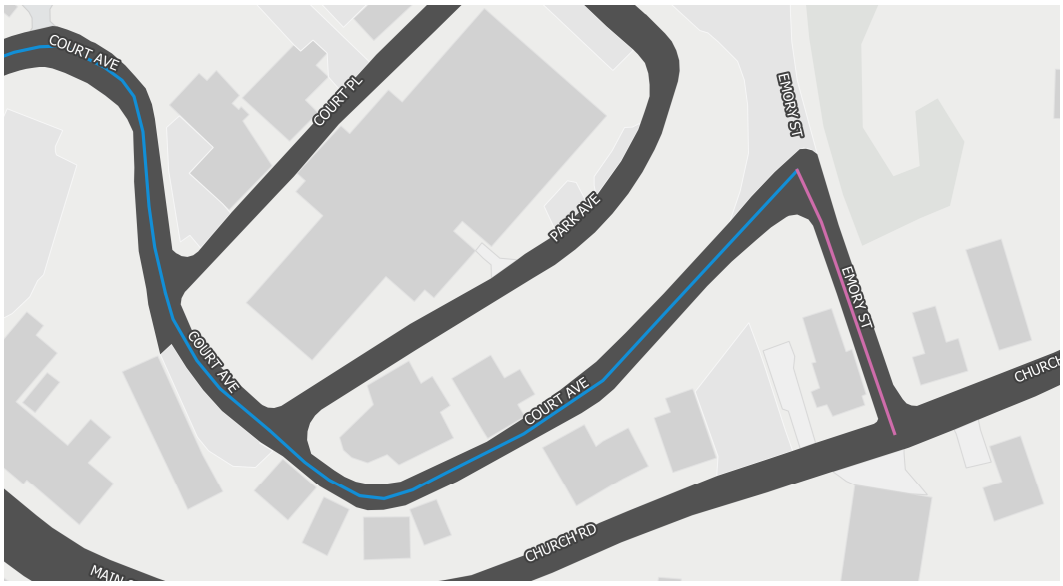


Figure 20: An intersection between Court Avenue (blue) and Emory Street (purple).

Definition: *For two geographic objects that intersect each other, we define the* ***intersection-based relevance score*** $\alpha_{int} \in [0,1]$ *between two geographic objects* g_i *and* g_j *as:*

$$\alpha_{int}(g_i, g_j) = \begin{cases} 1, & g_i \cap g_j \\ 0, & otherwise \end{cases}$$

Overlap

When a line overlaps with a polygon, or when two polygons overlap, the similarity between the two objects depends on their overlap. If the two objects have identical geometries (they are on top of each other) then the relevance score should be 1 and if they have no overlap the relevance score should be 0.

Definition: ***For two geographic objects that overlap,*** *we define the* ***overlap-based relevance score*** $\alpha_{ovr} \in [0,1]$ *between two lines or polygons* g_i *and* g_j *as the intersection of the two objects (calculated as length or area) divided by the length or area of the first object:*

$$\alpha_{ovr}(g_i, g_j) = \frac{g_i \cap g_j}{g_i}$$

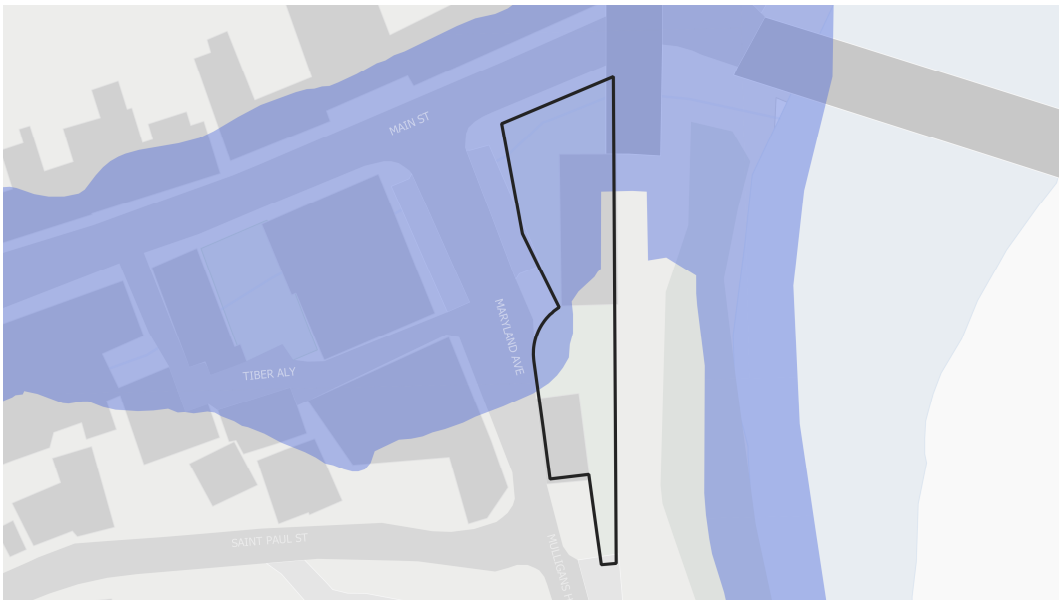


Figure 21: The B&O Museum Property (outlined in black) and the floodplain polygon (in blue).

For example, in Figure 21 we highlight the property of the B&O Railroad Museum (in black) and we overlay the county floodplain (in blue). The total area of the highlighted property is 13,327 square feet and the area of the floodplain intersecting that polygon (the blue area inside the white polygon) is 7,538 square feet. Therefore, we calculate the overlap as:

$$\alpha_{ovr}(g_i, g_j) = \frac{g_i \cap g_j}{g_i} = \frac{7538}{13327} = 0.56$$

Adjacency

Two polygons are adjacent when a part of their boundaries intersect each other. When two objects are adjacent to each other, we treat it as a binary relationship. Either two polygons are adjacent to each other, or they are not.

Definition: *For two geographic objects that are adjacent to each other, we define the* ***adjacency-based relevance score*** $\alpha_{adj} \in [0,1]$ *between the two polygons* g_i *and* g_j *as:*

$$\alpha_{adj}(g_i, g_j) = \begin{cases} 1, & edges\ overlap \\ 0, & otherwise \end{cases}$$

The next section explores the calculation of similarity between objects based on distance.

3.2.3 Distance Relationships between Geospatial Objects

In most situations, the closer a geographic object is to another, the stronger the relationship between them and the higher the relevance score between the two objects. For example, given a series of parks, two parks closer to each other are considered more similar than parks farther apart. This distance is generally considered to be a measure representing the length of the shortest line segment that can be drawn between two points. If we represent our domain as a two-dimensional Cartesian plane, then this distance is a straight line between the two points.

Figure 22: Calculating the distance between A and B: we are only interested in the shortest line segment connecting two objects (pink line) and not others (grey lines).

Linear Distance Similarity

We used a simplified linear distance between objects in the same layer to determine their relevancy to the user's location of interest. However, any GIS layer might need a different metric based on distance (or some other measure such as adjacency) and we plan to explore other metrics.

Definition: *The **linear distance similarity** α_{dist} between two objects in a domain is the complement of the proportion of the distance between two objects g_i and g_j compared to the maximum distance possible in the domain (a city or other geographic area of interest) d_{max}.*

$$\alpha_{dist}(g_i, g_j) = 1 - \frac{d(g_i, g_j)}{d_{max}}$$

With this formula, if two objects are geographically in the same location the similarity value is one and if two objects are the maximum possible distance from each

other (within the domain of interest) then their distance value is zero. For example, if the maximum distance between any two points in Baltimore City is 15 miles, and the distance between two objects is 5 miles, then we calculate the similarity

$$\alpha(g_i, g_j) = 1 - \frac{5}{15} = 0.67$$

Normalized Distance Similarity

While this calculation fits into our framework it does not consider the contextual definitions of distances between layers [104]. For example, if a user queries nearby addresses versus nearby parks, using the formula above could potentially provide thousands of nearby addresses for every park.

In Figure 23 we have identified the nearest addresses and parks to the "Mall in Columbia," a central landmark in Howard County. In this example, the average distance to the nearest 18 addresses is approximately 1,000 feet whereas the average distance to the nearest 5 parks is about 8,000 feet.

Figure 23: With "The Mall in Columbia" as the location of interest (red), the nearest addresses are shown in purple and the nearest parks in green.

An alternative way to limit the number of addresses that are returned in the situation above would be to replace the maximum possible distance with the typical distance between objects. By looking at the distribution of distances between any two objects in a GIS layer we can use the mean and standard deviation of that distribution to calculate distance. We assume that any distance more than three standard deviations past the mean distance for the population of objects in that layer is not similar enough to be returned in an information retrieval query.

When we examine the population of geographic objects in a layer, this formula needs to be modified to include what a user might reasonably consider "nearby." This means that while d_μ and d_σ represent the distribution of distances for all points in the layer, we are only interested in objects within a threshold distance. For example, we are only interested in the closest addresses to other addresses when evaluating the distribution of distances based on the distribution of distances between all objects in the relevant layer or domain. Using a threshold for nearby objects results in $d_{\mu*}$ and $d_{\sigma*}$ which are the values used in our similarity calculation.

Definition: *If $d(g_i, g_j)$ is the distance between any two objects in a layer, $d_{\mu*}$ is the mean distance between objects in that layer, and $d_{\sigma*}$ is the standard deviation for object distances in that layer, we define the* ***normalized distance similarity*** *α as:*

$$\alpha(g_i, g_j) = \begin{cases} 1 - \dfrac{d(g_i, g_j)}{d_{\mu*} + 3d_{\sigma*}}, & d(g_i, g_j) < 3d_{\sigma*} + d_{\mu*} \\ 0, & d(g_i, g_j) \geq 3d_{\sigma*} + d_{\mu*} \end{cases}$$

The next step is to mathematically define the relationship between two layers.

3.2.4 Relationships Between Geographic Layers

We group geographic objects of the same type into layers. For example, we define all parks in a domain as a layer and all ponds in the domain as a separate layer. These layers can have relationships defined between them. In our earlier example, park benches that exist within a park are said to belong to that park. This relationship between objects in two different layers can already be defined as a topological relationship, as described in Section 3.2.2. However, just because two layers can have a distance or topological relationship defined between the objects contained in those layers does not imply that those layers are relevant to each other. For example, when a user is at a park, they are likely more interested in nearby park benches than the nearby pond in the park since the park bench is an amenity used in the park, whereas the pond is a physical artifact that is a consequence of the park's location.

From Section 3.2.1, we defined a layer L as a grouping of objects of a similar type and is a subset of all objects in a domain. For example, one layer can contain all park benches and a different layer can contain all ponds in a domain of interest.

$$L \subseteq D$$

$\mathcal{G}_L$ is the set of geographic objects within that layer.

$$\mathcal{G}_L = \{g_1, g_2, \dots, g_n \in \mathcal{G}_D\}$$

Definition: *The **layer relevance score** (or layer similarity) λ between two layers L_p and L_q is a probability defining the strength of the relationship between those two layers.*

$$\lambda(L_p, L_q) \in [0,1]$$

A value of 0 implies no relationship between the two layers and a value of 1 implies the strongest possible relationship between two layers.

The relevance between any two geographic objects can be defined as the product of the relevance score between the layers containing those objects and the distance or topology between the two individual geographic objects independent of the layers that they are contained in. In the case when the two objects are in the same layer, the value of λ is equal to 1.

Definition: *If g_i is a geographic object belonging to L_p and g_j is a geographic object belonging to L_q then the* ***adjusted relevance score*** *α' between the two objects is the product of the layer relevance λ and the original relevance score α:*

$$\alpha'(g_i, g_j) = \alpha(g_i, g_j) * \lambda(L_p, L_q)$$

Since λ and α are both probability values, the product of both produces another probability value $\alpha' \in [0,1]$. Having defined the relationships between individual geographic objects and geographic layers, the next step is to use the principles behind multiplex networks to create a mathematical model of all geographic objects in a domain and the relationships between them.

3.2.5 Context Profiles

Since different users have diverse ideas as to which layers are relevant to others, we introduce the idea of a context profile, which is a grouping of layers that are relevant to a user's context.

Definition: *A **context profile** is a set of layers in the domain of interest and a set of mathematically defined layer relationships for those layers.*

$$\rho = \{L,\ \lambda(L_p, L_q) \mid \mathcal{G}_L \in \mathcal{G}, L_p, L_q \in L\}$$

The same layer can be involved in multiple context profiles. Context profiles will be used to focus the processing of the framework described in Chapter 4 and evaluated in Chapter 5.

3.3 Creating a Multilayer Network

A multilayer network contains relationships between layers (as described in Section 2.4), and relationships among objects in the same layer.

A Multilayer Semantic Link Network (mSLN) can be defined as a graph representation of the relationships between nodes among several different layers. Specifically, the mSLN models two types of relationships: relationships between layers and relationships between objects in the same layer. An mSLN is represented as a matrix of matrices with two types of components: Each individual layer of the mSLN is a graph represented as an adjacency matrix which contains the relationships between nodes in the same layer. Different layers are connected, and these connections are represented as inter-layer correlation matrices, which identify the relationships that exist in our ontology. In this section we detail the mathematical foundation and steps necessary to create a multilayer SLN.

Definition: *A **multiplex network** with M layers is a supra-adjacency matrix where* $\boldsymbol{A_m}$ *is the adjacency matrix for layer m and* $\boldsymbol{C_{ab}}$ *is the inter-layer correlation matrix between layers a and b:*

$$\mathcal{A} = \begin{bmatrix} A_1 & \cdots & C_{1M} \\ \vdots & \ddots & \vdots \\ C_{M1} & \cdots & A_M \end{bmatrix}$$

The following sections summarize the construction of both the adjacency matrices and the inter-layer correlation matrices.

3.3.1 Modeling all Objects in One Layer (Adjacency Matrices)

After defining the relationships between individual objects (Section 3.2.2 and Section 3.2.3) and the relationships between individual layers (Section 3.2.4), a mathematical structure based on the principles of multiplex networks can help define the relationships of objects in our domain. This formulation models the relationships between all objects in a layer and the relationships between all layers in a domain to create a model of the relationships between all objects in a domain.

Mathematically, the goal is to combine two types of matrices, layer adjacency matrices (modeling the relationships between objects in a layer) and inter-layer correlation matrices (modeling the relationships between layers themselves) to create a combined matrix known as a supra-adjacency matrix.

Starting with a layer L with n objects in it as defined in Section 3.2.1:

$$\mathcal{G}_L = \{g_1, g_2, \ldots, g_n \in \mathcal{G}_D\}$$

We can define each layer as a directed graph G with n nodes, with one node for each geographic object (or table row) in the original dataset. This graph can be represented as an adjacency matrix.

Definition: *Given a layer $\mathcal{G}_L$ with n objects, we define the* ***layer adjacency matrix*** $A \in \mathbb{R}^{nxn}$ *to represent the relationships between every object in $\mathcal{G}_L$ [131]. We use the similarity values α to define the values in this matrix.*

$$A_L = \begin{bmatrix} \alpha_{11} & \cdots & \alpha_{1n} \\ \vdots & \ddots & \vdots \\ \alpha_{n1} & \cdots & \alpha_{nn} \end{bmatrix}$$

In this example, $\alpha_{ij} = \alpha(g_i, g_j)$, the individual relevance score value between geographic objects g_i and g_j. Each matrix cell $a_{ij} \in A$ represents the strength of the relationship, or **similarity**, between the two nodes for a specified attribute. The relevance score value is between 0 and 1 and represents the probability that j is reachable from i. If the relevance score value is 1 then the nodes are identical for that attribute and if the relevance score value is 0 then there is no relationship between the nodes [159].

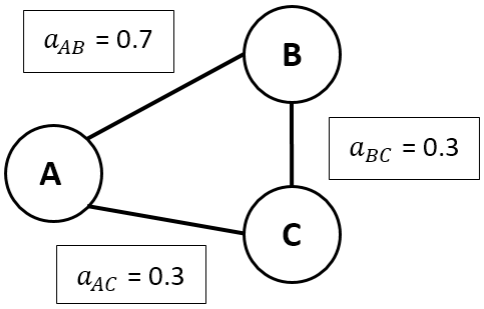


Figure 24: Three nodes and the similarity scores between them.

In Figure 24 we have three nodes: A, B and C and three sample relevance scores behind them. In the equation below we have these values translated into an adjacency matrix.

$$A = \begin{bmatrix} 0 & 0.7 & 0.3 \\ 0.7 & 0 & 0.3 \\ 0.3 & 0.3 & 0 \end{bmatrix}$$

The diagonal values $a_{AA} = a_{BB} = a_{CC} = 0$ because we are not interested in the relationship between a node and itself.

3.3.2 Modeling Layer Relevance (Inter-Layer Correlation Matrices)

To model the relationships between layers, we define another type of adjacency matrix called the **inter-layer correlation matrix**, which represents the strength of relationship when moving between layers when traversing the graph.

For this calculation, we include every geographic object in the network irrespective of the layer that contains that object since we can calculate the similarity α' between any two objects in any layer. N represents the total number of objects in our domain.

$$N = \text{count(g)} | \text{ g} \in \mathcal{G}_D$$

We also assume that we can only traverse the network between layers at the same node. This is equivalent of saying that the calculation of α and λ to produce α' are independent calculations. the inter-layer correlation matrices can be defined as:

$$C_{ab} = \begin{bmatrix} \omega & 0 & 0 \\ 0 & \cdots & 0 \\ 0 & 0 & \omega \end{bmatrix}$$

$\omega_n \mid n \in N$ is the penalty of moving from layer a to layer b at node n. As described in Chapter 2, we assign a penalty because objects in different layers describe distinct types of objects. For example, one park is similar to other parks since they have the

same type but a nearby police station is less relevant since police stations and parks are not commonly associated with each other.

We define ω as the result of the relevance calculation for the two layers.

$$\omega = \lambda(L_p, L_q)$$

3.3.3 Generating the Supra-Adjacency Matrix

In this section, we define the supra-adjacency matrix for geographic objects based on the definitions from earlier sections.

Definition: *A multiplex network with M layers can be modeled in a matrix form called a* ***supra-adjacency matrix*** $\mathcal{A}$ *where* $\boldsymbol{A_m}$ *is the adjacency matrix for layer m and* $\boldsymbol{C_{ab}}$ *is the inter-layer correlation matrix between layers a and b*:

$$\mathcal{A} = \begin{bmatrix} \mathrm{A}_1 & \cdots & \mathrm{C}_{1\mathrm{M}} \\ \vdots & \ddots & \vdots \\ \mathrm{C}_{\mathrm{M}1} & \cdots & \mathrm{A}_{\mathrm{M}} \end{bmatrix}$$

The supra-adjacency matrix $\mathcal{A}$ acts as its own adjacency matrix and structurally this implies that a multiplex network shares similar properties to individual networks [142]. The existence of a value in the matrix represents a relationship between nodes in any layer and in fact treats the equivalent node in multiple layers as separate nodes. For example, assuming N = 5 nodes and M = 3 layers (each with its own 5 x 5 adjacency matrix) the supra-adjacency matrix would be a 15 x 15 matrix. In a sense, this multiplex network is similar in structure to having 15 nodes in one layer. This intuitively makes sense because a network of networks (like a multilayer network) is just a larger network.

Since ω represents a penalty, given any starting layer (like streets), any objects in layers that have strong relationships with that starting layer (the two layers have a relationship with high ω) will show up in search results with higher probability than objects in layers with a weaker relationship with the starting layer.

To complete the construction of the supra-adjacency matrix we must calculate the inter-layer correlation matrices C_{ab} and then take the layer adjacency matrices and the inter-layer correlation matrices and combine them together.

The supra-adjacency matrix combines probabilities representing relationships between objects in the same layer (a_{ij} in A) and relationships between layers (ω in C). The next step is to model transitive relationships between objects.

3.3.4 Multilayer SLN Schema

Our goal is to use the supra-adjacency matrix $\mathcal{A}$ and generate an mSLN that defines the relationships between objects among differing layers. For example, we want to define the relationship between a given park and a given street.

With our construction, we only allow movement from layer to layer at the same node, as shown in Figure 25. That is, if we want to move from node *i* in one layer to node *k* in a second layer where the corresponding inter-layer correlation matrix node is represented by ω in our SLN schema we mean the following:

$$n_i \xrightarrow{\alpha} n_{j1} \quad n_{j1} \xrightarrow{\omega} n_{\mathrm{j2}}\,, \quad n_{\mathrm{j2}} \xrightarrow{\beta} n_k \implies n_i \xrightarrow{\gamma} n_k$$

$$|\ \alpha, \beta, \omega, \gamma \text{ are weights on semantic links and } \alpha \cdot \omega \cdot \beta = \gamma$$

This generalized scheme employs a connector node j that exists in both layers and represents a link between the two nodes i and k and accounts for the different types of

objects represented in a GIS. This link accommodates the possibility that the two layers have different topological relationships as defined in Section 3.3.2. For example:

- **Distance** - If the two objects are point objects and we are interested in the distance between the two objects to calculate similarity, then j is the median between the two points. In this situation we can simplify the schema to only include α to represent the distance relationship between the two points and ω to represent the strength of relationship between the two layers containing the points. If both points being compared are in the same layer, then $\omega = 1$ and α becomes the only determinant of the relationship between the two objects.
- **Polygon Distance** - If the original object i is in a point layer and the second object k is in a polygon layer, we can project object i to the topology of object k, or vice-versa. For example, if we want to find the nearest park to an input house, we can convert the input house to a polygon representation and then calculate the distance. Conversely, we can take all parks in second layer, convert them to points to match the topology of the original house and then calculate distance. In this scenario, j can represent either the equivalent polygon representation of the point coordinate in the second layer or the polygon that includes the first point in the second layer.
- **Point in Polygon** – Most GIS systems and spatial-enabled relational databases determine whether or not a point is inside a polygon (determining which school district contains a house) by using a ray casting algorithm. In this scenario j represents the intermediate rays used in the "point in polygon" calculation.

In these examples, j_1 and j_2 represent the projection of the original node i (which exists in the first layer) to the topology of the second layer containing destination node j. This schema shows the most generalized schema of the transition from node i to node k to conceptually determine the relationships between geospatial objects in different layers. However, in many cases (including the examples above) the schema can be simplified. The practical application of this schema uses common database functions to perform most of these calculations.

The probability γ represents the generalized probability of moving from one geospatial object to another when the two objects are in different layers. γ is made up of these components: α and β represent the distance between the two nodes geographically as derived from each layer's adjacency matrices and ω represents the strength of the relationship between the two layers containing the two objects of interest.

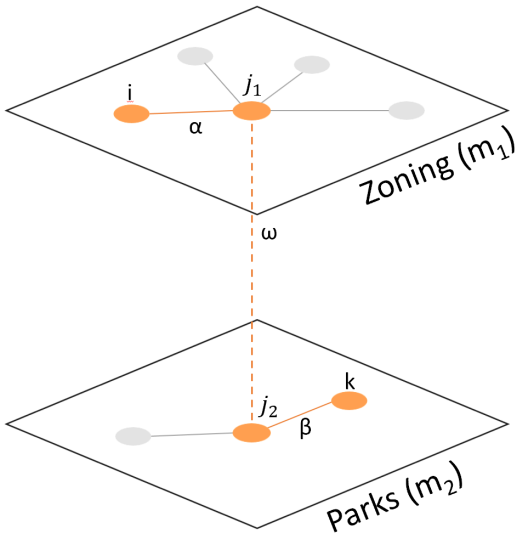


Figure 25: Multiplex SLN Schema illustrating the relationship between parks and zoning.

The mSLN schema defines the relationships between nodes in one direction. That is, we define an origin node and layer and a destination node and layer. There are many situations in which the relationship between two nodes in one direction can have a different meaning than the relationship in the opposite direction. For example, if we look at houses and school districts, there is only one school district for any given house, but there can be many houses contained in one school district.

This schema describes the transition from one node to another node across layers. Since the supra-adjacency matrix describes the topological relevance between all nodes in our domain and the relevance between layers, we can apply matrix operations to the supra-adjacency matrix $\mathcal{A}'$ to generate our mSLN. The steps are outlined in the next section.

3.3.5 Generating mSLN

There are two steps needed to convert the supra-adjacency matrix $\mathcal{A}$ to a reasoning-closed mSLN. First, we need to generate an initial mSLN by normalizing the row probabilities of $\mathcal{A}$. Second, we need to apply reasoning rules to the initial mSLN to reach its reasoning-closure. This section details those steps.

Generating Initial mSLN $\mathcal{A}'$.

Using the supra-adjacency matrix $\mathcal{A}$ we need to generate an mSLN. We follow the procedure outlined in [160, 161].

First, we need to convert the values in $\mathcal{A}$ into probabilities by normalizing them to produce a right stochastic matrix $\mathcal{A}'$. If $\mathcal{A}$ is a p × p matrix (where p = N × M):

$$\mathcal{A}'_{ij} = \frac{\mathcal{A}_{ij}}{\sum_{m=1}^{p} \mathcal{A}_{im}}$$

$\mathcal{A}'$ represents a Markov chain's transition matrix describing the probability of navigating between any two nodes. The initial SLN $\mathcal{A}'_{ij}$ represents the probability of moving from node i to node j. Therefore, for any node i the highest value of $\mathcal{A}'_{ij}$ (where j is any other another node in the graph) implies the strongest and most likely node to traverse. Additionally, the sum of all probabilities in a single row (representing the probability of moving from one node to any other node) sums to 1:

$$\sum_j \mathcal{A}'_{ij} = 1$$

Applying Reasoning Rules

We apply reasoning rules to the initial SLN to discover the implicit relationships between pairs of nodes. After applying reasoning rules, we should have a modified SLN that better reflects the strength of relationship between two nodes.

We use the reasoning rule defined earlier:

$$n_i \overset{\alpha}{\rightarrow} n_j, n_j \overset{\omega}{\rightarrow} n_{j*}, n_{j*} \overset{\beta}{\rightarrow} n_{k*} \implies n_i \overset{\gamma}{\rightarrow} n_{k*}$$

$$| \ \alpha, \beta, \omega, \gamma \text{ are weights on semantic links and } \alpha \cdot \omega \cdot \beta = \gamma$$

γ represents the **relevance score** between the two objects i and k. This rule says that if we want to traverse between two different geographic objects across two different layers, we multiply the probabilities of traversing between the two points in each layer (α and β) with the probability of traversing between the two layers (ω) to get the actual probability of traversing between the two points (γ). Since we have constructed a normalized supra-adjacency matrix $\mathcal{A}'$ we can use matrix multiplication to define the indirect links between all geographic objects [162].

Theorem 1: *If $\mathcal{A}'$ represents the initial distribution of transition probabilities for a stochastic process then the probability distribution at time t is* $(\mathcal{A}')^t$.

Proof: Let q(0) represent the initial distribution of probabilities from an initial node and q(t) represent the distribution of probabilities from that node at time t. This also represents a single row in $\mathcal{A}'$ corresponding to that initial node. We will prove this proposition by induction.

Base Step: For t = 0, the supra-adjacency matrix to the 0^{th} power is the identity matrix:

$$(\mathcal{A}')^0 = I$$

Induction Step: Next, we assume

$$q(t-1) = q(0) \cdot (\mathcal{A}')^{t-1}$$

If we multiply by $\mathcal{A}'$ then

$$q(t-1) \cdot \mathcal{A}' = q(0) \cdot (\mathcal{A}')^t$$

We then need to prove that $q(t) = q(t-1) \cdot \mathcal{A}'$. We look at the individual rows of row q. X_t represents the state of a traveler along the nodes in our Markov chain at time t.

$$(q(t-1) \cdot \mathcal{A}')_i = \sum_{j=1}^{n} q(t-1)_j \cdot p_{ji}$$

$$(q(t-1) \cdot \mathcal{A}')_i = \sum_{j=1}^{n} P(X_{t-1} = j) \cdot P(X_t = i | X_{t-1} = j)$$

$$(q(t-1) \cdot \mathcal{A}')_i = \sum_{j=1}^{n} P(X_t = i | X_{t-1} = j)$$

$$(q(t-1) \cdot \mathcal{A}')_i = q(t)_i$$

Therefore, the distribution of probabilities after t steps can be determined by taking the initial distribution and multiplying it by $(\mathcal{A}')^t$. This means that $(\mathcal{A}')^t{}_{ij}$ represents the probability of being at node j after t steps given that the starting node is i. ■

Theorem 2: *Multiplying right-stochastic matrix $\mathcal{A}'$ by itself produces another right-stochastic matrix which continues to represent a Markov chain representing the probability of navigating from node to node.*

Proof: From the definition of the initial mSLN we have a right stochastic matrix $\mathcal{A}'$, so all of the values in any individual row $j \in \{1 \ldots . n\}$ sums to 1. If we multiply $\mathcal{A}'$ by itself:

$$\sum_i (\mathcal{A}'\mathcal{A}')_{ij} = \sum_i \left(\sum_k \mathcal{A}'_{jk} \mathcal{A}'_{ki} \right)$$

$$= \sum_k \left(\mathcal{A}'_{jk} \quad \left(\sum_i \mathcal{A}'_{ki} \right) \right)$$

$$= \sum_k (\mathcal{A}'_{jk} \cdot 1) = 1$$

This means that every time we multiply the initial mSLN we get another right stochastic matrix. Each multiplication represents an additional walk on the graph. Squaring the initial mSLN represents all two-walk patches between any two nodes (as represented by their cell position on the product matrix) and multiplying the initial mSLN three times represents all three-walk paths between any two nodes.

Theorem 3: *If $\mathcal{A}'$ is a right-stochastic matrix then any power of $\mathcal{A}'$ of the form $(\mathcal{A}')^{\boldsymbol{k}}$ is also a right-stochastic matrix.*

Proof: This can be proved by induction.

Base Step: $\mathcal{A}'$ is a right-stochastic matrix and we have already shown that $(\mathcal{A}')^2$ is a right-stochastic matrix (see Theorem 2).

Induction Step: If we assume that $(\mathcal{A}')^{k-1}$ is a right-stochastic matrix and we multiply it by $(\mathcal{A}')^k$ then we get $\mathcal{A}'(\mathcal{A}')^{k-1} = (\mathcal{A}')^k$ therefore $(\mathcal{A}')^k$ is a right-stochastic matrix. ■

Theorem 3 demonstrates that multiplying the initial SLN multiple times preserves its right-stochastic properties and continues to represent a Markov chain establishing the probabilities of moving between any node in the network.

The goal is to calculate all potential walks between any two geographic objects in the domain to identify which walks between those objects produces the strongest edge weight (probability). Algorithm 1 multiplies the initial mSLN n times (once for each of the total number of rows or objects in our domain) and we limit ourselves to n matrix multiplication operations because the longest potential walk between any two nodes in a graph is the total number of nodes in the graph.

ALGORITHM 1: Applying Reasoning Rules on the mSLN [162]

***Input**: supra-adjacency matrix $\mathcal{A}'$*
***Output**: supra-adjacency matrix with reasoning rules applied $\mathcal{A}^*$*
$O \leftarrow$ supra-adjacency matrix $\mathcal{A}'$
P *$\leftarrow$ supra-adjacency matrix $\mathcal{A}'$*
$n \leftarrow$ length of $\mathcal{A}'$
for t = 2 to k - 1 **do**
    $O \leftarrow (\mathcal{A}')^t$
    $P \leftarrow (\mathcal{A}')^t$
    for *each* *i* *in* *n* **do**
        for *each* *j* *in* *n* **do**
            if $O_{ij} > P_{ij}$ *and* $i \,!= j$
                $P_{ij} = O_{ij}$
        end
    end
end
set $\mathcal{A}^*$ = P

return $\mathcal{A}^*$

Corollary: *Algorithm 1 converges the Markov chain between all geographic objects in the domain to a quiescent state representing the optimization and reasoning-closure of* $\mathcal{A}'$.

In Algorithm 1, we take the original supra-adjacency matrix $\mathcal{A}'$ and multiply it by itself k - 1 times, where k is the length of $\mathcal{A}'$. The length k of $\mathcal{A}'$ also represents the total number of geographic objects across all layers. During each multiplication, we compare the values of the new matrix with that of the previous iteration and if any of the matrix values are higher, we replace that value in the original matrix with the higher value. Replacing a value in the matrix implies that we have established a stronger transitive link between two objects than the original direct link between the two objects.

We multiply $\mathcal{A}'$ by itself a maximum of k - 1 times to reach a state of quiescence and find the highest of the k – 1 values of $(\mathcal{A}')^k$ (Step 1). We choose k - 1 because multiplying $\mathcal{A}'$ by itself k – 1 times creates the reasoning closure of the SLN and multiplying the reasoning-complete SLN by $\mathcal{A}'$ would not provide any new semantic links [3]. If we have a matrix with k nodes, then the longest possible sequence of edges from any node has length k – 1 and any further iteration would create a reasoning rule that represents a cyclical path on the network.

The result of Algorithm 1 is a reasoning-closed mSLN $\mathcal{A}^*$ which represents all of the optimized links between all geographic objects in our domain and still accounts for all of the topological and layer relationships established during the initial creation of

the mSLN. $\mathcal{A}^*$ serves as a basis for the mathematical approach outlined in the next section.

