UHASSELT

KNOWLEDGE IN ACTION

School of Transportation Sciences

Master of Transportation Sciences

Master's thesis

Geospatial AI: State of the art inventory and showcase

Andrew Bowen

Thesis presented in fulfillment of the requirements for the degree of Master of Transportation Sciences, specialization Transport Policy and Planning

SUPERVISOR :

Prof. dr. ir. Tom BELLEMANS

MENTOR :

De heer Farhan JAMIL

2024

2025

UHASSELT

KNOWLEDGE IN ACTION

www.uhasselt.be
Universiteit Hasselt
Campus Hasselt:
Martelarenlaan 42 | 3500 Hasselt
Campus Diepenbeek:
Agoralaan Gebouw D | 3590 Diepenbeek

UHASSELT

KNOWLEDGE IN ACTION

School of Transportation Sciences

Master of Transportation Sciences

Master's thesis

Geospatial AI: State of the art inventory and showcase

Andrew Bowen

Thesis presented in fulfillment of the requirements for the degree of Master of Transportation Sciences, specialization Transport Policy and Planning

SUPERVISOR :

Prof. dr. ir. Tom BELLEMANS

MENTOR :

De heer Farhan JAMIL

2024-2025

School of Transportation Sciences

Master of Transportation Sciences

Master's Thesis

Geospatial AI: State-of-the-Art Inventory and Showcase

Supervisor:

Prof. dr. Tom Bellemans

Mentor:

Mr. Farhan Jamil

Student Name:

Andrew Bowen

Thesis presented in partial fulfilment of the requirements for the degree of Master of Transportation Sciences

Acknowledgments

I would like to thank all my professors for encouraging me to learn in various ways throughout my study at Hasselt University. I would particularly like to thank my supervisor, Dr. Tom Bellemans, for his thoughtful feedback each time we discussed this thesis.

In addition, I would like to thank my girlfriend, Ms. Mary Schooler, and my parents, Carol and Jim Bowen, for their constant love and support.

Abstract

Recently, advances in artificial intelligence (AI) have prompted new interest in applying machine learning approaches to scientific problems. This is true in geographic AI (GeoAI), where many approaches are available. This thesis gives a review of recent advances in AI as well as the important considerations of spatial explicitness in GeoAI models. Then, a case study of bus arrival time (BAT) prediction for a bus route in Boston, MA, USA is described. Publicly available schedules and bus movement data were obtained over a three-month period. Two neural network models were built using gated recurrent units (GRU) and were trained and tested on the bus data then compared with two baseline historical average models. The non-spatially explicit Time Series GRU model performed better than its baseline on each performance measure by 8.6 – 11.2%. In contrast, the spatially explicit Seq2Seq GRU model performed generally worse than baseline. Explanations for the model performance in this instance are suggested, and implications of the chosen spatial representations in the models are discussed. Future research directions and improvements to models are proposed.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AGI – Artificial General Intelligence

AI – Artificial Intelligence

AVL – Automated Vehicle Location

BAT – Bus Arrival Time

CNN – Convolutional Neural Network

GNN – Graph Neural Network

GRU – Gated Recurrent Unit

GTFS – General Transit Feed Specification

k-NN – k-Nearest Neighbors

LLM – Large Language Model

LSTM – Long Short-Term Memory

MAE – Mean Absolute Error

MAPE – Mean Absolute Percentage Error

RMSE – Root Mean Squared Error

RNN – Recurrent Neural Network

RTPI – Real-time Passenger Information

Seq2Seq – Sequence to Sequence

SIRI – Service Interface for Real Time Information

SVM – Support Vector Machine

AI statement

For the initial stages of programming, downloading and compiling protocol buffer software, Microsoft Copilot was used to suggest Python code and command line prompts. All code written by Copilot was manually verified.

1. INTRODUCTION

1.1 Motivation

The shock that occurred in December 2022 with the public release of GPT-3 has led to continual adaptation as artificial intelligence (AI) pervades more and more spheres of human life. People who study economics and technology consider AI to possibly be the newest "general-purpose technology," a term used for wide-ranging, multi-use inventions like the steam engine, electricity, or computers (Goldfarb, 2024; Crafts, 2021). Such technologies result in major changes on multiple levels in how intellectual work is accomplished. As AI gets cheaper and easier to use, we can expect that it will be used in more places. Applications we cannot imagine now will be found (Goldfarb, 2024). As a result, it is not an exaggeration to say that AI has a good chance to reorder our society.

Geography and spatial science are affected by these changes. While not entirely novel, the use of AI in geography – now dubbed GeoAI – is exploding in popularity. The number of papers published on AI applications in human geography more than doubled from 2020 to 2022 (Wang et al., 2024). Researchers are finding that AI and machine learning enable them to tackle research questions in new ways (Liu & Biljecki, 2022; Wang et al., 2024). But the total impact on science is difficult to evaluate. The use of AI brings up questions about how theory ought to work in scientific discovery. In fact, it sometimes is mentioned that, though AI is being brought into scientific fields, the scientific fields are not being brought into AI (Liu & Biljecki, 2022). Therefore, it is incumbent for holders of this topic knowledge – scientists – to understand the new technologies, so that they can put themselves in the best position to carry on hard-won knowledge in their fields.

This thesis aims to review the changes that have occurred in AI in the past few years, to provide a frame for thinking about the recent AI explosion. This includes discussion of the capabilities and foundations of large language models (LLMs), which are the core of this explosion. Then, we will provide a surface level discussion of machine learning, its definitions, and some models, as well as its limitations. A review of GeoAI will aim to explain the historical context of the field, name the open problems and debates in GeoAI, list some applications, and provide takeaways for researchers. This will help to position the experimental state-of-the-art application to follow within the larger GeoAI field. We will also discuss the use of machine learning models in GeoAI in a more technical sense, with a particular focus towards GeoAI applications towards the activity on transport networks.

After the review, the state-of-the-art application will address the geospatial transportation phenomenon of public transport with an exploration of the spatial and temporal dynamics of public transport. Specifically, the bus arrival time (BAT) prediction application will be formulated as a problem that can be addressed, and its significance to transport users discussed (Kumar et al., 2025). Then, relating to the debates in GeoAI described in the review, two models will be introduced that approach the BAT prediction problem in different ways, albeit with comparable machine learning structures. These will be

compared to a non-machine learning baseline model. The training and testing of the models will be described. The results of the models will be displayed. Then, the discussion will center on the relative performance of the models, the conceptual differences between them, and what this experiment can tell us about how to proceed with future BAT prediction and GeoAI research.

1.2 Recent Developments in AI

Defining AI

Definitions of intelligence are controversial. Therefore, definitions of AI are controversial (Gillain, 2022). One of the most common practices is to define intelligence with reference to human capabilities. This is what Alan Turing did in his famous formulation, the Turing Test, which defines some agent as intelligent when it is capable of convincing a human into thinking the agent is human as well. Thus, the Enlightenment maxim "man is the measure of all things" is often followed.

In line with human capabilities, intelligence is often understood as consisting of many tasks-specific skills (Gillain, 2022). Think of an example of a common human activity: preparing a meal. Trying to list the competencies that a person uses in this common task is quite dizzying:

- Reading and comprehension (reading and understanding a recipe)
- Numerical and arithmetic skills (adjusting quantities)
- Declarative memory (remembering what ingredients are owned)
- Planning (breaking down objectives and ordering actions to take)
- Procedural memory (performing specific tasks: opening cans, etc.)
- Motor skills (physically performing movements needed for food preparation)
- Sensory perception (tasting and evaluating food)
- Creativity (improvising; "what will make this food taste better?")

Typically, AI is described as "narrow AI" if it addresses a specific competency but not the complete capabilities needed to qualify for intelligence. The vast majority of AI systems are forms of narrow AI (Gillain, 2022). Most applications described in scientific journals are narrow AI, since they describe specific applications in limited contexts. For full artificial general intelligence (AGI), AI needs to be adaptable to a variety of applications without human intervention. Humans have the ability to jump between using different competencies, so flexibility is a key component.

Definitions of AGI are contentious (Gillain, 2022). For the focus of this review, it is not necessary to define AGI. Rather, we will talk about narrow AI directed towards geospatial applications, in line with scientific literature on the subject of GeoAI.

The "AI Effect" Strikes (Again)

One aspect of definitions of AI is commonly observed: the common perception of "what makes AI" changes over time. The changing perception of AI is referred to as the AI effect.

Generally, this is a monotonously upward-rising expectation as capabilities increase. People move the goalposts by considering something that appeared intelligent to actually be simply "computation."

Nowadays, most historians of the field call Perceptrons as the first AI systems. These were single-layer neural networks used for binary classification of visual patterns. They were intended for identification of certain objects in aerial images, for military applications. However, eventually it was proven that single-layer neural networks are only able to learn linear equations, which led to a delay in the field for many years. Nevertheless, for a time, the perceptron was considered to be a success. In addition, it was the nucleus of pronouncements about the future of AI.

This is instructive to show how what qualifies as AI changes over time. Even though the perceptron had lesser capabilities than a bar code scanner, the New York Times reported it to be "the embryo of an electronic computer that [the Navy] expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence."

It is unclear whether this statement was a product of overzealous reporting or simply "military intelligence." But it falls into a common pattern where people assess AI based not only on its current performance, but also what people imagine it will do one day. This is true even if how the AI will grow these new capabilities goes undefined.

With the benefit of six-and-a-half decades of perspective, we can acknowledge that people often make grand pronouncements about AI that go unfulfilled. The situation nowadays is definitely different. LLMs and other generative models really do rival human capabilities. Like the Perceptron, however, the excitement about LLMs comes both from their current capabilities and what people imagine they will be capable of in the future.

Large Language Models and Related Developments

Within the past three years, large language models (LLMs) have taken a dominant role in the AI space in several respects. LLMs are the basis for disruption to knowledge-based and creative professions. They also threaten to destabilize mass communications, as they change the way in which information is relayed, and social relations, as individuals spend more of their time interacting with chatbots, instead of humans. LLMs derive their power from the central place of language in human capabilities. The seemingly simple task they perform in predicting the next word in a written document results in an incredible range of practical applications.

However, it is important to ground an understanding of the capabilities of LLMs with the knowledge that these models are based on technology available to the general public. The models and techniques used to make ChatGPT, Gemini, and Claude are of the same type used in many applications to non-language related aspects, including in the GeoAI field. In broad strokes, a LLM calculates and responds to human language in the same way a GeoAI algorithm using neural networks and transformers predicts some geographic phenomenon.

It is true that commercial LLMs are technologically more sophisticated and have been trained much more extensively than what an individual can do. LLMs are developed and trained as business propositions using enormous sums of money and volumes of expertise. Doubtless, companies like OpenAI and Google have some of the best AI researchers in their camp, and that helps them maintain an edge. But behind LLMs is a broader growth in processing power that helps all users of AI, not only large corporations.

Processing power continues to grow on a roughly exponential curve. This is intuitively understandable in the short-to-medium term. But progress beyond a few years out is hard to imagine or grasp. For example, the processing power available at a given price has grown by a factor of 1000 over the past 20 years, according to Moore's Law. Exponential growth can roughly intuitively explain why AI's progress seems so explosive.

On the high end, this progress has enabled LLMs, which use amounts of processing power that would have been impossible 10 years ago. But this advancement in tech also means that some machine learning types are now much more available to the layperson than back then (Gillain, 2022). Moore's Law refers specifically to processing power *per dollar*. Anyone with a computer, basic understanding of how to run computer programs, and time can experiment with deep neural networks and other machine learning processes. This has a few implications. First, individuals do not have to imagine that AI capabilities are only the domain of large corporations with multi-billion dollar capitalization. Second, researchers can use investigations in their own fields as a gateway to understanding how AI works more broadly.

Nowadays, the focus on LLMs means less scrutiny and acclaim falls on the types of technologies that were more cutting-edge a decade ago. Those would be, for example, recommendation engines for consumers on digital platforms, or computer vision that could identify faces. Were these two technologies AI? Certainly, they were (and still are) replacing humans in jobs like, respectively, salesperson or security guard. Their capabilities were powerful and flexible. Even though they get less attention than LLMs now, they still have a stake in AI.

The same will be true about many algorithms and methods that are explored in AI-applications in geography and transportation. They do not have the generality or world-shaping appearance as LLMs, yet they still have a level of flexibility and autonomy that exceeds what simpler methods brought. They still have a connection to the field of AI.

1.3 Basics of Machine Learning

At this point, it is time to change gears to stop talking about a class of capabilities – AI – and to begin to talk more in-depth about the technologies that make AI possible. In these sorts of applications, the capabilities of AI are generally associated with a certain domain of technology known as machine learning.

How Do Machines Learn?

Machine learning is also a contentious definition, but it generally refers to when a machine can increase its performance on a task or to gain knowledge through interaction with the world and without direct human intervention (Blockeel, 2022; Mitchell, 1997). The relation of AI to machine learning is such that that AI performs tasks via a model, which is defined and/or refined through data, or observations (Blockeel, 2022). In other words, machine learning is really about developing a model that represents something real. The machine represents the reality inside it, and perhaps that is where it stores the "knowledge" it has gained from learning.

In terms of specific methods, there is ambiguity about machine learning, because a variety of systems and statistical procedures can effectively increase task performance through exposure to data. Machine learning has been referred to as a branch of computational statistics, which emphasizes the importance of complex data and intensive computing procedures in what distinguishes it from other statistics (Goldfarb, 2024; Wegman, 1988). Machine learning is not generally thought of assuming the underlying probability distributions that underlie statistical theory (Blockeel, 2022); the tension between theory-based and data-driven methods is a common theme (Gillain, 2022). Machine learning is certainly used as a term to refer to techniques that offer higher levels of flexibility, and it is used in contrast to those methods considered "traditional" or "statistical" procedures. However, in practice, machine learning usually refers to non-parametric methods. Non-parametric methods mean that any type of function can be selected based on the data. Parametric methods set the form of the function to be selected in advance (Patil et al., 2022).

Types of Machine Learning

Supervised learning is a group of methods that learn using labeled data. Labeled data means you have observations of your dependent variable in the dataset. A more technical way of describing is that each relationship between a set of independent variables and a set of dependent variables has a truth value (Gillain, 2022). In other words, when the machine makes a prediction, the dataset ought to be able to say the machine is right or wrong. This is the "supervision:" telling the machine whether a combination of variables is correct or incorrect.

In supervised classification, the dependent variables are categories, and in supervised regression or prediction, the dependent variables are numeric values (Gillain, 2022).

Depending on the type of machine learning model, the labeled data is used differently. Neural networks will make predictions or classifications using independent variables during training. If these predictions or classifications do not match the observed dependent variable, it is an error, which triggers the neural network to change its model. Other machine learning types, like support vector machines, behave more analytically, but it still revolves around being able to identify clear, observed, true values from which to learn from.

Many of the difficulties in using machine learning for these purposes come from a lack of sufficient labeled data, which must be huge and varied. Otherwise, the machine cannot correctly learn the relationship between variables.

Another type of learning is unsupervised learning. This is where the data have no labels or clearly defined independent and dependent variables. Rather, the machine learning algorithm is trying to infer conclusions about structures in the data.

A good example is a clustering algorithm that attempts to put records from a dataset into a finite number of groups. The unsupervised learning algorithm will use some aspects of the dataset to decide which records to put where. But, there is no data available to say whether any given classification is correct or not correct. Another type of unsupervised learning is dimensionality reduction (Gillain, 2022).

A third type of learning is reinforcement learning. In this paradigm, the machine receives some sort of response related to its action. Based on receiving positive or negative reinforcement, it will be more or less likely, respectively, to repeat the action. Reinforcement learning similar to supervised learning in that the machine receives a “good” or “bad” responses. The key difference in reinforcement learning is that the set of responses in the future is changed (possible irrevocably) based on the action in the present.

An ideal example is a machine learning to play a game. In this situation, the actions of the machine to make a move results in a different set of responses of its opponent. It is not as simple as having labeled data where the “good” or “bad” response is immediately known.

Therefore AI is a capability that machine learning can produce. While it is also theoretical that AI could be produced by techniques other than machine learning, for this review and paper, we will consider AI to be primarily the purview of machine learning techniques.

The need of deep neural networks is sufficient labeled data for training purposes. In some applications, this is unworkable. For example, an application on geographic duplicate detection chose random forests precisely because the expectation of labeled data was unrealistic in that domain (Acheson, 2019). Other researchers have suggested that certain techniques may be better for uses in traffic forecasting (Koushik et al., 2023).

Data Needs for AI

Data-driven methods, as may be obvious, need a lot of data to work. When a very powerful neural network is asked to train on data, it is essentially trying to pick a single hypothesis out of a very large hypothesis space. It can eventually recreate almost any pattern of data, but it needs lots of training data to prevent overfitting to the precise input given to it (Mai, 2022). This is the basic challenge with trying to address a complex problem with data-driven methods. Simpler models are more generalizable with less data, but ultimately struggle to make accurate predictions when the underlying pattern of data is complex.

This means that they need a lot of data, because they have to identify enough of the relationships in the phenomena of interest to make good predictions. Researchers need to have labeled data of sufficient quality and quantity. In some cases, this limits what problems researchers can tackle (Gillain, 2022).

1.4 A Few Types of Machine Learning Models

Machine learning models differ in terms of their data requirements, their computational requirements, their comprehensibility, and the problems that they model best. A description of a few common machine learning models illustrates some of these differences.

Decision Tree and Random Forest

A basic, baby's first procedure for data mining. It produces good classification results, can take in messy, unscaled data or irrelevant features, and the results are inspectable. However, they tend to overfit to the available data, to an extent that they are seldom useful on their own. They learn very strange patterns in a way that they have low bias and high variance.

A random forest is a classification that uses a multitude of decision trees together. Essentially, a classification is decided on if the classification is picked by the most trees. Uses many trees to help reduce the weakness of decision trees whereby they tend to overfit to the available data. First formalized in 1995 by Tin Kam Ho. Unsupervised, they can make some sort of dissimilarity measure between observations. Relative to decision trees, there is some increase in bias and a decrease in explainability, but the model is generally much more predictive. This is considered to be a type of ensemble technique, where a number of algorithms are used to produce a better estimate than a single algorithm could produce.

K-nearest Neighbor (k-NN)

Formalized in 1951 by Evelyn Fix and Joseph Hodges. Comes in classification and regressional flavors. In classification, an element to be classified is compared to the k elements with the most similar values on the chosen variables ("nearest neighbors" or "neighbors"). Then the element is assigned to classification already held by a plurality of the k nearest neighbors. In k-NN regression, there is a "property value" assigned to each element, and the property value of the target element is set to the average of the property values of the k nearest neighbors. Can also be used for anomaly detection (called k-NN outlier). Interpretable by looking at the variables used and decision boundary learned during the training.

Data reduction and dimension reduction are required to work with large datasets. Approximate k-NN search can be used using locality sensitive hashing – this can allow it to do e.g. a similarity search on live video streams.

Support Vector Machines

These are based on the procedures required for modeling binary classification of data by, finding a line or higher dimensional object that best separates the labeled categories. But when the data dimension is higher, it is still possible for the computational techniques to separate the data using this technique. It is relatively computationally simple with those higher dimensional pieces of data. On the other hand, it takes a lot of computational power to handle a high volume of data. Thus the technique is relatively suitable, compared with other machine learning approaches, when the problem in question has a high number of data dimensions relative to the number of observations (Blockeel, 2022).

Artificial Neural Networks

A type of learning machine inspired by the human brain. The human brain is made up of neurons which connect to each other in a directional manner, transmitting and aggregating information as they go. Artificial neural networks are the same, but the entire physical sequence is replaced by mathematical functions.

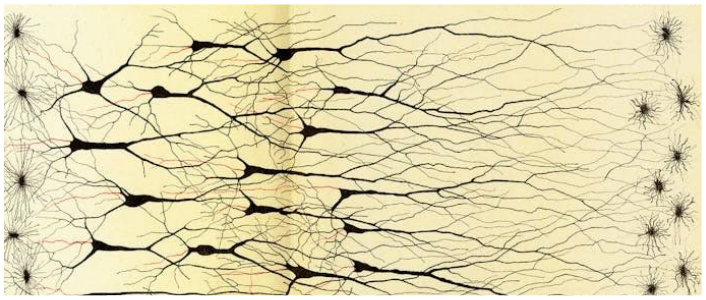


FIGURE 1. Drawing of stained neurons from a section of hippocampus (Golgi, 1885).

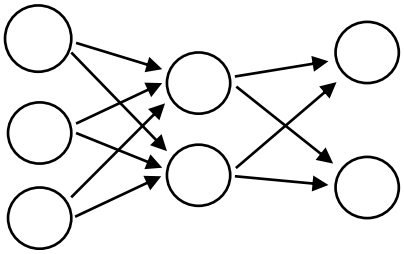


FIGURE 2. Diagram of a simple neural network.

The structure of the ANN consists of nodes and directional links: idealized neurons and directional signals. Each node has a weight, and the value (or activation) of each node is determined by the weighted average of the nodes giving input to it, followed by an activation function.

Neural networks are mathematical models that simulate the function of the human brain. At core, there is an input layer, an output layer, and some hidden layers of neurons, each of which are related to each other mathematically (Liu, 2022). As the model trains, some feedback mechanism adjusts the connections between neurons to allow the outputs to change in alignment with the data (Liu, 2022).

As with the human brain, the architecture of the neural network means that very complex tasks can be performed using relatively simple techniques at the cell level. The resulting structure that can predict relations between variables that have complex or non-linear relationships between them.

The architecture is generally defined in advance, while the weights of each node are refined in training. The weights are adjusted in a procedure called gradient descent. When in training mode, when the output of the neural network differs from the labeled value in the data, backpropagation is performed, which finds "where the network went wrong," so to speak. Using calculus, it is possible to computationally infer what change of weights would have moved the output closer to the labeled value. This procedure will only change the network a small amount, so lots of data are required to train neural networks. Often, the data are shown to the network multiple times – each repetition is called an epoch. After training is complete, the model weights are set and no longer change.

In this way, ANNs are not dissimilar from directed acyclic graphs (DAG) or outputs of structural equation modeling (SEM) approaches, which are commonly used in scientific disciplines. In contrast, DAGs and SEM approaches clearly assign meaning to nodes in the graph.

In ANNs, on the other hand, the activation levels of individual nodes are not usually of interest; only the inputs and final outputs are of interest. In addition, ANNs can have huge numbers of layers and nodes; and the nodes do not necessarily correspond to any particular conceptual or physical phenomenon – they are really just there to serve the predictive power of the algorithm. This quality is why neural networks lack explainability. It is very difficult to determine why a neural network makes a certain classification or prediction (Gillain, 2022).

ANNs are extremely flexible because they can be adapted in many ways. One way is it is possible to link neural networks to each other. This is why so many methods exist that are sub-categories of neural networks. Some of these examples will be addressed in Chapter 2. However, this also means that neural networks are free to be expanded beyond the extent to which humans can comprehend what is going on. This ultimately can compound the challenge of explainability.

2. REVIEW OF GEOAI

GeoAI is a field of applications – ML or AI techniques applied to the field of geography. As one would expect, we therefore inform work in GeoAI with methods and principles from both fields.

2.1 Developments in GeoAI

Geographic AI has a history that extends over 40 years. The introduction of AI into geography followed closely after quantitative geography became common. Geographers wished to formalize concepts in geography into principles and laws that could be consistently applied to spatial data. Geographers were also interested in techniques that could overcome methodological limitations in applying traditional statistical techniques in geography. (Anselin, 1989).

At that time, geographers were also beginning to make distinctions between model-driven approaches, which commence from some theoretical base, and data-driven approaches, which minimize the use of theories *a priori* before confronting the data (Anselin, 1989).

Also back then, researchers noted that both approaches had their difficulties in application. In the model-driven camp, formalizing how space fits into theory was the challenge. In the data-driven camp, it was finding statistical techniques that would not struggle to deal with the types of dependence found in spatial data. However, in general, the potential for GeoAI was seen as the possibility of bringing theory into computational geography.

GeoAI has been empowered by a combination of changes including an increase in computational power available, new AI techniques of deep learning, and the availability of high-quality geographic data (Janowicz, 2019). These are the same trends discussed in Chapter 1 as enabling LLMs and AI across-the-board.

Data collected and data made public, mostly through mobile sensors and IoT devices, has increased. The enabling of "social sensing," or using user-generated data to examine human activities on a short time scale, has affected geography (Janowicz et al., 2019). The tides of the data revolution and AI have fed and reinforced each other (Janowicz, et al., 2019; Liu, 2022). Geographers have done a good job of using new techniques that have emerged and in imagining the possibilities of big data and AI, which is good, since about 80% of "big data" is spatial (Liu, 2022).

Nowadays, AI is mainstream. In a twist to the situation in the 1980s, now that computational scientists of all kinds are working with spatial data, geographers are attempting to remind others that there is "something special about spatial."

Theory-Driven vs. Data-Driven

The relative belonging of theory versus empiricism have continued to preoccupy GeoAI since. Particularly with the beginning of data-driven geography, a distinction between two categories of approaches was described (Jiang & Luo, 2022). The top-down or model-

driven approach is where expert knowledge is used to construct the research questions and hypotheses, and then data is used to answer the questions and to support or reject the hypotheses (Anselin, 1989). On the other hand is the data-driven approach, which is where the data itself generates the hypothesis (Anselin, 1989; Goodchild, 2022; Jiang & Luo, 2022).

Early movements of geography into the AI field were written by Smith (1984). At that time, Smith observed that AI "lacks a basis of generally acceptable theory," rather being dominated by empirical exploration of the efficacy and efficiency" of programming techniques in problem-solving situations. This placed AI clearly in the data-driven tradition of quantitative geography.

In general, machine learning is a data-driven process (Koushik et al., 2023), but early approaches of GeoAI used expert knowledge as a type of top-down guide. Primarily, this was because of the limitations of the tools and processing power available at the time. Since the limitations of the perceptron became apparent, AI systems were significantly focused on developing expert systems. These were AI systems based on experts formalizing rules from their own experience and programming them into a computer.

Nowadays, as machine learning techniques have come so far, there is again a debate about model-driven versus data-driven GeoAI. The questions are, what geographic principles ought to be built into machine learning models doing GeoAI? The risks are that naive machine learning approaches may forget the principles at the core of geography, particularly spatial dependence and spatial heterogeneity. This is an understandable risk because GeoAI is increasingly being carried out by experts and researchers without a geography background (instead having more of a computational background).

Spatial Explicitness

Geographers have summarized their advice on how to carry out GeoAI by defining the category of spatially explicit GeoAI.

Spatial dependence refers to the fact that spatial objects or phenomena near each other are related to each other. This means that identical objects that are in different locations should not be treated as equivalent. (Anselin, 1989; Goodchild, 2022). A separate principle, spatial variability, states that relations between phenomena, including spatial phenomena, in turn depend on space. This is to say that a relation that is found in a particular area of land may not necessarily hold in another area of land. (Anselin, 1989; Goodchild, 2022).

Geographers argue that spatially explicit approaches increase the predictive power of models and are better grounded in geographic principles; therefore, spatially explicit approaches should be preferred where possible (Mai et al., 2022). In contrast, "non-spatially explicit" denotes uses of AI in the service of some geographic research question without using spatial variables in the artificial intelligence process itself. Surprisingly, this branch of GeoAI is actually more common for the time being (Li, 2020; Liu 2022).

Non-spatially explicit approaches have the downside of excluding spatial data when they could be crucially explanatory. Models of this kind will likely be less explainable using geographic theories. On the other hand, spatially explicit approaches require inputting geographical information (often called “embedding” or “encoding”) as input into machine learning.

2.2 Spatial Representations in GeoAI

Assuming that a spatially explicit approach is chosen, the question follows: how should spatial information be represented in the model? A review of two competing types of neural networks shows how answers to this question can and should vary with the problem at hand.

There are many ML models used in GeoAI, including k-NN classification, kernel-based models, hybrid models, support vector machines, and Bayesian networks (Jiang & Luo, 2022; Koushik et al., 2020). However, neural networks are where most progress in GeoAI has been made in the last few years. According to geographers, modern neural networks are valuable in GeoAI because they can represent complex, non-linear functions; they can use many different types of data and benefit from huge quantities; they are resilient to multicollinearity and other causers of difficulty in traditional statistical approaches; and they can grow to predictive accuracy higher than other ML approaches (Acheson, 2019; Koushik et al., 2020).

Within neural networks, there is a divide into two main types that represent relationships in space: convolutional neural networks (CNNs) and graph neural networks (GNNs).

CNNs use overlapping receptive fields to analyze data. This makes them good for analyzing raster data like satellite images (Liu, 2022; Mai, 2022). CNNs work quite similarly to how mammalian visual systems function. Within the retina, there are photoreceptors, biological sensors of light. These photoreceptors are wired to cells that integrate information from the photoreceptors. An example is seen below.

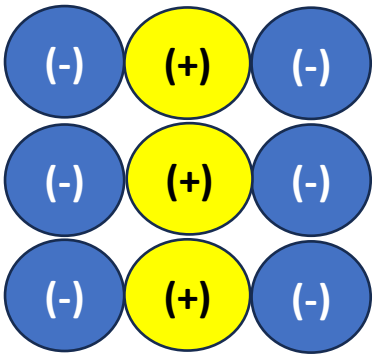
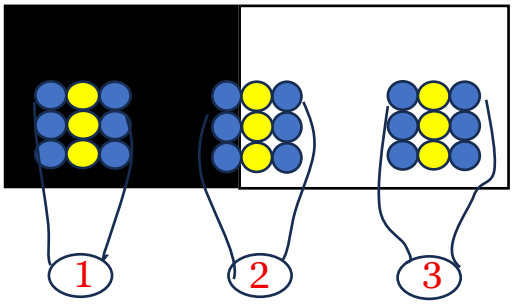


FIGURE 3. “Edge detector” cells in the mammalian visual system have combinations of excitatory (+) and inhibitory (-) connections to light-sensitive photoreceptors. This connection pattern leads the cell to respond most strongly to visual edges, as with cell 2 in the diagram.



CNNs have artificial neurons that are connected in the same way. The difference is that the CNN units can learn convolutional operations through exposure to data. Just as some downstream cells in the mammalian visual system are specialized to detect edges and corners, downstream neurons in CNNs learn to detect features that have a relation to the prediction or classification task at hand.