3.4 Formal Statement of the Problem

The purpose of this dissertation is to create a system that can allow an end-user to query a GIS database and to have that database contribute answers that are relevant to that query by retrieving relevant geographic objects based on this query. GIS queries generally originate from a user's context (such as their location or an initial address of interest).

Definition: *The **initial search node** g_x is the node in the graph representing the user's context when interfacing with the system.*

In the earlier sections we defined a set of geographic objects g_i within a domain of interest $\mathcal{G}_D$. We then defined the distinct types of relevance scores between these geographic objects α. We also defined groups of geographic objects of a common type called a layer L containing those objects and the relevance scores between those layers λ. Combining these create a measure γ identifying the total relevance between two geographic objects.

While identifying the most relevant single object in our domain to an initial object is helpful, the purpose of this research is a database query resulting in a set of results that reach a minimum relevance criterion.

Definition: *For any query, the **search (query) threshold** Θ is the minimum relevance score necessary to provide useful results to a user. The search threshold also refers to the minimum probability of walking to any geographic object in the graph representing the objects in our geographic domain.*

The search threshold is a percentage that determines the minimum relevance score a geographic object must have with the initial search object to be returned to a user. This can be defined by the user to fine-tune the number of results returned or can default to a value specified by the system.

Definition: *Given an initial object of interest g_x, a series of edge weights between geographic objects γ, a context profile ρ, and a search threshold Θ, the* ***result set*** G_R *is the set of all objects within the geospatial domain where the edge weights between that initial object and all objects in the domain is greater than that threshold.*

$$G_R \subseteq G \; where \; \exists \; g_r \in G_R \mid \gamma(g_r, g_x) \geq \; \theta$$

That is, given an initial geographic object of interest g_x (a set of coordinates or another geographic object) we want to identify the set of most relevant geographic objects in the form of a result set G_R where the relevance score γ between objects within that set and the initial point of interest is greater than a fixed query threshold θ.

3.5 Solution to the Problem

Given an initial starting location g_x, identifying the most relevant objects involves querying this mSLN for the objects with the highest relevance score relative to the initial objects of interest. Based on our mSLN schema, we want to maximize γ.

Restating the Problem

We constructed a supra-adjacency matrix $\mathcal{A}'$ that models the relationships between objects in a geographic domain considering their geographic (distance and/or topology) relationships as well as the relationships between the layers containing those objects.

Since the mSLN is a matrix-based model of the optimized relationships between all geographic domains and Algorithm 1 is a modification of Warshall's algorithm to find the shortest path between any two nodes in a directed graph, we can redefine the statement of the problem in terms of the mSLN:

$$G_R \subseteq G \; where \; \exists \; g_r \in \; \mathcal{A}^* \text{ where } \mathcal{A}^*_{ij} \geq \; \theta$$

That is, given an initial geographic object of interest g_x (a set of coordinates or another geographic object) we want to identify the set of most relevant geographic objects in the form of a result set G_R where the relevance score γ between objects within that set and the initial point of interest is greater than a fixed query threshold θ.

Identifying an Initial Search Node and Layer

Querying the mSLN involves finding a starting geographic object g_x. This could occur in one of the following example scenarios:

A user with a mobile device uses the device's GPS coordinates to serve as a starting location and then finds the nearest address to that starting location to serve as the starting node. If an address is not feasible given the user's context (for example, the user is in the middle of a state or national park far enough away from a useable address) an alternative starting layer could be used. In this situation the GPS coordinates of the user would be treated as an isolated layer with one node. If we create a virtual node g_{gps} and identify an initial layer L we can identify the closest node as follows:

$$g_x = \; g_r \mid where \; d(g_{gps}, g_r) = \min\left(d(g_{gps}, g_i)\right) \; \forall \; g_i \in \; \mathcal{G}_L$$

A user enters in an address into a form (common in many GIS applications). The address layer is the starting layer, and the entered address is the starting node.

$$g_x = \; g_{address} \; \in \; \mathcal{G}_{addresses}$$

A GIS interactive map user selects any geographic object displayed on the map. Based on the layer and object selected relevant geographic objects can be selected.

$$g_x = g_{selected} \in \mathcal{G}$$

Given an input data row (to make a prediction on) we first generate an initial prediction, which is used to reveal additional predictions from the mSLN as described below.

Definition: *Starting with an incoming query* $Q = \{g_x, m_j\}$*, an* ***initial prediction*** *is the location of the point of interest as it is represented in an appropriate node* $g_x \in N$ *and an appropriate layer* $m_j \in M$.

For example, if a user types in an address, then the initial node is the address entered, and the starting layer is the "address points" layer. In most situations the initial node is implied. For example, if the initial node is a searched address (one's home, for example) then it is implied that the initial starting layer is the address points layer.

Traversing the mSLN

Once the initial node has been identified, we need to discover relevant matches by traversing the mSLN based on a fixed threshold θ. We traverse nodes in the mSLN where the relationship value on the link is higher than this threshold. In other words, given an initial node $g_x \in N$ and a search threshold θ we identify all $g_i \in \mathcal{G}_D$ where $\mathcal{A}^*_{g_x,g_i} \geq \theta$.

Searching mSLN Within One Layer

The simplest example of this algorithm involves searching for relevant geographic objects within the initial starting layer. For example, given an input address, finding the closest addresses to that address.

ALGORITHM 2A: Searching mSLN Within One Layer

$m_x \leftarrow$ *initial search layer*
$n_x \leftarrow$ *initial node on that layer*
$\Theta \leftarrow$ *search probability threshold*
for n_i in m_x **do**
    if $\alpha(n_i, n_x) \geq \Theta$
        $R \leftarrow$ *set of search results (relevant geographic objects)*
    end if
end
sort R **by** $\gamma(n_i, n_x)$ descending
return R

In Algorithm 2A, the use of α (the adjacency values) and γ (the optimized mSLN values) are interchangeable since we are not interested in the inter-layer correlation values.

Searching Multiple Layers Through a Single Point

There are some scenarios where it is necessary, once an initial search location is identified, to identify the one relevant feature in other layers to that initial object. In local government GIS, these types of applications are known as "My Neighborhood" applications

ALGORITHM 2B: Using mSLN to Identify All Features at Same Location

$m_x \leftarrow$ *initial search layer*
$n_x \leftarrow$ *initial node on that layer*
$\Theta \leftarrow$ *search probability threshold*
$M_x \leftarrow$ *all layers ontologically related to* m_x *sorted by* ω *descending*
for n_i in m_x **do**
    for ***each*** M_j ***in*** M_x **do**
        if $\omega(m_x, M_j) \geq \Theta$
            for ***each*** k ***in*** M_j **do**
                if $\alpha(n_x, k) * \omega(m_x, M_j) \geq \Theta$
                $R \leftarrow R + k$
            end
        end if

```
        end
end
sort R by γ(n_i, n_x) descending
return R
```

Searching for All Relevant Objects in All Layers

To facilitate a more practical, time-efficient, and real-time search (where the relevance scores in a context profile are not pre-calculated), we propose a search algorithm (Algorithm 2C) that respects the spirit of the supra-adjacency matrix calculations while remaining "live" to facilitate the types of searches a user would expect.

ALGORITHM 2C: Live "Cold-Start" Searching mSLN

```
m_x ← initial search layer
n_x ← initial node on that layer
Θ ← search probability threshold
M_x ← all layers ontologically related to m_x sorted by ω descending
for n_i in m_x do
        if α(n_i, n_x) ≥ Θ
                R ← set of search results (relevant geographic objects)
        end if
        for each M_j in M_x do
                if ω(m_x, M_j) ≥ Θ
                    for each k in M_j do
                        if α(n_x,k) * ω(m_x, M_j) ≥ Θ
                            R ← R + k
                    end
                end if
         end
end
sort R by γ(n_i, n_x) descending
return R
```

Any search requires four parameters. First, the initial search layer from which to start with (for example, if a search involves a user selecting an address, the initial layer should be the address layer). Second, an initial node on that layer. Third, a probability threshold. Fourth, a context profile defining which layers are relevant to the user. All features in all layers above the probability threshold in that context profile will be returned.

In other words, to conduct a search, the algorithm looks for all objects within the probability threshold Θ in the same layer. For example, when the user is looking at a park then the search will find all other parks within the threshold first. Then it will look at the other layers in the ontology M_x and if the strength of relationship between the selected layer and the original layer is greater than the threshold, it will look at the features in the layer. Because of the transitive nature of the mSLN, we multiply the probability representing the relationship between the two layers with the probability representing the distance between the initial search node and the objects in the second layer. This algorithm is represented in Figure 26 below.

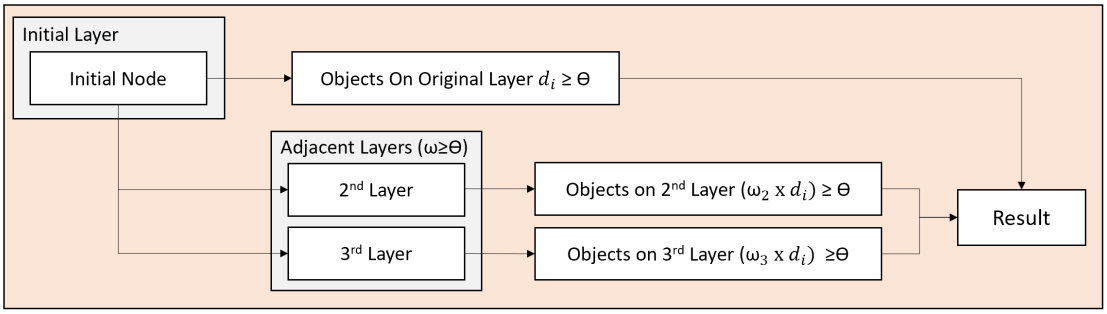


Figure 26: General Search Algorithm (simplified version of Algorithm 2C)

3.6 Algorithmic Complexity

Table 4 shows the different calculation steps involved in the generation of the mSLN and the computational complexity involved in each. Note that when given a dataset with N nodes and M layers, N will almost always be the larger number and will dominate any complexity calculation.

Table 4: Performance (Big-O) calculations for generating mSLN.

Step	Rough Calculation	Complexity
Create Layer Adjacency Matrices	M x N x N	$O(n^2)$
Create Inter-Layer Correlation Matrices	M x N	O(n)
Generate Supra-Adjacency Matrix	M x N	O(n)
Generate Initial SLN	MxN x (N+M–1)	$O(n^2)$
Generate Optimized SLN	(Initial SLN)^N	$O(n^4)$

Given that all operations in the algorithm can be broken down into chunks that can be independently computed, we feel that this algorithm would benefit from the processing capabilities of a big data node. For example, matrix multiplication for a 10 x 10 matrix can be broken down into 100 independent calculations, one each for the 100 cells of the resultant matrix.

Notes on Scalability

While a typical local government (such as the Howard County example used in the evaluation) can have about 200,000 geographic objects stored in their GIS database, there are concerns about how the system would perform at a state/province or federal level. For example, a dataset of addresses for the United States would include hundreds

of millions of records which could prove problematic considering the performance estimates provided in Table 4.

However, the mathematical operations documented in this Chapter (such as matrix multiplication) are parallelizable and can be calculated using a distributed computing architecture.

Additionally, the use of context profiles allows relevance to be calculated on a subset of objects which minimizes the number of necessary calculations.

Notes on Sparsity

While the construction of the supra-adjacency matrix for M layers and N nodes consists of a (M x N) x (M x N) matrix, most of these values are zero since not every feature exists in every layer.

$$\mathcal{A} = \begin{bmatrix} \boldsymbol{A_1} & \dots & \boldsymbol{C_{1M}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{C_{M1}} & \dots & \boldsymbol{A_M} \end{bmatrix}$$

In the example above we can choose any individual row. Let us choose the first one. This row consists of N values in $\boldsymbol{A_1}$ and then there are M – 1 inter-correlation values since each C in that row consists of only one ω and there are M – 1 $\boldsymbol{C_{1m}}$ matrices. Given N = 5000 and M = 5, the first row in this construction consists of 5004 actual values and 19996 zeroes. The more layers M the greater the sparsity of the matrix.

While the mathematical foundations require a substantial number of matrix multiplications, the practical implementation of this model can be completed with some shortcuts, notably:

- When all records are stored in a database (as described in Chapter 4) we can focus on calculations for the rows that exist. This removes the problem of sparsity.
- In a database environment we can utilize the spatial indexing functions of the DBMS. For some functions, notably finding the "nearest" objects, this reduces the runtime from O(n) to O(log n).
- We split the processing into a static component and a dynamic component. The static component pre-generates the mSLN ahead of time and reduces the effort of using the mSLN to a simple database lookup.

3.7 Summary

Chapter 3 described a mathematical model of geographic objects, the potential relationships between these objects, and the creation of an mSLN that models the implicit semantic relationships between these geographic objects. The next chapter translates this mathematical model into a practical application system that can be implemented in an organization using existing database and GIS technologies.

Chapter 4: General System Architecture

Chapter 3 tackled the mathematical problem of presenting relevant information to a user query in a geospatial environment. The purpose is to provide a user with additional and relevant information in an intelligent manner that enhances a GIS user's experience and provides new and unexpected information based on an analysis of historical user interactions with the GIS and the current user's stated intentions with the system.

To make the system useful to organizations that have implemented existing geographical information systems with a database backend, it is necessary to outline the necessary steps to implement this mathematical model using existing database and machine learning technologies. Many of these components already exist in modern geospatial infrastructures and can be easily adapted to build this model.

This chapter takes this mathematical framework and translates it into a practical framework that reflects those principles and can be practically implemented in an organization called SAM-GIS (Semantic Augmentation Model for GIS). This framework examines the properties of geospatial data from various sources (primarily relational databases), creates a model of geographic relevance, and sets up an interface to that model that facilitates user queries. A diagram of this architecture is shown in Figure 27.

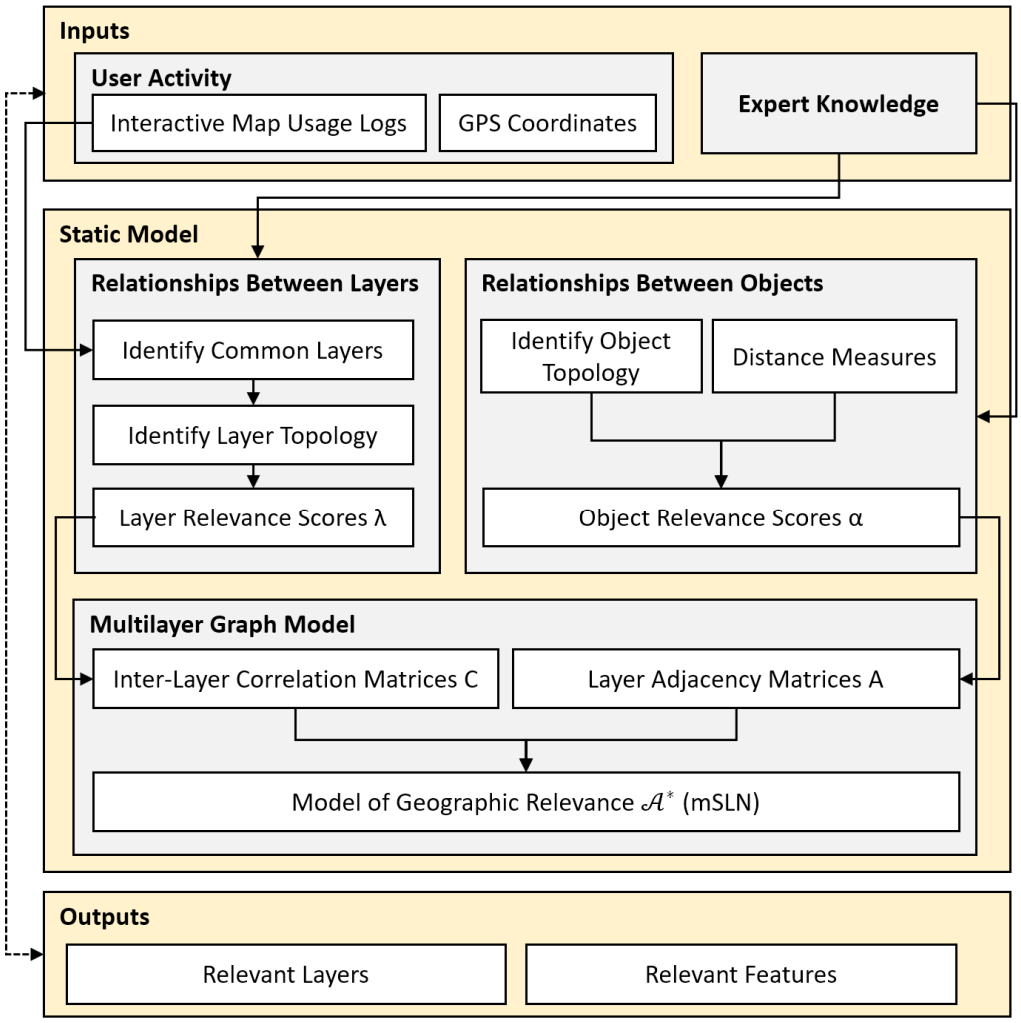


Figure 27: An outline of SAM-GIS representing the approach of this research.

To integrate this framework into existing GIS workflows it must respect the mathematical model developed in Chapter 3 and the types of GIS technologies and workflows currently used in industry.

Describing this framework requires understanding the types of GIS inputs that feed into the model (Section 4.1), a static model of the relationships between geographic objects (Section 4.2), and a dynamic query interface that provides relevant geospatial objects based on a user's query (Section 4.3).

The complete model includes three components:

- The model needs to be easily implementable by organizations. Section 4.1 describes a **series of inputs** to the model that reflect the readily available GIS technologies available today.
- A data structure needs to be developed that can reflect the mathematical model of relevance described in Chapter 3. Section 4.2 describes a **static model** of geographic relevance between geographic objects and outlines the processes needed to create this model.
- Applications need to easily query the static model in a way that reflects user input and the user's context. Section 4.3 outlines a **dynamic query interface** that queries the static model to provide users relevant geographic objects (**outputs**) based on their current situation.

4.1 System Inputs

Since the proposed system is an extension of the types of technologies used in modern GIS infrastructures, a complete description of this system should describe the type of inputs that will be used to create the static model and as inputs to the dynamic query interface.

In this section we describe the various inputs to the system described in Chapter 4. First, we describe the nature of modern tiled interactive maps, how users use these types of maps, and how the logs of this usage can be used as an input to the static model. Second, we talk about a user's current location as represented by GPS coordinates and how these coordinates can be used to infer a user's context and intention. Finally, we

discuss the role of expert knowledge to fill-in relationships between layers that cannot be easily computed automatically.

Section 4.2 will examine how these inputs will be used to create a static model describing the relationships between geographic objects in our domain. Section 4.3 will use these inputs to guide queries to the system.

4.1.1 User Activity

The usage of various GIS applications can inform a system of the user's intentions. We examine two common user interactions with a GIS: Online interactive map activity and the user's GPS coordinates. By understanding the behavior of users of a GIS we can try to predict and suggest geographic layers and objects of interest to the user.

Interactive Map Usage

This section describes the nature of interactive maps and the logs that can be extracted from them and then outlines a workflow to pre-process these logs into an input suitable for the static model described in Section 4.2.

Most modern interactive mapping system such as Google or Bing maps or OpenStreetMap are based on map tiles. These organizations divide the Earth into standardized tiles and load relevant data bounded by these tile extents into the user's web browser. While the sources of these tiles can vary (generated from a database or loaded as preset images) these tiles conform to a consistent standard. This allows multiple layers to be presented and be overlaid perfectly on an interactive map without the need to worry about the source of the data.

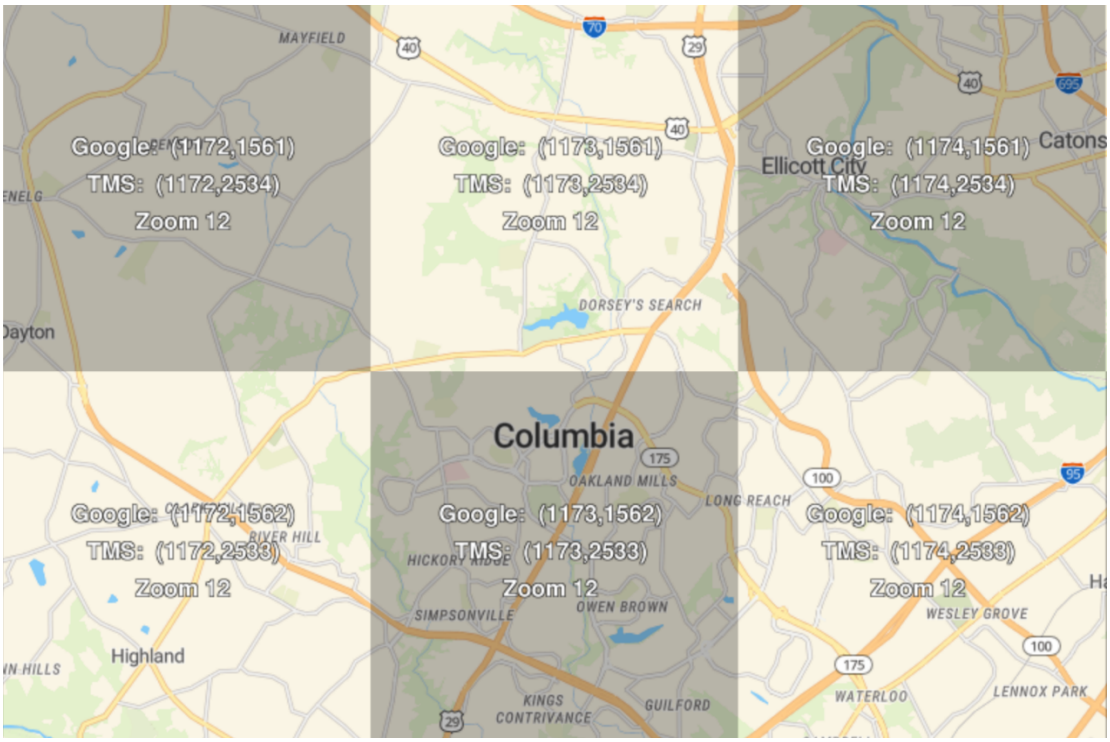


Figure 28: An example of standard tile numbering schemes. From https://www.maptiler.com/google-maps-coordinates-tile-bounds-projection/

Several standards exist for tile requests [163]. These standards were created so that any request for map data from any server that conforms to the standard would consistently show the data in the same location. Generally, these tiling standards represent a tile as a triplet (X, Y, Z) where X and Y denote the tile map's coordinate center and Z represents a preset zoom level on the map.

An example of a tiled web map is **Web Map Service** (WMS), an international standard for generating and distributing map information on the Internet. A web server that supports the WMS standard takes an incoming web service request and generates an image tile that is returned to the user. Since WMS requests are web service calls, we can look at the nature of these requests by investigating the web server logs of an internet web server that processes WMS requests.

A WMS request is a web service request to a server that supports the WMS standard and generates a map image. A typical WMS request includes the layer of interest and a bounding box defining the area that data from that layer should be returned to the user. For example, the following WMS request asks for an image of all streets within a box defined by a given set of coordinates:

https://data.howardcountymd.gov/geoserver/wms?SERVICE=WMS&VERSION=1.1.1&REQUEST=GetMap&FORMAT=image%2Fpng&TRANSPARENT=true&LAYERS=general:Zoning&WIDTH=256&HEIGHT=256&SRS=EPSG:3857&STYLES=&BBOX=-8548717.243414111,4757440.64046937,-8547494.250961548,4758663.632921933

The request above asks the server for the zoning GIS layer (the LAYERS query string in bold) within the given bounding box (the BBOX query string also in bold). The results of this request are shown in Figure 29.

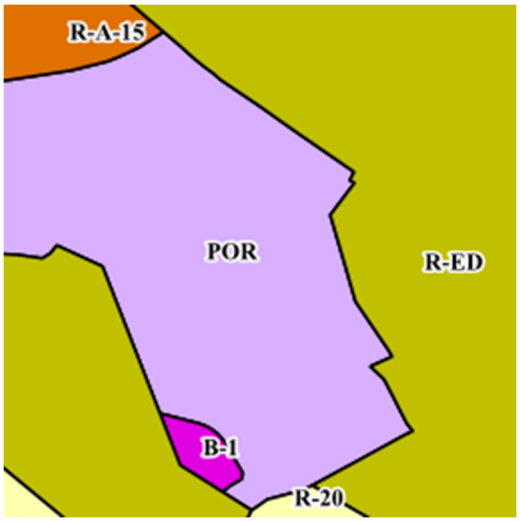


Figure 29: Example image tile of zoning generated from WMS request.

Each bounding box defines a tile on the map conforming to one of the tile schemes used by modern GIS interactive maps. This type of "tiled" map displays interactive

map graphics as a series of tiles that are stitched together to provide a cohesive map experience for users. When a user pans the map, tiles associated with the newly revealed coordinates are loaded from the server on the fly. This type of tiled map was popularized by the Google Maps platform.

An outline of the process of taking users' interactive map usage activity and converting this activity into preprocessed data that can be used to create a static model in Section 4.2 is described in Figure 30 within the "Inputs" component of the system architecture.

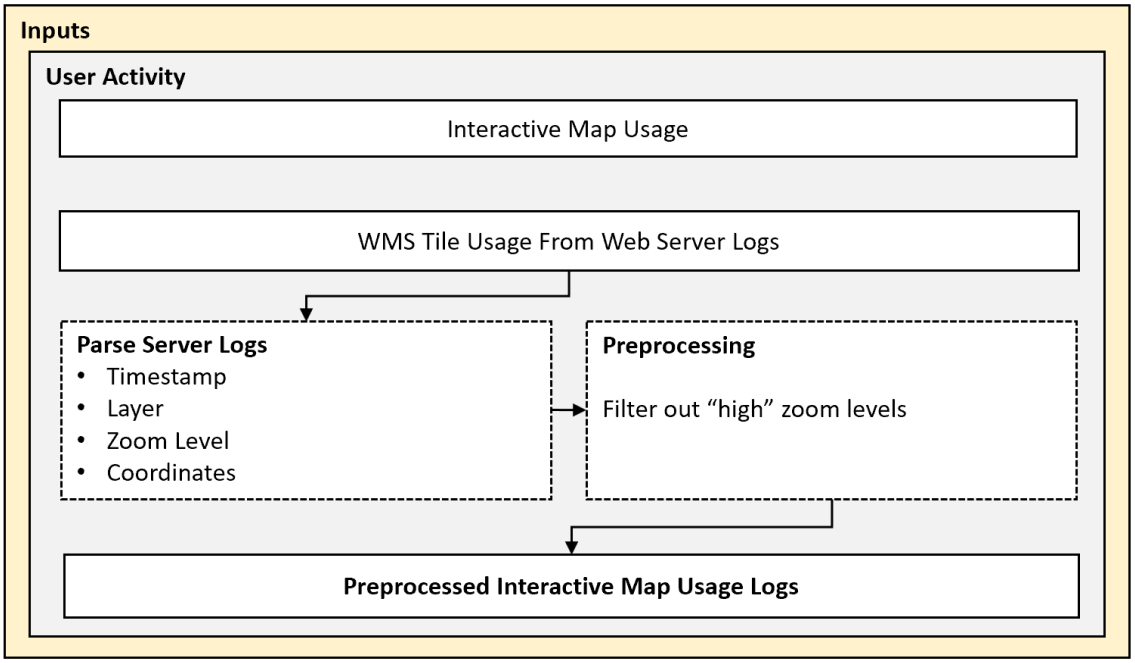


Figure 30: Preprocessing Interactive Map Usage

Parse Server Logs. Raw web server log files need to be converted into a format suitable for the analysis outlined in later steps. Given a series of log entries from a web server, we identify which of these log files represent WMS entries. We then parse the text of these entries to identify fields that will be relevant to our analysis.

Web servers that host OGC-client web services such as WMS can also include other types of web requests, including erroneous web requests. These non-WMS web server log entries are not relevant to our analysis, so we first filter out any non-WMS related web requests and then we filter out any WMS requests that are not of the type "Get Map." These records are loaded into a database for further processing and filtering.

Pre-Processing - This is the process that filters out WMS log entries at specific zoom levels because we are only interested in localized activity (a user zooms into a neighborhood of interest on an interactive map).

The WMS log files are loaded into a database table where one row contains a log item. That is, we take a web request string and convert into tabular data with the following columns:

- *Timestamp* – Common to all server logs, this includes the date and time the user requested a specified map tile. The timestamp information helps us identify simultaneously accessed layers.
- *Layer* – The map layer represented by the requested tile. In the WMS "Get Map" request, this is the text after the LAYERS query string identifier. Layers are identified by a workspace and layer name, so in our example the string is general:Street_Centerline. The workspace is "general", and the layer is "Street_Centerline." Workspaces in WMS layers are usually created as an organizational tool for security purposes and are not relevant to our analysis. We care about the layer name itself, in this case, the street centerline because one of the outputs of later steps is an ontology representing the relationships between these layers.

- The *bounding box* of a WMS request specifies a minimum and maximum set of coordinates that define a rectangular (usually square) geographic area [164]. Based on the bounding box query string BBOX, we can derive zoom level and coordinates:

- *Zoom Level* – The bounding box coordinates can infer the dimensions of the requested tile in real-world coordinates and since WMS requests adhere to discrete zoom levels globally, the zoom level can be inferred by calculating the X and Y dimensions and then comparing these values to a lookup table with the standard dimensions for each zoom level. In the example above, the bounding box coordinates are (-8548717.24, 4757440.64, -8547494.25, 4758663.63). This forms a bounding box that is 1123 feet by 1123 feet (8548717 – 8547494 = 1123, 4758663 – 4757440 = 1123), which implies a zoom level of 15 when referenced against the standard zoom level tile dimensions as described in Table 5.
- *Coordinates* – By taking the bounding box coordinates we can calculate the centroid of the tile. In the example above, we can derive a centroid of (-8548105, 4758051). We will use these point coordinates to identify clusters of requests in localized areas.

Table 5: A list of zoom levels and their corresponding tile dimensions.

Zoom Level	Approximate Tile Dimensions
11	19,568 ft. x 19,568 ft.
12	9,784 ft. x 9,784 ft.
13	4,892 ft. x 4,892 ft.
14	2,446 ft. x 2,446 ft.
15	1,223 ft. x 1,223 ft.
16	611 ft. x 611 ft.
17	305 ft. x 305 ft.
18	153 ft. x 153 ft.
19	76 ft. x 76 ft.

The result of this parsing is a table for each web request with the attributes listed in Table 6.

Table 6: Parsed WMS transaction logs in tabular format

Timestamp	Layer	Zoom	Coordinate X	Coordinate Y
2018-04-11 12:09:35	Zoning	15	-8548105	4758051
2018-04-11 12:23:00	Parks	15	-8546271	4740318
2018-04-11 04:28:00	Property	17	-8556590	4757669

GPS Coordinates and User Selected Features

A user's GPS coordinates can provide context into a user's situation and therefore can serve as an input into our framework. While the immediate coordinates of a user

when they are interacting with a GIS are not as important in the creation of the static model, they are important during a user's query.

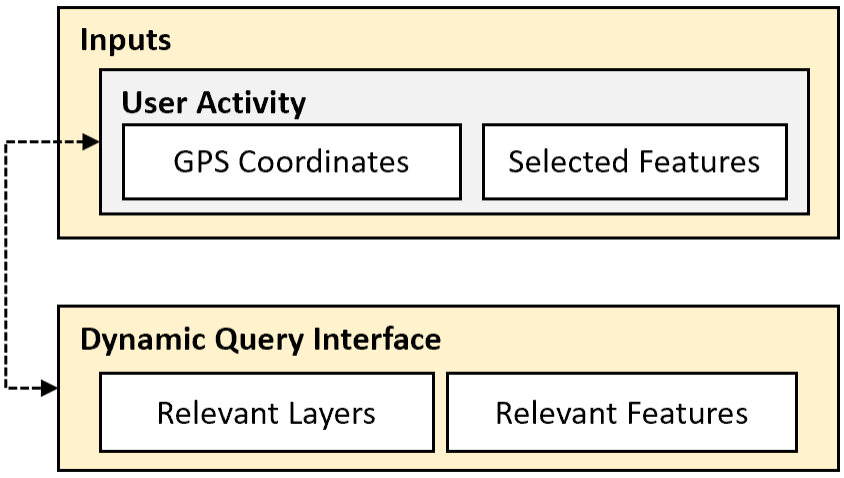


Figure 31: A User's GPS Coordinates and Selected Features are potential inputs to a Dynamic Query Interface.

Similarly, when a user selects a feature that user has indicated that this feature is an object of interest to that user and might be interested in relevant features to the selected feature. The extraction of relevant features to a user's selection will be described in detail in Section 4.3.

4.1.2 Expert Knowledge

The goal of processing user activity is to set up mathematical relationships between geographic objects: Both for the creation of a static model of those relationships and between a user's interactions with a GIS and the relevant features the static model attempts to reveal to the user based on those interactions.