As can be seen from the figures, a CNN imposes a grid-like spatial relationship on whatever input data is given to it. This works well with raster data, which already has a grid structure: pixels in the data can map to CNN input cells. However, it works less naturally on spatial or geographic relationships that do not have a grid structure. For example, transport networks are more often represented by graphs which take into account accessibility relations of areas. That is a different sense than the Euclidian adjacency that CNNs are dealing with.

GNNs, naturally, are specialized in analyzing data structured in graphs, which makes them particularly useful for transportation uses (Mai, 2022). They share some similarities to traditional approaches in geographic information analysis that represent geographies in terms of graphs with spatial weights (Liu et al., 2024). Graphs embedded in GNNs can be directed or undirected depending on the use. Like CNNs, they work through a convolutional process. This means that each neuron takes input from neurons to which they are adjacent, as defined by the edges of the embedded graph structure.

When a GNN makes a prediction, it is given a set of input data in the graphical form. The embedded graph influences the calculation so, whatever output is taken, it will be partially determined by the adjacency defined by the graph. During training, backpropagation infers the upstream origins of prediction errors and can retune those weights if desired. Therefore, the researcher can have the training phase reweight the adjacency in the embedded graph. Or, they can leave the embedded graph and update other weights in the model, if they want to take the adjacency as a given.

As can be imagined, GNNs allow some testing of theories around spatial relations. If an adjacency graph generated through theory cannot reproduce observed geographic phenomena, what does it mean? It requires decoding aspects of the GNN to understand why the model is failing to learn. In this aspect, explainability, GNNs are still in an early phase of development (Liu et al., 2024).

So far, GNNs have found use in traffic forecasting and navigation (Mai, 2022; Zhou et al., 2019). GNNs are relatively new and still in an early phase of development (Jiang & Luo, 2022).

Perhaps a reader will notice that each neural network implies a certain structure of relationships will exist in the data. This is where theory comes into the process. In fact, there are many options available on how to do this. The crucial step is the geographic encoding, which embeds the geographic information in the data in question into a form that can be used by the machine learning approach of choice. This can be as simple as

directly appending the coordinates of geographic features to the input dataset, or it can involve some sort of transformation.

Making predictions using vector-structured data has also occurred, with traffic forecasting being a large area of focus (Jiang & Luo, 2022; Koushik et al., 2020). AI is being used to "fuse" datasets that may contain related data, but differ somewhat structurally or contain duplicates (Acheson, 2019; Jiang & Luo, 2022). AI can also be used to interpolate missing figures in datasets. In addition, the use of natural language processing is also found in geography, as it can be used for classification of user-generated data that is geotagged (Koushik et al., 2023; Li, 2020). Question answering has also been linked to geography, such as to provide information on a user's surroundings (Mai, 2022).

Make use of datasets – It is advantageous for geographers to have a variety of high quality datasets, such as remote sensing datasets available from LandSat, as well as a range of lower quality datasets (Janowicz, 2019). In addition, however, researcher commenters have noted that one of the potentialities of artificial intelligence is to integrate, repair, and ultimately use data that could be low-quality, collected for a different purpose, or incomplete. AI also provides the potential to interpolate one type of data from a quality source into another, and therefore there possibility to dodge the risks of using lower-quality data. There is also possibility to use an easier-to-obtain dataset for a hard-to-obtain dataset, opening up new research possibilities (Janowicz, 2019).

In a very tangible and physical way, the importance of joining disparate datasets together in order to create a more informed whole is illustrated in geographic analysis, where data can literally be layered on top of another (Acheson, 2019). By using as much data as possible, researchers can gain the maximum amount of context (Janowicz, 2019). AI techniques can help integrate all these data to make more accurate predictions (Jiang & Luo, 2022).

Alongside processing power and the actual machine learning model development, increased availability of data reinforced the use of AI. Obviously, LLMs are successfully because f the massive amount of data they are trained on. Geospatial scientists are familiar with this data development because so much new data available is spatial in the last 20 years. The innovations responsible for spatial data proliferation are the smartphone, whose sensors create the means to follow human travel patterns; social media, which allows individuals to share data with others that often includes space; and the data economy, which provides financial incentives for firms to collect, store, analyze, reprocess, and export spatial data on massive scale.

Perhaps a theme of applications using GeoAI is using data from different sources. Many AI techniques, including random forests and neural networks (NNs), can use multiple types of data easily. With the ability of AI to also expand the datasets available to researchers through data fusion and interpolation, there are many research questions that can be answered (Janowicz, 2019; Sparks, 2019).

2.3 GeoAI Applications

So far, transportation is one of the most important focus fields in the use of GeoID to explain human behavior. Understandably for a field studying the movement of people and goods, the majority of studies use some sort of spatial-temporal flow data, among which data received from transportation networks and smart phone apps are the most frequent uses. The flip side of this is that other data types and sources are comparatively underused. For example, the use of government census data is reportedly used less in transport studies than in other fields in human geography. Using video data for transport purposes is also rare, although this is generally true in most fields of ai application (Wang et al., 2024).

There are numerous studies that attempt to apply the state-of-the-art techniques of graph neural networks to issues of traffic prediction.

3. CASE STUDY: BUS ARRIVAL TIME PREDICTION USING GEOAI

3.1 The Utility of Real-Time Bus Information

Every minute, thousands of buses arrive at bus stops around the world. When they do so is often dictated by a schedule, but in reality, all public transport vehicles are subject to variations around this schedule due to complexities in operations. These include time spent to load and unload passengers, road or right-of-way conditions, vehicle limitations, and more (Aemmer, 2022). This is particularly true for buses, who typically deal with variability stemming from bus stops and from traveling in mixed traffic with other vehicles.

Bus arrival time (BAT) prediction is considered to be both a part of intelligent transportation systems (ITS) and smart cities (Patil et al., 2022). Knowing when a bus is going to arrive in advance enables a number of additional functionalities. A bus that has tremendous variability in its arrival times is not easy for passengers to plan around. But accurate real-time enable passenger planning even when buses are off-schedule. If that information is provided to passengers, it enables trip planning. Bus agencies that know trip trajectories in advance can adjust their operations in real-time to improve service, such as by dispatching additional vehicles (Alexandre et al., 2022; Huang et al., 2021).

In situations where real-time passenger information is available, there is another dimension. In practice, many passengers rely on GTFS-realtime for their trip planning (Abusalim, 2020; Newmark, 2024). Therefore, the accuracy and availability of that information becomes a part of service quality. As schedule adherence makes the reliability component of service quality in the physical realm, accuracy of real-time information makes up the reliability component in the digital realm.

In bus systems where real-time passenger information is available, passengers benefit. A survey taken in a US city in 2012 showed that about 90% of respondents reported themselves to be much more or somewhat more satisfied with public transport as result of real-time passenger information (Gooze et al., 2013). Multiple surveys have supported that passengers using RTI have lower perceived wait times and/or lower actual wait times at the bus stop than those using traditional schedules (Mishalani et al., 2006; Watkins, 2011). Another study showed that RTI was attributable for a bus ridership increase of 1.7% in New York City (Brakewood et al., 2015).

There are reasons to seek improvement in RTI in systems where it already exists. Increasing the accuracy of real-time predictions has the potential to further reduce travel times for all passengers. A study in Stockholm suggested that an increase in RTI accuracy was equivalent to a 45% increase in service frequency, or 20 additional bus trips per hour, in terms of the effect on passenger waiting times (Cats & Loutos, 2016). Researchers have also found that people vary in their tolerance for errors in RTPI. A survey taken in a US city in 2012 had the median respondent consider the prediction deviance at which they

would personally consider it to be an error to be 4-5 minutes (Gooze et al., 2013). This suggests that increasing accuracy of RTPI can satisfy and attract more passengers.

3.2 How Real-Time Information is Provided

In economically developed countries, transit companies often have automated vehicle location (AVL) systems installed in their vehicles. These systems broadcast information about the vehicle's current state to the companies' headquarters on a regular time interval (Adorno, 2023). Transit companies use this system to monitor their buses, organize their operations, to evaluate their performance, and for computer-aided dispatching (Kumar et al., 2025).

The AVL information can also be used as input to real-time passenger information systems. Traditionally, real-time passenger information uses AVL in a simple manner. Using the AVL, the transit company can determine when a vehicle arrives at a stop "time point" then can calculate the vehicle's deviation – early, late, or on-time – in relation to the vehicle's schedule.

Methods for Prediction

The most commonly reported method is for the transit company to assume that this schedule deviation will remain the same throughout the rest of the vehicle's trip. If a bus is now two minutes behind schedule, the bus will be predicted to arrive at each downstream stop two minutes past the scheduled time. This is implicitly the same as assuming the bus will travel the remainder of the route at the average speeds implied by the schedule. This is the most basic form of bus arrival time (BAT) prediction possible, and it is used by many transit agencies, including Minneapolis, Stockholm, Auckland, and it is provided for by public transport management software commonly used in the industry (Cats & Loutos, 2015; Crudden & Berrebi, 2023; Elliott & Lumley, 2020). This model is simple and explainable, but has limited explanatory power, and is tied to the static schedule; therefore, it cannot respond to conditions that change faster than the schedule does.

Reportedly, another method that has been used is to use a weighted average of the travel times of the three previous vehicles to predict the travel time for the current vehicle, inferring arrival times through summation (Horbury, 1999; Jabez, 1993). Reports of agencies using more sophisticated prediction methods are not evident.

Technology and Data Formats

Transit agencies' AVL data structures are often proprietary and not mutually intelligible (Adorno, 2023). Depending on the intended use of the real-time information, it may require various transformations out of the initial AVL format.

A common format for sharing information with passengers is General Transit Feed Specification (GTFS). GTFS-realtime is the subset of GTFS used for relaying real-time

information. It has been reported that transit agencies usually employ outside vendors to translate their AVL systems into a usable GTFS-realtime feed (Newmark, 2024).

The feeds are made publicly available to passengers and, more commonly, to app developers who in turn display the information to users. In Europe, there is also the Service Interface for Real Time Information (SIRI). This specification was introduced in 2006, and it provides for interagency communications and other functions, while the SIRI Lite protocol allows relaying SIRI information to end-users, similar to GTFS (Adorno, 2023).

3.3 Academic Studies on BAT Prediction

Data Sources for BAT Studies

Bus location data used for research is either publicly available data in a standard format like GTFS-realtime, or it is provided in a proprietary format directly by the transport agency. A specific deficit is GTFS does not have any provision for providing actual arrival times at stops (Aemmer et al., 2024). Researchers that want to have ground truth data for stop times must derive it by processing GTFS-realtime data (Adorno, 2023; Barbeau et al., 2020; Wessel et al, 2017). As such, data preparation takes up much of the time used to accomplish these studies (Alexandre et al., 2022).

Other studies of BAT use datasets that already have arrival times (Cats & Loutos, 2016; Mazloumi et al., 2011). It is generally known that transit agencies have AVL systems that do indeed mark definite arrival times. Indeed, this is necessary to evaluate physical bus performance: most bus agencies rate an “on-time” percentage that comes from a comparison of actual arrival times with the scheduled times (Aemmer et al., 2022).

For studies aiming to predict arrival time, it is necessary to have valid and reliable ground-truth arrival times at stops. Therefore, an accurate calculation method is needed if GTFS-realtime or another data source lacking arrival times is used.

In the literature research, there are generally two ways to derive stop times. The first uses the TripUpdates file of GTFS-realtime and takes the last prediction as the actual arrival time (Newmark, 2024). The idea for this is the last trip update likely has an error small enough to be acceptable. The alternative is processing GTFS-realtime VehiclePositions. This method can be further into proximity methods and interpolation methods. Proximity methods take the timestamp of the closest vehicle position to the stop (Adorno, 2023) or a timestamp of one position within a given proximity to the stop (Aemmer, 2024; Zhou et al, 2024) as the actual arrival time. The interpolation method infers the actual arrival time through linear interpolation of the closest-reported positions upstream and downstream of the stop and their associated timestamps (Barbeau et al., n.d.; Wessel et al., 2017). Usually this requires referencing the route shape along with the stop location.

Machine Learning Models Used for BAT

Traditional time-series models are capable of capturing cyclical variations in observations in the bus space using linear regression techniques and exponential smoothing (Shen et al., 2025). On the other hand, they struggle with dealing with unsmoothed time series data and time lag, and multicollinearity in the data is also a concern (Shen et al., 2025).

The cyclical pattern of travel times is a challenge to time series techniques. In the machine learning realm, the simplest RNNs struggle to find consistent associations between distant time points that characterize cycles, although they are suitable for noticing patterns between observations on a short time-scale (Olah, 2015). This makes them rather unsuitable for strongly cyclical contexts like BAT prediction, and that weakness is shared with several non-ML time series techniques.

However, more advanced RNNs have been designed that are effective at discovering long-term relationships. This category, which includes Long Short-Term Memory (LSTM) networks and Gated Recurrent Unit (GRU) networks, makes use of gates that decide which information inherited from previous states is relevant to the present (Bhutani et al., 2022; Olah, 2015). These gates are themselves sub-networks: they learn parameters through the training process that allow them to associate distant time points. LSTM and GRU networks are commonly used and successful at a wide variety of time-series prediction tasks (Petersen et al., 2019). In addition, transformer models have been used to replicate the functionality of RNNs in the time dimension in BAT studies (Jeong et al., 2024).

Because bus networks and road network are easily representable in a graph form, it is common to use graph neural networks (GNN) for the geographic aspect. Graph neural networks (GNNs) are defined by containing some sort of graph representation is input, or embedded, into the neural network used for predictions. However, there are significant variations within this category, because there are many different ways to use graphs in the model.

Physics-based methods have been highlighted in a review as a promising area of focus in BAT prediction for being able to make predictions with less historical data (Kumar et al., 2025). These would be a more top-down approach than data-driven alternatives. There are examples of machine learning BAT prediction attempts that explicitly attempt to consider patterns of congestion in traffic flow (Bhutani et al., 2024; Julio et al., 2016; Yuan et al., 2021)

Other machine learning methods used have included k-NN algorithms and SVR (Alexandre et al., 2022). The most used methods mentioned in that paper for bus travel time prediction were SVR, gradient boosted decision trees, and multilayer perceptrons, which are a type of CNN (Alexandre et al., 2022).

Evaluating Results of BAT Prediction Studies

	Baseline Type	Baseline Performance, MAPE	Type, Best Performing Model	Best Performing Model Performance, MAPE	Percent Reduction of Baseline
Achar et al., 2020	Historical Average	30%	Spatial Kalman Filter	20%	33%
Amaris et al., 2021	Historical Average	23.14%	Bi-LSTM	12.5%	46.0%
Aemmer et al., 2024	Historical Average	29%, 38%	GRU	11%, 13%	62%, 66%
Dhivyabharathi et al., 2017	Historical Average	31.76%, 19.40%	Particle Filter	17.18%, 13.01%	46%, 33%
Julio et al., 2016	Historical Average	14.7-32.4%	ANN	17.9 – 21.9%	7.5– 44.8%
Ma et al., 2022	Historical Average	12.66%	Multi-Attention GNN	3.98%	68.6%
Petersen et al., 2019	Historical Average	10.62%, 8.28%	Convolutional LSTM	4.04%, 5.61%	62.0%, 32.2%

TABLE 1. *BAT studies with historical average baseline and relative performance measures.*

A primary difficulty in evaluating the body of research on BAT prediction is caused by differences in bus systems. Bus operations differ in their level of unreliability. Some routes are more variable and some systems are more variable. Prediction difficulty increases with the variability of the events to be predicted. A bus network that is always perfectly on schedule is trivial to predict – in fact, the schedule does it perfectly. On the other end, a bus route that is heavily congested at all hours and subject to all manner of random events is extremely difficult to predict. Unfortunately, the common measures used to predict machine learning accuracy do not take note of the underlying variability. Therefore, it is difficult to compare studies.

Table 1 above shows a novel rudimentary method for quantifying model performance across different bus networks. By using a common baseline of historical average, the improvement in a given performance measure can be quantified, and thus an implicit adjustment for the difficulty of the BAT problem is obtained. Unfortunately, only a subset of BAT studies report the performance of consistent baseline like historical average model or the scheduled arrival time. More commonly used are competing machine learning models. However, it is difficult to know how to interpret these, particularly because there are so many potential variations in machine learning methods, and specifics are often unreported in the studies. Of course, the preferences and knowledge of the scientist affect how any of their models is built, and so there may be some bias towards favoring one's own model (Gillain, 2022). This is why it is so important that the field move towards consistent baseline measures.

In the lack of these measures, we rely on within-study comparisons of model types. However, as Table 2 shows, some models are shown to be superior in one situation, then

shown to be inferior in another. It is difficult to draw durable conclusions from this state of literature.

Source	Best Performing Model	Other Models Tested
Achar et al., 2020	Spatial Kalman filter	Random forest, space discretization, historical average
Amaris et al., 2021	Bi-LSTM	LSTM, CNN
Aemmer et al., 2024	GRU	Transformer, CNN, ANN, RNN+CNN mixed approach
Bhutani et al., 2024	Bidirectional LSTM	Kalman Filter, SV Kalman Filter, Other RNN and LSTM approaches
Chondrodima et al., 2022	Multi-layer perceptron	k-NN, linear regression, additive regression, reduced error pruning tree
Dhivyabharathi et al., 2017	Particle Filter	Historical Average
Jeong et al., 2024	Transformer model	GRU, LSTM, Bi-LSTM, ANN, a different transformer model
Julio et al., 2016	ANN	Support Vector Regression, SVM/k-means/ANN, historical average
Ma et al., 2022	Multi-Attention GNN	Historical average, ANN, SVR, k-NN, LSTM, Geoconvolutional layer plus LSTM, Convolutional LSTM
Petersen et al., 2019	Convolutional LSTM	LSTM, Google Traffic, Historical Average, Current model from agency
Qiu et al., 2024	Transformer with GNN	CNN, LSTM, Transformer, CNN with GCN, LSTM and GCN
Shanthi et al., 2022	Support Vector Regression	Random Forest Regression, Decision Tree Regressor, k-NN Regressor, Gradient Boosting Regressor, XGB Regressor, AdaBoost Regressor
Xie et al., 2021	Convolutional LSTM	GRU, multiple other variations of LSTM
Yu et al., 2011	SVM	k-NN, ANN, linear regression

TABLE 2. *Best-performing models in selected BAT literature, by study.*

3.4 Specifications of the BAT Prediction Problem

It seems clear enough that the specifications of the BAT prediction problem ought to be related to the practical use case for the solutions. For real-time passenger information, predictions should be made on a fairly long time horizon, because passengers will plan their trips some time in advance. Bus bunching detection was another application of BAT prediction mentioned in the literature, and this probably also benefits from a longer

prediction horizon (Alexandre et al., 2022). Predictions used for transit signal priority may require only a shorter time horizon.

This time horizon results in a distinction between single-step ahead and multi-step ahead predictions. In some cases, studies limit themselves to the former. However, a single-step, which usually means one stop ahead of the current position, is not very far. Systems that can produce accurate predictions multiple steps ahead are preferable.

Public transport movements are highly cyclical. The vehicles repeat the same trips on a recurring basis, at the same times, day after day and week after week, according to schedules. Schedules are not usually changed on a day-to-day basis, so these cycles persist for many days at least. In addition, public transport vehicles that move on the road network (e.g. buses, trams) are affected by recurring patterns in traffic conditions. There is therefore a strong time-dependency in the phenomenon of bus arrivals that ought to be modeled somehow in BAT prediction solutions.

Techniques used to deal with bus arrival times generally try to capture geographical and temporal aspects of the problem. Because historic and real-time bus data consist of a series of observations over time, the problem can be thought of as a time-series prediction problem. The time series construction depends on a key assumption: that travel times that are temporally close to one another are also close to each other in travel times.

For bus operations, this is true to an extent, because of two primary reasons. First, bus operations are influenced by road congestions, which take some time to dissipate and therefore temporally adjacent buses may be influenced by the same congestion events (Bhutani et al., 2024; Petersen et al., 2019). Second, transportation has a cyclical pattern tied to human activities. Rush hour comes at the same time every day, subject to recurring congestion (Petersen et al., 2019).

Geographic relationships are also something that naturally are relevant to the BAT prediction problem. Geographic information can give some baseline guidelines to prediction of the movement of buses. For example, the length of segments can be associated with the travel times. But also segments that are in physical proximity to each other, or that connect to each other, may be more similar to each other than distant segments. Congestion propagation happens along network segments (Bhutani et al., 2024; Petersen et al., 2019)

3.5 Proposed Approach

In Chapter 2, the importance of building GeoAI models that can capture spatial dependency was discussed. Here, we explore the choices that researchers have in model building by comparing and contrasting two experimental models. Both models use the same underlying RNN structure, the gated recurrent unit (GRU), as a building block. However, the two models differ in how they employ these GRUs. In the first model, the sequence defined for the GRU exists in the time dimension. It is similar to a traditional time-series prediction problem in this sense, where a long series of past observations of the same phenomenon is used as input for predictions to be made for the near future. In the second model, the GRU is used in the spatial dimension. The sequence of travel times for ordered segments in a

single bus trip is given as input, and the predictions to be made are for segments downstream of the current bus location. Results are displayed and discussed.

The train set was used to train the model, adjusting the weights in the model using a standard method. After predictions are made, the resulting predictions are rated using two error measures: mean absolute error and root mean squared error. Predictions are compared to those from a simple model based on historical averages.

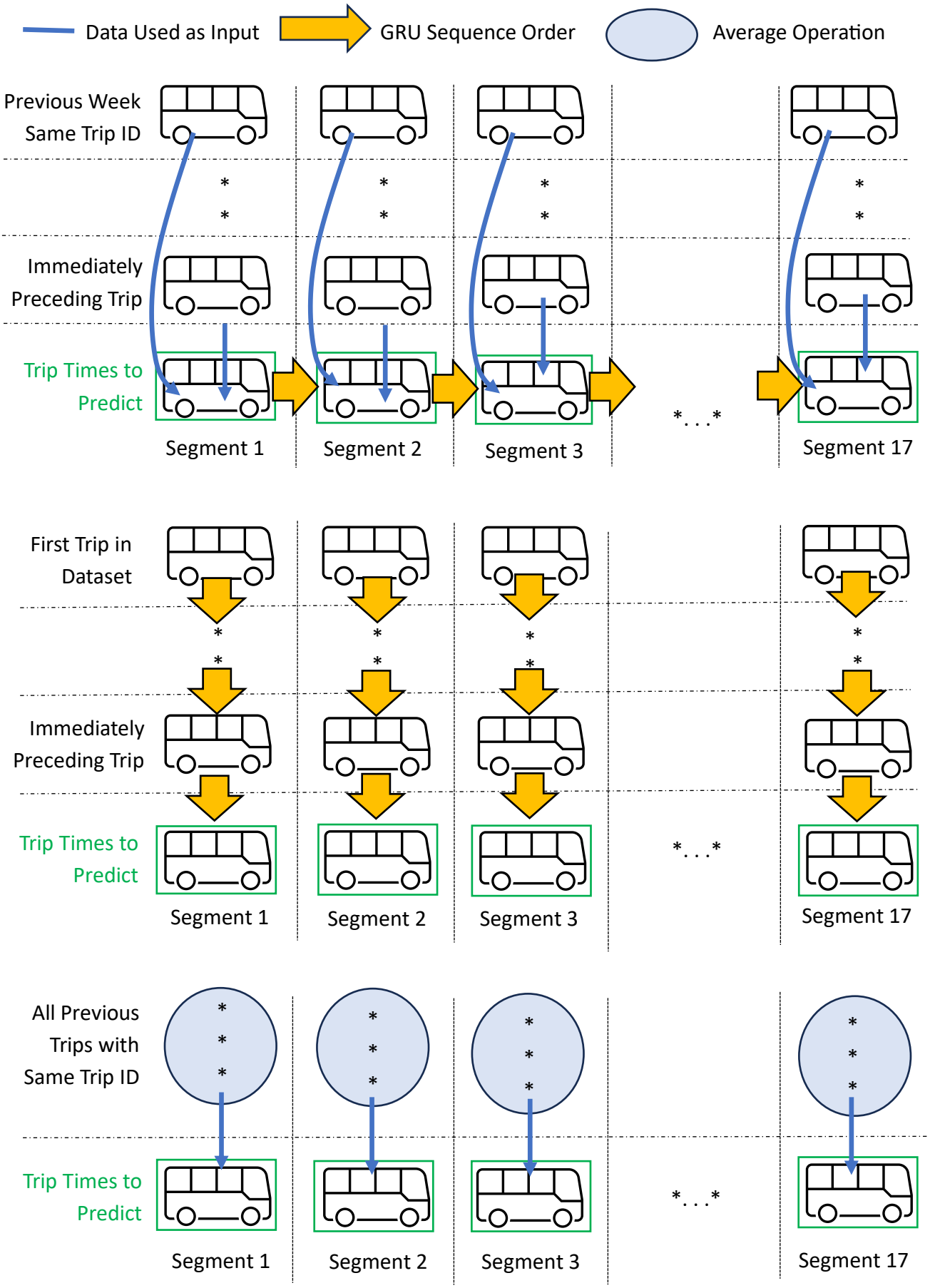
Gated Recurrent Unit defines a neural network of recurrent or sequential cells, which means that it operates in a sequential mode, where some output of the model is fed back into the model as input at each step in the sequence. GRU, similarly to Long Short-Term Memory, is distinguished from basic RNN units by the presence of gates (Olah, 2015). In a normal RNN, the output is received directly from the previous cell, but in GRU and LSTM, the gates define what information is taken as input. The existence of the gates makes it more possible for the cell to learn long-term relationships in the data (Olah, 2015; Petersen et al., 2019). In turn, this means that the GRU can learn what information it needs to pick up from each step in the sequence to pass on.

The wrinkle in GRU and LSTM is that the sequencing does not have to be mapped to the time dimension. Rather, anything that is predictably sequential will work. This means that it is possible to map a spatial sequence to the GRU (Bhutani et al., 2024). This includes a situation of predictable sequences like a given bus route with a set number of segments, where the presentation of the segments is the same every time. On the other hand, in the traditional approach, the LSTM is matched to the time dimension, although with bus prediction and unlike traffic prediction, bus data is obtained sparsely and not evenly distributed in time (Bhutani et al., 2024; Petersen et al., 2019). That means that neither choice is necessarily perfect in predicting bus arrival time.

In the spatial sequential case, the idea is that some information is transmitted in the spatial realm – that is, some phenomenon that passes through space – is detected, and that is what is passed on in the hidden states that each cell passes to the next in the sequence (Bhutani et al., 2024). In the temporal sequential case, we expect some phenomenon that passes through time to be detected and transmitted through the progression of hidden states. This has the benefit in that it allows that detections of congestion propagation could be used to inform predictions (Bhutani et al., 2024).

This uses a sequential method in order to capture relationships between certain phenomena that change in a repeatable way over time and space. It follows a type of problem where the goal is to predict a future state given some sequence of previous states (Gillain, 2022).

(reverse) **FIGURE 4.** Proposed models for application. From top: Seq2Seq GRU applying recurrence in the spatial dimension, along the route segments; Time Series GRU applying recurrence in the time dimension, separately by segment; historical average using averaging by segment and trip_id.


Data Used as Input
GRU Sequence Order
Average Operation
Previous Week
Same Trip ID
Immediately
Preceding Trip
Trip Times to
Predict
Segment 1
Segment 2
Segment 3
Segment 17
First Trip in
Dataset
Immediately
Preceding Trip
Trip Times to
Predict
Segment 1
Segment 2
Segment 3
Segment 17
All Previous
Trips with
Same Trip ID
Trip Times to
Predict
Segment 1
Segment 2
Segment 3
Segment 17

4. METHODS

4.1 Data Description

GTFS-Static

General Transit Feed Specification (GTFS) is a data specification designed to standardize how public transport agencies share information about their services (Newmark, 2024). The feed is a set of comma-separated values (.csv) files, with each file containing information on a different type of entity. GTFS feeds always contain a file of routes, a file of trips, a file of stops, and a file of stop times. Other files, such as a calendar file that describes which days each trip occurs, and a shapes file that geographically delineates the path of each trip, are often also included (Chondrodima, 2022).

In this case study, we will make use of five GTFS-static files: routes, trips, stops, stop times, and shapes. Routes represent a named public transport route – for example, "route 1." Trips describes a timed and repeated series of bus trips – for example, the trip that begins at 5:00 every weekday. Each trip is associated with one route. Stops describe the geographic points of bus stops or stations. Stop times are the timings when specific trips are scheduled to arrive at specific stops. Finally, shapes describe the geographic pattern that each trip is meant to take (Chondrodima, 2022). Note that routes are not described with geometry, so shapes are needed.

Records in these files are linked to each other by their use of common identifiers (Newmark, 2024). For example, to find all stop times at a given bus stop, one can search the stop times file for records where the "stop_id" matches the stop of interest.

To distinguish GTFS from GTFS-realtime, which is separate, we will refer to GTFS as "GTFS-static" (Chondrodima, 2022; Newmark, 2024). One GTFS-static feed is in effect at a time. GTFS-static feeds are updated on a regular basis, but never more than daily. This means that the operational uncertainty of the bus system – delays, service disruptions – is generally beneath GTFS-static's attention.

GTFS-Realtime

GTFS-realtime feeds report on the actual movement of the public transport vehicles as they occur. These are usually updated on a frequency of every 10, 15, or 60 seconds, depending on the transit agency and specific route (Chondrodima et al., 2022; Newmark et al., 2024).

Three file types are provided in GTFS-Realtime streams. The VehiclePositions file contains the current positions of transport vehicles. Predicted arrival times at future stops are described in the TripUpdates file. In addition, ServiceAlerts provides general warnings and notices to passengers regarding service that are too emergent or short-lived to be represented in GTFS-Static. Crucially, each record in a GTFS-realtime file is timestamped with the most recent time the information was updated.

Data is provided in the protocol buffer (.pb) format. This is a data format designed to efficiently transmit information on large numbers of relatively small units. It is not a human readable format, so it must be processed before use. Each file contains information on many units – for example, one VehiclePositions file provides the location of every vehicle in service at the time (plus some which are not in service).