While the relationships described above and in Section 4.1.1 are automated processes that, there are some aspects of the static model that cannot easily be

calculated automatically. For example, one of our goals in Section 4.2 is to calculate the strength of the relationships between objects and layers as a probability, a computer algorithm cannot figure out the nature of those relationships with 100% certainty. For example, there is no uniformly programmatic way to figure out how a police station and school are related. Therefore, we take the layer relationships derived from the WMS logs and manually assign the nature of the relationships using expert knowledge.

Objects in GIS layers can take many forms and based on these forms relationships between layers can be identified. Examples of GIS layer relationships include:

- **Object near object** – Objects in Layer A are close in distance to objects in Layer B. For example, when comparing police stations (represented as points) and addresses (also represented as points), we can define a relationship between addresses and police stations where police stations close to a given address are more relevant than those that are farther away.
- **Point inside polygon** – Point objects in Layer A are contained within polygon objects in Layer B. For example, we can define a relationship between address points and election precincts where addresses can be contained within election precincts (represented as polygons). We can use this relationship to define a user's election precinct based on the user's address.

Non-spatial relationships – GIS layers can be associated with each other and with tables that do not contain geographic information by database foreign key relationships. For example, a table that contains spatially relevant keywords (like town names or addresses) can be linked to geographic objects even if this table does not specifically contain geographic objects.

Other types of relationships for use in geospatial ontologies have been explored such as adjacency (between two polygons) and connectedness (between two linear objects), however, in this research we focus on the "proximity" and "contains" relationships [67]. Once we have a listing of relationships between GIS layers, we have created a geospatial layer ontology in the form of relationships between layers.

A further examination of the topological relationships between objects in different layers but expanded to include relationships between objects in the same layer was outlined in Section 3.2.

4.2 Static Model

A user of the framework (SAM-GIS) described in this chapter wants to enhance their usage of a GIS by taking inputs such as their GPS coordinates or a selected GIS object to find relevant objects within that GIS. Storing a series of geographic objects and measures of relevancy between those objects can make it easier for a user to identify relevant objects based on the user's query to the GIS.

This section describes a database-based static model of the relationships between geographic objects using the inputs described in Section 4.1 and a GIS containing a series of geographic objects represented in layers. This static model is created by taking some of the inputs described in Section 4.1, notably pre-processed interactive map logs and expert knowledge and producing a data structure that reflects these relationships and makes it easy to query this model.

Sections 3.2 and 3.3 described the potential mathematical relationships between models and this section will describe calculation of those relationships using existing database structures and data mining techniques.

This static model contains three components as shown in Figure 32:

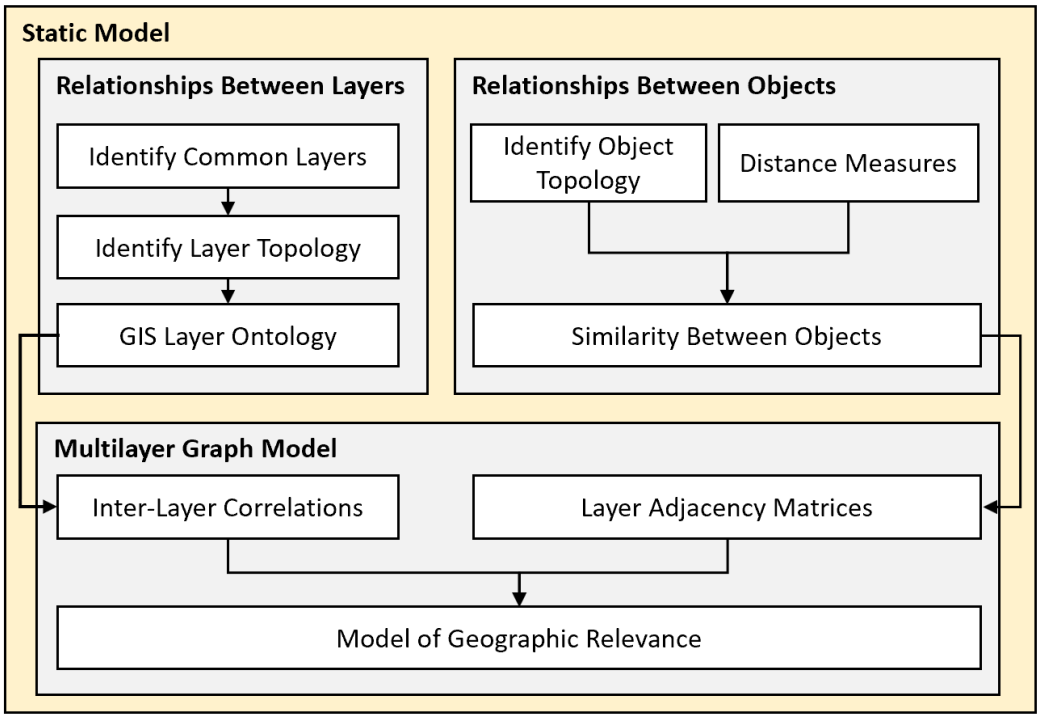


Figure 32: The Static Model in SAM-GIS

Relationships Between Layers (Ontology). Geographic objects in a GIS are typically organized into layers and the relationships between objects in those layers can be defined by the layers containing those objects. For example, when comparing "parks" to "streets" we can define an adjacency relationship between them (parks are adjacent to streets since they typically need vehicle access). When comparing any given park to any given street, knowing the layers containing those two objects can define the relationship (adjacency) between the two objects. Therefore, the first step in creating a static model of relevance between two objects is to identify the type and strength of relationship between the layers containing those objects.

Determining the types of relationships between layers requires the creation of a geospatial ontology that represents the relationships between different layers, such as

the relationship between addresses and school districts. This includes identifying layers that are commonly accessed in a GIS, identifying topological relationship between the objects in those layers as well as identifying the strength of the relationship between those layers. This is an implementation of the mathematical structure defined in Section 3.2.4 and is discussed in detail in Section 4.2.1.

Relationships Between Objects. Storing the relationships between individual geographic objects in a GIS will allow those relationships to be easily queried by a user of the GIS. After establishing the relationships between layers, relationships between geospatial objects needs to be mathematically defined, calculated and stored. We calculate the strength of relationships (similarity) between objects in the same layer. This also requires identifying the topological structure of the geospatial objects in that layer and using the topology to define the relevance between two objects in different layers and/or different geographic structures. For example, addresses are more relevant if their distance is smaller, but school districts are more relevant if they are adjacent to each other. This is an implementation of the mathematical structures defined in Sections 3.2.2 and 3.2.3 and is discussed in Section 4.2.2.

4.2.1 Relationships Between Layers

The primary objective of this research is to create a computational framework that can provide a user with relevant geographic objects when presented with a query. The layers that contain the geographic objects being compared to determine their relevance can identify the nature of the relationship between the two objects as well as the strength of the relationship between the two objects. For example, “parks and “schools” might be relevant to each other but “parks” and “fire stations” might not be.

Therefore, we want to identify the types of objects that are relevant to a user querying the GIS. For example, we need to discover that a user is interested in parks or schools before attempting to identify relevant objects that satisfy the query. This mimics the typical GIS workflow where relevant layers are loaded before querying objects in those layers or the typical problem-solving workflow where relevant datasets are identified and loaded before performing any data-driven analysis.

This section describes a process for identifying layers that are common to a user's interactive map usage by performing various data mining techniques on the pre-processed web server logs generated from those interactions from Section 4.1.1. Once we have established that a relationship exists between two commonly accessed interactive map layers, we use expert knowledge to define those topological relationships (a distance relationship, a point inside polygon relationship, etc.).

We identify which layers are relevant to our multilayer construction by creating an ontology, which we use to provide augmented and relevant answers to all user queries. An ontology represents the type of relationship between spatial layers. We first create a basis for our ontology by identifying common layer interactions during interactive map usage. We then use those layer interactions to define ontological relationships between any two layers. For example, if users commonly view schools and parks on an interactive map at the same time, we can infer that a future search for geographic information should include those two layers. The set of such relationships between different layers generates an ontology, as described in Figure 33.

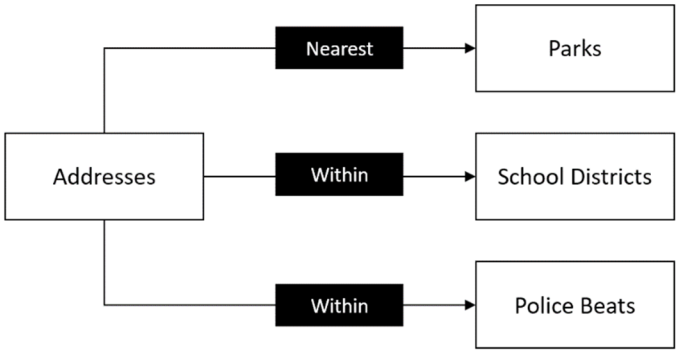


Figure 33: An example of a geospatial ontology describing the relationships between spatial layers.

Performing a series of mathematical operations on the pre-processed interactive map logs from Section 4.1.1 and the expert knowledge discussed in Section 4.1.2 will help produce a geospatial ontology.

Investigate Common Tiled Map Interactions

There are three methods that we use to investigate interactive map usage from WMS logs, as depicted in Figure 34. First, we use association rule mining on interactive map requests to identify layers commonly used as map overlays. The purpose is to reveal which layers are simultaneously relevant to the user, therefore building the user's context. We use the standard WMS web service schema to identify the nature of these requests.

Second, we look at WMS usage to find GIS layers that are commonly relevant based on a user's interactive map zoom and coordinates. Third, we identify common contexts of interactive map usage.

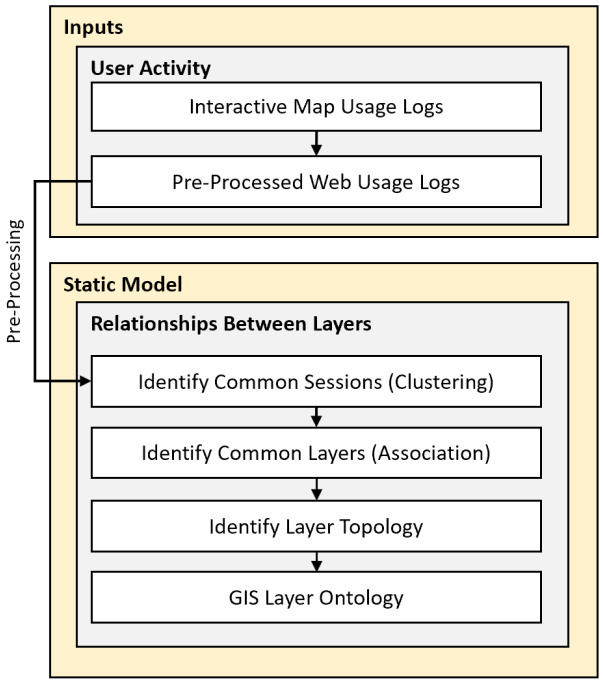


Figure 34: Techniques to derive a GIS Layer Ontology from interactive map usage

Since WMS requests are standard web server GET requests and since most web servers log web service requests by IP address, we use the source IP addresses to identify a session and the layers associated with that session. We do this under the assumption that when users access an interactive map they are doing so for a specific purpose. If we can identify the layers commonly accessed in a typical user's interactive map session, we can create a profile of relevant datasets for users accessing that interactive map in the future. This can also inform our model of the types of layers commonly accessed together and therefore are relevant to each other.

Figure 35 outlines the steps necessary to extract GIS layer associations from WMS logs:

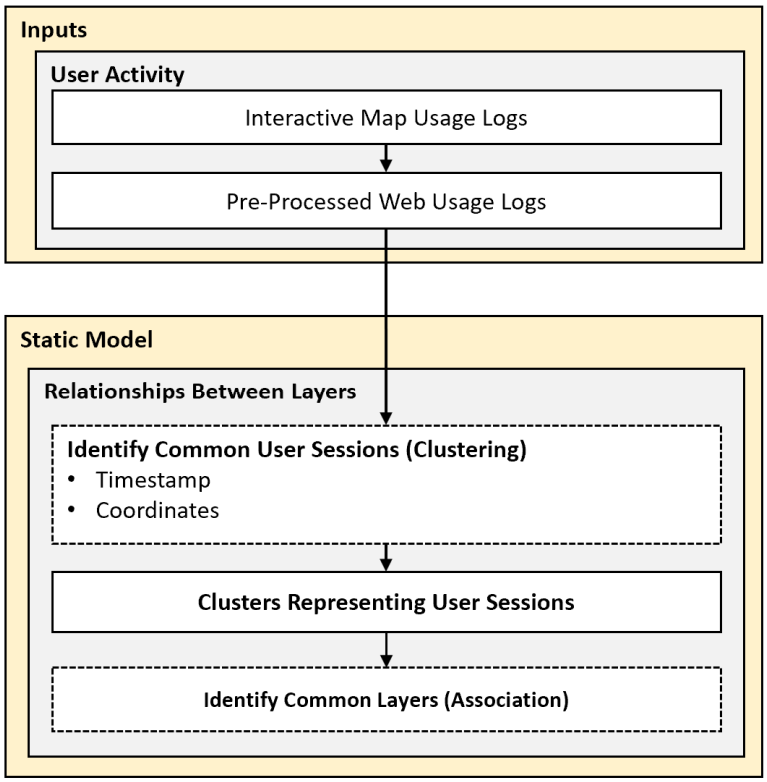


Figure 35: Data Flow Process to Extract Layer relationships from Interactive Map Usage Log Files

Identify Common Users (Clustering) – Given anonymized WMS interactive map usage logs, we need to attempt to identify individual user sessions since we have no identifiable information in the logs themselves.

Specifically, given a set of timestamps and coordinates, we want to identify user sessions as a set of layers accessed in a localized area on the map during a similar time period. For example, a user of an interactive map accesses the parks layer and the property layer between 9:00 and 9:15 and zoomed into the location of one park of interest. However, looking at the logs alone can make it difficult for several reasons:

- If an interactive map application and its corresponding OGC web server are behind a proxy, the web logs might not reveal the actual IP address of the user accessing the interactive map. This makes it difficult to identify sessions.

- If it is desirable to perform analysis on WMS usage while preserving privacy, deriving sessions from web logs without knowing the user's IP address can be helpful.
- If a user with the same IP address used an interactive map in one local area (town A) and then switched to a different local area (town B) we would like to treat these as two separate user contexts. This might be difficult if we based session information from a user's IP address.

Therefore, we try to identify user sessions from the behavior inferred from the WMS logs themselves. We do this by using spatiotemporal clustering techniques on the timestamp and the X and Y coordinates of the centroid of requested map tile. The goal is to identify clusters of activity in localized areas within a reasonable window of time. This will assign a label to each log entry with the identified session.

Spatio-temporal clustering identifies groups of objects based on their spatial and temporal similarity [165]. In our case, one cluster includes WMS log activity that occurs within the same timeframe (a user session) as well as a specific zoom and location the activity occurred in. The result of this clustering is a series of identified sessions with each session containing the list of layers the user accessed in that session.

Identify Common Layers (Association) – Once a set of user sessions has been identified (represented as clusters from the previous section), we identify the layers associated with those clusters. If the same set of layers is commonly associated together then we can establish relationships between those layers, as illustrated in Figure 36.

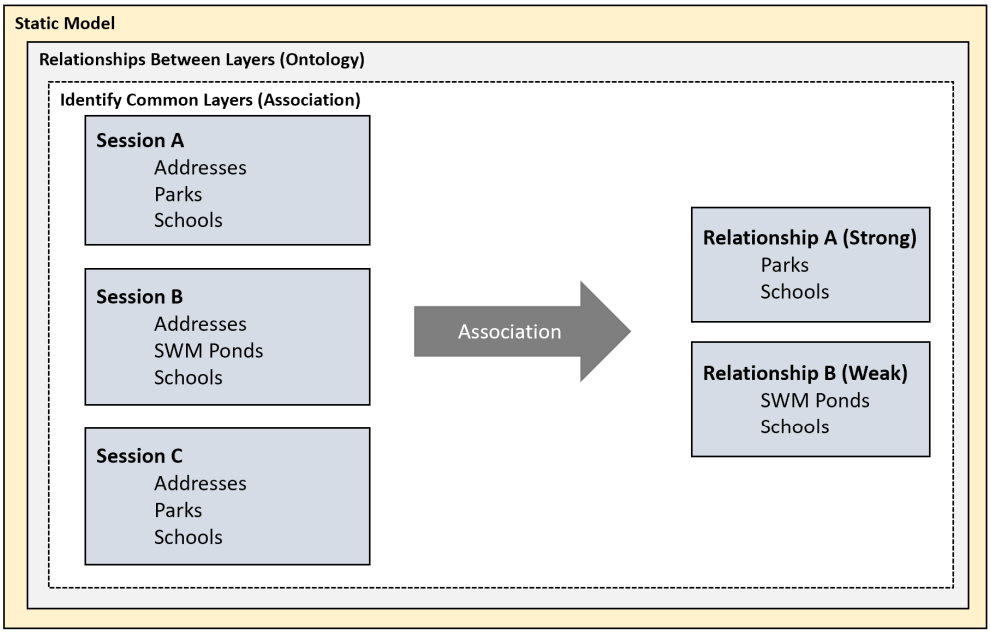


Figure 36: Using Association Rules to Identify Layer Relationships

The result of this association mining are association rules defining the strength between two layers. This provides a listing of layer relationships and a probability value representing the strength of that relationship.

Use Expert Knowledge to Identify Layer Relationship Types

This process provides a series of probabilities representing the strength of relationships between two layers, but it does not tell us the types of relationships between objects in different layers. Specifically, we do not know the topological relationship between two layers as reflected in Sections 3.2.2 and 3.2.3.

While a computer could attempt to automatically determine these relationships based the types of objects in each layer (if one layer is polygons and a second layer is points, we could assume a point-within-polygon relationship between the objects in both layers) this would not be a realistic solution without the approval of expert GIS

professionals and domain professionals who may fine-tune the specific layer relationships based on their own knowledge.

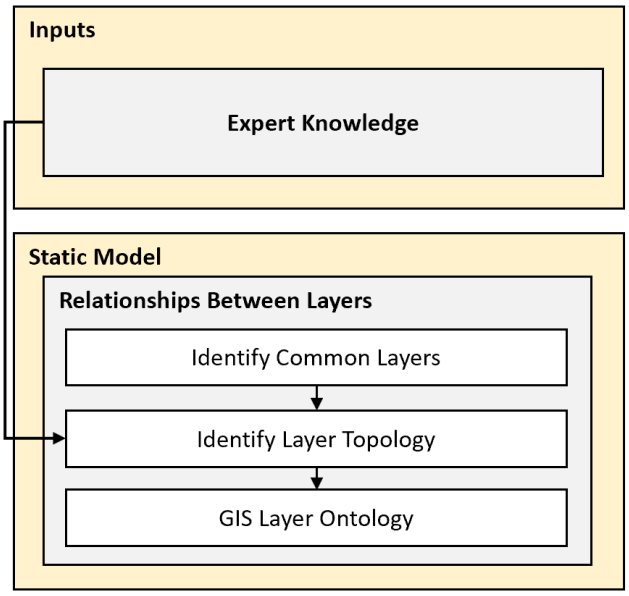


Figure 37: Identifying Layer Topology Using Expert Knowledge

Given a set of layers and the relationships between those layers (the strength represented as a probability and the topological nature of those relationships) we establish a layer ontology.

4.2.2 Relationships between Objects

If a user wants to identify relevant geographic objects in a GIS system, creating a record of all geographic objects in a GIS, the types of relationship between them, and the strength of the relationship between them provides an easy-to-query database of these objects.

This section outlines the process of creating a record of the geographic objects in a GIS and the strength and nature of the relationship between those objects. The result

will be a practical implementation of the mSLN mathematical concept outlined in Section 3.3.

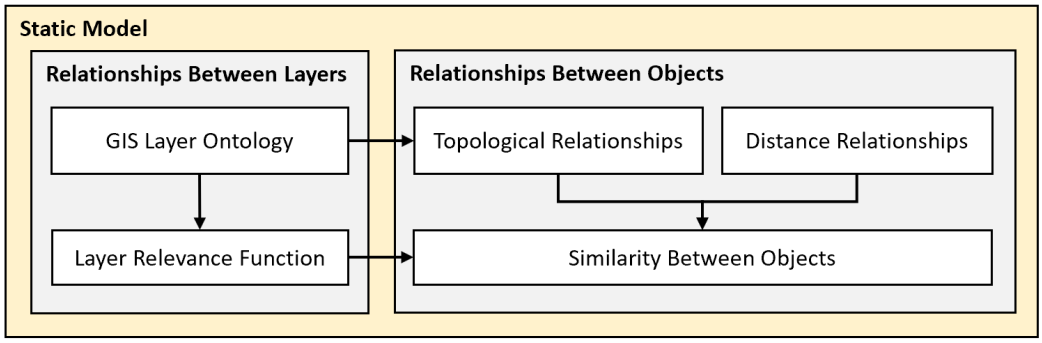


Figure 38: Calculating the similarity between geographic objects.

As shown in Figure 38, there several processing steps necessary in the creation of an mSLN using common database queries, data mining techniques and common geospatial operations. First, we take the topological relationships from the ontology derived in Section 4.1 and translate them into OGC standard topological functions. We then identify calculations of distance between geographic objects in a GIS using those OGC functions.

We combine both the topological relationships and the distance relationships between geographic objects to create a layer relevance function. The output will be a central function where, given two layers in a GIS as inputs, computes the relevance (or strength of relationship) between the two layers.

Once this function is created, we create a lookup table of these relationships that can be easily queried by a user. This lookup table will be a database representation of the ontology representing the relationship between any two layers in the GIS.

Finally, an object similarity measure between any two geographic objects is created that reflects both the layer relevance between the layers containing those objects and the object relationships (distance or topology) between the two objects.

The following sections are based on Microsoft SQL Server code (the testbed will be described in Chapter 5 and a full whitepaper is in the appendix) however the database types will conform to ANSI SQL when possible and will generalize database types otherwise. For example, most enterprise database systems support a geometry database type to store spatial information but the implementation of that functionality and syntax differs between these systems.

Register Topological Relationships Between Layers

The first step is to take the results of the analysis of Section 4.1 and represent them as a database table. We set up two tables:

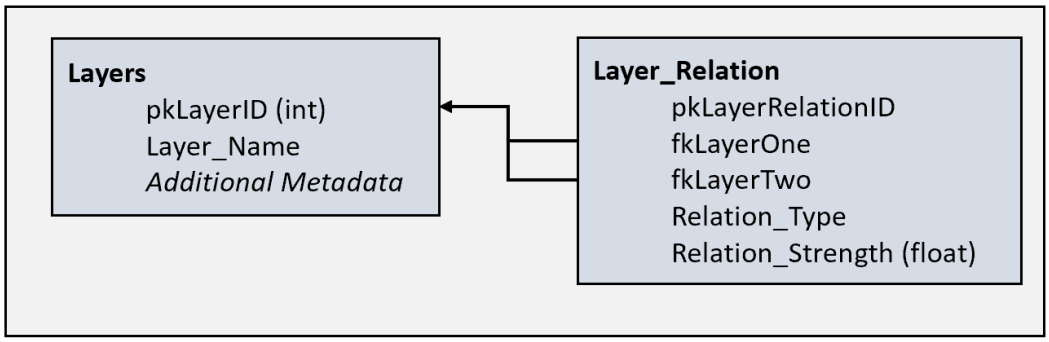


Figure 39: The Layer and Layer_Relation Tables

- The **Layers** table represents every GIS layer in our system.

Layers	
pkLayerID	Layer_Name
1	Parks
2	Park Pavilions

Figure 40: Detail of Layers Table

- The **Layer_Relation** table represents potential relationships between GIS Layers in our system. At a minimum, a record in this table requires two foreign keys identifying the two layers, a field describing the nature of the relationship, and a probability value (a floating-point number) representing the strength of the relationship (or similarity value) calculated in Section 4.1.

Layer_Relation				
pkLayerRelationID	fkLayerOne	fkLayerTwo	Relation_Type	Relation_Strength
1	1	2	Contains (Intersection)	0.8
2	2	1	Within (Intersection)	0.8

Figure 41: Detail of Layer_Relation Table

We represent the same relationship between park and park pavilions twice. If we are looking at a park, then we are interested in all park pavilions within that park, so we use the "contains" relationship. If we are looking at a park pavilion, then we want to identify the park that the pavilion is "within." The Layer_Relation assumes that one layer is the primary layer (containing the object we are referring to) and the second layer contains objects that are related to that primary object. While computationally these are both the intersection of two objects, the semantic meaning of the relationship depends on which geographic object is the starting point (or subject of our search).

We explore the different types of relationships between geographic objects in the next section.

Register Distance Relationships Between Layers

Sections 3.2.2 and 3.2.3 described the different types of geometric relationships between geographic objects from a mathematical perspective. However, this can only produce a calculated value but does not include the semantic nature of these relationships.

In the previous section we outlined that while a park pavilion and a park can have a relationship that is calculated by the intersection of those two objects, this calculation does not fully describe the nature of the relationship between the two objects.

The Layer_Relation table contains a field describing the nature of the relationship between the two layers. Table 7 below describes these relationships, the types of objects that utilize those relationship types, and the OGC standard functions (implemented by most enterprise databases) that would be used to compute these relationships.

Table 7: Object Relationship Types

Relationship_Type	Object Type 1	Object Type 2	OGC Function
Distance	Any	Any	ST_Distance
Intersection (Contains)	Polygon	Point or Line	ST_Contains
Intersection (Within)	Point or Line	Polygon	ST_Within
Intersection (Generic)	Any	Any	ST_Intersects
Overlap	Line or Polygon	Line or Polygon	ST_Overlaps
Adjacency	Polygon	Polygon	ST_Touches
Feature	Any	Any	SQL Join

It is important to delineate the relationship types between different layers and the objects within them because a calculation alone would not describe the nature of the relationship between the two objects.

Relationships Based on Features

While the focus of this work is on relationships between objects based on their geographic properties, sometimes features in GIS layers are related based on the properties of those geographic objects. For example, two locations might be similar because they share the same zoning code, even if the two locations are not geographically close. Shopping centers are similar to each other despite being in different locations on a map.

We add an additional relationship type to Table 7 to represent these relationships.

Relationships to Non-Geographic Objects

We want to account for relationships between the geographic objects represented in our network and objects that are not geographic. For example, if a document contains location information (like an address) in its text we should be able to link it to other documents based on that location information.

Create Layer Relevance Function

Once relationships are determined we can calculate similarity measures between those objects. This involves taking two input layers and calculating the similarity between those two layers based on the calculations from Section 4.1.

Given an input layer, the result of the layer similarity function would be a sorted list the most similar layers based on the similarity scores in those layers and how those

layers are related to the input layer. For example, if the input layer is “parks” then one result would be “park pavilions” and “contains” with a relevance score.

Creating a Table of Ontological Relationships

Given a series of layers, the types of relationships possible between those layers and the strength of the relationship between those layers produces an ontology in tabular format, Table_Relation.

Create Similarity Measures

Once the two tables reflecting the layers and relationships between those layers (both the relationship type and the strength of the relationship) we create two tables to represent individual geographic objects and the relationship between those objects. These calculations reflect the layers containing those relationships and the topological measures outlined in Sections 3.2.2 and 3.2.3.

Two new tables are created to represent geographic objects (features) and the relationships between them.

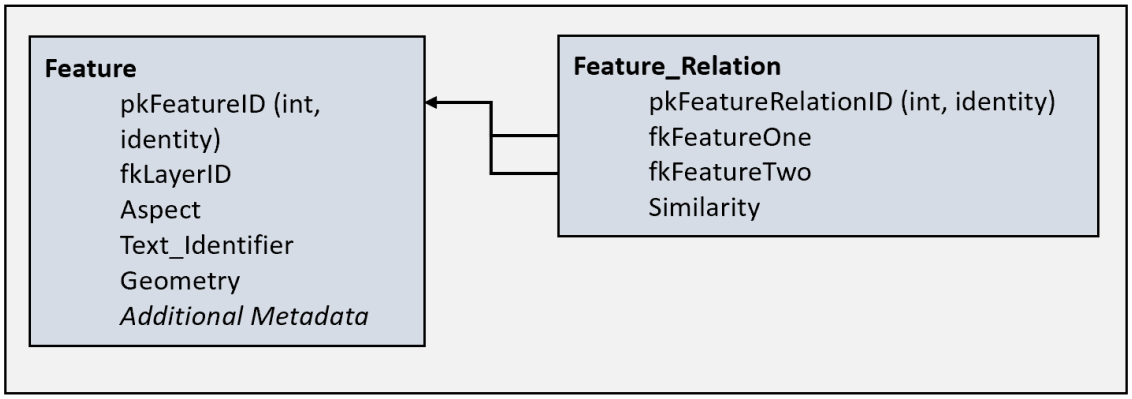


Figure 42: Feature and Feature Relation Tables

- The **Feature** table includes the foreign key representing the layer (from the Layers table) containing the feature. A feature can be represented multiple

times in this table, and the aspect represents which part of the feature we want to base the identity of the feature on. A text identifier allows a keyword search of these features, and the geometry is extracted from the source data, stored and indexed so that the OGR functions for the distance and topological relationships can be run on them.

- The **Feature_Relation** table includes foreign keys identifying the two features involved in the relationship and a similarity score between the two.

The Feature table represents an index table similar to ones used in a simple star schema data warehouse. In a production environment, both the Feature and Feature_Relation tables need to be periodically refreshed as records are added, updated and deleted from the source databases.

4.2.3 Static Model of Geographic Relevance

When a user initiates a query to identify relevant geographic objects the time to execute that query will be faster (close to real-time) if the relevance between those geographic objects has been calculated ahead of time as a static component.

To complete the static model, we combine these mathematically defined relationships (both the ontology for relationships between layers and the similarity measurements for relationships within a layer) into a database structure that will form the foundation for user queries and analysis.

When a user wants to identify relevant geographic objects, they will query a database containing information abouts layers, the geographic objects in those layers, and the relevance between those objects. In the previous section four tables were created to store this information: Layer, Layer_Relation, Feature, Feature_Relation.

This section describes the process of creating a database representation of the mSLN that will be queried by a user trying to find relevant geographic objects based on a query input. A more detailed white paper explaining the table structures and stored procedures involved in this process can be found in the appendix.

First, using the tables defined in Section 4.2.2, a basic data warehouse is created. This data warehouse is a database representation of the supra-adjacency matrix described in Section 3.3.3. Second, the reasoning rules outlined in Sections 3.3.4 and 3.3.5 are translated into database statements and be used to create a new table in our database representing the relationships between geospatial objects in the GIS with the reasoning rules applied. The result of these operations will be a static model in a database that corresponds to the mSLN outlined in Section 3.3.

A Geospatial Data Warehouse

The creation of a data warehouse typically involves copying critical tables from source databases for two purposes. First, by copying relevant data from operational databases to the warehouse, any queries to the data warehouse will not interrupt or slow down any transactions performed on the source databases. Second, we hope to organize the data warehouse such that the reasoning rules can be applied efficiently.

Since this data warehouse is part of the static model it is assumed that any data copied from diverse sources will be copied on a periodic basis, for example, a nightly process to copy the source GIS tables to a central location and then to perform mathematical operations on those tables to create the mSLN.

The process outlined below can be adapted for a virtual warehouse with minor tweaks. However, we assume that the relevant source tables are copied on a periodic basis.

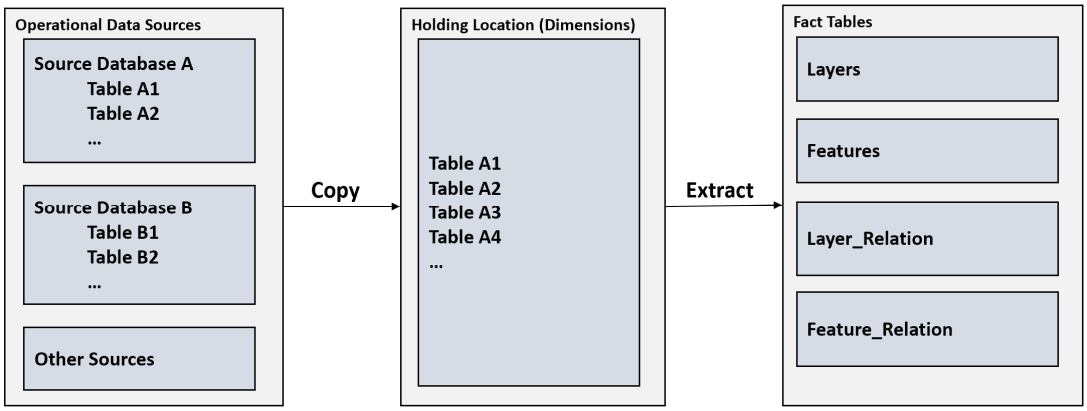


Figure 43: Source data is copied into a data warehouse and then features are extracted into the tables described in Sections 4.2.1 and 4.2.3.