In this study, we will only use the VehiclePositions file to produce BAT predictions. TripUpdates are also sometimes used by researchers to produce BAT predictions and to evaluate BAT predictions provided by the agency (Newmark, 2024).

4.2 Data Collection

GTFS-static feeds were downloaded directly from the MBTA website. The website indicates the days that each feed is active. All feeds active during the study period (4 March to 15 June) were downloaded.

For GTFS-realtime, a Python script was written to request VehiclePositions every 10 seconds. This resulted in receiving roughly 26.000 .pb files per day. Data collection occurred roughly continuously from 4 March to 15 June 2025. Howeve, two types of interruptions occurred. First, planned interruptions occurred in order to restart the program or computer; these, as much as possible, were timed to happen during the lowest point of daily bus operations, which is between 3:00 and 4:00 Eastern US time. In addition, unplanned interruptions occurred if the computer shut down or was put to sleep, due to error. As a result, only

4.3 Data Processing

Pre-Processing

Some of these data are duplicates that occur when the frequency of requests is faster than the data is refreshed in the system. Additionally, some data – particularly rail trips, which are also received from the query – are not used in this study. Note that that trips that were cancelled, not scheduled for that day, or not broadcasting were not analyzed.

Inferring Arrival Times

For each day, a list of scheduled stop times was generated by taking a list of bus trips found in that day's GTFS-realtime file, then querying all stop times from GTFS-static matching those trips.

Determination of arrival times at stops went using a two-stage process. The first stage of this matching was done by proximity. For each scheduled stop time, the earliest vehicle position within 20 meters of the stop location specified in GTFS-static was used. Exceptionally, for the first stop in each trip, the last vehicle position within proximity was used, because vehicles often arrive at the first stop far in advance of the trip beginning. In effect, this means we have the departure time for the first stop, in contrast with all other stops. The second stage was to use linear interpolation for scheduled stops that did not

have any vehicle positions within proximity. All vehicle positions within 100 meters of the trip path from GTFS-static were transferred to the closest point along the trip path. Then, the two vehicle positions on either side of the scheduled stop location were used to linearly interpolate the stop time. The procedure falls closest to that done by Chondrodima and colleagues (2022), who likewise used proximity, then interpolation to infer arrival times.

Segmentation

The approach segments the route so that dwell time at one stop plus the travel time to the next stop is the time value to be predicted. The exception is the first segment, in which the segment time includes only the travel time. Route 31 outbound has 18 stops, so the segmentation results in 17 segments for which to predict times. This discretization scheme was chosen because the resulting times can easily be added to the current time to predict arrival times at stops.

With this definition of segments, subtracting the final arrival time from the initial arrival time results in the segment time.

After this operation, some issues appeared. Some segment times were calculated as zero or negative. These segments were removed from the dataset. As a result, not all trips were complete.

In addition, preliminary data analysis showed that, for unknown reasons, trips starting after midnight (labeled hour >=25 in the dataset) had a higher mean segment time than any other start time in the dataset. It seemed to indicate a data or calculation issue, since the pattern was entirely inconsistent with the rest of the dataset. Because of this, these trips, which made up a relatively small proportion of the dataset, were removed from the test data set.

4.4 Model Architectures

Time Series GRU Model

This model treated each of the 17 segments in Route 31 outbound as separate time-series prediction problems. This means 17 different GRU neural networks were trained, each applying to one segment. All of the networks used identical architecture, but they developed different weights during the training process to fit the pattern of their respective segment. In this model, it is important to note that no information is passed between the 17 models. In this way, the geographic nature of the phenomenon of interest is largely ignored. As such, it is a non-spatially explicit implementation of GeoAI.

The input data for each segment was the entire time-sorted series of observations over the entire study period. That constituted a sequence of length between 7478 and 7900 observations, depending on the segment. Sequence lengths differed because some observations were discarded, as described in 4.3 above, if their segment times were calculated to be zero or negative, or if they began after midnight, due to issues with data or the segment time calculation. The input vector for each observed segment time consisted of the time of

day in decimal form; the elapsed time since the last observation; and the day of the week in one-hot coded form, which constituted seven binary variables. This made vectors of length 9.

GRUs work as follows: each observation gives its vector to the GRU. For each observation, the GRU calculates a "hidden state" based on the input data and the hidden state from the previous observation (or an initialized hidden state, for the first observation in the sequence). The calculation of the hidden state is done via multiple "gates:" these are neural network layers that control the updating and outputting of information from the GRU. In the training process, these neural network layers are the part of the GRU that changes, as the parameters of the neurons are adjusted in response to errors. After the calculation, the new hidden state is fed back into the GRU, ready for the next observation in the sequence.

In this case, the new hidden state was used to calculate the predicted segment time of that observation. To arrive at the predicted value, the new hidden state was fed through a ReLU layer then a fully connected neural network layer. The ReLU layer provides a non-linear function that allows networks to model non-linearity. The neural network layer converted the hidden state, which was a vector of some length, into a scalar value that was used as the prediction. The neural network layer also has weights that are adjusted in response to errors.

The length of the hidden state vector is a hyperparameter. Generally, larger "hidden sizes" allow more information to be passed through the hidden layer. It was set to 96 based on informal trial-and-error tuning. Additional hyperparameters included the number of neurons in the fully-connected layer, as well as the learning rate and the number of epochs.

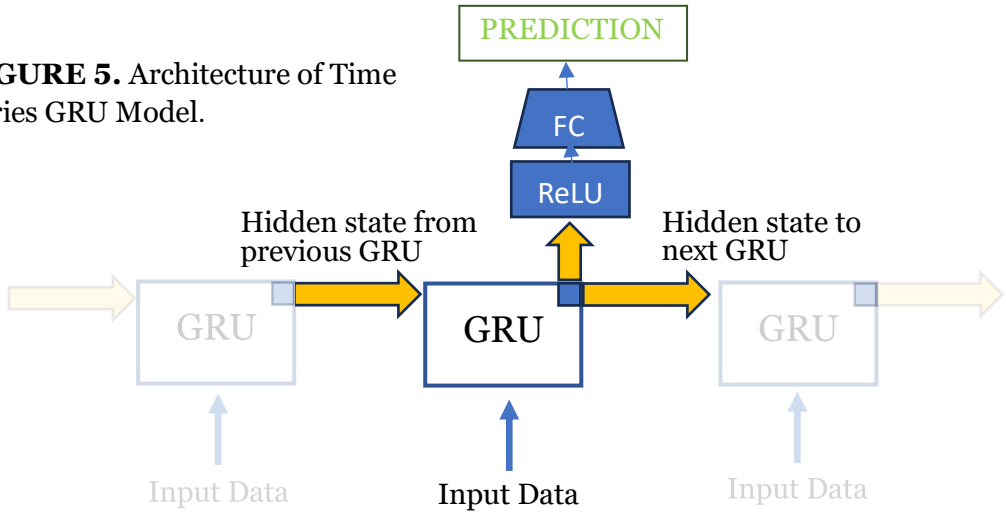


FIGURE 5. Architecture of Time Series GRU Model.

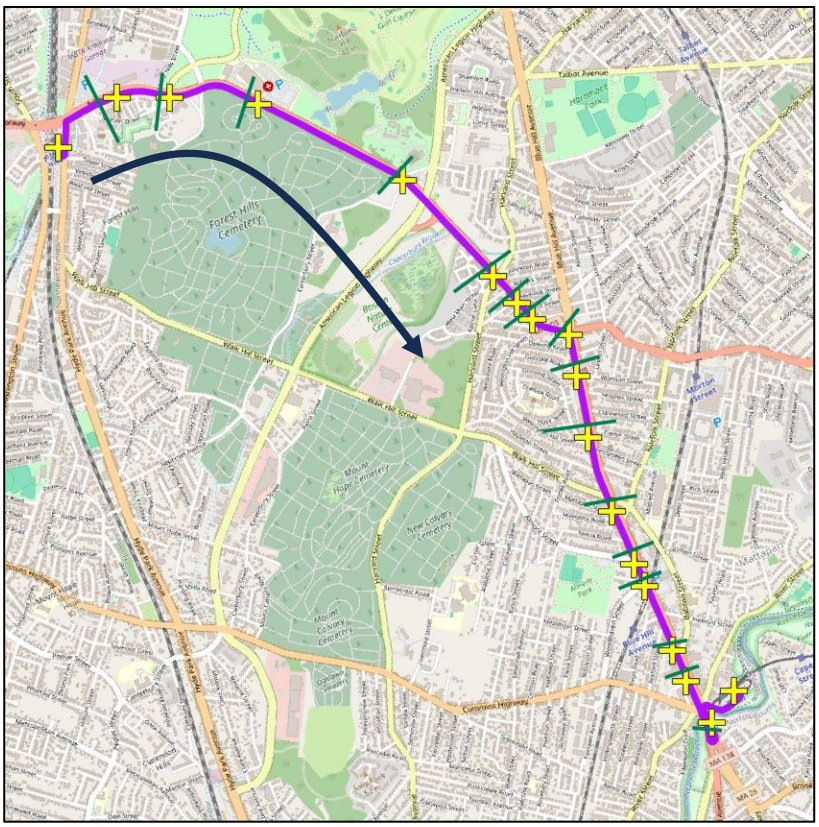


FIGURE 6. Route 31 (Outbound) in Boston, MA, USA. The purple line is the bus route, the black arrow indicates the direction of travel, and the yellow markers are stops. The green lines indicate approximate boundaries between the 17 segments, which are the basis for the prediction scheme.

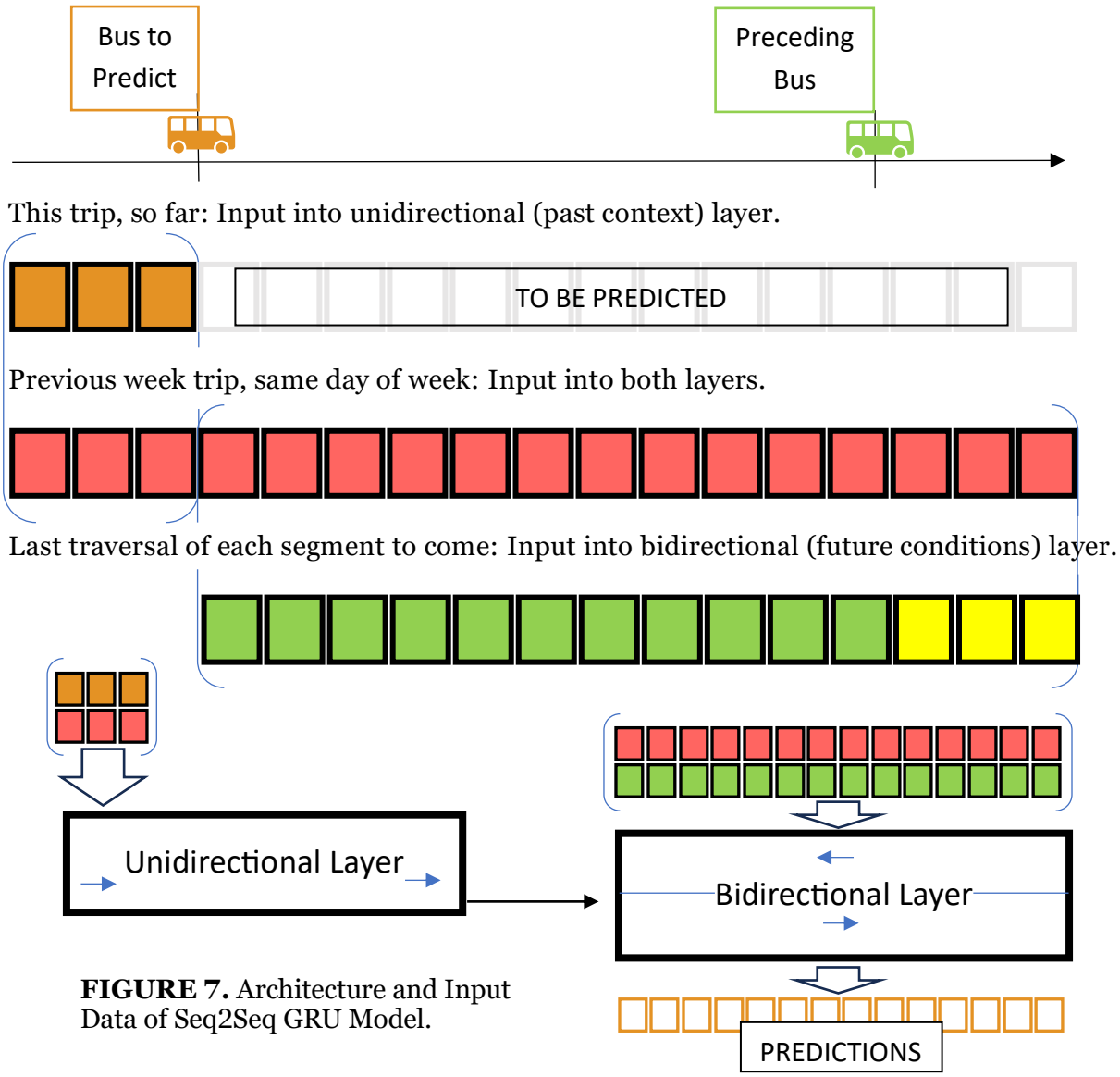


FIGURE 7. Architecture and Input Data of Seq2Seq GRU Model.

Seq2Seq GRU Model

This model, which was based on the architecture developed by Bhutani et al. (2024), worked on a trip level and used two major modules:

- A uni-directional GRU layer meant to aggregate information about previous segments traveled by the current trip and
- A bi-directional GRU layer meant to aggregate information downstream.

The fundamental sequence of data here is the trip, with length of 17 segments. For each arrival at a stop, the model can make a prediction based on the previously traveled segments. In addition, this data is supplemented by matching the trip to a trip from a previous week on the same day of the week, as well as the immediately preceding traversal of each segment. From all these trips, the input is the time of entry into in each segment and the

The unidirectional layer took the sequence of previously traveled segment observations as input data. Each of those observations was a vector consisting of the time of entry at the

segment in decimal form and the segment time, as well as the time of day and segment time of the previous week trip. The final hidden state from this layer was passed to the bi-directional layer.

The bidirectional layer had an input data sequence corresponding to the segments that remained on the trip, with one element being one segment. The input vector was the data from both previous trips – both the time of day and the travel time. In addition, it took the hidden state input from the previous layer as input for every cell.

In the machine learning steps, a model is built to learn one set of parameters for route 31, direction 0 across all times and weekdays. Input data, which consisted of roughly 5000 trips, were divided into training and test sets on a 5:1 ratio (~83%/17%).

In the machine learning steps, the model is built using a gated recurrent unit (GRU) layer to combine context found in the times of the previously traveled segments of the trip to predict. Then the output of that layer is fed into a separate, bidirectional GRU layer which predicts the travel time on each untraveled segment to come. The bidirectional layer means that each segment takes information from downstream segments and upstream segments.

Baseline Models 1 and 2: Historical Average

Two historical average baseline models were used. The only difference between the models is the specific training and test sets. Baseline Model 1 uses the same training and test data as the Time Series GRU. Baseline Model 2 uses the same training and test data as the Seq2Seq GRU.