The process of creating the data warehouse involves some preparation work in the source databases and the creation of various server jobs that will copy the data from those source databases. The steps are outlined below:

- The source databases are investigated to identify relevant layers. While some GIS databases create one table per layer and can be easily copied other databases have structures that are optimal for their transactional nature. The relevant data from those databases need to be transformed into a format where one table represents one layer. Typically, this is done by creating a view in the database to represent that one layer and then copying that.

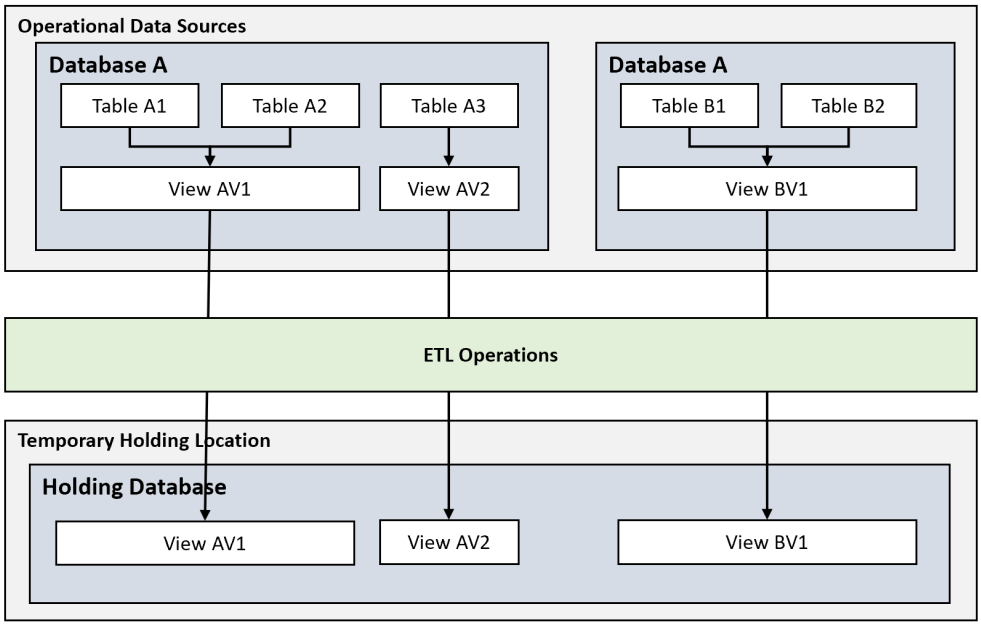


Figure 44: Views are created in the source databases and then ETL operations copy these views to a holding table.

- A job is run on a periodic basis (at least nightly) to copy these tables to a holding location. For our purposes, this holding table is a separate database in the same DB instance, but it can be included in the main table. This is shown in Figure 44.

Extract Features

The next step is to populate our operational database with the tables stored in the holding location. This process is a series of database stored procedures that combines the data from the holding locations with the outputs from previous steps. An outline of this process is shown in Figure 45.

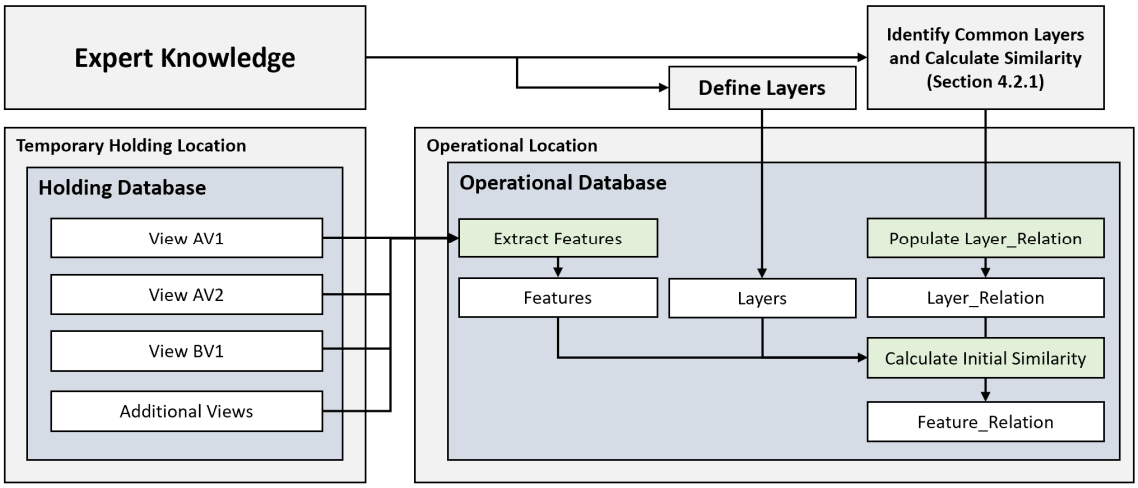


Figure 45: Populating base tables from data in holding location.

Two outputs from Section 4.2.1 are used as inputs. First, experts need to define the tables in the holding location so that the system knows which tables to extract features from. The **Layers** column is manually populated with these tables and any relevant metadata associated with those layers are entered.

Second, layer relationship information (including the type of topological relationship and a similarity score defining the strength of relationship) are imported into the **Layer_Relation** table. This is a result of the classification and association operations outlined in Section 4.2.1 (using prior interactive map activity to determine layer relevance) as well as any manually defined relationships entered in by experts.

Once the layers have been populated into a database, a stored procedure, Extract_Features queries the fields in the holding database and places them in the **Features** table. As a data warehouse, the feature table acts as a star schema's fact table.

This job also extracts relevant index information (both searchable fields and the geometries of the features in the tables) and copies them to the Feature table. Many database systems have specialized update functions that identify the records that have

changed between the data warehouse and the source tables so that only those records are updated or added to the data warehouse, minimizing the number of write and re-index operations on those tables.

The next step is to calculate the relevance between all these features and to use the results of those calculations to populate the **Feature_Relation** table.

Generating Initial Supra-Adjacency Matrix

After populating the Layer, Layer_Relation and Feature tables, the similarity calculations outlined in Section 4.2.2 need to be applied to the records in that table. The goal of these calculations is to produce the **Feature_Relation** table that reflects the relevance of each geospatial object to another. If this table is indexed, then user queries to find relevant geographic objects can run considerably faster for the user.

This process is outlined in Figure 46 below.

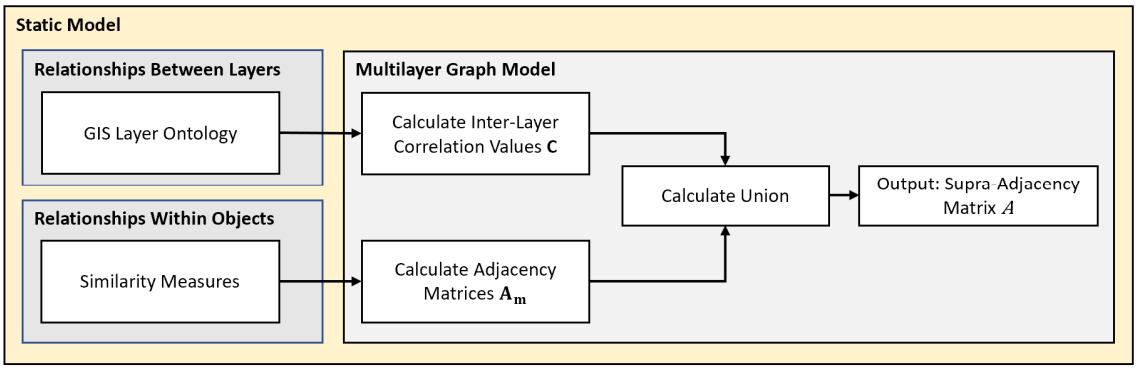


Figure 46: Generating supra-adjacency matrix.

Calculating and representing the supra-adjacency matrix using the database sources outlined earlier requires the following steps:

- The layer association probabilities calculated in Section 4.2.1 are used to populate the Layer_Relation table. This corresponds to the inter-layer correlation values C as described in Section 3.3.2.

- When new layers are added to the system or when new objects are added to the existing layers in the system (for example, if a new fire station is built) then the similarity calculations for those new features (comparing them to all existing features in the network). This similarity score is a product of the similarity calculations between features described in Section 4.2.2 and the layer similarity values described in Section 4.2.1.

The result of these two steps is a listing of all features in the GIS with an initial similarity score calculated between them. This table, Feature_Relation is a database representation of the supra-adjacency matrix.

Implementing Reasoning Rules

Once the Feature_Relation table has been created or updated, a final operation needs to be run to implement the reasoning rules described in Sections 3.3.4 and 3.3.5. Given an initial listing of all features in the GIS and their similarity based on both the layer similarity between them and the topology rules defined by the user we calculate the relevance between any two objects in the system. An outline of this process is shown in Figure 47.

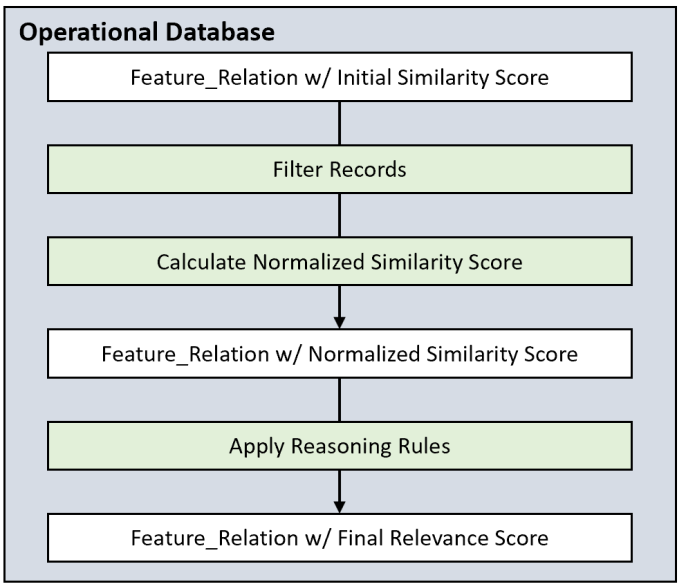


Figure 47: Applying Reasoning Rules

All these calculations happen in the Feature_Relation table and all the procedures described are update operations on fields in that table. These steps reflect the calculations outlined in Section 3.3.5. More details on the SQL implementation of these calculations can be found in the Appendix.

First, an initial threshold is determined so that only features with a similarity score above this threshold are included in the calculations. This threshold is a judgment made by experts. The purpose is to prevent the table from being populated with sparse records that reflect weak relationships between objects. Additionally, the calculations involved in applying the reasoning rules are computationally intense and the fewer records involved in those calculations, the better.

Next, the similarity scores are normalized so that the sum of probabilities between one feature and all other features in the domain sum to 1. The result of these calculations is stored in the Feature_Relation table's **Similarity_Normalized** column.

Finally, an algorithm to replicate the matrix calculations outlined in Algorithm 2C is run on the normalized records. The result of these calculations is stored in the Feature_Relation table's **Relevance_Score** column.

The result of this section is a database with four core tables populated with data: Layers representing every layer of interest in the system, Layer_Relation representing the type of topological relationship and similarity between any two layers of interest in the system, Feature representing all the geospatial objects in the domain, and Feature_Relation representing the relevance between any two objects in the domain.

Querying this system primarily involves the Feature_Relation table and this will be described in the next section.

4.3 Dynamic Query Interface

When a user wants to query the model, they will provide an input (such as their GPS location or a point of interest) and the system will provide layers and features relevant to that input. This section describes how these queries are handled by the system.

4.3.1 Identify Relevant Layers

We currently examine spatial clusters to define a relationship between two layers, however we do not consider the context of the user's location of interest. For example, a Floodplain layer might only be relevant to a user near a river (and not so relevant farther away from a river), so a future modification to our system should take into consideration the location in a user's query.

The primary application would involve mobile devices (with GPS) accessing a GIS database. Given an input GPS location, any layers that are not relevant to the user's location should automatically filtered out. Only layers that consist of specific boundaries should be considered for this treatment. A summary of the process is shown in Figure 48.

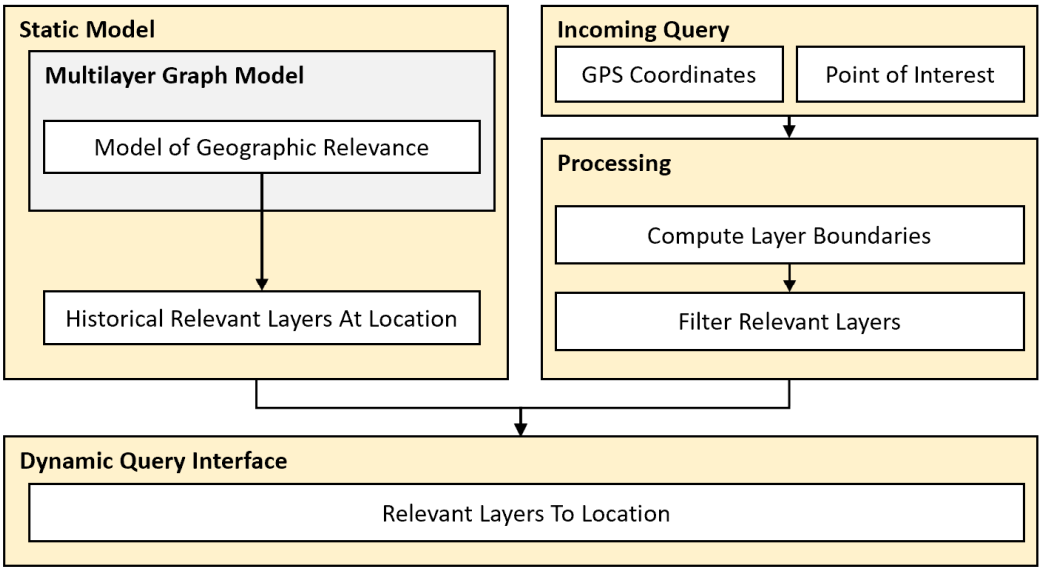


Figure 48: Deriving contextual layer relevance

This process involves the following components and steps from our model:

- An incoming query involves either the user's GPS coordinates or the coordinates of a point of interest specified by the user.
- The layer boundaries are queried to filter out any layers that could not possibly be related to the point of interest. For example, layers containing information about bodies of water would not be relevant if the user's location is not close or in one of those bodies of water. The result of this query is a list of relevant layers.

- The GIS layer ontology is queried to identify relevant layers to the layers remaining after this filter operation.
- We synthesize the two results to identify the most relevant layers to the user's position. From there we identify the relevant objects within those layers.

An outline of this workflow from the perspective of the database described in Section 4.2.3 is shown in Figure 49.

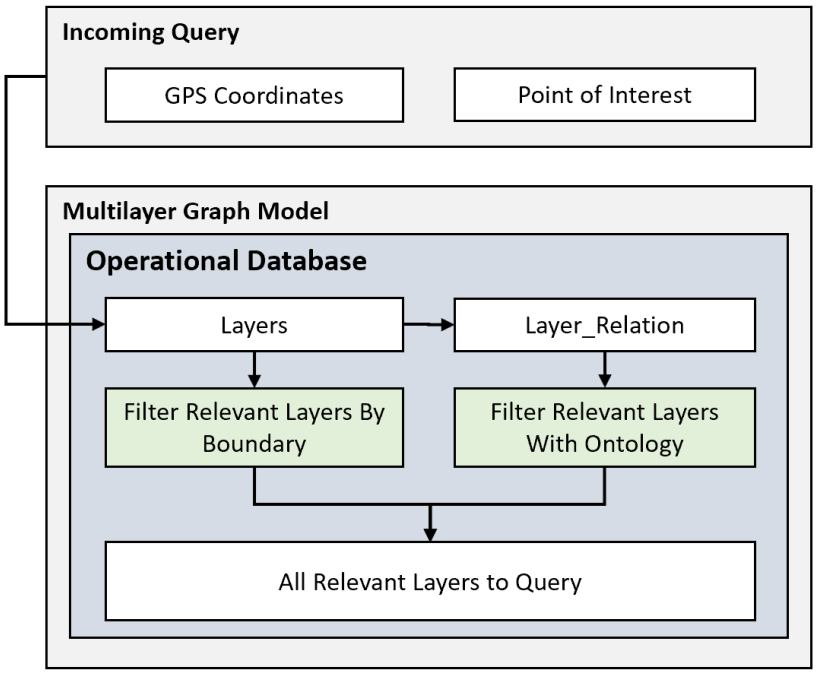


Figure 49: Querying relevant layers from the database.

Identifying layer boundaries helps filter out any unnecessary layers from future processing steps and can help to speed up the identification of relevant layers from both a computing time perspective and from the user's perspective. There are also situations in GIS where a user is interested in relevant layers only and do not directly need to query features and this workflow can accommodate that need.

The next section outlines the process for facilitating user queries to the system.

4.3.2 Identify Relevant Features

When a user queries the model to identify relevant features (for example, anything relevant to their current GPS location) the user provides some input (the GPS coordinates or a selected point or feature of interest) and then a series of SQL stored procedures are executed to generate a list of relevant geospatial features to the user. The process of querying relevant features is outlined in Figure 50.

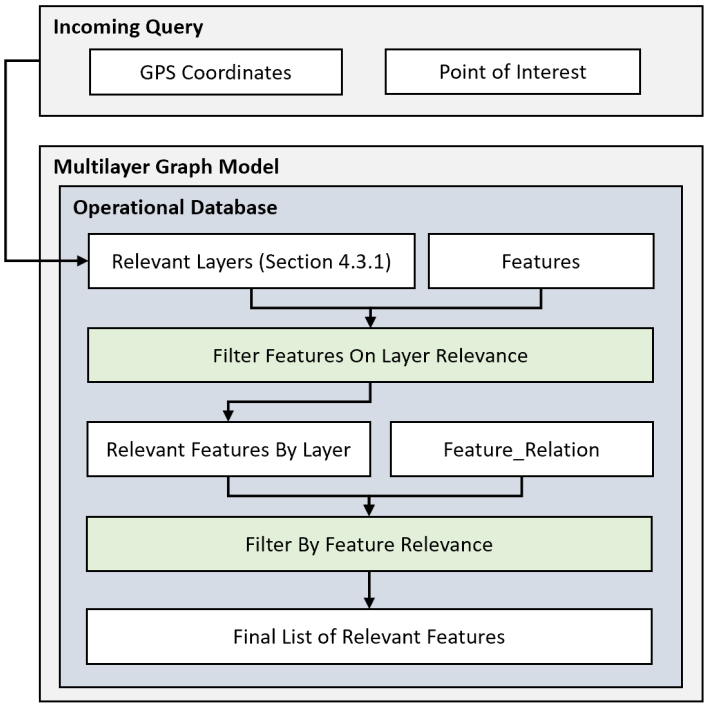


Figure 50: Querying Relevant Features

The pre-processing steps outlined in Section 4.2 to build the model reduces the process of querying this model to a series of simple SQL queries to filter records from the Feature_Relation table. Every step is designed to progressively filter out geospatial objects that the model finds to not be relevant to the user. This architecture is designed to maximize performance.

First, the relevant layers identified in Section 4.3.1 are used to filter on the Features table so that only Features from those relevant layers shows up in the results. Only these features will be included in the next step.

Second, the Feature_Relation table is queried to identify all features relevant to the user's input. Any features filtered out in the previous step are not included in this query. This produces a list of relevant features.

At this point the resulting dataset can be sorted by relevance and only a subset of records can be returned to the user (the TOP 100 results, for example) if an application using this model needs it. For example, web-based applications prefer to minimize the amount of data transferred to the client to maximize browser performance.

Summary

Chapter 4 outlines a process to implement the mathematical model described in Chapter 3 using standard GIS and database technologies and standards. This includes creating a model of geospatial relevant, implementing it in a database system (SAM-GIS), and creating methods (SQL stored procedures) to query this model. A more detailed explanation of the database structure that represents the model and the stored procedures used to query the model is provided in an appendix.

Chapter 5: Experimental Evaluation

The aim of this research is to create a general model of geographic relevance to a given query that can extend typical results produced by existing GIS databases. This approach generates expanded search results relevant to a user and their context.

This chapter outlines the experimental evaluation of the mathematical model presented in Chapter 3 with the system architecture and implementation (SAM-GIS) described in Chapter 4. First, the primary source data for the model is described. Second, intermediate results related to the construction of the model is discussed. Third, evaluations of the model are documented. Finally, case studies showing practical applications of this model will be outlined.

5.1 Source Data

To evaluate the practicality and the usefulness of these measures, we examine sample queries on a prototype system mimicking a typical local government GIS and database environment. SAM-GIS runs on Microsoft SQL Server and the geometries are stored in SQL Server's native format. SAM-GIS uses the database's built-in geometry calculations. The datasets used from Howard County, Maryland's online GIS portal at https://data.howardcountymd.gov, their open data portal at https://opendata.howardcountymd.gov, and their County Council site at https://cc.howardcountymd.gov.

Howard County provides approximately 150 geospatial datasets to the public through a data portal (https://data.howardcountymd.gov) in multiple formats, as shown in Figure 51. Ultimately, 120 of these datasets were imported into a holding database in SQL Server using the GDAL/OGR (www.gdal.org) data translation library.

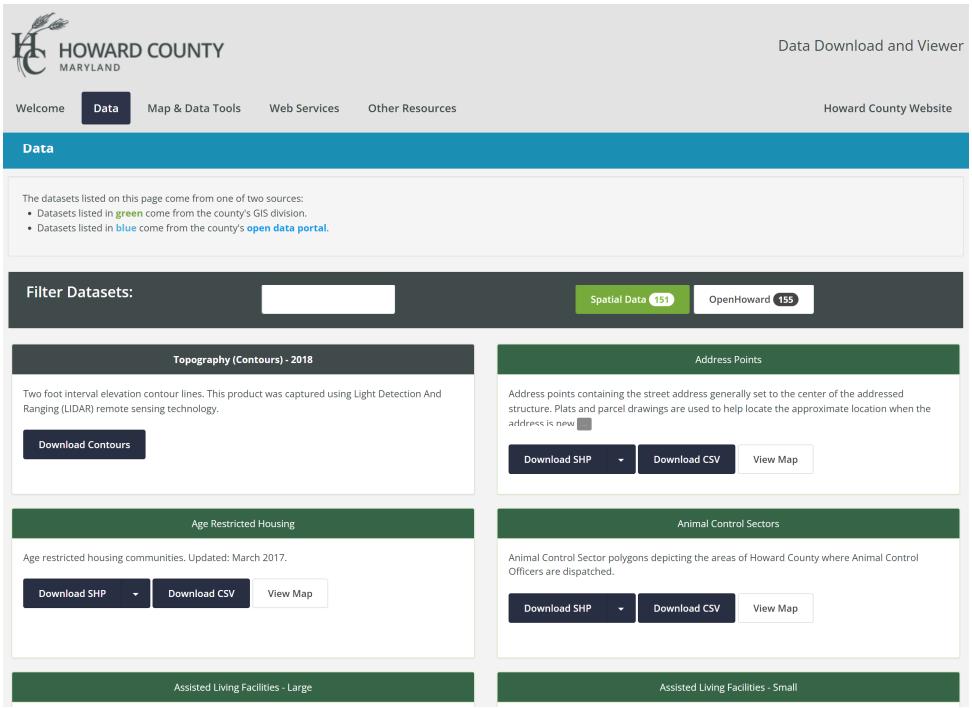
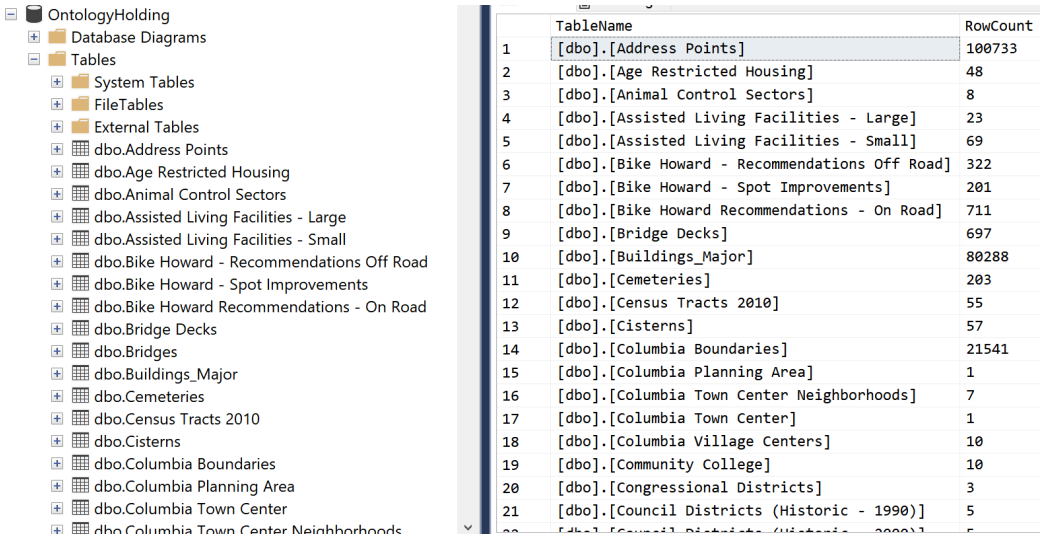


Figure 51: Howard County's open data portal (https://data.howardcountymd.gov)

Once loaded into the holding table, the datasets appear as individual database tables, each representing one GIS layer, as shown in Figure 52.



	TableName	RowCount
1	[dbo].[Address Points]	100733
2	[dbo].[Age Restricted Housing]	48
3	[dbo].[Animal Control Sectors]	8
4	[dbo].[Assisted Living Facilities - Large]	23
5	[dbo].[Assisted Living Facilities - Small]	69
6	[dbo].[Bike Howard - Recommendations Off Road]	322
7	[dbo].[Bike Howard - Spot Improvements]	201
8	[dbo].[Bike Howard Recommendations - On Road]	711
9	[dbo].[Bridge Decks]	697
10	[dbo].[Buildings_Major]	80288
11	[dbo].[Cemeteries]	203
12	[dbo].[Census Tracts 2010]	55
13	[dbo].[Cisterns]	57
14	[dbo].[Columbia Boundaries]	21541
15	[dbo].[Columbia Planning Area]	1
16	[dbo].[Columbia Town Center Neighborhoods]	7
17	[dbo].[Columbia Town Center]	1
18	[dbo].[Columbia Village Centers]	10
19	[dbo].[Community College]	10
20	[dbo].[Congressional Districts]	3
21	[dbo].[Council Districts (Historic - 1990)]	5

Figure 52: Datasets loaded into Holding Table with Row Counts

The 120 layers represent 667,625 geographic objects (table rows) imported into our system.

A separate database, "Ontology," contains the data structure necessary for our experiments. A detailed discussion of the structure of this database and related functions and stored procedures is covered in the appendix.

5.2 Model Construction

This section uses the source data outlined in Section 5.1, describes a practical implementation of the architecture described in Chapter 4 and reports observations on the intermediate results from the steps involved in the construction of our model.

5.2.1 Automated Ontology Construction from Interactive Map Usage

There are two methods we can use to describe the relationships between GIS layers: Observing user activity and using expert knowledge. In this section we outline the process of using online activity logs to infer the strength of relationship between GIS layers.

Identify WMS user Sessions (Clustering)

The layers that users commonly access when using GIS resources such as online interactive maps can help define the ontological relationships between layers and the strength of those relationships.

Section 4.2.1 described a process to take WMS logs and apply a clustering algorithm to identify user sessions. These user sessions allow us to understand what layers were commonly accessed by an individual user during a web interactive map

session. When a user chooses to activate a series of layers during an interactive map session then they infer a meaningful relationship to those layers.

The data provided by Howard County anonymizes the origin (normally identified by IP address) therefore user sessions need to be estimated from the time of a user's activity and the locations the user is searching for.

WMS logs were provided for April 11th, 2018, and contain 181,734 log entries. After only including entries relevant to WMS map tile requests and dropping any tile requests for zoom levels higher than 16 (we do not wish to include "global" map tile requests), we reduce the number of log entries to 134,476.

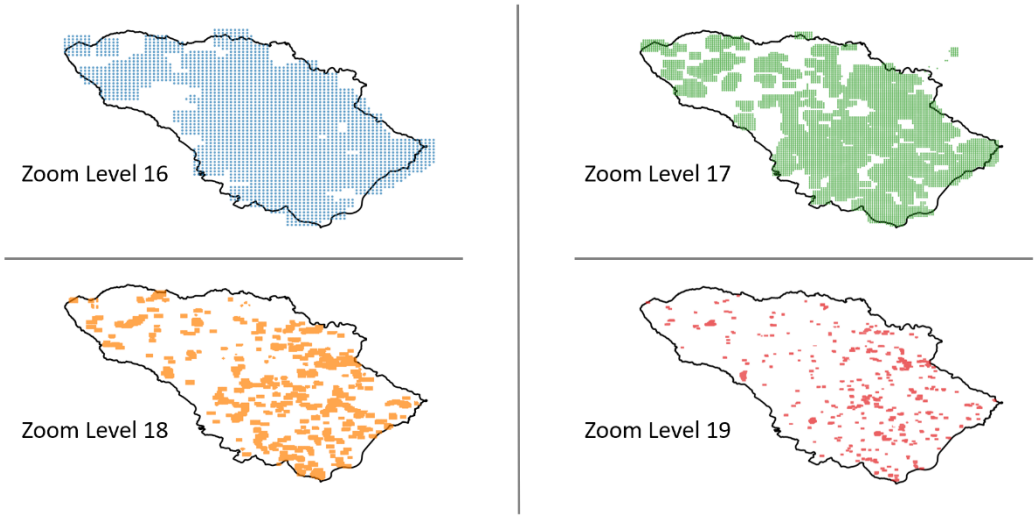


Figure 53: One day of WMS tile requests at various zoom levels

Maps displaying the WMS activity for that day are displayed in Figure 53. As the user zooms in to an area of interest, the tiles requested become more localized. Zoom level 16 shows a single cluster in light blue, while zoom level 17 displays several clusters in green, and so on. Since we are looking for local clusters of activity that we

can identify as user sessions it makes more sense to focus on zoom levels 18 and 19 to find these sessions.

We use the ST-DBSCAN [166] algorithm to identify clusters across time and space. We want to distinguish between distinct user sessions and if we ignore the time element in our analysis then a cluster could include two different users who zoomed into the same neighborhood at two separate times of the day. For example, a user could zoom into the "Sanctuary" neighborhood at 8:30 a.m. and load the "floodplain" and "hydric soils" layer and a different user could zoom into the same neighborhood at 4:30 p.m. and load the "historic district" and "historic sites" layers. These users loaded layers that do not have a lot in common and if we did not take time into account then our system would assume a relationship between "floodplain" and "historic sites" that is stronger than it should be.

We use the Python implementation of this algorithm at (https://github.com/eubr-bigsea/py-st-dbscan). This is an extension of the DBSCAN algorithm which takes two parameters. The first (ε_1) is a distance along the two-dimensional geographic plane. The second (ε_2) is an interval (temporal distance) along the temporal dimension. With these two parameters we can find clusters of WMS activity within a specific geographic area and distance (interval) of time.

ε_1 = spatial threshold (meters)	ε_2 = temporal threshold (seconds)	# of clusters
1000	900	48
1000	1800	49
1000	**3600**	**52**
1000	7200	45
2000	900	49
2000	1800	46
2000	3600	46
2000	7200	27

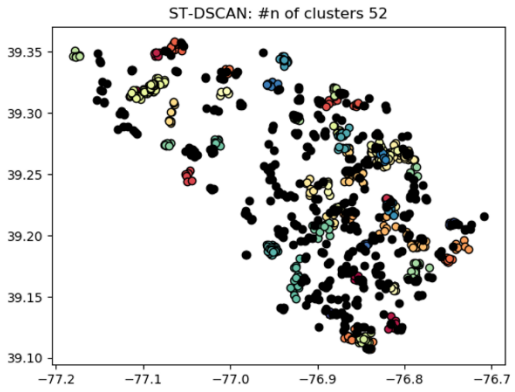


Figure 54: Exploring different ST-DBSCAN Parameters and the resulting clusters.

In Figure 54 we explore different values of ε_1 and ε_2. These values are reasonable expectations of what would define a user session (the cluster we are looking for). In the figure, a spatial threshold of 1000 meters and a temporal threshold of 3600 seconds (one hour) tries to form clusters of WMS activity under the assumption that any localized interactive map activity within the same hour should represent a session. For this example (April 11th) there were 52 clusters of WMS user activity generated.

We will examine two of these clusters. The first, shown in Figure 55 represents a user session that focused on an area in Clarksville, Maryland. The dots represent the centroids of the map tiles loaded in this user session. Common layers on the map include the "property" layer in blue and the "street centerline" layer in purple. More distinctive layers (specific to this session) include "growth tiers" (designations by the government as to what kind of development can happen on designated plots of land) in red, "hydric soils" (soils that were created through a process of erosion) in brown, and the "floodplain" (areas prone to flooding) in green. While we do not know this user's

exact intention in loading these layers, we know that the three special layers had a connection valuable to the user's context.

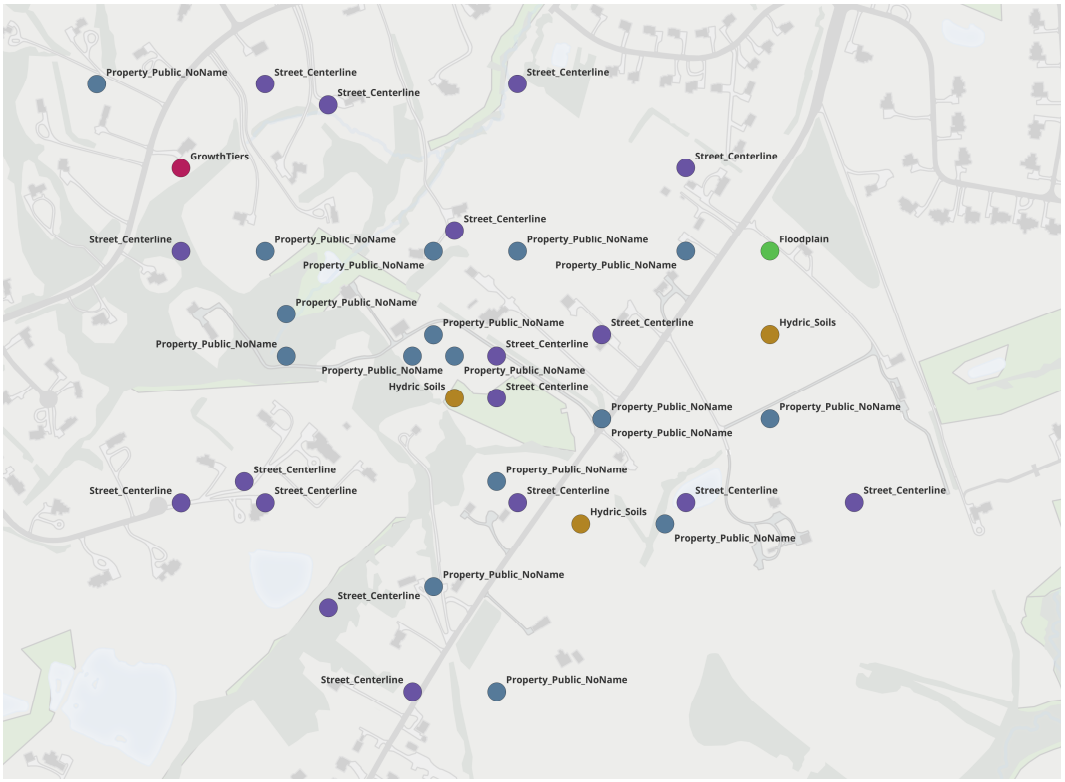


Figure 55: A sample cluster in Clarksville, MD representing a user session.

The second cluster, as shown in Figure 56, represents a user session where they zoomed into Elkridge, Maryland. This session includes soils in magenta, directional signs in green, addresses in blue, and designated places (like the growth tiers in the previous example) and the floodplain. While some of the layers are similar to the first example, the user in this second example was interested in soils and designated places at the same time.

Every session has a distinct "basket" of layers that were loaded by the user and in the next section we use association rule mining to identify those commonly associated layers.

Figure 56: A sample cluster in Elkridge, MD representing a user session.

Identify Commonly Requested Layers (Association)

Given clusters of localized WMS activity, each cluster containing a "basket" of layers, we perform an association rule analysis. While the FP-Growth algorithm was ultimately chosen for future steps, it is important to explore some intermediate attempts at association with the Apriori algorithm to illustrate why this decision was made and to provide some insight into the nature of how users access the layers in a GIS interactive map.

Some sample baskets from our dataset are shown in Table 8 including the two baskets representing the two clusters shown in the previous section.

Table 8: Sample "Baskets" of Layers for Each Session

Session ID	Basket of Layers
2	Address Points, Historic Sites, Street Centerline
11 (Figure 55)	Property Lines, Streets, Growth Tiers, Hydric Soils, Floodplain
23 (Figure 56)	Soils, Directional Signs, Address Points, Designated Places

We do not want the association rules generated to reflect layers that are accessed in most interactive map sessions therefore we prioritize lift in these calculations. Running the Apriori algorithm in Weka (https://www.cs.waikato.ac.nz/ml/weka/) with a minimum lift of 1.1 produces the results shown in Table 9. While the algorithm was run to show the top 50 rules, only the top 10 are shown.

Table 9: Running Apriori Algorithm on User Session Clusters

Rule	Lift	Confidence
Address Points => Scanned Drawings, Zoning	2.03	0.63
Zoning => Address Points, Scanned Drawings	2.03	0.83
Property, Scanned Drawings => Zoning	2.03	0.63
Zoning => Property, Scanned Drawings	2.03	0.83
Scanned Drawings => Zoning	1.99	0.61
Zoning => Scanned Drawings	1.99	0.92
Scanned Drawings => Address Points, Zoning	1.97	0.56
Scanned Drawings => Property, Zoning	1.97	0.56
Address Points, Zoning => Scanned Drawings	1.97	0.91
Property, Zoning => Scanned Drawings	1.97	0.91

There are two issues illustrated in this example. First, even using the lift metric to rank the results produces results that are dominated by the same algorithms. In typical GIS interactive map usage, most users will choose to activate the property layer (to find their own property or to find information on the property owners in the area of interest) or the address points (a common method of identifying locations on a map). Other than Zoning, the other layers shown in Table 9 are essentially "utility" layers common to most interactive map usage. Ranking by lift, the only other non-utility layer included in a rule is "Floodplain" which only shows up in the 47th ranked rule.

The second issue is that we are primarily interested in one-to-one relationships between layers so that we can assign a single probability to those layers and in Weka, the Apriori algorithm does not provide that choice.

To solve both issues a second attempt at finding associations between layers in the user session clusters was done using the FP-Growth algorithm. In Weka, FP-Growth provides an option to only return one-to-one relationships. Additionally, FP-Growth is considered a more modern and performant algorithm over Apriori. In this run, popular layers such as address points and property lines are removed. This resulted in 34 rules, of which the top 10 are shown in Table 10.

Table 10: Result of FP-Growth algorithm on session clusters without "utility" layers

Rule	Lift	Confidence
Land Use => Designated Places	3.9	0.5
Designated Places => Land Use	3.9	1.0
Streams => Floodplain	3.25	0.83
Floodplain => Streams	3.25	0.5
Floodplain => Designated Places	3.12	0.4
Designated Places => Floodplain	3.12	0.8
DNR Open Space => Land Use	3.03	0.78
Land Use => DNR Open Space	3.03	0.7
Zoning => Streams	2.71	0.42
Streams => Zoning	2.71	0.83

The top ten results show mirrored relationships between layers. For example, the top two entries both show a relationship between land use and designated places. The distinction is that the confidence of the second rule is higher, mostly likely because land use is a more popular layer, and it is more likely that a user who has loaded designated places would want to also load land use rather than the other way around (since land use is a more general-purpose layer).

Ultimately the result of both the clustering (to identify distinct user sessions) and association is to identify layers that are commonly associated with each other during the utilization of GIS systems (interactive map usage in this case). There is a question whether to use lift or confidence as the metric for the ω values between layers that is

used in our model. We choose to use confidence since it is already a probability value and since it reflects a bidirectional relationship between any two layers.

Summary of Model Construction

There are two intermediate results worth discussing as they relate to creating a mathematical model that reflects how GIS users work. These results were developed mathematically in Section 3.2.

Most GIS activity is localized. A user zooms into a specific location, typically at a neighborhood level and then loads the relevant layers that they need. This was demonstrated when attempts to cluster the WMS logs did not produce distinct clusters at higher zoom levels. This is not to say that there is no GIS activity at higher zoom levels (for example, someone wanting to view legislative districts might want to do so at a higher zoom level) but that most usage of these systems focuses on local activity.

Some layers are more commonly viewed than others and this sometimes leads to a unidirectional path to these common layers. Our initial association rule mining results were initially dominated by utility layers such as address points and the property layer. Even after removing those layers from the analysis, some association rules had a stronger confidence in one direction. For example, people who load designated places are more likely to load the land use layer than the other way around.

5.2.2 Using Expert Knowledge to Identify Relationships

Other than the automated generation of these relationships described in the previous section, a subject-matter expert (SME) can arbitrarily define relationship relationships between layers. This is done by converting the needs of a GIS application into a series

of layers and associated probability values that are entered into the "context profiles" table described in the appendix.

5.2.3 Evaluating Adjusted Distance Measures

As explained in Section 3.2.3, what is considered a close geographic object depends on the distribution of the points on that layer. A nearby property or a nearby fire hydrant might be a couple hundred feet from a user's location of interest but a nearby park can be 1-2 miles from that user. This needs to be reflected in our model and it is done by approximating the typical distance between objects in any given layer.

As explained in the Appendix (Section A.2.1) this is conducted by taking several samples of features from a given layer, identifying the 100 closest features from that layer to those sample features and then calculating the average and standard deviation of those calculated distances. Based on Section 3.2.3, we use this to calculate an adjusted distance with which to evaluate closeness.

Table 11 shows a sampling of layers in Howard County and the adjusted mean and standard deviation for those layers.

Table 11: Average distances for sample layers (in feet)

	Mean Distance (μ)	Standard Deviation (σ)	Max distance (μ*3σ)
Addresses	941 ft.	579 ft.	1,699 ft.
Parks	5,079 ft.	3,499 ft.	15,576 ft.
Cemeteries	9,441 ft.	4,453 ft.	22,800 ft.

Based on these distances we can limit how much we expand the search for any given query. We can also supply an intermediate result:

Near things are more related to each other than far things, but what is considered "near" depends on the type of thing. Tobler's law was defined before the advent of modern GIS systems that group geographic objects into layers. This updated definition attempts to contextualize it and verify it mathematically.

5.3 Sample Queries

This section shows the utility of SAM-GIS starting with some sample queries. Then it evaluates the effect specific parameters have on those queries.

While most queries of local government GIS systems such as Howard County's revolve around residential interests, most of these queries will originate from the Thomas Isaac Log Cabin (https://visitoldellicottcity.com/item/thomas-isaac-log-cabin/) at 8394 Main Street, Ellicott City, Maryland.

5.3.1 Historic Sites Example

We are interested in historic features that are near the Thomas Isaac Log Cabin. This can include historic sites as inventoried by Howard County, the two historic districts within the County, and cemeteries located within the County.

In a typical GIS application a user would use a dedicated Historic Sites application (such as Howard County's at https://data.howardcountymd.gov/InteractiveMap.html?Workspace=HistoricSitesViewer) to determine the closest historic site to their location. We can find the Thomas

Isaac Log Cabin is in the “Historic Sites” layer and is represented in our model with Feature ID = 1635067.

	pkFeatureKeywordID	fkLayerKeywordID	fkFeatureID	KeywordName
910	274303	22	1625686	Thompson House
911	252030	22	1635067	Thos Issac's Log Cabin
912	273885	22	1625405	Tiber Crossing Partnershi

Figure 57: Finding the Feature ID of the Thomas Isaac Log Cabin

We create a context profile for Historic Site listing all the layers that might be relevant to a historian with the type of relationship we’re interested in and the strength of those relationships.

Table 12: Context Profile for Historical Items

	Relationship Strength	Type of Relationship
Historical District	1.00	Within
Cemeteries	0.90	Distance
Properties	0.20	Distance

A shown in Table 12, given a historic site of interest, we strongly care if the site is located within a historical district and, if it does, the name of that historic district (although sometimes historic sites are not located within a district). People interested in history might be interested in nearby cemeteries although not as strongly as they are interested in other historical sites. Finally, since the user starts at the log cabin, they might be interested in nearby properties, but we set the relationship strength to 0.20 because this is not critical information for the user.

Results

Running our algorithm (Algorithm 2C from Section 3.5) produces the following set of geographic objects with assigned relevance scores, as shown on map in Figure 58.

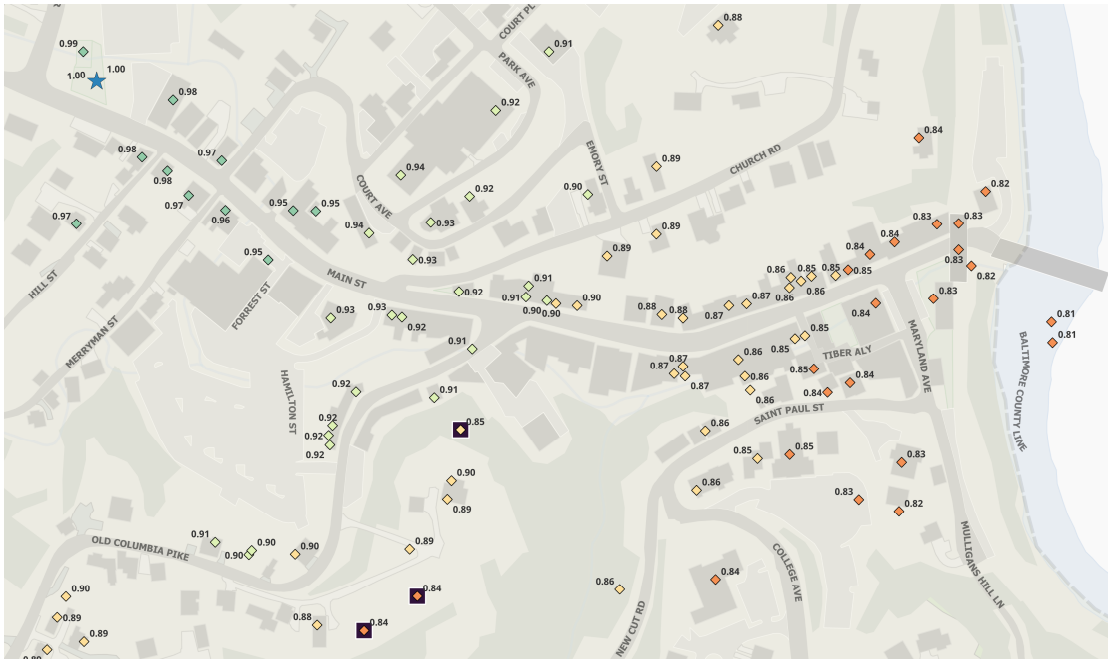


Figure 58: Map showing relevance scores for geographic objects near the Thomas Isaac Log Cabin (the blue star on the left).

A count of these properties is displayed in Table 13.

Table 13: Count of historically relevant geographic objects near the Thomas Isaac Log Cabin.

Threshold ϴ	Historical Sites	Cemeteries	Properties
95	15		1
90	45		1
85	90	1	1
80	119	5	1
75	134	5	1

There are a few observations about these results. First, the only geographic object related to a property is the property where the log cabin is located. This is because the relationship strength set in our context profile was set to 0.20, a small number. If we

had set the value higher than we would have received result for more properties. In fact, if the relationship strength had been set to 1, we would have received every property in the general area. Ultimately the relationship strength between layers is a parameter that can be adjusted depending on user needs.

The second observation is about the penalty on the cemeteries. This context profile included cemeteries as being relevant but there was a slight penalty on this layer because they were not as relevant as the historical sites themselves. There are a few cemeteries shown in the results in Figure 59 as black boxes.

Figure 59: Several cemeteries (outlined in black) were returned in addition to the historic sites.

Note that the relevance scores are significantly lower than those of the surrounding historic sites. This is the result of the layer significance, or penalty, specified in the context profile.

Evaluation

To evaluate SAM-GIS, we need to identify the ground truth for this particular experiment and compare our results to this ground truth. A user interested in Historic Ellicott City, especially those interested in a "haunted" perspective of history, might be interested in the 53 locations documented on the "Visit Ellicott City" website brochure (https://visitoldellicottcity.com/places/), which constitutes the ground truth. We compare the results of SAM-GIS to the data from this ground truth. For this evaluation, a "true positive" is a location from this brochure that was included in the result set in SAM-GIS.

This analysis compares the results of a search for historic locations in a traditional GIS system, specifically the GIS system operated by the local government of Howard County, Maryland, over a single-layer with the results returned by SAM-GIS.

Accuracy is the proportion of the correctly classified records: Records that match what should have been found and not including records that should not have been found. Since the search returns many records, the accuracy will be low in general and the more we lower the threshold Θ the lower the accuracy. However, for the expanded search there is an improvement in accuracy since we are including cemeteries in our search that would not have been included otherwise. The accuracy of our model in this scenario is shown in Figure 60.

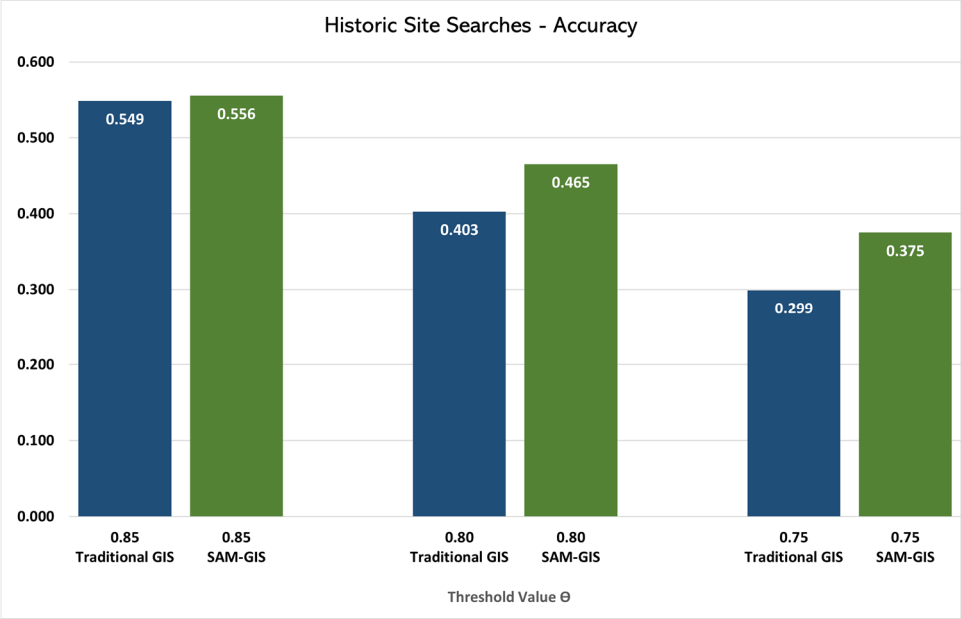


Figure 60: Comparing the accuracy of SAM-GIS versus traditional GIS searches for historic sites.

Precision is the ratio of correctly found records divided by the total number of records returned. In this case it is the number of historically relevant geographic objects found divided by the total number of records found. As we expand our search the number of records returned increases and the probability of more "false positives" increases lowering this score. There is a noticeable improvement in precision when we expand the search. The precision of our model in this scenario is shown in Figure 61.

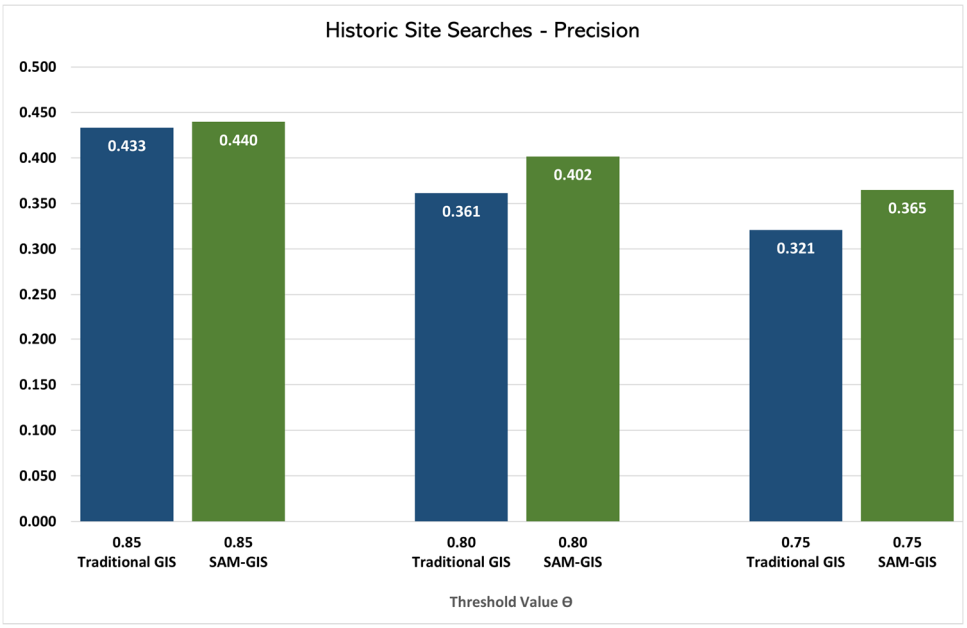


Figure 61: Comparing the precision of SAM-GIS versus traditional GIS searches for historic sites.

Recall is the ratio of correctly identified records divided by the sum of correctly identified records and incorrectly classified negatives. In our scenario, it is the proportion of correctly identified historic sites divided by the sum of those incorrectly identified historic sites and the sites that did not show up in the search but should have appeared in the search result. Recall improves as the search expands (the threshold Θ goes down). Recall is increased when we expand the search to multiple layers, as shown in Figure 62.

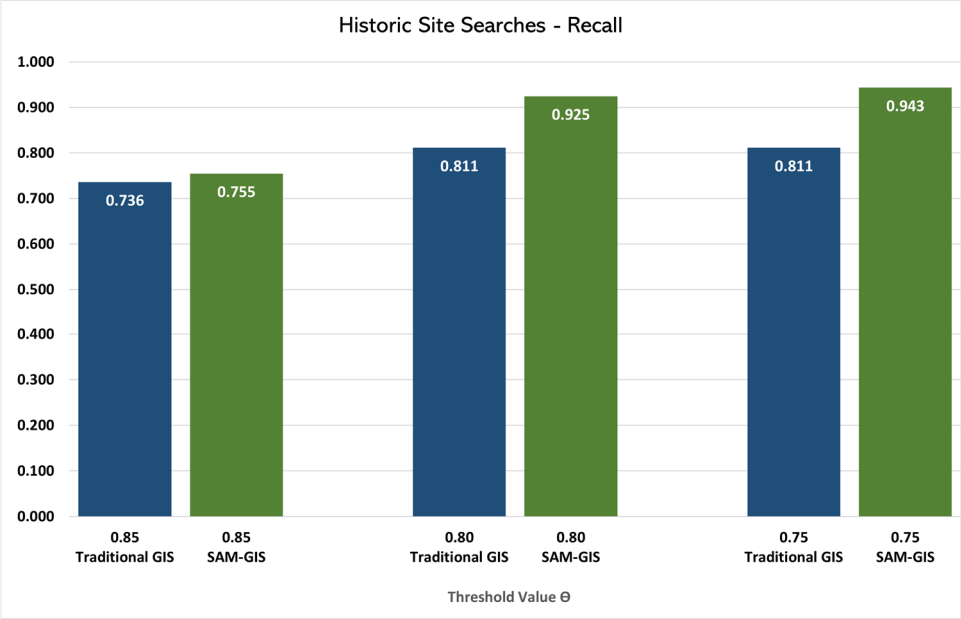


Figure 62: Comparing the recall of SAM-GIS versus traditional GIS searches for historic sites.

The F-Measure (or F-Score) is an aggregate measure that balances the differing priorities of the precision and recall scores.

Figure 63: Comparing the F-Measure of SAM-GIS versus traditional GIS searches for historic sites.

As shown in Figure 63 the F-Measure is improved with SAM-GIS compared to traditional GIS searches.

Comparison to Publicly Available Map Searches

It is quite common that many consumers use commercial web-based mapping services such as Google Maps (https://maps.google.com) to also perform these types of searches.

A search for "Historic Locations near Thomas Isaac Log Cabin" in Google Maps only returns four results: The Thomas Isaac Log Cabin itself, The Ellicott City Historic District, the Benson-Hammond House, and the Historic Ellicott City non-profit. Only two of these objects are historic landmarks. This search does not return the other 51 historic locations in the ground truth.

Figure 64: A search in Google Maps for "Historic Sites in Ellicott City"

A search in Google Maps that provides a better result is "Historic Sites in Ellicott City." A screenshot of these results can be found in Figure 64. Of the top 20 results returned on the first page in Google Maps, 8 were correctly identified historic sites

from the "Visit EC" brochure. Three were incorrectly classified non-profit organizations that serve the historic district but are not actual historic landmarks. These three results most likely reflect Google's propensity to search locations using text labels (a search that includes the words "historic" and "Ellicott City") in addition to their geospatial relevance. This also might explain the 9 returned results that are not in the Historic Ellicott city district, such as "Dorsey Hall" which is a historic location several miles away but technically is located in the modern "Ellicott City" zip codes and would not be included in the brochure.

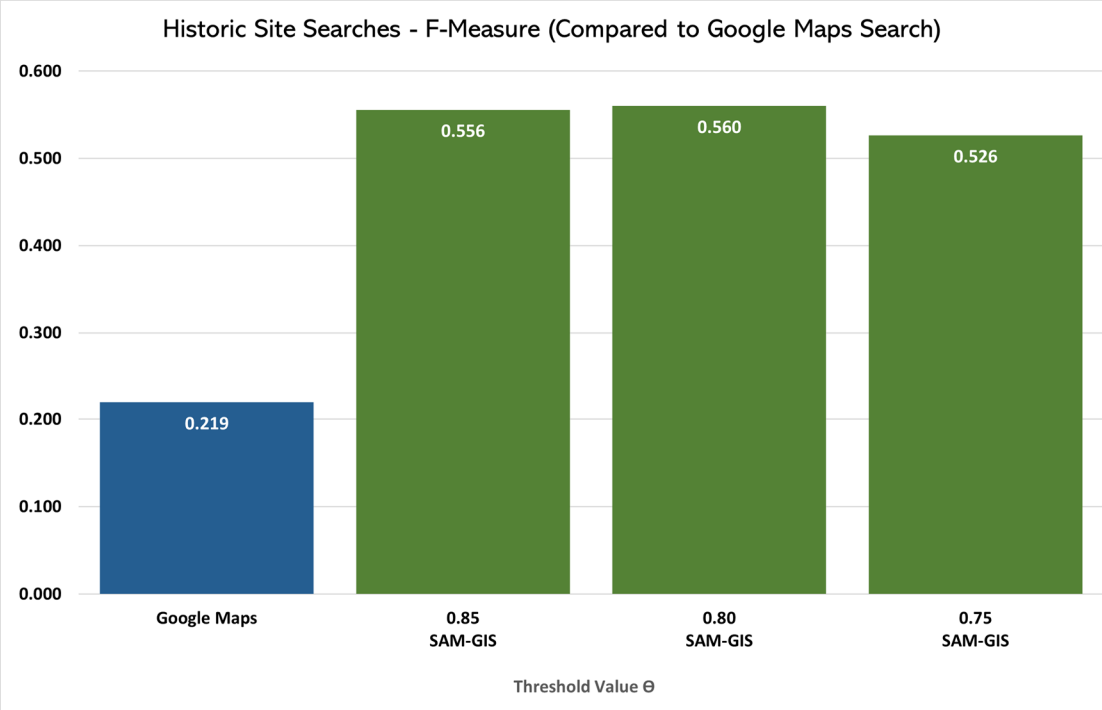


Figure 65: Comparing the F-Measure from SAM-GIS versus a Google Maps Search for "Historic Sites in Ellicott City"

As shown in Figure 65, the F-Measure for the results from SAM-GIS greatly improve upon the results from the search in Google Maps. Recall is greatly improved with a minimal decrease in Precision.

The key difference between a system like SAM-GIS and the map search results produced by Google Maps is that Google searches tend to be unfocused and emphasize locations as keywords rather than the geographic properties of these objects and their semantics.

Sparse Search Results

One aspect of GIS applications is that they typically return to the user many more results than they need. In the historic sites example, a user might only be interested in a handful of historic sites, but a map (paper or interactive) will display more locations than they can realistically visit. This has always been a characteristic of mapping applications resulting in a typically low levels of accuracy and precision. This is an expected outcome in this type of GIS applications.

To show this, we establish a smaller dataset of tourism-friendly historic sites as established on the "Visit Historic Ellicott City" site with only 13 locations.

Figure 66: Results of a sparse search on common tourist locations in Historic Ellicott City. Note that the SAM-GIS results are consistently better than a traditional GIS search.

A few observations on the results of a sparse search (shown in Figure 66). Since we are only interested in 13 potential tourist locations, when the GIS search returns a map with over 100 sites (many which are not in the brochure) this increases the "false positive" counts, and this is reflected in the levels of accuracy and precision, although it is noticeable that the recall score is quite representative of the expected query results.

5.3.2 Flood Risk Example

Someone interested in the Thomas Isaac Log Cabin for historical purposes would also be concerned about flooding in the area. The log cabin is in the Ellicott City historical district which has a history of frequent flooding. Is the log cabin in danger of

flooding? What relevant geographic objects can help educate someone about this danger?

For this query we create another context profile that includes layers that can inform the user about the flood risk in their area based on the user's address as the initial location.

Table 14: Context Profile for Historical Items

	Relationship Strength	Type of Relationship
Floodplain	1.00	Distance
Frequently Flooded Roads	1.00	Distance
Impervious	0.80	Distance

We take three layers and identify nearby objects by their distance to the address of interest. The floodplain is a layer that determines the likelihood of a property being flooded based on that property's proximity to rivers and their flows. Frequently Flooded Roads are also based on their proximity to bodies of water but tend to be in older and more rural areas where the roads were not built to modern standards. The key difference is that while a frequently flooded road can be rebuilt to no longer flood, a floodplain describes the likelihood of a property to flood independent of any engineering improvements made to the property. Finally, a house is more likely to flood based on the amount of impervious surface immediately surrounding it, however, this relationship is not as strong as the other two layers since many properties have parking lots or driveways and are in no real danger of flooding.

Retrieving Relevant Information from this Context Profile

Ten addresses were selected to evaluate the model. The ground truth for these ten addresses can be found at the Howard County website reflecting FEMA insurance rate data (https://data.howardcountymd.gov/GDFIRM/main_Web.aspx). Six were classified as normal properties but four are found within the FEMA "insurance adjustment" rate map. SAM-GIS was run on each of these addresses with a threshold value Θ of 0.75.

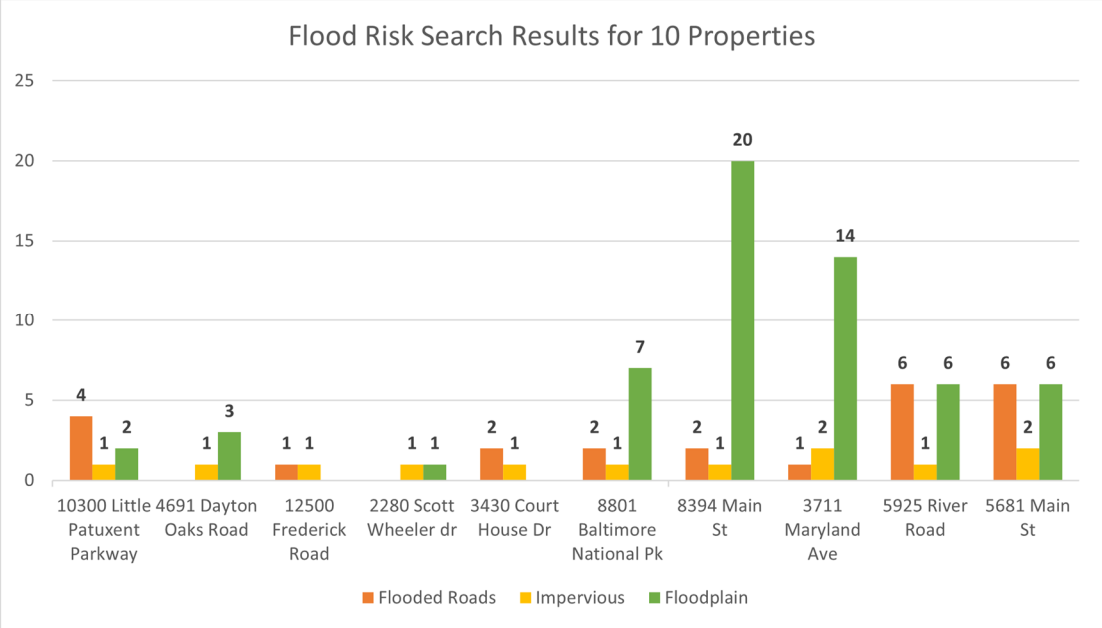


Figure 67: Flood Risk Search Results for 10 Properties (the y axis represents the count of the measure: flooded roads, impervious, floodplain)

As shown in Figure 67, each one of these addresses has their own distribution of relevant geographic objects based on their location. For example, 10300 Little Patuxent Parkway, despite not being near many floodplain tiles, is near 4 frequently flooded roads.