For each trip_id-segment combination in the training set, average segment times are calculated. Any trip_id-segment combination in the test set takes the corresponding average from the training set as its prediction.

This scheme was chosen because it accounts for the temporal and spatial aspects of the bus travel in a non-machine learning method. The trip_id refers to trips that happen at the same time on similar days (weekdays, for example), so the predictions make sense from a time perspective. The segments are of different lengths and average traversal times, so they need to be treated separately. Therefore, the segment ids ensure that each segment time is only compared to other recorded times on the same segment.

4.5 Training and Testing

Allocation of Training and Test Sets

The Time Series GRU model depends on a continuous, time-based chain of GRU units. The model may not perform correctly if it were reinitiated for the test phase, and randomly assigning days to training or test would disorder the sequence of observations. Hence, the training and test sets were divided on the 5:1 ratio so that the test phase followed the training phase. This meant all data from 4 March to 28 May was taken as training data and all data from 29 May to 15 June were taken as test data.

For the Seq2Seq model, in contrast, we took a random sample of trips that had complete data. To have complete data, the trip to predict had to have at least one complete prior trip during that day and one complete trip from the same day of the week, on a previous week, with the same trip_id. The trip to predict needed to be complete as well. From trips that met these criteria, we divided them randomly into training and test trips on the 5:1 ratio.

Training Process

Programming of the neural network models was done using Python 3.11 with PyTorch for the machine learning components. Following the process separately for each model, the training data was administered to the models. All models used a batch size of 1, meaning that model weights were revised after each item in all input sequences. Absolute error, also known as L1 error, was used as the loss function for the training process.

For the Time Series GRU model, each model was trained separately only on observations for the corresponding segment. All segments' models were trained for 136 epochs. After the first epoch and again after every additional 15 epochs, target values and predictions for the training set were saved. Then the model made predictions for the test set, without modifying the model weights, and those values and predictions were saved as well. This process meant that training and testing essentially happened in parallel; of course, the model was not allowed to learn from the test data, preserving the integrity of the train/test process.

The Seq2Seq GRU model was trained for 60 epochs. The same process was done for this model, except the first saving happened after 10 epochs and then every 10 epochs thereafter.

Note that the Seq2Seq GRU is different in that it produces multiple predictions for most segments, because each trip updates its predictions with every new segment traversed. That means that the last segment is predicted 17 separate times, the second to last 16 times, and so on. To make it more comparable to the other models, which only produce one prediction per segment, the list of predictions for each segment was averaged. This produced a composite prediction that reflected the overall accuracy in predicting that segment time.

Using the training set outputs, we generated training curves using the principal error measures, mean absolute error (MAE) and root mean squared error (RMSE), which are commonly used in the machine learning literature. We selected to use the model parameters from the most

successful epoch (lowest error measures) on MAE. Separate choices were allowed for each segment model in the Time Series GRU, since they were separate neural networks.

Testing Process

Once each model was chosen using the training measures, we took the values that exact model produced during its test run and calculated the error measures. Data was also merged on a record-by-record basis to allow analysis by time-of-day and segment. These analyses, as well as top-level measures, are presented in the results section.

5. RESULTS

5.1 Training Results

The resulting error measures for these test runs is shown below as curves. This comparison shows that some variation exists in the number of training epochs after which the model is most accurate on the test data. Most segment models reached maximum test accuracy (minimum error) after 15 or 30 epochs.

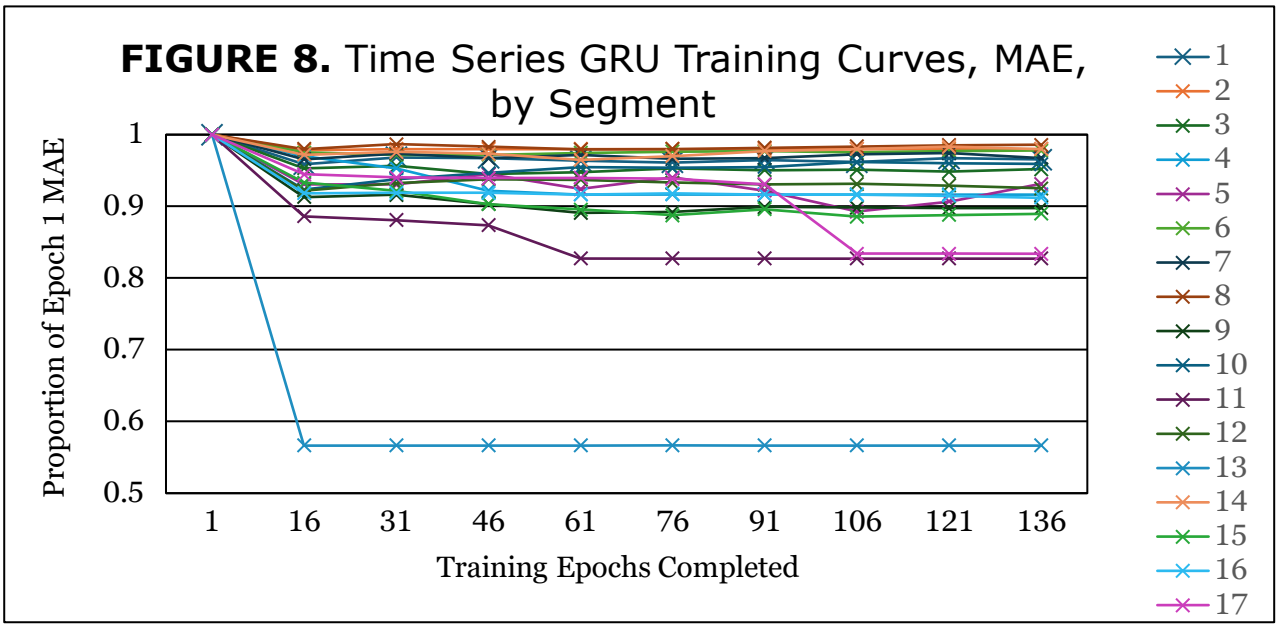


FIGURE 8. Time Series GRU Training Curves, MAE, by Segment

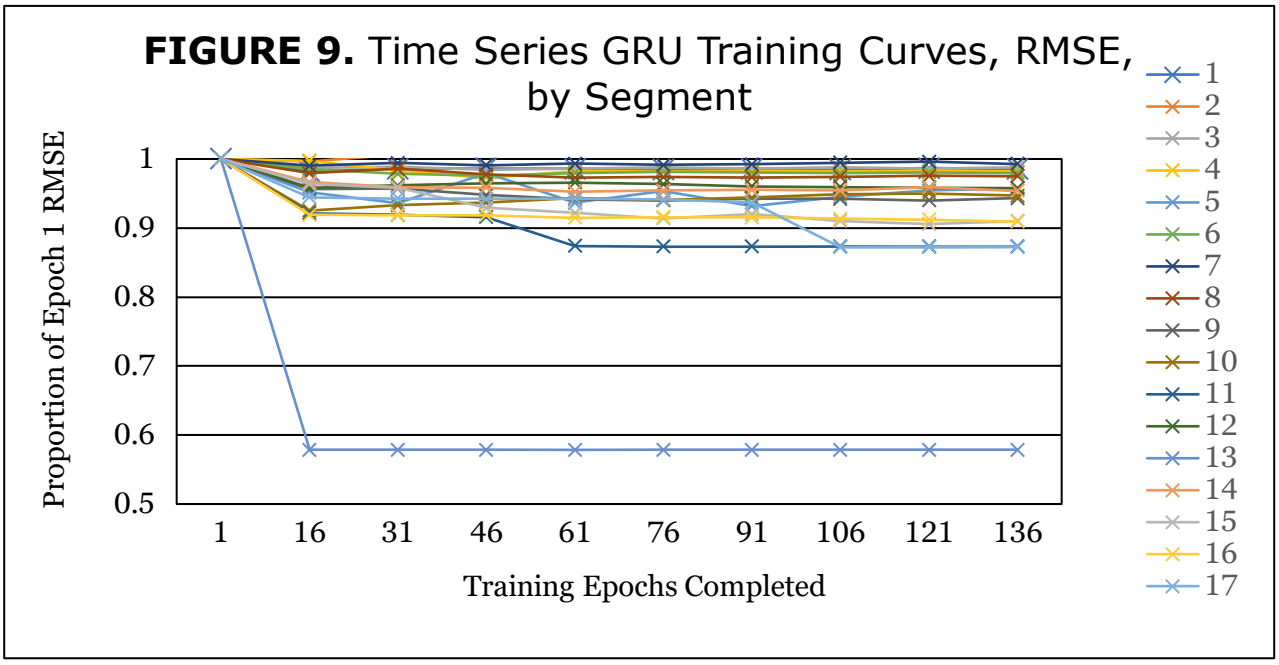


FIGURE 9. Time Series GRU Training Curves, RMSE, by Segment

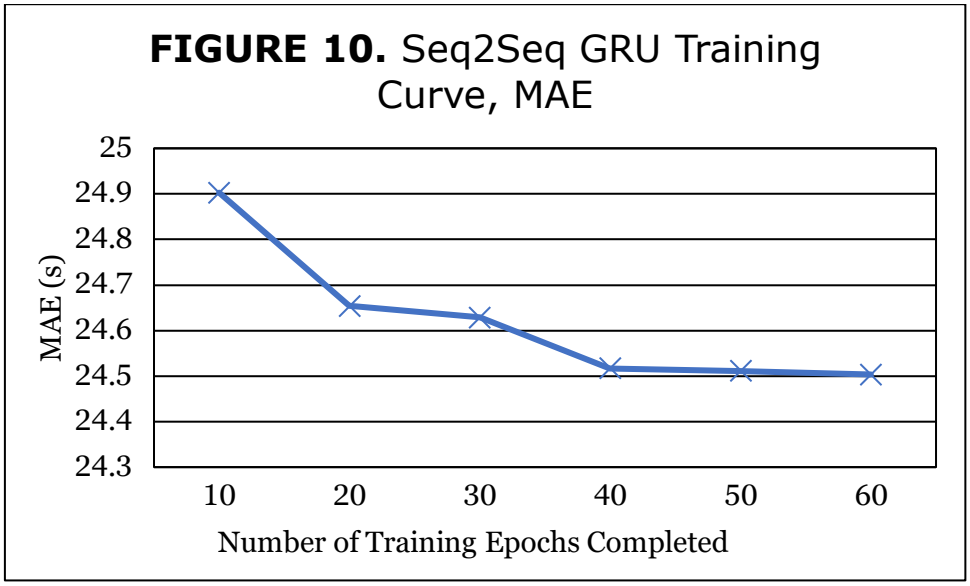


FIGURE 10. Seq2Seq GRU Training Curve, MAE

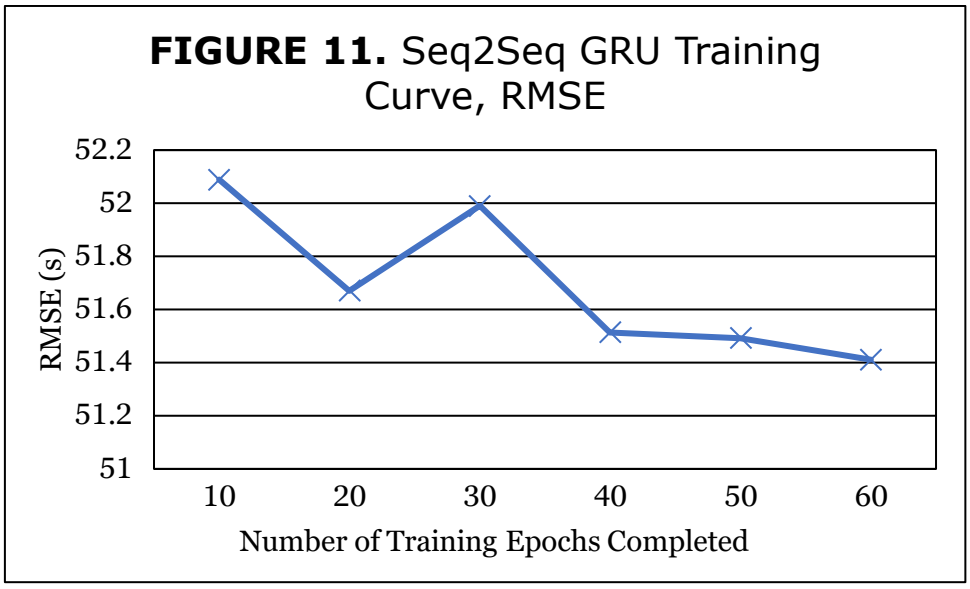


FIGURE 11. Seq2Seq GRU Training Curve, RMSE

The lowest level of MAE and RMSE were found after 60 epochs of training, so that model was chosen.

5.2 Testing Results

Overall Performance Measures

Performance Measures Averaged Across All Segments, by Model (Lower is Better)			
	RMSE	MAE	MAPE
Time Series GRU	46.0 s	26.5 s	47.7%
Baseline Model 1	51.8 s	29.0 s	52.2%
Seq2Seq GRU	55.5 s	26.7 s	41.4 %
Baseline Model 2	49.1 s	26.0 s	46.6 %