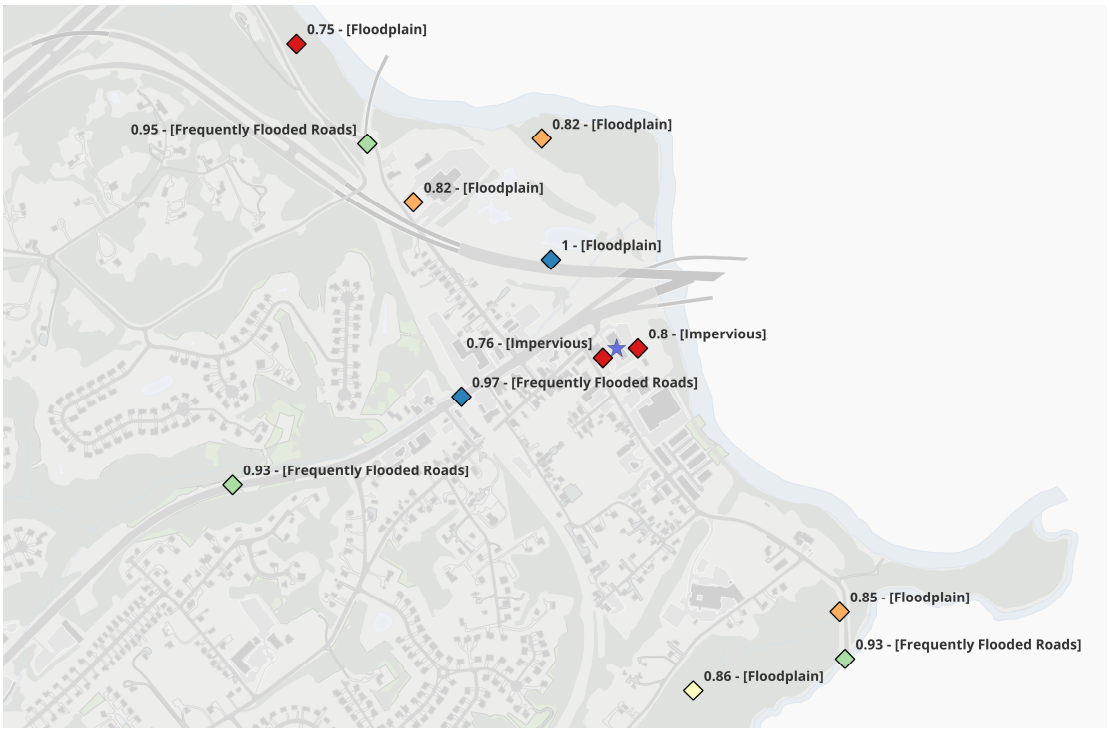


Figure 68: Results of a flood risk search for 5681 Main Street, Elkridge, MD

A search in SAM-GIS for flood risk for 5681 Main Street (Figure 68) returns 6 floodplain blocks, 2 impervious service blocks and 6 frequently flooded roads (only four shown in the figure).

Evaluation

While the primary purpose of creating a system to return any "flood" related geographic objects will simply return and display those objects, we can attempt to evaluate how well this works, and use it as a classifier to predict properties subject to adjusted insurance rates due to flooding versus those that do not need additional flood insurance.

To do this we score each of the ten properties based on the sum of all relevance scores for all objects returned for each property. In the case of 5681 Main Street, there were 14 objects returned and the total relevance score for those 14 objects is 12.02.

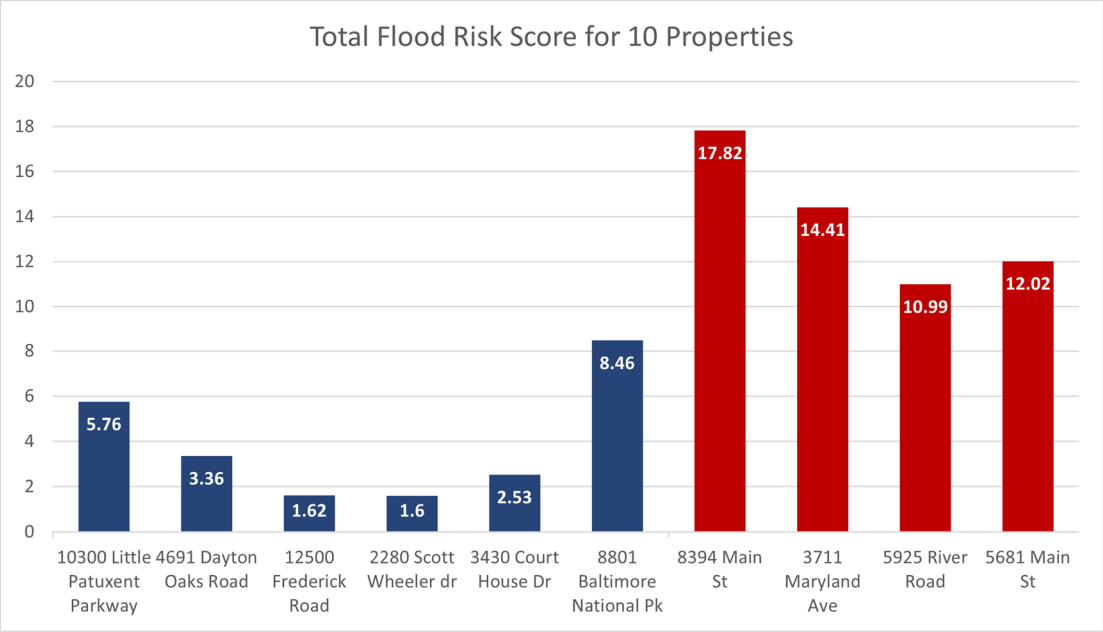


Figure 69: Calculating an aggregate flood risk score for the ten properties.

As shown in Figure 69, the six properties not in a FEMA insurance adjustment zone (the six bars in blue) have distinctly lower flood risk scores than the four properties that are in the adjustment zone (the four properties in red). If we split the scores at a fixed value between the highest non-FEMA property which has a score of 8.46 and the lowest FEMA property which has a score of 10.99, we can create a simple classifier.

Beyond creating a score to develop a classifier (a user enters in their address receives a score that corresponds to a FEMA classification), knowing what relevant flood objects that exist near a given property is helpful to the people who own that property. In a traditional GIS system, this would have been accomplished with a custom-built interactive map but with SAM-GIS, a non-technical user can enter a series of parameters (the context profile) to automatically get similar results.

This was an example of a simple classifier where a user could define an arbitrary threshold to determine which locations were subject to adjusted insurance rates. As

with any classifier, the choice of a threshold can also be computationally generated and in more complex scenarios the threshold might not be so clear-cut leading to "false positive" records during classification. Nevertheless, by running several experiments with different splits, we can observe and choose the value that represents the optimal split for these types of experiments, that would clearly differentiate between the two outcomes of the classifier. For this experiment we chose a split of 10 resulting in a perfect separation between properties subject to FEMA flood insurance rates and those that are not.

Investigating the use of the mSLN model and adapting SAM-GIS for use in common data mining tasks (such as classification and clustering) is listed as future work in Chapter 6.

5.3.3 Legislation Example

This case study examines the application of our geospatial model to a government council investigating the relationship between two pieces of legislation that affects similar land properties.

Background

In this example, a local government council discusses pending legislation affecting a neighborhood. During this discussion one council member remembers that there was earlier legislation affecting this neighborhood from a few years ago but cannot remember the legislation number. In this scenario, a method to query related heterogeneous data (legislation, zoning, streets) would improve productivity in a way that most effectively provides that council member with relevant answers.

This search for relevant information is an example of geographic information retrieval. It highlights the desire to retrieve relevant legislation from the perspective of a user investigating a neighborhood and its history. Artifacts from other layers, such as nearby parks or zoning ordinances could also be relevant and desirable to be included in the query results.

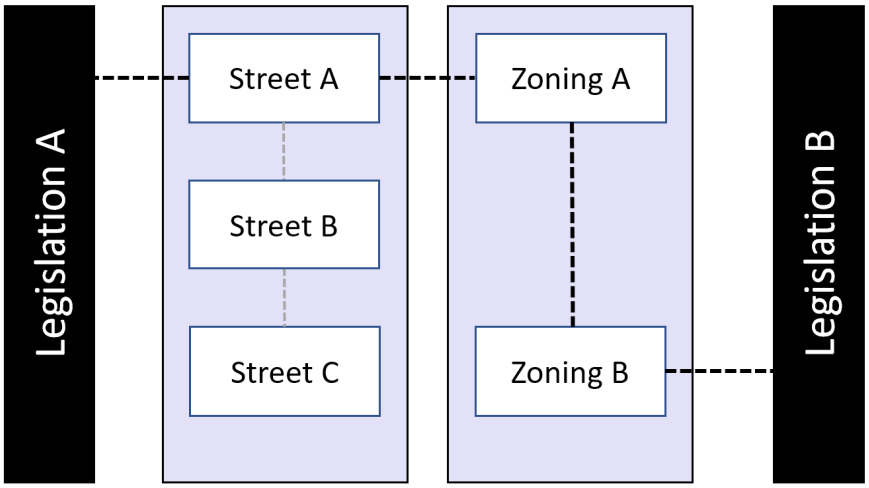


Figure 70: An example of how two pieces of legislation can be related through nearby relevant geographic artifacts.

In addition to incorporating multiple metrics describing the relationships between any two objects, GIR research generally assumes that the user only wishes to find relevant geographic artifacts in one table when there could be data in multiple tables and databases relevant to the query. For example, in addition to searching for relevant legislation, the council might be interested in construction projects, zoning ordinances or permits issued in that neighborhood.

This case study examines the potential of using our multilayer framework to identify relationships between text corpora based on the geographic relevance between objects identified in the text. Our motivating example compares the text of government

legislation to find relationships between those legislation based on the geographic references found in the text of that legislation.

We use publicly accessible data from Howard County, Maryland for the analysis of our results. Following our motivating example, we look at legislation passed by the county government to identify legislation that might be related based on the geographic locations they describe.

Datasets

We use the following datasets to construct our mSLN: A dataset with legislation related information, another one with zoning information and finally a third dataset containing street segments (see Table 15).

Table 15: Description of Datasets used to construct mSLN

Dataset Name	Dataset Size
Legislation	154 pieces of legislation
Zoning	531 zone polygons
Streets	12,139 local street segments

Following our motivating example, the legislation is one of our test datasets. This includes legislation (government laws such as council bills and resolutions) from January 2006 to May 2017 that includes a reference to a street in its description. The two records in Table 16 show two pieces of legislation that reference streets in their short description. They both reference a street closing. Given the 154 pieces of legislation in our original dataset 38 of them were tagged as sharing a common theme. These themes included closing streets, agricultural preservation, and zoning issues.

Table 16: Sample rows from legislation dataset with GIS keywords.

Legislation Number	GIS Keyword	Short Description
CR35-2017	FOREST AVE	A RESOLUTION to close a portion of Forest Avenue…
CR1-2017	WINTER THICKET RD	A RESOLUTION to close all of Winter Thicket Road…

This data can be accessed at https://cc.howardcountymd.gov. The GIS datasets can be downloaded at https://data.howardcountymd.gov.

Submitting Original Queries

Query: Given the piece of legislation as an input "CR35-2017" find all geographically relevant legislation.

For a Howard County resident or employee, the solution to this query requires the use of two independent tools that are not integrated (either through a user interface or through the underlying databases that support those applications): The County's Legislation Search Tool at https://apps.howardcountymd.gov/olis/ (as shown in Figure 71) and the standard GIS tools described in Section 2.1.

The legislation search tool itself provides bespoke search capabilities. A user can search legislation by keyword and by various attributes of the legislation (the sponsors of the legislation, whether it passed, the budget year it was passed, etc.). However, this tool is not tied to the county's GIS system, and there is no publicly accessible GIS representation of legislation (for example, a map showing that CR35-2017 affects Forest Avenue).

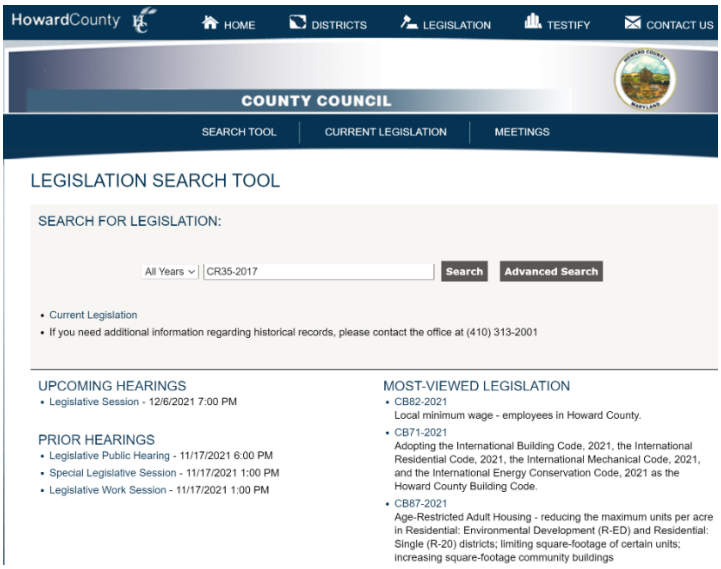


Figure 71: Howard County's Legislation Search Tool

Query Results that are Geographically Relevant and Augmented

We evaluated two scenarios: One comparing legislation using a single layer (streets) and another using two layers, the first representing streets and the second layer representing zoning information (forming a multiplex construction with two layers). With 38 pieces of legislation tagged with streets as GIS keywords (see Table 16) there are 1,444 possible relationships formed. Of those, 96 were manually tagged as having reasonably legitimate relationships. Those 96 relationships represent the ground truth of our scenarios.

As a baseline, we tried to identify relevant legislation based on the streets identified in their text alone. We varied the search radius and the threshold Θ limiting the scope of the results that would return for any given legislation.

We then submitted the same query using two layers, streets and zoning, to identify relevant legislation that would not be identifiable otherwise. For example, two pieces of legislation might be related because they both refer to agricultural preservation and a comparison of street proximity alone might not reveal this relationship. However,

since agricultural preservation only applies to properties that are agriculturally zoned, we can infer this relationship.

Legislation is tagged with streets and streets are related to zoning in that a street segment is contained inside a zone. We choose a search radius of 4000 ft. on the zoning layer since for zoning, unlike streets, we are not as interested in distance.

As an example of the benefits of using two layers we focus on an individual piece of legislation. The legislation with number "CR15-2013" describes an effort by Howard County to purchase farmland for agricultural preservation. Using streets alone, the only two matches the query returns with a threshold $\Theta > 0.8$ are "CR67-2015" which authorizes the county to sell property it owns and "CR64-2009" which describes issuing municipal bonds. Neither legislation reference agricultural preservation. When we use two layers (adding zoning) with the same threshold the query returns 5 additional matches that also relate to agricultural preservation.

The results of this experiment are shown in Figure 72. While precision is reduced with SAM-GIS (due to the expanded search results) recall is increased to a greater degree, resulting in higher F-measures for SAM-GIS for all three threshold values.

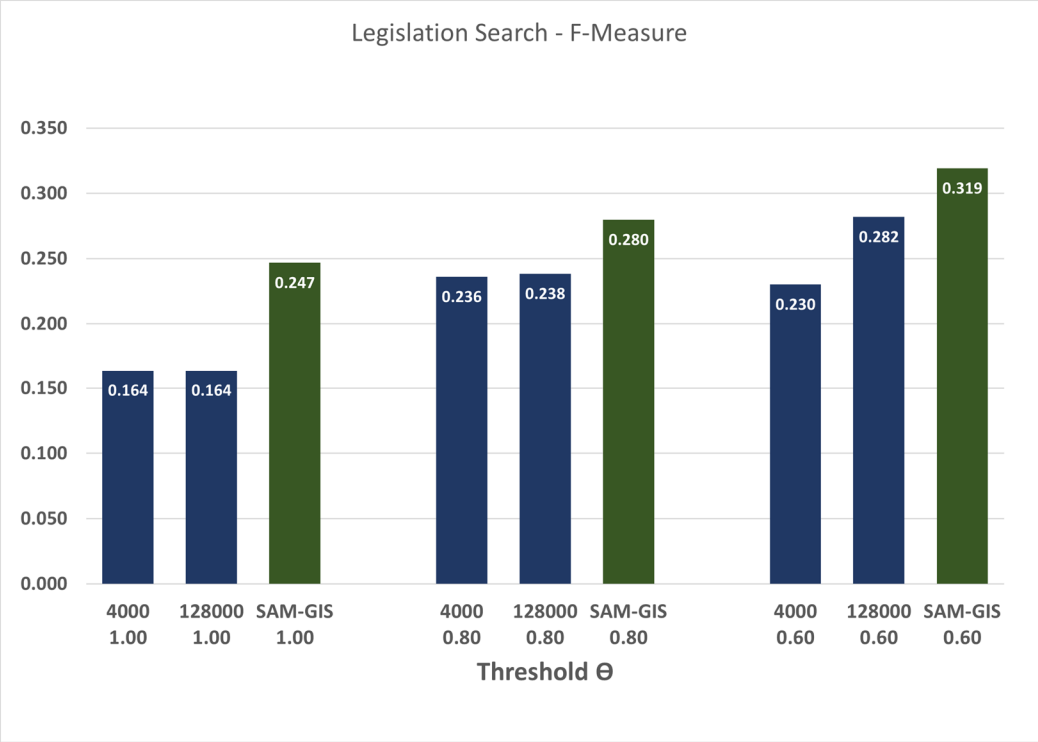


Figure 72: Comparing the F-Measure of SAM-GIS versus a traditional GIS search for our legislation example.

Performance Advantages

One key difference between SAM-GIS and the traditional searches is in the number of calculations. Calculating similarity for thousands of street segments is more computationally intensive than calculating similarity for hundreds of zoning polygons. The reduced amount of computation provides a boost to the performance of the system while at the same time it does not sacrifice any relevant information from the results.

For example, one of the first steps is to calculate the similarity between the geographic objects linked to the legislation. As shown in Table 17, if we compare streets, even at a radius of 4000 ft. with our dataset we must make about 2.3 million comparisons. This increases to a maximum of about 147 million comparisons to compute the similarity between all 12,139 segments. Performing the comparison with

zoning with a radius of 4000 ft. only requires about 12,000 comparisons. Finding the zone associated with a street in a standard geospatial database is a trivial calculation assuming both tables have proper spatial indexes but calculating the similarity for both tables is a brute force calculation.

Table 17: Effect of radius on similarity calculations required.

Radius	Street Comparisons	Zoning Comparisons
4000 ft.	~2.3 million	12,119
16000 ft.	~24.6 million	61,545
64000 ft.	~132 million	257,193
128000 ft.	~147 million	281,693

5.3.4 Summary of Findings from Sample Queries

This section described three potential applications of the SAM-GIS model. The first explored a user who wished to find historically relevant locations. The second evaluated the flood risk associated with properties. The third identified relevant legislation based on the geographic locations written into the legislation text.

The results from these examples provide the following observations:

- **Lowering the threshold reduces accuracy and precision**. This is especially true when a user searches for few items among a large list. These types of searches are common among users of GIS applications. A lower threshold strongly affects accuracy and lowers precision to a lesser degree.
- **Lowering the threshold increases recall**. One of the weaknesses identified in traditional GIS applications is their inability to search multiple related

classes (or layers) of GIS objects, therefore increasing the number features that can be retrieved increases the overall recall of the application.

- **SAM-GIS improves over traditional GIS searches in all evaluation measures of interest**. The expanded search provides increases in accuracy, precision, recall and F-Measure.

The next section will briefly outline some observations about the performance of SAM-GIS on the test hardware.

5.4 Performance Evaluation

This section evaluates the performance of SAM-GIS, notably in the time it requires to build the model and to conduct searches. There are two methods of querying objects in SAM-GIS.

Cold Start Search. The first is a "cold start" search that directly queries a series of layers in a context profile as described in Algorithm 2C from section 3.3.5. The performance of this search will be documented in Section 5.4.2.

Static Model Search. The second method searches for features in a pre-generated (static) model of relevance for the layers and objects for an associated context profile. For a given context profile, the static model only needs to be created once (static part) and any queries of the model can be done quickly since it is a simple database lookup (dynamic part). The creation of a static model was documented in Section 4.2 and the search is based on Algorithm 1 from Section 3.3.5. The performance of generating the static model and queries off that model will be described in Section 5.4.3.

5.4.1 Performance of Pre-Requisites

Whether we use the static version of SAM-GIS or the dynamic version, there are several variables and records that need to be calculated ahead of time. These utility values can be calculated ahead of time.

A summary of the features in the test system and the computational time required to pre-generate summary information related to the imported features are outlined in Table 18.

Table 18: Summary of objects used in sample queries

Total Layers in Model	120 layers
Total Features in Model	667,625 features
Initial Import of Features from Holding DB	15 seconds
Update Features from Holding DB	9 seconds
Calculate Normalized Distances for all Layers	91 seconds
Calculate Normalized Distance for one layer (average)	1-2 seconds

5.4.2 Cold-start Queries

This section outlines the performance of SAM-GIS utilizing a direct query of the system. To evaluate the performance of the system at run-time, several context profiles were created with different numbers of layers and different numbers of features. A description of these context profile is described in Table 19. Small layers have 10-40 features each, medium-sized layers have 10,000-30,000 features each, large layers have 30,000-65,000 features each, and very large layers have over 100,000 features each. These are prototypical layer sizes for a medium-sized city or county.

Table 19: Twelve different context profiles to evaluate performance. The cells describe the total number of features in each profile.

	Small	Medium	Large	Very Large
2 Layers	33	15166	39076	210459
3 Layers	75	19015	51811	305603
4 Layers	82	22325	63936	389770

Each context profile was run using the same starting point. The second layer was given a relationship strength of 0.9, the third was given a relationship strength of 0.8 and the fourth was given a relationship strength of 0.7.

Table 20 shows the number of time (in seconds) it takes for to query SAM-GIS from the same starting point for each context profile.

Table 20: Time to process a query from the same origin point across all twelve context profiles (in seconds)

	Small	Medium	Large	Very Large
2 Layers	10	11	13	18
3 Layers	10	11	13	22
4 Layers	10	11	13	26

The main conclusion is that the number of layers alone does not influence the processing time of a dynamic query to SAM-GIS, but the total number of features contained in those layers does. This relationship is linear, as shown in Figure 73.

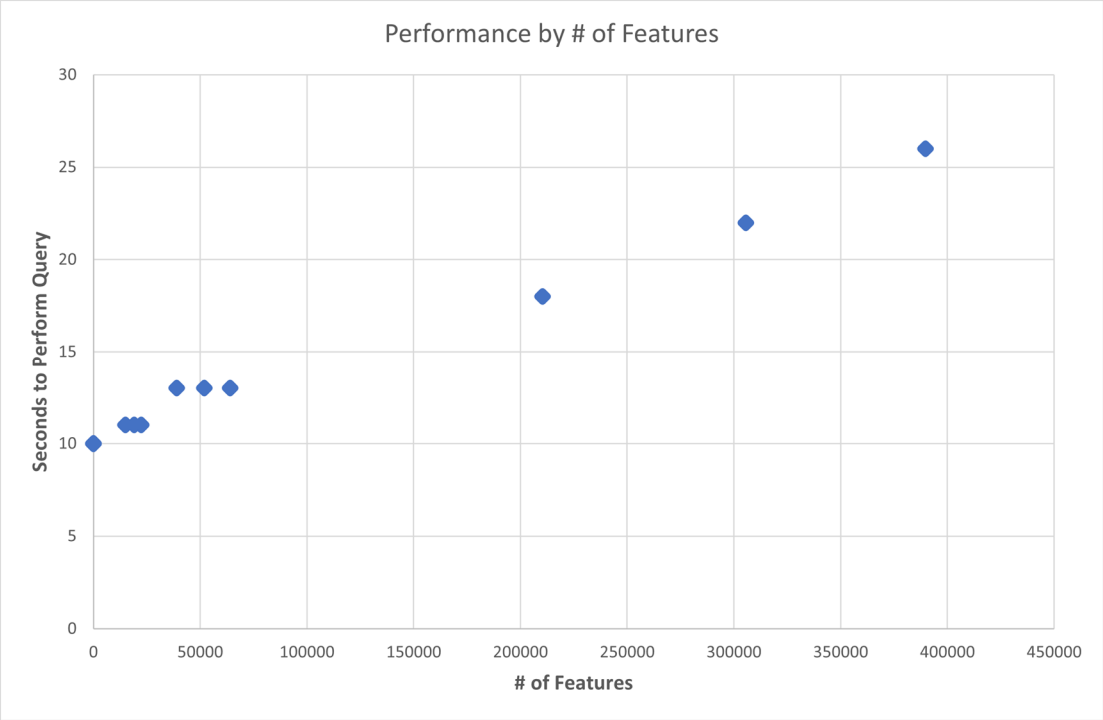


Figure 73: There is a linear relationship between the number of features in a context profile and the time to perform a same query.

The dynamic query algorithm (Algorithm 2C from Section 3.5) requires comparing the original feature to every feature in the context profile based on both the geographic properties of those features and that reflects in the linear relationship between the two. There is some overhead (about 10 seconds) that SQL Server needs to mechanically facilitate these queries but after that the linear relationship is reflected in the graph.

Additionally, SAM-GIS takes advantage of spatial indexes available in database management systems. In this case it is utilizing the SQL Server spatial index with the default settings on the “Feature” table.

5.4.3 Static Model Generation and Query Performance

A static model is created using Algorithm 1 (Section 3.3.5) based on the layers and features contained within a specific context profile. This process is also outlined in Section 4.2.

Pre-generating the model requires creating an initial SLN based on the features contained within a context profile and then identifying multi-step relationships based on the topological relationships between the features and the strength of relationship (similarity) between the layers in the context profile.

To evaluate this functionality, a simple context profile was created with two layers: Parks and Farms. Both layers combined have 170 features.

Table 21: Time to Create Static Model

Task	Time (minutes: seconds)
Initial SLN Creation Time	1:49
Two-Hop SLN Creation Time	50:52
Three-Hope SLN Creation Time	51:02

As shown in Table 21, while the time to create the initial SLN showing one-hop relationships between geographic objects is quick, there is a consistently lengthy time to create two-hop and three-hope relationships. This is because the initial SLN can be accomplished with simple database joins which takes advantage of indexes in the database.

Creating a two-hop and three-hope SLN requires comparing every element in the context profile to all the others. This is the database equivalent of a matrix multiplication of the initial SLN by itself, as described in Section 3.3.

We look at the number of relationships that are improved (provide a higher strength of relationship) in the two-hop and three-hop scenarios. One benefit of a semantic link network is that every time the SLN is multiplied by itself (using the reasoning rules) it reflects multi-hop relationships between nodes that are not apparent when we only look at single-hop relationships between nodes.

Table 22: Percent of node relationships improved with multi-hop scenarios for the parks and farms context profile.

Relationship strength between parks and farms	% Records Improved with Two-Hop	% Records Improved with Three-Hop
1.0	21.2%	1%
0.9	18.0%	0.9%
0.8	16.6%	0.9%
0.7	15.9%	0.8%

Table 22 shows the number of relationships that are improved (show higher probabilities) when we include two-hop and three-hop connections between nodes. These percentages depend on the nature of the layers included in a context profile as well as the relationship strength (or penalty) between the layers. In the parks and farms context profile the strength of relationship was changed between test runs and the table shows the percentage of records that showed improved relationships when including two-hop relationships and three-hop relationships.

There are diminishing returns to applying the reasoning rules in the mSLN. Including all two-hop relationships between geographic objects in the context profile improves between 15-21% of the relationships between nodes including relationship

scores that are above the search threshold. However, calculating all three-hop relationships only improves about 1% of the relationships between nodes. Since there is a considerable amount of time required in calculating each additional hop in the static model, there is little practical benefit to continuing to calculate additional hops.

5.4.4 Comparing execution of cold-start and dynamic queries

While pre-generating a static model takes a considerable amount of time (over 50 minutes in the example from Section 5.4.3), one benefit is that when a user needs to query this model the result is nearly instantaneous since this only requires a simple database lookup. Section 5.3.2 outlined the performance of the "cold start" model, and those queries took between 10 and 26 seconds depending on the size of the context profile. The static version of the model calculates the necessary edge weights ahead of time, eliminating this delay.

Table 23: Time to query relevant features with "cold start" queries and queries of the static model.

	Cold-Start Model (Algorithm 2C)	Static Model (Algorithm 1)
Small Context Profile	10 seconds	<1 second
Medium Context Profile	11 seconds	<1 second

5.5 Summary of Experimental Evaluation

Chapter 5 explored the evaluation of the SAM-GIS model described in this dissertation with real-world data derived from Howard County, Maryland's various open data portals. Multiple intermediate results were developed during the creation of

the model and several more results were revealed when SAM-GIS was used to model geographic relevance in three sample queries: Investigating historic sites, classifying flood risk, and finding relevant legislation based on their geographic impact. The performance of SAM-GIS using both dynamic "live" queries and static "pre-generated" queries was explored.

Chapter 6: Conclusion

Users of geographic information systems have traditionally relied on desktop or web-based applications to access relevant geographic objects stored in their related databases. This requires a developer to create an application to facilitate the user's needs. These applications have not been adaptable to the user's context and rely on traditional application development workflows.

Additionally, most GIS implementations have not adapted to utilize modern algorithmic techniques to augment these systems with the capability of providing additional results that are more relevant to the user.

This dissertation presented a mathematical model and a framework based on it that takes advantage of the multi-layered nature of GIS systems to create a consisted model of geographic relevance for objects. It also describes a practical system implementation, SAM-GIS that utilizes this framework and can be easily implemented by typical GIS organizations.

6.1 Summary of Contributions

This section details the contributions of this dissertation.

A mathematical model of geographic relevance. This research work defined common GIS terms mathematically and conceived, designed, implemented a cohesive mathematical formulation to tie common GIS concepts together and proved their correctness through theorems (Sections 3.1 through 3.3).

- Many GIS calculations revolve around topological rules and while these rules are encoded into many geographic information system and databases that utilize

spatial object types (through the OGR topological standards), this dissertation provided a mathematical formalization of these rules.

- This dissertation described an extension of Tobler's law: While near things are more important to the user than far things, what is considered "near" depends on a user's context. Additionally, we explored what geographic relevance means beyond the traditional GIS conception of distance by incorporating multiple layers.
- This mathematical model is generalizable to other domains. While this work was designed for GIS applications, the model itself describes a technique to identify relevance between objects in multi-layered graphs [167]. This work has already been adapted for use in cybersecurity applications. For GIS applications, any topological rule can be adapted for use in this framework. The next section outlines potential extensions to this model.

Algorithmic Approach for enhanced GIS query automation. This dissertation outlined several algorithms that use the mathematical frameworks described earlier to provide enhanced results to those provided by traditional GIS applications. This includes additional geographic objects that might be relevant to the users of these applications (Section 3.5).

- Multiple algorithms were outlined that utilized the mathematical concepts of geographic relevance (Chapter 3). These algorithms are adaptable to different use cases that might be relevant to GIS users.
- This work provided an implementation of multi-layer and semantic link networks that utilized multiple algorithms to identify objects that are

geographically relevant. The mathematical framework described in Chapter 3 was translated into a system implementation described in Chapter 4 with usable code provided in the Appendix.

Application of multi-layer and SLN graph theory. While research into multilayer networks and semantic link networks have increased in the past decade, the transformation of these theories into practical applications has been minimal. This dissertation provides a workable implementation of these concepts (Section 3.3).

- Existing research into semantic link networks acknowledge that the application of reasoning rules to develop the SLN is a computationally intensive process. This work implemented an SLN designed to interoperate with traditional database systems and to perform queries within a reasonable time frame.
- This work combined the concepts of semantic link networks and multi-layer networks into a combined concept: The multi-layer semantic link network (mSLN). The necessary mathematical foundations for this data structure were outlined (Section 3.4).

Generation of a geospatial ontology. This work facilitates the creation of an ontology both manually by subject matter experts and automatically through the usage of WMS logs. This ontology mathematically assigns values that reflect the relevance of the layers involved in the relationships (Section 4.1).

Consolidation of GIS concepts. Traditionally, geographic information systems are designed either based on ad-hoc querying or by application developers who use a traditional process methodology. When viewed though a common mathematical

framework, most GIS applications represent similar types of queries. This dissertation consolidates most GIS queries into a common framework (Sections 3.4 and 3.5).

The implementation of a prototype system. The mathematical framework has been incorporated into a set of algorithms, which can be implemented by any enterprise organization that utilizes GIS. It uses a star schema data warehouse as a basis and is designed to piggyback on existing GIS systems (Chapter 4 and Appendix).

An evaluation of the model and framework. The framework has been evaluated using the prototype system. This includes:

- Validating the mathematical model by proving theorems that derive from existing mathematical concepts and the definitions of concepts related to geographic objects (Chapter 3).
- Evaluating the framework using multiple real-life case studies relevant to local communities in Howard County, MD, USA by comparing the results of SAM-GIS with ground truth and examining the accuracy, precision and recall of the system (Chapter 5).
- Examining performance and network complexity for the system based on a typical local government population (Section 3.6 and Section 5.4).