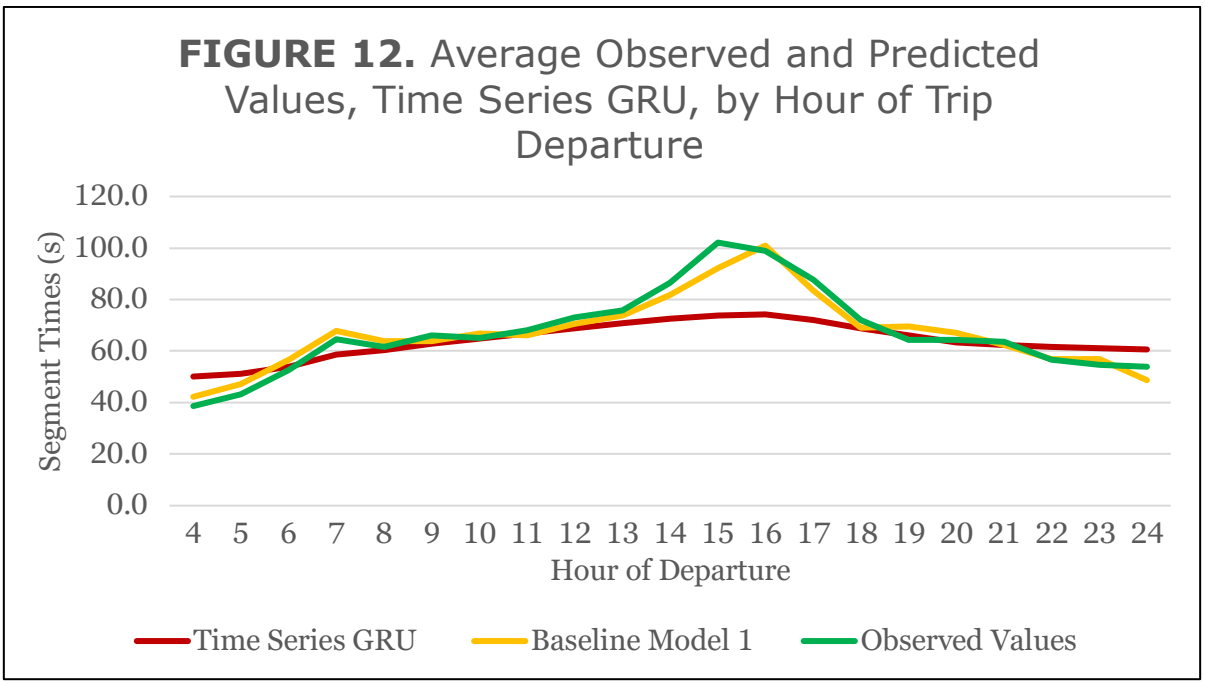
TABLE 3. Performance Measures, by Model

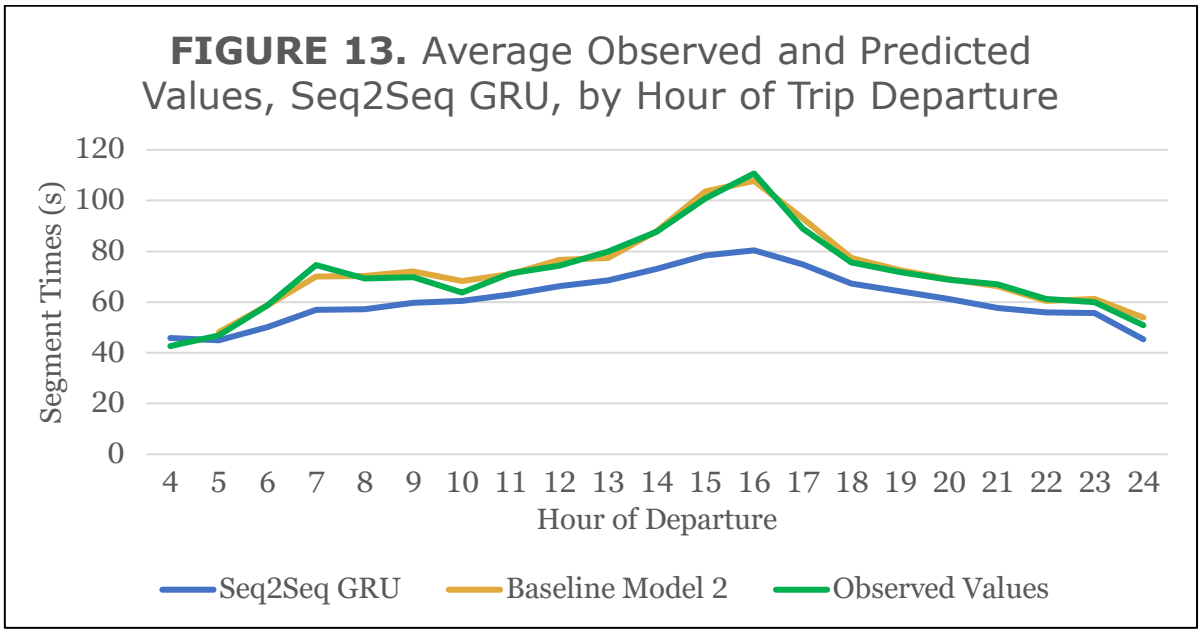
The Time Series GRU is the best model in terms of RMSE. The Baseline Model 2 is the best in terms of MAE, although both the Seq2Seq GRU and the Time Series GRU have comparable results on that measure. The Seq2Seq has clearly lower MAPE than the competing models. However, it also has the highest RMSE.

Comparing the two neural network models, the Time Series GRU performs better on the RMSE and MAE measures, while the Seq2Seq GRU performs better on MAPE.

In comparing the Baseline Model 1 and Model 2, Baseline Model 2 does substantially better on all performance measures.

Predicted Values and Target Values, By Time-of-Day





These line charts show the actual, average values of segment times as predicted by all models and as observed. Note that the observed values differ between the two charts; this is because the charts are sorted by the training/test set composition. The first chart uses the sequential training/test division (as do the contained models), while the second chart uses the randomly assigned training/test division.

Reading both charts, it is clear that the historical average (Baseline Models 1 + 2) closely track the daily pattern of segment times. This pattern consists of a minor peak centered at 7AM and a major peak centered on 3-4pm (hours 15-16). Notably, neither neural network model does a great job at tracking the daily time pattern. Each has a slope that imitates the slope of the observed time pattern, but is constantly less in terms of degree. For the

Time Series GRU, this means that the beginning and end of the day involve systematic overestimation of travel times, while the PM peak involves systematic underestimation. For the Seq2Seq GRU, another clear finding is a substantial underestimation of travel times across the day. This is particularly pronounced at the PM peak, where, like the Time Series GRU, it struggles to adapt to predict the high travel times.

Performance Measures, by Time-of-Day

Looking at the Time Series GRU results in detail, we see that the model outperforms the historical average baseline in most hours of the day, with the notable exception of the pm peak. For MAPE, the Time Series GRU performs better than the historical average baseline during most times, with the exception of the beginning and the end of the day.

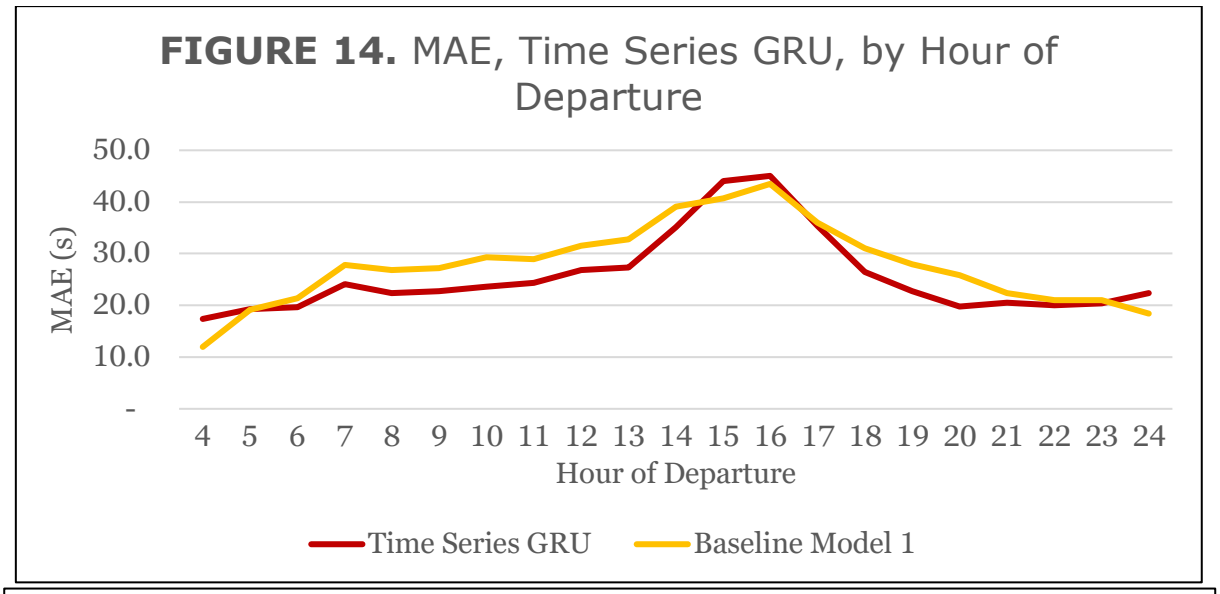


FIGURE 14. MAE, Time Series GRU, by Hour of Departure

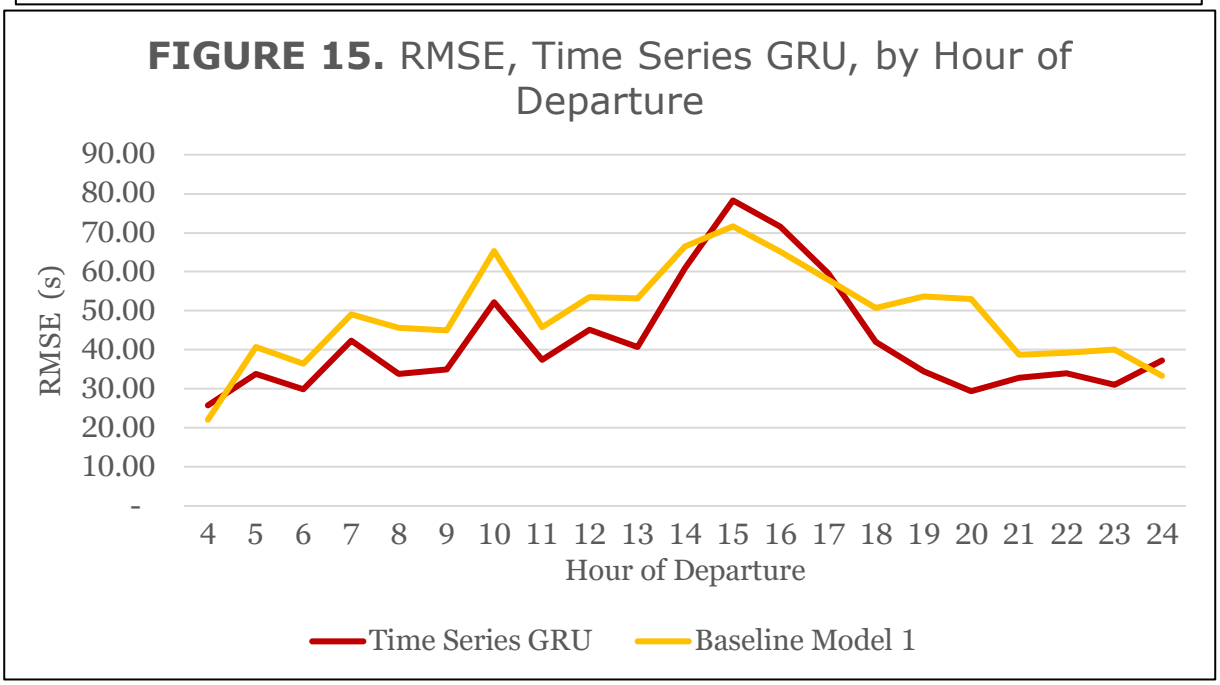
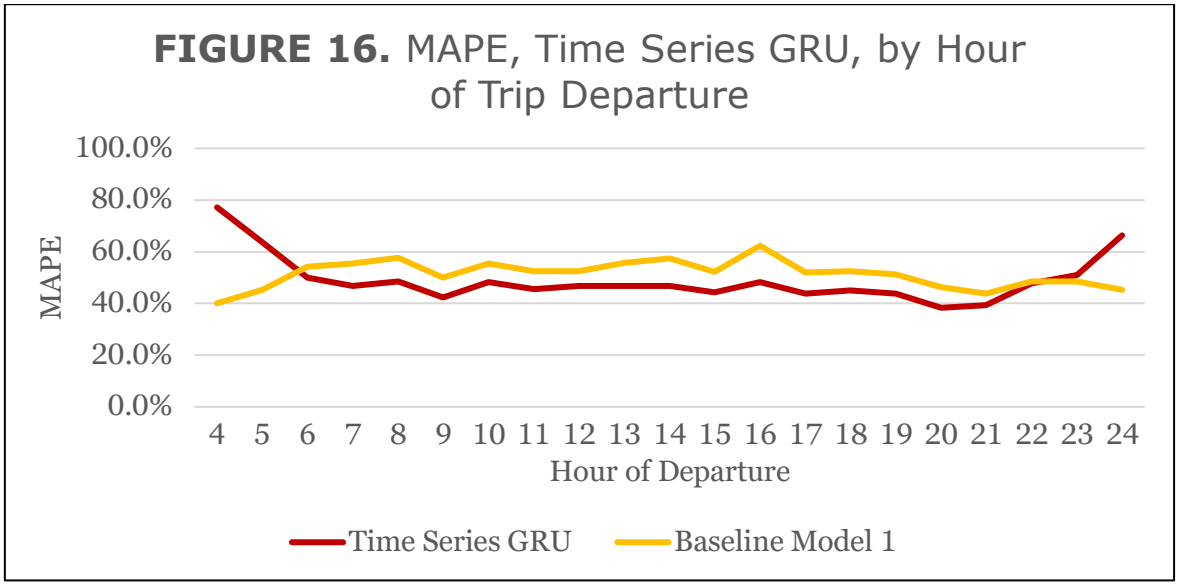


FIGURE 15. RMSE, Time Series GRU, by Hour of Departure



The Seq2Seq model, on the other hand, underperforms the historical average baseline for most of the day on RMSE and MAPE. For MAE, it is comparable to the baseline for most of the day, except for the PM peak, when it performs somewhat worse than the baseline.