6.2 Limitations and Future Work

This dissertation describes the conception, design, and implementation of a model of geographic relevance and a basic implementation of that model. This section outlines some limitations of this work and describes potential expansion or improvements to the system.

Spatial Heterogeneity. The mSLN-based model operates under the assumptions derived from Tobler's law: Certain topological relations between geographic objects are continuous and reflect a distance between those two objects. However, this model does not account for spatial heterogeneity in the sense that there are small and random variations between the properties of geographic objects immediately adjacent to each other. Future work could investigate how semantic heterogeneity manifests itself in the model.

Dynamic Updates to the System. While the performance of SAM-GIS is acceptable for medium-sized datasets at the local government (county) level, organizations are increasingly interested in the analysis of real-time or streaming data and this framework, as it is currently constructed, is not fully optimized for these types of updates. Identifying methods to optimize the ability for the framework to adapt to new information would be an interesting avenue for future research.

Parameter Tuning. This work described the construction of a model of geographic relevance based on parameters (such as layer relevance scores) either devised by experts or based on usage of GIS interactive maps, as described in Sections 4.2.1 and 5.2.1. Future research could investigate other methods of establishing these parameters including those used for text mining (such as the "distributional hypothesis" [168]) that could be used to interpret the probability distributions of geographic objects in a domain. For example, if two classes of geographic objects have a similar spatial distribution in the same domain, could both layers be reduced to one in a context profile?

Graph-based data mining. Since the product of the implementation of our model is a mathematical graph, an analyst could implement one of the available graph-based data mining identify patterns in the network of geographic objects.

Scalability. The performance of the system was satisfactory and usable at the local government level using a single desktop computer. However, GIS applications are prevalent at the state and federal level. GIS datasets at the federal level can include hundreds of millions of geospatial objects which can be problematic for a single workstation. Future research could investigate the ability to use distributing computing technologies to make the model construction feasible at this scale. The bulk of the work in the SAM-GIS framework is building the static model of geographic relevance and the most time-consuming portion of this involves matrix multiplication which has already been established as parallelizable task for distributed systems. Therefore, moving our environment into a distributed and/or parallel system is expected to have vast improvements and deal with geographic objects at a Federal level.

Continuous Improvement Based on User Interaction. The usage of GIS systems informs the structure and design of SAM-GIS. As users use GIS systems, increased interest in local areas can be fed back into the mSLN model to re-adjust weights automatically. In this sense, as the user interacts with the system, the system adapts to meet the user's needs in real-time. This expansion of the system would require a workflow to capture the user's approval of the recommendations made by SAM-GIS and a method to incorporate that feedback to improve the model.

Resiliency. Research into emergency situations such as natural disasters or public health emergencies encompass a field known as resiliency. If an organization had a

network of infrastructure with multiple layers (electrical grid layer, flood layer, emergency shelter layer) it would be interesting to investigate how the elimination of a node from the mSLN would affect the operation of the network, especially for emergency situations.

Equity in Geospatial Analysis. Any model of geospatial relevance depends on the source data used to build that model. Geospatial datasets are typically constructed by institutions such as government agencies and this can introduce biases (implicit and explicit) in the source data which can propagate to analyses based on this data. Future work could look at how these biases are generated in the original data propagate through the mSLN or even how such analysis could help detect biases in the placement of services and infrastructure in a community.

Implement the system in an organization. The system described in Chapter 4 and in the Appendix was designed to be implemented by organizations with an existing GIS infrastructure. A practical implementation of this system at such an organization would provide a baseline to help that organization solve problems related to geographic relevance.

Appendix: Implementation Whitepaper

This appendix outlines the practical implementation of the system proposed in this dissertation in the form of a whitepaper. Any organization with skilled SQL practitioners should be able to take the specification outlined below and implement the database structure outlined.

This appendix focuses on the database backend of the proposed system. It does not include user interface elements nor implementation in a desktop GIS system.

The code outlined below was designed for Microsoft SQL Server systems and was designed on a SQL Server 2016 Express server. The SQL code should be easily adaptable to other database systems.

The names assigned to entities are specific to the sample implementation used for this work. Any organization could change these names to suit their own purposes.

A.1 Importing Data

The basis for this work is a standard data warehouse and this requires importing data from sources. This work involved utilizing the WFS services provided by Howard County, Maryland imported into a SQL Server as a series of standalone tables in a "holding database" using the GDAL/OGR open-source ETL software. There are many methods to acquire datasets and many organizations will already have their data stored in established database servers. For example, if the source systems are pre-existing SQL Server databases, then an SSIS package would be the preferred method of data transfer. What is important is the ability to copy relevant datasets (layers) to the holding table using whatever methods are available to that organization. This requires

organizational knowledge about the systems holding datasets (GIS systems and other systems holding geographic datasets).

A.1.1 Ontology Holding Table

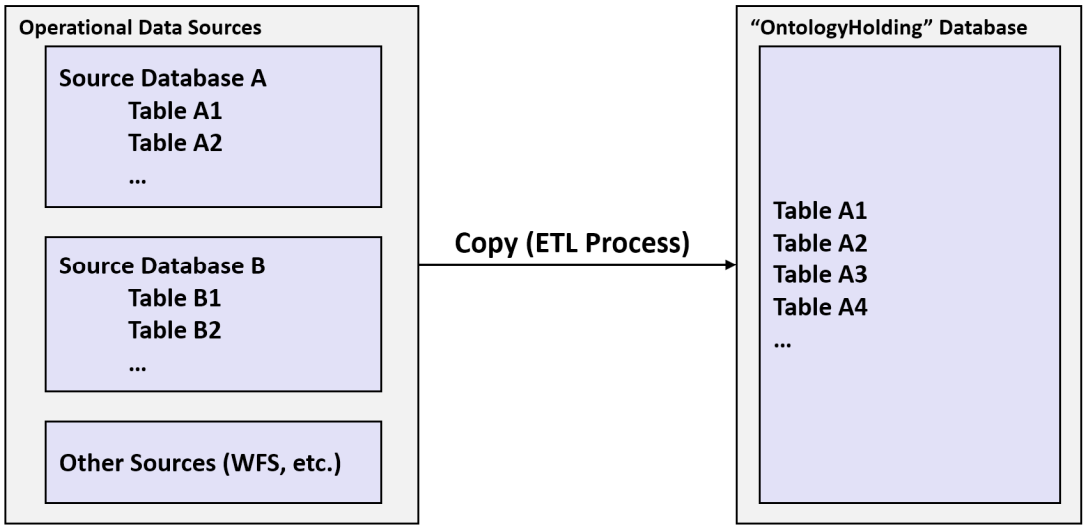


Figure 74: Loading data from operational data sources into holding database.

In our example data is imported into a holding table called **OntologyHolding** as shown in Figure 74. The tables located in this database (as shown in Figure 75) are unmodified from the table structure exposed by the organization in the original location. For example, the fields in the "Address Points" table are the same as those presented by Howard County on their WFS server.

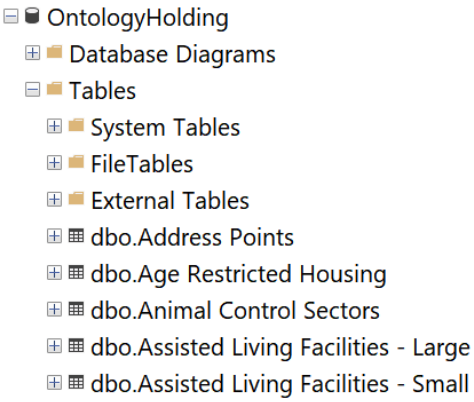


Figure 75: The Ontology Holding Table with Imported Tables

In this example, 120 tables were imported from Howard County's WFS server.

A.2.1 Layers Table (tblLayer)

In the main database we create a table **tblLayer** which contains a listing of all of the layers imported into the system. This is how the system knows of the existence of a layer to be included. This table is manually populated by the system's architect to ensure that only the relevant layers are included in the system.

Algorithm A1: Create tblLayer

```
CREATE TABLE [dbo].[tblLayer](
	[pkLayerID] [int] IDENTITY(1,1) NOT NULL,
	[TableName] [varchar](50) NULL,
	[PrimaryKeyColumn] [varchar](50) NULL,
	[GeometryColumn] [varchar](50) NULL,
	[DistanceAverage] [float] NULL,
	[DistanceStDev] [float] NULL,
	[IconURL] [varchar](200) NULL,
	[TableTitle] [varchar](200) NULL,
	[IsPublic] [bit] NULL,
	[Description] [varchar](max) NULL,
 CONSTRAINT [PK_tblTable] PRIMARY KEY CLUSTERED
(
	[pkLayerID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY
= OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
```

This table includes important relevant to the imported table in the holding table (TableName, PrimaryKeyColumn, GeometryColumn) and some additional information that will be useful when presenting this information to the user (IconURL, TableTitle, IsPublic, Description).

	pkLayerID	TableName	PrimaryKeyColumn	GeometryColumn	IconURL	TableTitle	IsPublic	Description
1	1	[Address Points]	qgs_fid	geom	NULL	Address Points	1	NULL
2	2	[Age Restricted Housing]	qgs_fid	geom	NULL	Age Restricted Housing	1	NULL
3	3	[Animal Control Sectors]	qgs_fid	geom	NULL	Animal Control Sectors	1	NULL
4	4	[Assisted Living Facilities - Large]	qgs_fid	geom	NULL	Assisted Living Facilities - Large	1	NULL
5	5	[Assisted Living Facilities - Small]	qgs_fid	geom	NULL	Assisted Living Facilities - Small	1	NULL
6	6	[Bike Howard - Recommendations Off Road]	qgs_fid	geom	NULL	Bike Howard - Recommendations Off Road	1	NULL
7	7	[Bike Howard - Spot Improvements]	qgs_fid	geom	NULL	Bike Howard - Spot Improvements	1	NULL
8	8	[Bike Howard Recommendations - On Road]	qgs_fid	geom	NULL	Bike Howard Recommendations - On Road	1	NULL
9	9	[Bridge Decks]	qgs_fid	geom	NULL	Bridge Decks	1	NULL
10	10	[Bridges]	qgs_fid	geom	NULL	Bridges	1	NULL
11	11	[Buildings_Major]	qgs_fid	geom	NULL	Buildings_Major	1	NULL

Figure 76: Layers table populated with layer information in the OntologyHolding table.

A.2 Extracting Indexes

Once all tables are loaded into the database, the next step is to start the process of linking these tables. We start by creating a basic index on the imported records. We will create two indexes: One for keywords and one for object geometries.

A.2.1 Loading Features (tblFeature)

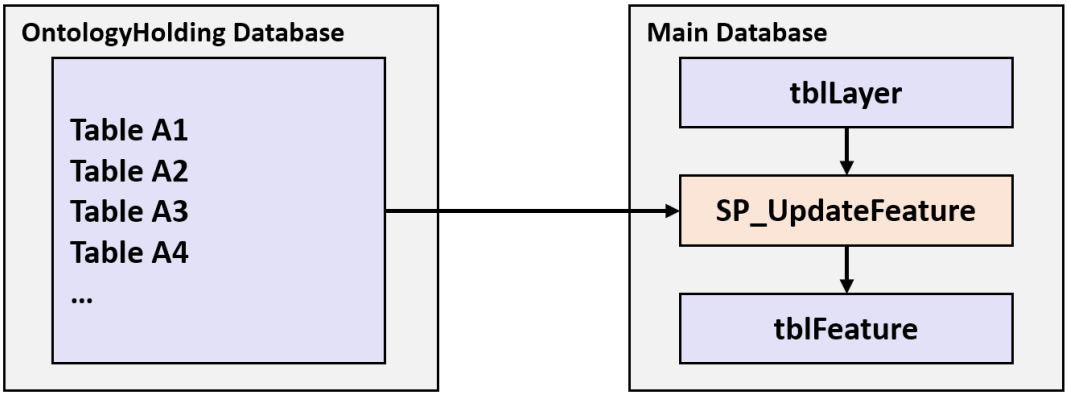
The **tblFeature** table includes every feature in the model. The 120 layers included in the model contain a total of 562,211 records (each representing a geographic object).

Algorithm A2: Create tblFeature

```
CREATE TABLE [dbo].[tblFeature](
	[pkFeatureID] [int] IDENTITY(1,1) NOT NULL,
	[fkLayerID] [int] NULL,
	[UniqueID] [uniqueidentifier] NULL,
	[PrimaryKey] [int] NULL,
	[Descriptor] [varchar](50) NULL,
	[geom] [geometry] NULL,
 CONSTRAINT [PK_tblObject] PRIMARY KEY CLUSTERED
(
	[pkFeatureID] ASC
```

```
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY
= OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
```

If the basis of the model is a simple data warehouse with a star schema then tblFeature acts as the fact table in this warehouse.



A stored procedure, **sp_UpdateFeatures** scans the rows in the holding table that should be imported (as designated in tblLayer) and uses SQL Server's merge function to efficiently load only new or updated records.

Algorithm A3: Updating Features (fact table)

```
CREATE PROCEDURE [dbo].[sp_UpdateFeatures]
AS
BEGIN
	SET NOCOUNT ON;

	CREATE TABLE #TempTable (fkLayerID int,PrimaryKey int,geom
geometry)

	DECLARE @TempQuery varchar(max);

	DECLARE @MyLayerID int;
	DECLARE @MyPrimaryKeyColumn varchar(100);
	DECLARE @MyGeometryColumn varchar(100);
	DECLARE @MyTableName varchar(100);

	DECLARE MyCursor CURSOR
	FOR
	SELECT pkLayerID, PrimaryKeyColumn, GeometryColumn,TableName
	FROM tblLayer
```

```
	OPEN MyCursor

	FETCH NEXT FROM MyCursor INTO @MyLayerID, @MyPrimaryKeyColumn,
@MyGeometryColumn, @MyTableName

	WHILE @@FETCH_STATUS = 0
		BEGIN
			SET @TempQuery = 'SELECT ' + STR(@MyLayerID) + ','
+ @MyPrimaryKeyColumn + ',' + @MyGeometryColumn + ' FROM
OntologyHolding.dbo.' + @MyTableName;
			PRINT @TempQuery;
			INSERT INTO #TempTable (fkLayerID,PrimaryKey,geom)
			EXECUTE(@TempQuery)

			FETCH NEXT FROM MyCursor INTO @MyLayerID,
@MyPrimaryKeyColumn, @MyGeometryColumn, @MyTableName;

		END;

	CLOSE MyCursor;
	DEALLOCATE MyCursor;

	MERGE tblFeature AS t
	USING (SELECT fkLayerID, PrimaryKey, geom FROM #TempTable) AS
s
	ON t.fkLayerID = s.fkLayerID and t.PrimaryKey = s.PrimaryKey
		--Update
		WHEN MATCHED AND NOT t.geom.STEquals(s.geom) = 1 THEN
			UPDATE SET t.geom = s.geom
		--Insert
		WHEN NOT MATCHED BY TARGET THEN
			INSERT (fkLayerID,UniqueID,PrimaryKey,geom) VALUES
(s.fkLayerID,NEWID(),s.PrimaryKey,s.geom)
		--Delete
		WHEN NOT MATCHED BY SOURCE THEN
			DELETE;

	DROP TABLE #TempTable
```

The stored procedure re-scans all of the tables in the ontology table and checks to see if there are any updated or added records when compared to the rows in tblFeature and only inserts or updates those records as necessary. If a record in tblFeature no longer has a counterpart in the source tables then it is removed from tblFeature.

The result is a table containing references to the features in the sources tables (in the holding database):

Results | Spatial results | Messages

	pkFeatureID	fkLayerID	UniqueID	PrimaryKey	Descriptor	geom
45...	450986	117	70B15C49-DC70-4B5B-A83A-C47813111F0F	533	NULL	0xC80800000104FD000000131371EB3FFC344141EF1180D...
45...	450987	117	702D2D8A-1EAC-4393-BE92-71DC64C329A0	536	NULL	0xC80800000104FFF20000CFA4A81011833441552347AA7...
45...	450988	117	72F206BF-300C-43C1-B0DD-55FDFF42AF80	534	NULL	0xC80800000104BF000000CA37363888F4344125C123858...
45...	450989	117	85F6AB32-1E4F-4AF1-9260-CA649956C170	537	NULL	0xC80800000104BA3B0000AF1544254967344 1A5B2306D5...
45...	450990	49	2C5E3431-B968-406D-9141-8B74B94A5F95	733	NULL	0xC8080000010409000000051A797F02CA634414028F7A81...
45...	450991	117	3496CE1D-22CA-4DD1-8DD8-23FC48A5DFD7	538	NULL	0xC8080000010439000000 9B5CE74E026A3441E75A0FFC8...
45...	450992	49	72D5BFBF-F2ED-4924-8090-7FE4387A940D	734	NULL	0xC8080000010408B0000001F190685519E3441EC3603296...
45...	450993	117	C238B39A-9B35-4650-8D1E-6968A4E39851	531	NULL	0xC8080000010408B010000C76C244A74EE344103441CAE2...
45...	450994	49	7A222145-069F-4989-A0EE-F7DA880A8274	732	NULL	0xC8080000010408000000064BAA531489A3441247A9DB9C...
45...	450995	49	FF9D71C9-1D03-429F-8CF8-5F1CF74697DE	737	NULL	0xC8080000010408A000000D44B296181BA344173013F330...
45...	450996	49	049B7789-787B-4607-B467-1B45B50F3A97	738	NULL	0xC8080000010405000000 1463B2F5D2783441357AB9CCF...
45...	450997	49	F68312CF-9E45-4BFE-8768-47A7E24A9D34	736	NULL	0xC8080000010409000000095DF7959C7B33441A008B9474...
45...	450998	117	64FD7BDA-4182-4F9B-ADB7-F0C243B40A18	535	NULL	0xC8080000010481000000058EA1872FEB34410902BFF59...
45...	450999	49	FD338A62-A480-45D1-A107-6FDFBB81B28E	731	NULL	0xC8080000010409000000 9E66581A75A53441D688E4489...

Figure 77: Sample rows from the Features Table.

Figure 77 shows records in tblFeature that reference the "Historic Sites" layer (fkLayerID = 49) and the Zoning layer (fkLayerID = 117). The pkFeatureID is specific to tblFeature and the PrimaryKey is the associated key from the source table. The geometry is carried over in the geom column.

It is recommended that the indexes of tblFeature, particularly the spatial index be rebuilt/reorganized periodically, particularly after a major update.

Updating Layer Distance Distributions

Once the features have been loaded into the databases it is necessary to calculate the distribution of distances for any given layer since different layers have differing concepts of "near." For example, two nearby properties might be hundreds of feet away but nearby parks might be a mile or two away from each other. This is generally a one-time calculation since adding one additional feature to a layer (such a new park) should not dramatically change the average distance between features.

This calculation is completed with **sp_CalculateLayerDistances** described below.

Algorithm A4: sp_CalculateLayerDistances

```
CREATE PROCEDURE sp_CalculateLayerDistances
      @MyLayerID int
AS
BEGIN

      SET NOCOUNT ON;

      DECLARE @MyAverage float;
      DECLARE @MyStDev float;

    select @MyAverage = Avg(x.geom.STDistance(y.geom)), @MyStDev =
StDev(x.geom.STDistance(y.geom))
      from (select top 1 * from dbo.tblFeature where fkLayerID =
@MyLayerID) x, (select top 100 * from dbo.tblFeature where fkLayerID
= @MyLayerID) y
      where x.pkFeatureID != y.pkFeatureID

      UPDATE tblLayer
      SET DistanceAverage = @MyAverage, DistanceStDev = @MyStDev
      WHERE pkLayerID = @MyLayerID


END
```

The result is tblLayer populated with the average and standard deviation of distances between two typical features inside that layer.

Results | Messages

	pkLayerID	TableName	PrimaryKeyColumn	GeometryColumn	DistanceAverage	DistanceStDev
1	1	[Address Points]	qgs_fid	geom	941.259675676536	579.534173875469
2	2	[Age Restricted Housing]	qgs_fid	geom	34043.2920608363	19239.4284603132
3	3	[Animal Control Sectors]	qgs_fid	geom	28156.1965187192	16501.3790327911
4	4	[Assisted Living Facilities - Large]	qgs_fid	geom	28925.7701350113	13415.146805427
5	5	[Assisted Living Facilities - Small]	qgs_fid	geom	38134.5719506551	20782.6808604961
6	6	[Bike Howard - Recommendations Off Road]	qgs_fid	geom	19442.1327675619	9486.45278072138
7	7	[Bike Howard - Spot Improvements]	qgs_fid	geom	23815.0488609441	18379.3493327661
8	8	[Bike Howard Recommendations - On Road]	qgs_fid	geom	29170.3118222755	20203.221928572
9	9	[Bridge Decks]	qgs_fid	geom	19503.5263520955	11142.3232912372

Figure 78: Layer Table populated with distance distribution information.

A.2.2 Keyword Index (tblFeature_Keyword)

The index for keywords serves as a method to quickly identify objects by name (since this is still an easy and common way to find objects). A user should be able

search for the name “Patapsco” and quickly receive references to geographic objects with that name including “Patapsco Middle School” or “Patapsco River.”

The table **tblLayer_Keyword**, which will contain the keywords in our system is defined below:

Algorithm A5: Create tblLayer_Keyword

```
CREATE TABLE [dbo].[tblLayer_Keyword](
	[pkLayerKeywordID] [int] IDENTITY(1,1) NOT NULL,
	[fkLayerID] [int] NULL,
	[fkFeatureID] [int] NULL,
	[KeywordColumn] [varchar](50) NULL,
 CONSTRAINT [PK_tblClassKeyword] PRIMARY KEY CLUSTERED
(
	[pkLayerKeywordID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY
= OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
```

This table includes a reference to the layer being indexed (fkLayerID as well as a reference the text field in the table that will be included in the index (KeywordColumn). It is manually created by subject matter experts who select the fields that are suitable for a keyword search. One layer can have multiple keyword columns indexed. In Figure 79, the zip code layer (fkLayerID = 116) has two fields marked as a keyword search: ZIPCODE representing the five-digit numerical code and AREANAME representing the city associated with that zip code.

Results | Messages

	pkLayerKeywordID	fkLayerID	fkFeatureID	KeywordColumn
44	44	103	NULL	StatArea20
45	45	106	NULL	ROADNAME
46	46	114	NULL	PRECINCT20
47	47	115	NULL	MDE8NAME
48	48	116	NULL	ZIPCODE
49	49	116	NULL	AREANAME
50	50	117	NULL	ZONE

Figure 79: Sample records from tblLayer_Keyword. The zip code layer has two entries.

Not every layer in the model has keywords associated with them. For example, a floodplain is geographically distinct but no names or labels are associated with floodplains. In our example, only 49 out of the 114 total layers have keywords associated with them and one has two fields indexed as keywords. This provides a total of 50 records in this example.

Another table, **tblFeature_Keyword** contains the keywords extracted from the layers based on the keyword column specified in tblLayer_Keyword.

Algorithm A6: Create tblFeature_Keyword

```
CREATE TABLE [dbo].[tblFeature_Keyword](
	[pkFeatureKeywordID] [int] IDENTITY(1,1) NOT NULL,
	[fkLayerKeywordID] [int] NULL,
	[fkFeatureID] [int] NULL,
	[KeywordName] [varchar](max) NULL,
 CONSTRAINT [PK_tblObjectKeyword] PRIMARY KEY CLUSTERED
(
	[pkFeatureKeywordID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY
= OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
```

This table includes a reference to tblLayer_Keyword (fkLayerKeywordID), a reference to the individual feature in tblFeature referenced by the keyword (fkFeatureID) and the actual text keyword that can be searched for (KeywordName).

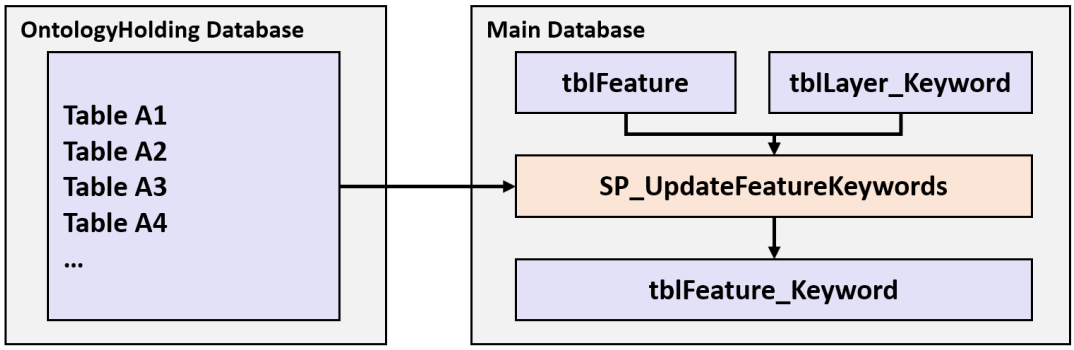


Figure 80: Loading feature keywords with sp_UpdateFeatureKeywords.

A stored procedure, **sp_UpdateFeatureKeywords** queries the source tables in the holding database for the keyword columns specified in tblLayer_Keyword and links them to the correct indexed feature in tblFeature.

Algorithm A7: Stored Procedure sp_UpdateFeatureKeywords

```
CREATE PROCEDURE [dbo].[sp_UpdateFeatureKeywords]
AS
BEGIN
	SET NOCOUNT ON;

	CREATE TABLE #TempTable (fkLayerKeywordID int,fkFeatureID
int,keywordname varchar(MAX))

	DECLARE @TempQuery varchar(max);

	DECLARE @MyLayerKeywordID int;
	DECLARE @MyLayerID int;
	DECLARE @MyKeywordColumn varchar(100);
	DECLARE @MyPrimaryKeyColumn varchar(100);
	DECLARE @MyTableName varchar(100);

	DECLARE MyCursor CURSOR
	FOR
	SELECT pkLayerKeywordID, fkLayerID,
KeywordColumn,TableName,PrimaryKeyColumn
	FROM tblLayer_Keyword
	LEFT OUTER JOIN tblLayer
		ON tblLayer.pkLayerID = tblLayer_Keyword.fkLayerID

	OPEN MyCursor

	FETCH NEXT FROM MyCursor INTO @MyLayerKeywordID, @MyLayerID,
@MyKeywordColumn, @MyTableName, @MyPrimaryKeyColumn

	WHILE @@FETCH_STATUS = 0
```

```
            BEGIN

                  SET @TempQuery = 'SELECT ' +
STR(@MyLayerKeywordID) + ',' + @MyKeywordColumn + ', + pkFeatureID
FROM OntologyHolding.dbo.' + @MyTableName + 'x LEFT OUTER JOIN
dbo.tblFeature f ON f.PrimaryKey = x. ' + @MyPrimaryKeyColumn + '
WHERE fkLayerID = ' + STR(@MyLayerID) + ' AND ' + @MyKeywordColumn +
' IS NOT NULL'

                  PRINT @TempQuery;
                  INSERT INTO #TempTable
(fklayerkeywordID,keywordname,fkFeatureID)
                  EXECUTE(@TempQuery)

                  FETCH NEXT FROM MyCursor INTO @MyLayerKeywordID,
@MyLayerID, @MyKeywordColumn, @MyTableName, @MyPrimaryKeyColumn;

            END;

      CLOSE MyCursor;
      DEALLOCATE MyCursor;

      MERGE tblFeature_Keyword AS t
      USING (SELECT fkLayerKeywordID, fkFeatureID, keywordname FROM
#TempTable) AS s
      ON t.fkLayerKeywordID = s.fkLayerKeywordID AND  t.fkFeatureID
= s.fkFeatureID
            --Update
            WHEN MATCHED AND NOT t.KeywordName = s.keywordname THEN
                  UPDATE SET t.KeywordName =s.keywordname
            --Insert
            WHEN NOT MATCHED BY TARGET THEN
                  INSERT (fkLayerKeywordID,fkFeatureID,KeywordName)
VALUES (s.fkLayerKeywordID,s.fkFeatureID,s.KeywordName)
            --Delete
            WHEN NOT MATCHED BY SOURCE THEN
                  DELETE;

      DROP TABLE #TempTable

      PRINT @@ROWCOUNT;

END
```

The result of this stored procedure is tblFeature_Keyword populated with a list of keywords associated with their IDs from tblFeature.

Results Messages

	pkFeatureKeywordID	fkLayerKeywordID	fkFeatureID	KeywordName
382	120924	40	108238	Howard
383	120934	40	108239	Centennial HS
384	120927	40	108240	Wilde Lake
385	120914	40	108243	Oakland Mills
386	120923	40	108246	Hammond
387	120925	40	108250	River Hill
388	120803	36	110700	Dayton Oaks ES
389	120804	36	110702	Cradlerock ES
390	120809	36	110740	Deep Run ES

Figure 81: tblFeature_Keyword populated with keywords.

Querying Features By Keyword

In a GIS application, if a user searches for a geographic object by feature, they can use the stored procedure **sp_QueryFeaturesByKeyword**.

Algorithm A8: sp_QueryFeaturesByKeyword

```
CREATE PROCEDURE [dbo].[sp_QueryFeatureByKeyword]
      @MySearchTerm varchar(50),
      @NumResults int
AS
BEGIN

      SET NOCOUNT ON;

    SELECT TOP (@NumResults) k.KeywordName, k.fkFeatureID,
l.TableName, l.TableTitle, f.geom as MyGeom
      FROM dbo.tblFeature_Keyword k
      LEFT OUTER JOIN dbo.tblFeature f
            on f.pkFeatureID = k.fkFeatureID
      LEFT OUTER JOIN dbo.tblLayer l
            on l.pkLayerID = f.fkLayerID
      WHERE KeywordName like '%' + @MySearchTerm + '%'
      ORDER BY fkFeatureID

END
```

This stored procedure takes two inputs: A search term designated by the user and the number of results they want returned to them. A search for “Patapsco” generates the results as shown in Figure 82.

Results | Spatial results | Messages

	KeywordName	fkFeatureID	TableName	TableTitle	MyGeom
1	Patapsco	136939	[Schools - Middle]	Schools - Middle	0xC8080000010C1E12A54F71D0344162109A173E222241
2	8914 MOUNT PATAPSCO CT	143042	[Address Points]	Address Points	0xC8080000010C95F78A6B61C434419FCA48611E492241
3	8905 MOUNT PATAPSCO CT	143043	[Address Points]	Address Points	0xC8080000010C4D6AC3B5DCC53441A8E053E1114A2241
4	8910 MOUNT PATAPSCO CT	143044	[Address Points]	Address Points	0xC8080000010C1205862B87C43441E68DF67A914A2241
5	16118 PATAPSCO OVERLOOK CT	153703	[Address Points]	Address Points	0xC8080000010C4B4B7ABD9FB53341F21F35DC2EBA2241
6	16101 PATAPSCO OVERLOOK CT	153742	[Address Points]	Address Points	0xC8080000010CFA4F0ED76EB133415DEAD6F5DCB62241
7	16125 PATAPSCO OVERLOOK CT	153749	[Address Points]	Address Points	0xC8080000010CFEDB7E5462B2334113680B8553BB2241
8	16112 PATAPSCO OVERLOOK CT	153750	[Address Points]	Address Points	0xC8080000010CBA5353F8EBB3334192FF2C0A58B92241
9	16119 PATAPSCO OVERLOOK CT	153758	[Address Points]	Address Points	0xC8080000010CB3F268268EB13341EAA76A66C0BB2241
10	16107 PATAPSCO OVERLOOK CT	153767	[Address Points]	Address Points	0xC8080000010CBB6AB707D7B133417588663D21B82241
11	16124 PATAPSCO OVERLOOK CT	153788	[Address Points]	Address Points	0xC8080000010C31257114B7B43341F84233856BBB2241
12	16113 PATAPSCO OVERLOOK CT	153823	[Address Points]	Address Points	0xC8080000010C33912D5C15B233411D190E00AAB92241
13	16130 PATAPSCO OVERLOOK CT	154131	[Address Points]	Address Points	0xC8080000010CC47D3FA1C5B33341F768DCA306BC2241
14	16106 PATAPSCO OVERLOOK CT	154133	[Address Points]	Address Points	0xC8080000010C0D3C10EE65B333415A0518051BB72241
15	16100 PATAPSCO OVERLOOK CT	154154	[Address Points]	Address Points	0xC8080000010C1291B31E56B333417032DC7AB9B52241

Figure 82: A keyword search for "Patapsco."

A.3 Ontology Creation

This section will outline the tables and procedures that describe the relationship between two layers.

A.3.1 Relationship Type (tblLayer_Relation_Type)

tblLayer_Relation_Type is a reference table describing the types of topological relationships between different layers as described in Section 4.2.1. It can be expanded to include new types of relationships that might be useful to an organization in the future. At the moment it is populated with the relationships described in the dissertation (the standard OGR topological relationships).

Algorithm A9: Create tblLayer_Relation_Type

```
CREATE TABLE [dbo].[tblLayer_Relation_Type](
	[pkRelationshipTypeID] [int] IDENTITY(1,1) NOT NULL,
	[RelationshipTypeName] [varchar](50) NULL,
 CONSTRAINT [PK_tblRelationshipType] PRIMARY KEY CLUSTERED
(
	[pkRelationshipTypeID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY
= OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
```

Results | Messages

	pkRelationshipTypeID	RelationshipTypeName
1	1	Distance
2	2	Contains
3	3	Within
4	4	Intersects
5	5	Overlaps
6	6	Touches
7	7	Feature

Figure 83: tblLayer_Relation_Type populated with the standard OGR topological relationships.

A.3.2 Context Profiles (tblLayer_ContextProfile)

Context profiles define the situation of a user (which informs the reason they're using the system) and are stored in **tblLayer_ContextProfile**.

Algorithm A10: Create tblLayer_ContextProfile

```
CREATE TABLE [dbo].[tblLayer_ContextProfile](
	[pkContextProfileID] [int] IDENTITY(1,1) NOT NULL,
	[ContextProfileName] [varchar](200) NULL,
 CONSTRAINT [PK_tblLayer_ContextProfile] PRIMARY KEY CLUSTERED
(
	[pkContextProfileID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
```

Context profiles can be created computationally (for example, by analyzing WMS logs as mentioned in Section 4.2.1) or they can be created manually based on user needs.

Results | Messages

	pkContextProfileID	ContextProfileName
1	1	Default
2	2	Looking for Parks
3	3	Looking to Move (Real Estate)

Figure 84: tblLayer_ContextProfile with some sample profiles

A.3.3 Layer Relationships (tblLayer_Relation)

An ontology describes the relationships between classes of objects and this information is stored in tblLayer_Relation. These relationships are tagged to a context profile (tblLayer_ContextProfile) meaning that the same two layers can have two different types of relationships depending on the user's context. All of this is stored in **tblLayer_Relation**.

Algorithm A11: Create tblLayer_Relation

```
CREATE TABLE [dbo].[tblLayer_Relation](
	[pkRelationshipID] [int] IDENTITY(1,1) NOT NULL,
	[fkContextProfileID] [int] NULL,
	[fkLayerOneID] [int] NULL,
	[fkLayerTwoID] [int] NULL,
	[fkRelationshipTypeID] [int] NULL,
	[RelationshipStrength] [float] NULL,
 CONSTRAINT [PK_tblRelationship] PRIMARY KEY CLUSTERED
(
	[pkRelationshipID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY
= OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
```

A record in tblLayer_Relation includes a reference to a context profile (fkContextProfileID), the two layers involved in the relationship (fkLayerOneID, fkLayerTwoID), the type of relationship defined in tblLayer_Relation_Type (fkRelationshipTypeID) and the strength of the relationship (RelationshipStrength) which is a number between 0 and 1 where a higher number signifies a stronger relationship between the layers.

Bibliography

[1] Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J. P., Moreno, Y. and Porter, M. A. Multilayer Networks. *arXiv.org*, physics.soc-ph (Sep 27 2013).

[2] Aleroud, A. Contextual information fusion for the detection of cyber-attacks. (2014).

[3] Zhuge, H. Communities and Emerging Semantics in Semantic Link Network: Discovery and Learning. *Knowledge and Data Engineering, IEEE Transactions on*, 21, 6 (2009), 785-799.

[4] Tobler, W. R. *A Computer Movie Simulating Urban Growth in the Detroit Region. Vol. 46*, 1970.

[5] Wieczorek, W. F. and Delmerico, A. M. Geographic Information Systems. *Comput Stat*, 1, 2 (2009), 167-186.

[6] ESRI. Retrieved from https://www.esri.com/en-us/home.

[7] Pitney Bowes, I. Mapinfo (2019).

[8] Manifold. Retrieved from http://www.manifold.net.

[9] Geoserver. 2019. Retrieved from http://geoserver.org.

[10] OpenLayers. 2019. Retrieved from https://openlayers.org.

[11] Bhattacharjee, S., Prasad, R. R., Dwivedi, A., Dasgupta, A. and Ghosh, S. K. Ontology based framework for semantic resolution of geospatial query. *ISDA* (2012), 437-442.

[12] Shapefile. 2019. Retrieved from https://en.wikipedia.org/wiki/Shapefile.

[13] Cohn, A. G. and Renz, J. Qualitative spatial representation and reasoning. *Foundations of Artificial Intelligence* (2008).

[14] Albrecht, J. Universal Analytical GIS Operations - A Task-Oriented Systematization of Data Structure-Independent GIS Functionality. *Geographic Information Research: Transatlantic Perspectives* (1998), 577-592.

[15] Chen, H. *An intelligent broker architecture for context-aware systems, A PhD*, 2003.

[16] Weiser, M. The computer for the 21st century. *Scientific american* (1991).

[17] Ayed, D., Delanote, D. and Berbers, Y. MDD Approach for the Development of Context-Aware Applications. *CONTEXT*, 4635, Chapter 2 (2007), 15-28.

[18] Context. Merriam-Webster, n.d. Retrieved from Merriam-Webster.com.

[19] Chen, G. and Kotz, D. A survey of context-aware mobile computing research (2000).

[20] Zimmermann, A., Lorenz, A. and Oppermann, R. An Operational Definition of Context. *CONTEXT*, 4635, Chapter 42 (2007), 558-571.

[21] Brown, P. J., Bovey, J. D. and Chen, X. Context-aware applications: from the laboratory to the marketplace. *IEEE personal communications* (1997).

[22] Gross, T. and Specht, M. *Awareness in Context-Aware Information Systems*. Vieweg+Teubner Verlag, City, 2001.

[23] Winograd, T. Architectures for Context. *Human-Computer Interaction*, 16, 2 (2001), 401-419.

[24] Brézillon, P. Using Context for Supporting Users Efficiently. *HICSS* (2003), 9 pp.

[25] Dey, A. K., Abowd, G. D. and Salber, D. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. *Human-Computer Interaction*, 16, 2 (2001), 97-166.

[26] Salber, D., Dey, A. K. and Abowd, G. D. The Context Toolkit - Aiding the Development of Context-Enabled Applications. *CHI* (1999), 434-441.

[27] Agre, P. E. Changing Places - Contexts of Awareness in Computing. *Human-Computer Interaction*, 16, 2 (2001), 177-192.

[28] Bazire, M. and Brézillon, P. Understanding Context Before Using It. *CONTEXT*, 3554, Chapter 3 (2005), 29-40.

[29] Erickson, T. Some problems with the notion of context-aware computing. *Commun. ACM*, 45, 2 (2002), 102-104.

[30] Chen, H. and Tolia, S. Steps towards creating a context-aware software agent system. *HP Laboratories Palo Alto HPL-2001* (2001).

[31] Hull, R., Neaves, P. and Bedford-Roberts, J. Towards situated computing. In *Proceedings of the Digest of Papers. First International Symposium on Wearable Computers* (1997). IEEE, [insert City of Publication],[insert 1997 of Publication].

[32] Hanssens, N., Kulkarni, A., Tuchida, R. and Horton, T. Building Agent-Based Intelligent Workspaces. *International Conference on Internet Computing* (2002).

[33] Dey, A. K. Providing architectural support for building context-aware applications (2000).

[34] Abowd, G. D., Dey, A. K., Brown, P. J., Davies, N., Smith, M. and Steggles, P. Towards a Better Understanding of Context and Context-Awareness. *HUC*, 1707, Chapter 29 (1999), 304-307.

[35] McCarthy, J. and Buvac, S. Formalizing context (expanded notes) (1997).

[36] Castro, P. and Muntz, R. Using context to assist in multimedia object retrieval. *First International Workshop on Multimedia Intelligent …* (1999).

[37] Schmidt, A., Beigl, M. and Gellersen, H. W. *There is more to Context than Location*.

[38] Vieira, V., Tedesco, P. A. and Salgado, A. C. Designing context-sensitive systems - An integrated approach. *Expert Syst. Appl.* (2011).

[39] Hong, J. I. and Landay, J. A. An Infrastructure Approach to Context-Aware Computing. *Human-Computer Interaction*, 16, 2 (2001), 287-303.

[40] Schilit, B., Adams, N. and Want, R. Context-aware computing applications. In *Proceedings of the Mobile Computing Systems and Applications, 1994. Proceedings., Workshop on* (1994). IEEE, [insert City of Publication],[insert 1994 of Publication].

[41] Perera, C., Zaslavsky, A., Christen, P. and Georgakopoulos, D. Context Aware Computing for The Internet of Things: A Survey. *IEEE Communications Surveys & Tutorials*, 16, 1 (Feb 05 2014), 414-454.

[42] Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A. and Riboni, D. A survey of context modelling and reasoning techniques. *Pervasive and Mobile Computing*, 6, 2 (2010), 161-180.

[43] Englemore, R. and Morgan, A. *Blackboard Systems; Edited by Robert Engelmore, Tony Morgan (the Insight Series in Artificial Intell*. Addison-Wesley Longman Publishing Co., Inc., 1988.

[44] Ontology (information science). Wikipedia, the free encyclopedia. Retrieved from https://en.wikipedia.org/wiki/Ontology_(information_science).

[45] Noy, N. F. and McGuinness, D. L. Ontology development 101: A guide to creating your first ontology (2001).

[46] Uschold, M. and Gruninger, M. Ontologies - principles, methods and applications. *Knowledge Eng. Review*, 11, 02 (1996), 93.

[47] Studer, R., Benjamins, V. R. and Fensel, D. Knowledge engineering: Principles and methods. *Data & Knowledge Engineering*, 25, 1-2 (1998), 161-197.

[48] YE, J., COYLE, L., DOBSON, S. and NIXON, P. Ontology-based models in pervasive computing systems. *The Knowledge Engineering Review*, 22, 04 (2007).

[49] Topcu, F. Context Modeling and Reasoning Techniques. *SNET Seminar in the ST* (2011).

[50] Finin, T., Fritzson, R., McKay, D. and McEntire, R. *KQML as an agent communication language*. ACM, New York, New York, USA, 1994.

[51] OWL Web Ontology Language Reference. Retrieved from http://www.w3.org/TR/owl-ref/.

[52] Wang, X., Zhang, D., Gu, T. and Pung, H. K. Ontology Based Context Modeling and Reasoning using OWL. *PerCom Workshops* (2004).

[53] Lenat, D. The dimensions of context-space. *available online at URL http://www casbah org/ …* (1998).

[54] Goodchild, M. F., Yuan, M. and Cova, T. J. Towards a general theory of geographic representation in GIS. *Int J Geogr Inf Sci*, 21, 3 (2007), 239-260.

[55] Kuhn, W. Geospatial Semantics - Why, of What, and How? *J Data Semantics III*, 3534, 1 (Jan 01 2005), 1-24.

[56] Smith, B. and Mark, D. M. Ontology and geographic kinds (1998).

[57] Navarrete, T. and Blat, J. An Algorithm for Merging Geographic Datasets Based on the Spatial Distributions of Their Values. *GeoS*, 4853, Chapter 5 (2007), 66-81.

[58] Hu, Y. Geospatial Semantics. *arXiv*, cs.CL (Jul 12 2017), 80-94.

[59] Van Laere, O., Quinn, J., Schockaert, S. and Dhoedt, B. Spatially Aware Term Selection for Geotagging. *IEEE Transactions on Knowledge and Data Engineering*, 26, 1 (2014), 221-234.

[60] Moura, T. H. V. M., Davis Jr., C. A. and Fonseca, F. T. Reference data enhancement for geographic information retrieval using linked data. *Transactions in GIS*, 21, 4 (Nov 02 2016), 683-700.

[61] Lacasta, J., Nogueras-Iso, J., Béjar, R., Muro-Medrano, P. R. and Zarazaga-Soria, F. J. A Web Ontology Service to facilitate interoperability within a Spatial Data Infrastructure: Applicability to discovery. *Data & knowledge engineering*, 63, 3 (2007), 947-971.

[62] Buccella, A., Cechich, A., Gendarmi, D. and Lanubile, F. Building a global normalized ontology for integrating geographic data sources. *Computers and Geosciences*, 37, 7 (2011).

[63] Laurini, R. A conceptual framework for geographic knowledge engineering. *Elsevier* (Oct 19 2013).

[64] Karalopoulos, A., Kokla, M. and Kavouras, M. *Comparing Representations of Geographic Knowledge Expressed as Conceptual Graphs*. Springer, Berlin, Heidelberg, City, 2005.

[65] Third, A., Bennett, B. and Mallenby, D. *Architecture for a Grounded Ontology of Geographic Information*. Springer, Berlin, Heidelberg, City, 2007.

[66] Sen, S. *Two Types of Hierarchies in Geospatial Ontologies*. Springer, Berlin, Heidelberg, City, 2007.

[67] Torres, M., Quintero, R., Moreno, M. and Fonseca, F. *Ontology-Driven Description of Spatial Data for Their Semantic Processing*. Springer, Berlin, Heidelberg, City, 2005.

[68] Buccella, A., Cechich, A., Gendarmi, D., Lanubile, F. and 2011 Building a global normalized ontology for integrating geographic data sources. *Computers and Geosciences* (Apr 08 2011).

[69] Stock, K. and Cialone, C. *An Approach to the Management of Multiple Aligned Multilingual Ontologies for a Geospatial Earth Observation System*. Springer, Berlin, Heidelberg, City, 2011.

[70] Jung, C. T., Sun, C. H. and Yuan, M. An ontology-enabled framework for a geospatial problem-solving environment. *Computers, Environment and Urban Systems* (Jan 11 2013).

[71] Zhao, T., Zhang, C., Wei, M. and Peng, Z.-R. Ontology-Based Geospatial Data Query and Integration. *GIScience* (2008).

[72] Randell, D. A., Cui, Z. and Cohn, A. G. A Spatial Logic based on Regions and Connection. *KR* (1992).

[73] Grenon, P. and Smith, B. SNAP and SPAN - Towards Dynamic Spatial Ontology. *Spatial Cognition & Computation*, 4, 1 (2004), 69-104.

[74] Duckham, M. and Worboys, M. An algebraic approach to automated geospatial information fusion. *Int J Geogr Inf Sci*, 19, 5 (2005), 537-557.

[75] Formica, A., Mazzei, M., Pourabbas, E. and Rafanelli, M. Enriching the semantics of the directed polyline–polygon topological relationships: the DLP-intersection matrix. *Journal of Geographical Systems* (Feb 27 2017).

[76] van den Brink, L., Janssen, P., Quak, W. and Stoter, J. Towards a high level of semantic harmonisation in the geospatial domain. *Computers, Environment and Urban Systems*, 62, C (Mar 01 2017), 233-242.

[77] Yue, P., Gong, J., Di, L., He, L. and Wei, Y. Integrating semantic web technologies and geospatial catalog services for geospatial information discovery and processing in cyberinfrastructure. *GeoInformatica*, 15, 2 (Oct 09 2009), 273-303.

[78] Nieland, S., Kleinschmit, B. and Förster, M. Using ontological inference and hierarchical matchmaking to overcome semantic heterogeneity in remote sensing-based biodiversity monitoring. *International Journal of Applied Earth Observation and Geoinformation* (Oct 16 2014).

[79] Han, Y. and Xu, W. An ontology-oriented decision support system for emergency management based on information fusion. In *Proceedings of the Proceedings of the 1st ACM SIGSPATIAL International …* (2015), [insert City of Publication],[insert 2015 of Publication].

[80] Chaabane, S. and Jaziri, W. A novel algorithm for fully automated mapping of geospatial ontologies. *Journal of Geographical Systems*, 20, 1 (2017).

[81] Janowicz, K., Raubal, M. and Kuhn, W. The semantics of similarity in geographic information retrieval. *Journal of Spatial Information Science*, 2 (May 25 2011).

[82] Reichenbacher, T. and De Sabbata, S. Geographic relevance - different notions of geographies and relevancies. *SIGSPATIAL Special*, 3, 2 (2011), 67-70.

[83] Moreno-Ibarra, M. *Semantic Similarity Applied to Generalization of Geospatial Data*. Springer, Berlin, Heidelberg, City, 2007.

[84] Perry, M., Sheth, A. P., Hakimpour, F. and Jain, P. *Supporting Complex Thematic, Spatial and Temporal Queries over Semantic Web Data*. Springer, Berlin, Heidelberg, City, 2007.

[85] Schlieder, C. *Representing the Meaning of Spatial Behavior by Spatially Grounded Intentional Systems*. Springer, Berlin, Heidelberg, City, 2005.

[86] Adabala, N. and Toyama, K. *Purpose-Driven Navigation*. Springer, Berlin, Heidelberg, City, 2005.

[87] Matyas, C. and Schlieder, C. *A Spatial User Similarity Measure for Geographic Recommender Systems*. Springer, Berlin, Heidelberg, City, 2009.

[88] Janowicz, K. and Wilkes, M. *SIM-DL~A: A Novel Semantic Similarity Measure for Description Logics Reducing Inter-concept to Inter-instance Similarity*. Berlin; Springer, City, 2009.

[89] Zaila, Y. L. and Montesi, D. Geographic information extraction, disambiguation and ranking techniques. In *Proceedings of the Proceedings of the 9th Workshop on Geographic Information Retrieval - GIR '15* (Paris, France, 2015), [insert City of Publication],[insert 2015 of Publication].

[90] George, B. and Shekhar, S. Modeling Spatio-temporal Network Computations - A Summary of Results. *GeoS*, 4853, Chapter 12 (2007), 177-194.

[91] Raper, J. Geographic relevance [information retrieval systems]. *Journal of Documentation*, 63, 6 (2007), 836-852.

[92] De Sabbata, S. and Reichenbacher, T. *A probabilistic model of geographic relevance*. ACM, New York, New York, USA, 2010.

[93] Liu, L., Gao, Y., Lin, X., Guo, X. and Li, H. A framework and implementation for qualitative geographic information retrieval. *Geoinformatics* (2013), 1-4.

[94] Palacio, D., Cabanac, G., Hubert, G., Pinel-Sauvagnat, K. and Sallaberry, C. Prototyping a personalized contextual retrieval framework. In *Proceedings of the Proceedings of the 7th Workshop on Geographic Information Retrieval - GIR '13* (2013), [insert City of Publication],[insert 2013 of Publication].

[95] Wallgrün, J. O., Klippel, A. and Karimzadeh, M. Towards contextualized models of spatial relations. In *Proceedings of the Proceedings of the 9th Workshop on Geographic …* (2015), [insert City of Publication],[insert 2015 of Publication].

[96] Brisaboa, N. R., Luaces, M. R., Places, Á. S. and Seco, D. Exploiting geographic references of documents in a geographical information retrieval system using an ontology-based index. *GeoInformatica*, 14, 3 (2010), 307-331.

[97] Sabino, A. and Rodrigues, A. Indirect location recommendation. In *Proceedings of the Proceedings of the 8th Workshop on Geographic Information Retrieval - GIR '14* (2014), [insert City of Publication],[insert 2014 of Publication].

[98] Mata, F. Geographic Information Retrieval by Topological, Geographical, and Conceptual Matching. *GeoS*, 4853, Chapter 7 (2007), 98-113.

[99] Andrade, F. G. d., Baptista, C. d. S. and Davis, C. A. Improving geographic information retrieval in spatial data infrastructures. *GeoInformatica*, 18, 4 (2014), 793 - 818.

[100] Mata, F. iRank - Ranking Geographical Information by Conceptual, Geographic and Topologic Similarity. *GeoS*, 5892, Chapter 10 (2009), 159-174.

[101] Schwering, A. and Raubal, M. *Measuring Semantic Similarity Between Geospatial Conceptual Regions*. Springer, Berlin, Heidelberg, City, 2005.

[102] Arpinar, I. B., Sheth, A. P., Ramakrishnan, C., Usery, E. L., Azami, M. and Kwan, M.-P. Geospatial Ontology Development and Semantic Analytics. *Trans. GIS*, 10, 4 (2006), 551-575.

[103] Baglioni, M., Masserotti, M. V., Renso, C. and Spinsanti, L. Improving geodatabase semantic querying exploiting. In *Proceedings of the GeoS'11:*

Proceedings of the 4th international conference on GeoSpatial semantics (May, 2011). Springer-Verlag, [insert City of Publication],[insert 2011 of Publication].

[104] Renteria-Agualimpia, W. and Levashkin, S. Multi-criteria Geographic Information Retrieval Model Based on Geospatial Semantic Integration. *GeoS*, 6631, 3 (2011), 166-181.

[105] Li, W., Goodchild, M. F. and Raskin, R. Towards geospatial semantic search: exploiting latent semantic relations in geospatial data. *International Journal of Digital Earth*, 7, 1 (Jan 20 2014), 17-37.

[106] Ahlqvist, O. *Using Semantic Similarity Metrics to Uncover Category and Land Cover Change*. Springer, Berlin, Heidelberg, City, 2005.

[107] Farnaghi, M. and Mansourian, A. Disaster planning using automated composition of semantic OGC web services: A case study in sheltering. *Computers, Environment and Urban Systems*, 41, C (Sep 01 2013), 204-218.

[108] Zhao, P. and Di, L. Semantic web service based geospatial knowledge discovery. … *of 2006 IEEE International Geoscience And …* (2006), 3490-3493.

[109] Chauhan, L. P. S. and Shekhar, S. GeoAI – Accelerating a Virtuous Cycle between AI and Geo. *2021 Thirteen Int Conf Contemp Comput Ic3-2021* (2021), 355-370.

[110] Shekhar, S. What is special about spatial data science and Geo-AI? *33rd Int Conf Sci Statistical Database Management* (2021), 271-271.

[111] Birant, D. and Engineering, A. K. D. K. ST-DBSCAN: An algorithm for clustering spatial–temporal data. *Elsevier*, 60, 1 (2006).

[112] Cai, J., Xie, Y., Deng, M., Tang, X., Li, Y. and Shekhar, S. Significant spatial co-distribution pattern discovery. *Computers, Environment and Urban Systems*, 84 (2020), 101543.

[113] Li, Y. and Shekhar, S. *Local Co-location Pattern Detection: A Summary of Results*. City, 2018.

[114] Yoo, J. S. and Shekhar, S. A Joinless Approach for Mining Spatial Colocation Patterns. *IEEE Transactions on Knowledge and Data Engineering*, 18, 10 (2006), 1323-1337.

[115] Cai, J., Deng, M., Guo, Y., Xie, Y. and Shekhar, S. Discovering regions of anomalous spatial co-locations. *Int J Geogr Inf Sci*, 35, 5 (2020), 1-25.

[116] Xie, Y., Bao, H., Li, Y. and Shekhar, S. Discovering Spatial Mixture Patterns of Interest. *Proc 28th Int Conf Adv Geogr Information Syst* (2020), 608-617.

[117] Tang, X., Gupta, J. and Shekhar, S. Linear Hotspot Discovery on All Simple Paths. *Proc 27th Acm Sigspatial Int Conf Adv Geogr Information Syst* (2019), 476-479.

[118] Shi, L. and Janeja, V. P. Anomalous window discovery through scan statistics for linear intersecting paths (SSLIP). *Proc 15th Acm Sigkdd Int Conf Knowl Discov Data Min - Kdd '09* (2009), 767-776.

[119] Sainani, H., Namayanja, J. M., Sharma, G., Misal, V. and Janeja, V. P. IP Reputation Scoring with Geo-Contextual Feature Augmentation. *Acm Transactions Management Information Syst Tmis*, 11, 4 (2020), 1-29.

[120] Janeja, V. P., Adam, N. R., Atluri, V. and Vaidya, J. Spatial neighborhood based anomaly detection in sensor datasets. *Data Min Knowl Disc*, 20, 2 (2010), 221-258.

[121] Li, Y., Kotwal, P., Wang, P., Shekhar, S. and Northrop, W. Trajectory-aware Lowest-cost Path Selection. *Proc 16th Int Symposium Spatial Temporal Databases* (2019), 61-69.

[122] Xie, Y., Zhou, X. and Shekhar, S. Discovering Interesting Subpaths with Statistical Significance from Spatiotemporal Datasets. *Acm Transactions Intelligent Syst Technology Tist*, 11, 1 (2020), 1-24.

[123] Barabasi, A. L. and Albert, R. Emergence of scaling in random networks. *Science*, 286, 5439 (Oct 15 1999), 509-512.

[124] Dorogovtsev, S. N., Goltsev, A. V. and Mendes, J. F. F. Critical phenomena in complex networks. *arXiv.org*, cond-mat.stat-mech, 4 (Apr 30 2007), 1275-1335.

[125] Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D. and Alon, U. Network motifs: simple building blocks of complex networks. *Science*, 298, 5594 (Oct 25 2002), 824-827.

[126] Kenett, D. Y., Gao, J., Huang, X., Shao, S., Vodenska, I., Buldyrev, S. V., Paul, G., Stanley, H. E. and Havlin, S. Network of Interdependent Networks: Overview of Theory and Applications. *Networks of Networks: The Last Frontier of Complexity*, Chapter 1 (2014), 3-.

[127] Havlin, S., Kenett, D. Y., Ben-Jacob, E., Bunde, A., Cohen, R., Hermann, H., Kantelhardt, J. W., Kertész, J., Kirkpatrick, S., Kurths, J., Portugali, J. and Solomon, S. Challenges in network science: Applications to infrastructures, climate, social systems and economics. *The European Physical Journal Special Topics*, 214 (Nov 2012), 273-293.

[128] De Domenico, M., Sole-Ribalta, A., Omodei, E., Gomez, S. and Arenas, A. Centrality in Interconnected Multilayer Networks. *arXiv.org*, physics.soc-ph (Nov 12 2013), 6868.

[129] Min, B., Do Yi, S., Lee, K.-M. and Goh, K. I. Network robustness of multiplex networks with interlayer degree correlations. *arXiv.org*, physics.soc-ph (Jul 04 2013).

[130] Sole-Ribalta, A., De Domenico, M., Gomez, S. and Arenas, A. Centrality rankings in multiplex networks. In *Proceedings of the WebSci '14: Proceedings of the 2014 ACM conference on Web science* (Jun, 2014). ACM Request Permissions, [insert City of Publication],[insert 2014 of Publication].

[131] Estrada, E. and Higham, D. J. Network Properties Revealed through Matrix Functions. *SIAM review*, 52, 4 (2010), 696-714.

[132] Horvat, E.-A. and Zweig, K. A. One-mode Projection of Multiplex Bipartite Graphs. In *Proceedings of the Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference on* (2012), [insert City of Publication],[insert 2012 of Publication].

[133] Estrada, E. and Gómez-Gardeñes, J. Communicability reveals a transition to coordinated behavior in multiplex networks. *Physical Review E*, 89, 4 (Apr 30 2014), 042819.

[134] Kurant, M. and Thiran, P. Layered Complex Networks. *Physical Review Letters*, 96, 1 (Apr 2006), 138701.

[135] De Domenico, M., Nicosia, V., Arenas, A. and Latora, V. Layer aggregation and reducibility of multilayer interconnected networks. *CoRR abs/1409.5253*, physics.soc-ph (2014), 6864.

[136] Dorogovtsev, S. N., Mendes, J. F. F., Samukhin, A. N. and Zyuzin, A. Y. Organization of modular networks. *arXiv.org*, cond-mat.stat-mech, 5 (Mar 24 2008), 056106.

[137] De Domenico, M., Sole-Ribalta, A., Cozzo, E., Kivelä, M., Moreno, Y., Porter, M. A., Gomez, S. and Arenas, A. Mathematical Formulation of Multi-Layer Networks. *arXiv.org*, physics.soc-ph, 4 (Jul 18 2013), 041022.

[138] Lee, K.-M., Kim, J. Y., Cho, W.-k., Goh, K. I. and Kim, I. M. Correlated multiplexity and connectivity of multiplex random networks. *arXiv.org*, physics.soc-ph (Oct 31 2011).

[139] Gao, J., Buldyrev, S. V., Stanley, H. E. and Havlin, S. Networks formed from interdependent networks. *Nat Phys*, 8, 1 (01//print 2012), 40-48.

[140] Min, B., Gwak, S. H., Lee, N. and Goh, K. I. Layer-switching cost and optimality in information spreading on multiplex networks. *Scientific reports*, 6 (2016), 21392.

[141] De Domenico, M., Sole, A., Gomez, S. and Arenas, A. Random Walks on Multiplex Networks. *arXiv.org*, physics.soc-ph, 23 (Jun 03 2013), 8351-8356.

[142] Newman, M. E., Strogatz, S. H. and Watts, D. J. Random graphs with arbitrary degree distributions and their applications. *Phys Rev E Stat Nonlin Soft Matter Phys*, 64, 2 Pt 2 (Aug 2001), 026118.

[143] Newman, M. *Networks: An Introduction*. Oxford University Press, Inc., 2010.

[144] Solá, L., Romance, M., Criado, R. and Flores, J. Eigenvector centrality of nodes in multiplex networks. … *Journal of Nonlinear* …, 23, 3 (2013), 033131.

[145] Estrada, E., Hatano, N. and Benzi, M. The physics of communicability in complex networks. *Physics Reports*, 514, 3 (May 01 2012), 89-119.

[146] Estrada, E. and Hatano, N. Communicability in complex networks. *arXiv.org*, physics.soc-ph, 3 (Jul 05 2007), 036111.

[147] Nicosia, V., Bianconi, G., Latora, V. and Barthelemy, M. Growing Multiplex Networks. *Physical Review Letters*, 111, 5 (Jul 31 2013), 058701.

[148] Morris, R. G. and Barthelemy, M. Transport on coupled spatial networks. *arXiv.org*, cond-mat.dis-nn (May 12 2012).

[149] Gomez, S., Diaz-Guilera, A., Gómez-Gardeñes, J., Perez-Vicente, C. J., Moreno, Y. and Arenas, A. Diffusion dynamics on multiplex networks. *arXiv.org*, physics.soc-ph, 2 (Jul 11 2012), 028701.

[150] Kolda, T. G. and Bader, B. W. Tensor Decompositions and Applications. *SIAM review*, 51, 3 (2009), 455-500.

[151] Sole-Ribalta, A., De Domenico, M., Kouvaris, N. E., Diaz-Guilera, A., Gomez, S. and Arenas, A. Spectral properties of the Laplacian of multiplex networks. *arXiv.org*, physics.soc-ph, 3 (Jul 08 2013), 032807.

[152] Berners-Lee, T., Hendler, J. and Lassila, O. The Semantic Web. *Scientific american*, 284, 5 (2001), 34-43.

[153] Zhuge, H. Autonomous semantic link networking model for the Knowledge Grid. *Concurrency and Computation - Practice and Experience*, 19, 7 (2007), 1065-1085.

[154] Zhuge, H. *The Knowledge Grid: Toward Cyber-Physical Society*. World Scientific Publishing Co., Inc., 2012.

[155] Aleman-Meza, B., Nagarajan, M., Ramakrishnan, C., Ding, L., Kolari, P., Sheth, A. P., Arpinar, I. B., Joshi, A. and Finin, T. Semantic analytics on social networks - experiences in addressing the problem of conflict of interest detection. *WWW* (2006), 407.

[156] Pons, A. P. Object prefetching using semantic links. *ACM SIGMIS Database: the DATABASE for Advances in Information Systems*, 37, 1 (Jan 31 2006), 97-109.

[157] AlEroud, A. *Contextual information fusion for the detection of cyberattack*. UNIVERSITY OF MARYLAND, BALTIMORE COUNTY, 2014.

[158] Ahlqvist, O. On the (Limited) Difference between Feature and Geometric Semantic Similarity Models. *GeoS*, 6631, Chapter 8 (2011), 124-132.

[159] Lee, K.-M., Min, B. and Goh, K.-I. Towards real-world complexity: an introduction to multiplex networks. *CoRR abs/1502.03909*, 88, 2 (2015), 48.

[160] Radicchi, F. and Arenas, A. Abrupt transition in the structural formation of interconnected networks. *arXiv.org*, physics.soc-ph (Jul 17 2013).

[161] Aleroud, A., Karabatis, G., Sharma, P. and He, P. Context and semantics for detection of cyber attacks. *International Journal of Information and Computer Security*, 6, 1 (Mar 2014), 63.

[162] Karabatis, G., Chen, Z., Janeja, V. P., Lobo, T., Advani, M., Lindvall, M. and Feldmann, R. L. Using Semantic Networks and Context in Search for Relevant Software Engineering Artifacts. *J. Data Semantics ()*, 5880, Chapter 3 (2009), 74-104.

[163] Tiled web map. 2018. Wikipedia, the free encyclopedia. Retrieved from https://en.wikipedia.org/wiki/Tiled_web_map.

[164] OpenGIS® Web Map Server Implementation Specification. 2018. Retrieved from http://portal.opengeospatial.org/files/?artifact_id=14416.

[165] Kisilevich, S., Mansmann, F., Nanni, M. and Rinzivillo, S. *Spatio-temporal clustering*. Springer, Boston, MA, City, 2009.

[166] Birant, D. and Kut, A. ST-DBSCAN: An algorithm for clustering spatial–temporal data. *Data & Knowledge Engineering*, 60, 1 (2007), 208-221.

[167] Pillai, M. and Karabatis, G. *Using Multiplex Networks to Model Cybersecurity Attack Profiles*. Springer International Publishing, City, 2016.

[168] Sahlgren, M. The Distributional Hypothesis. *The Italian Journal of Linguistics*, 20 (2008), 33-54.