FIGURE 17. MAE, Seq2Seq GRU, by Hour of Trip Departure

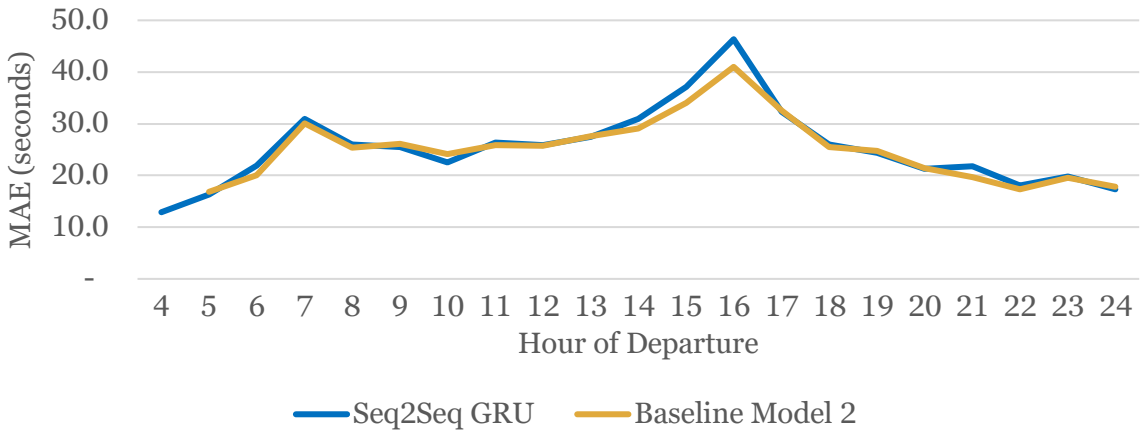


FIGURE 18. RMSE, Seq2Seq GRU, by Hour of Trip Departure

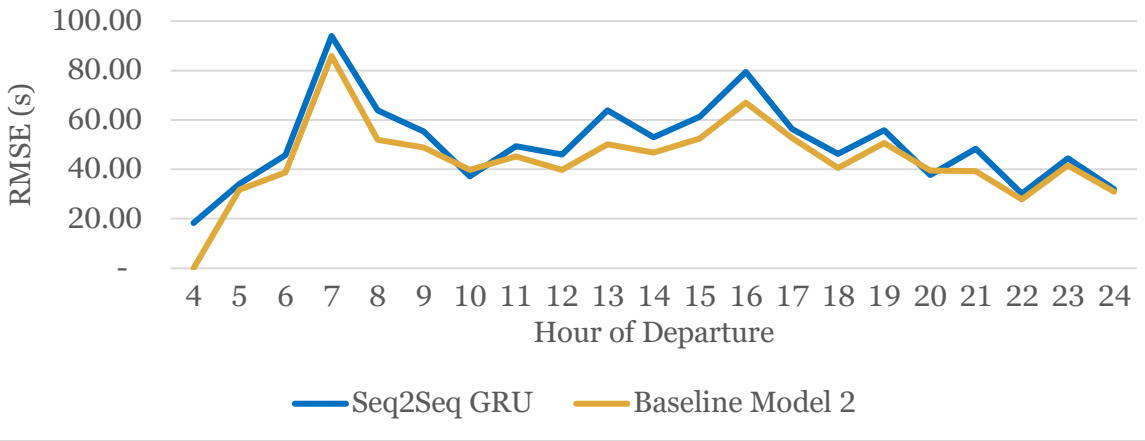
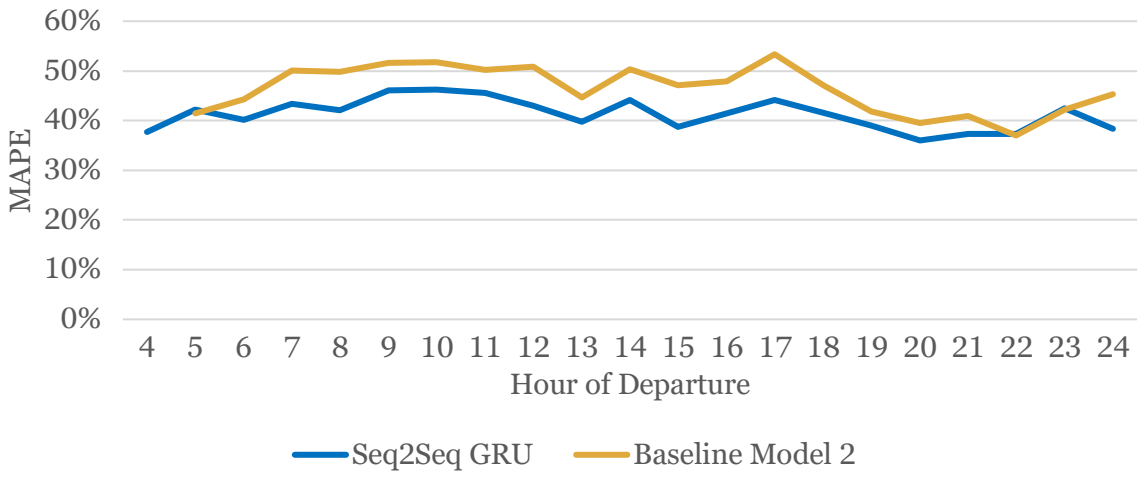
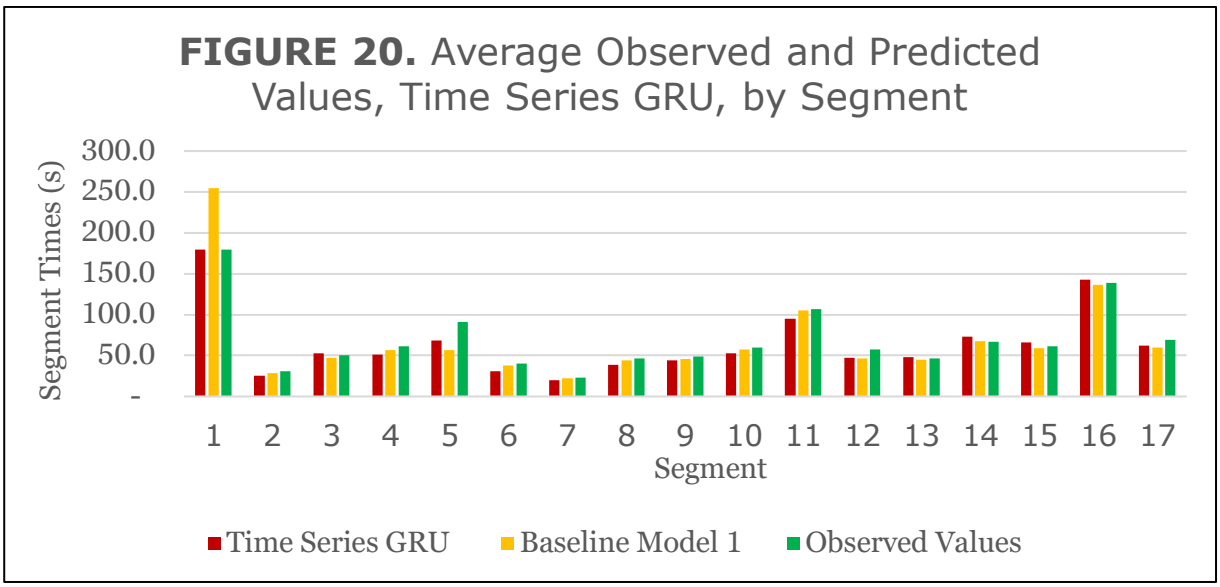
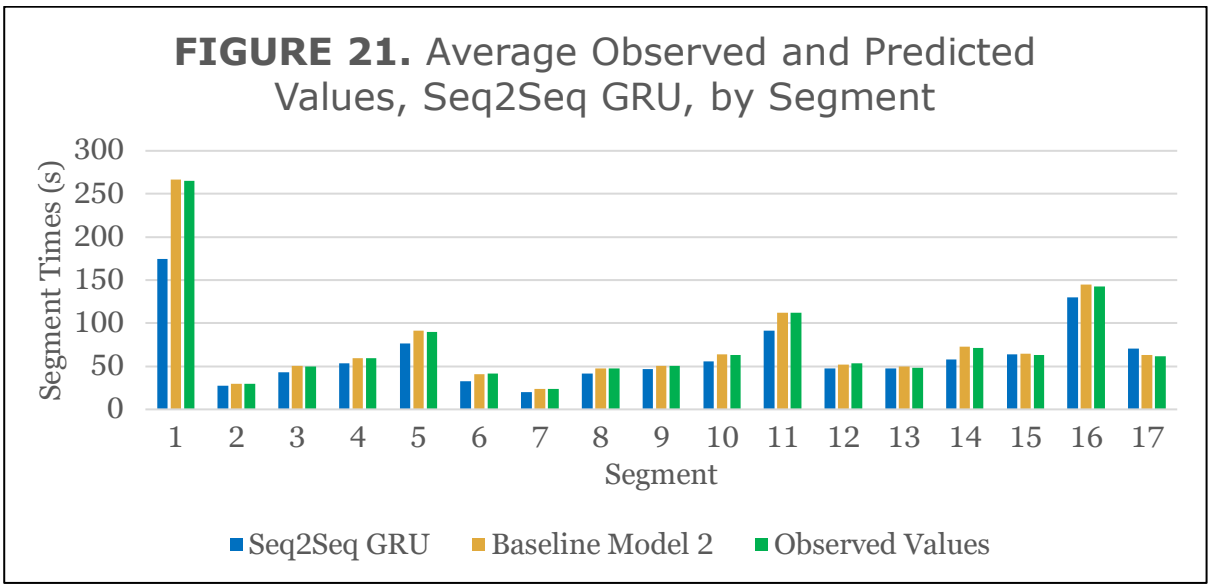


FIGURE 19. MAPE, Seq2Seq GRU, by Hour of Trip Departure



Predicted Values and Target Values, By Segment





These bar charts display the predicted and observed values by segment. We see, overall, that observed travel time ranges quite a bit between segments, with segment 7 being the lowest at roughly 25 seconds and segment 1 being the highest around 175 seconds in the observed values in the sequential training/test division (top chart) and about 275 seconds in the random training/test division (bottom chart). The substantial difference between the segment 1 figures is noteworthy.

We see that the Time Series GRU has a slight tendency to underestimate values, but is otherwise consistently near the observed values. The Seq2Seq GRU tends to underestimate travel times for all segments, but particularly segment 1.

Performance Measures, By Segment

FIGURE 22. MAE, Time Series GRU Model, by Segment

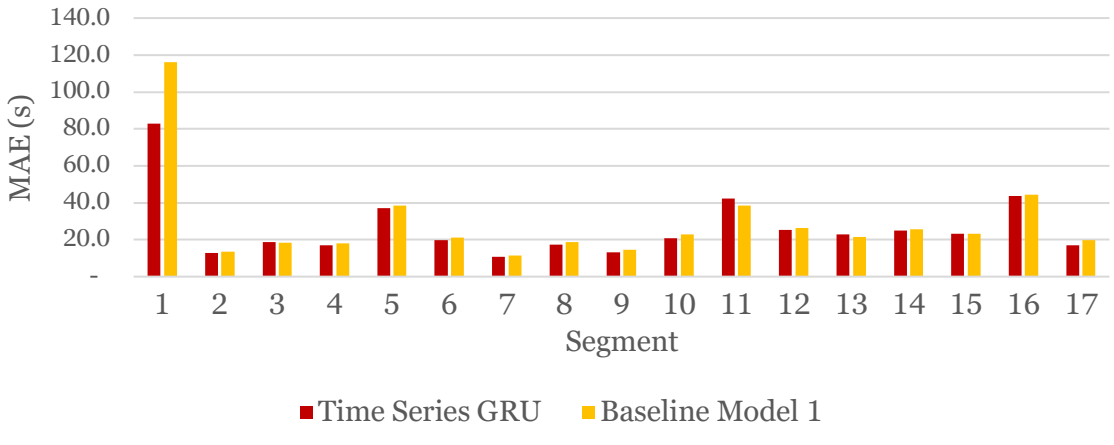


FIGURE 23. RMSE, Time Series GRU, by Segment

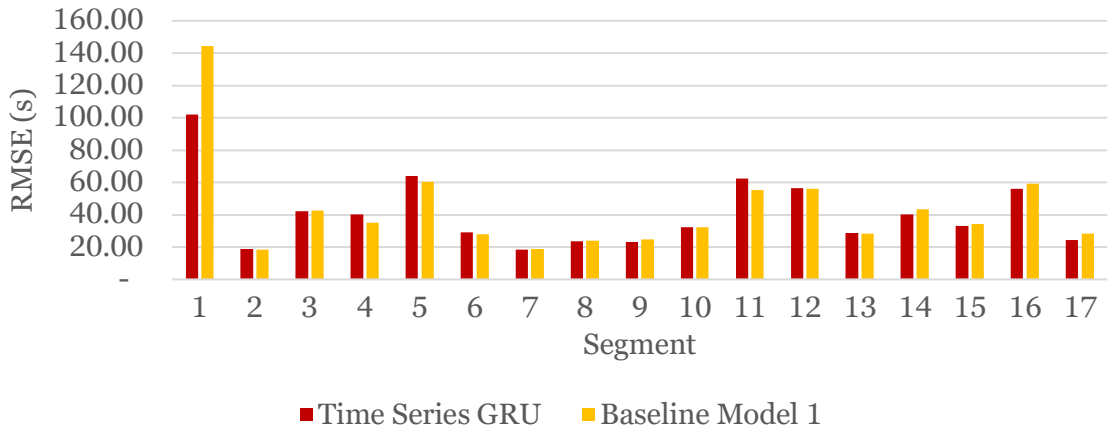
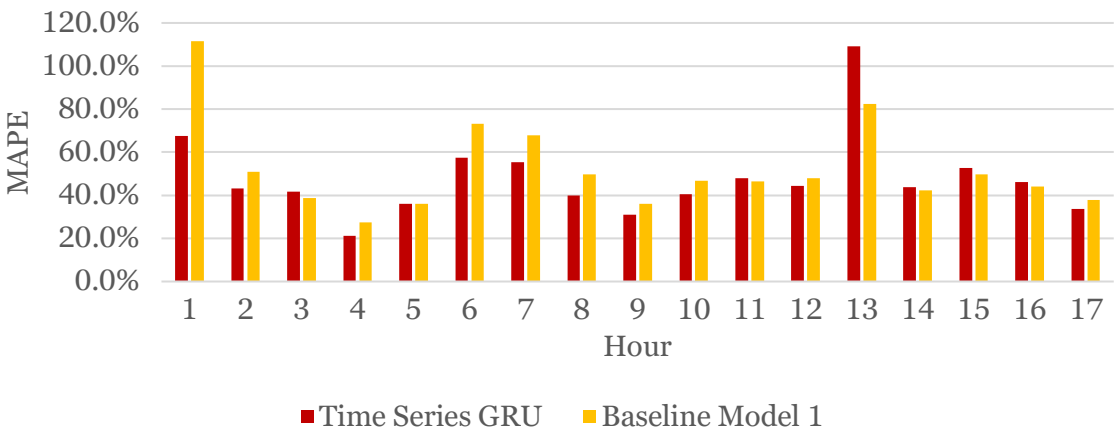


FIGURE 24. MAPE, Time Series GRU, by Segment



Moving onto performance measures, interestingly, the Time Series GRU is comparable to the performance of the baseline model for most segments. The notable exception is segment 1, where the Time Series GRU significantly outperforms the baseline model. For MAPE, the Time Series GRU is generally better than the baseline, with the exception of segment 13, where the Time Series GRU underperforms the baseline model.

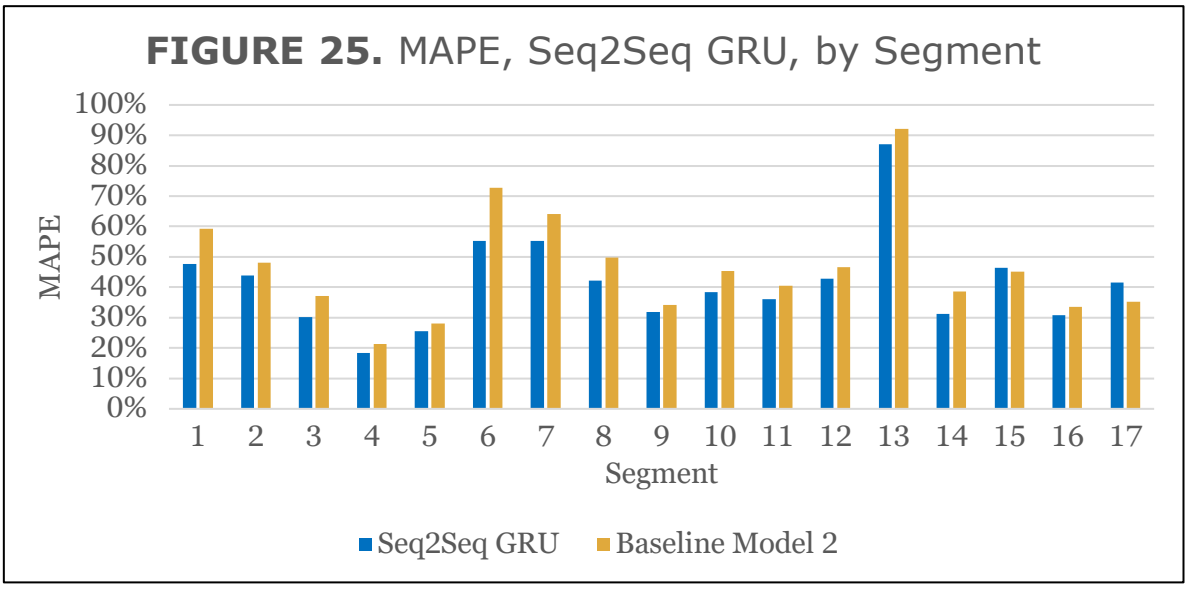


FIGURE 25. MAPE, Seq2Seq GRU, by Segment

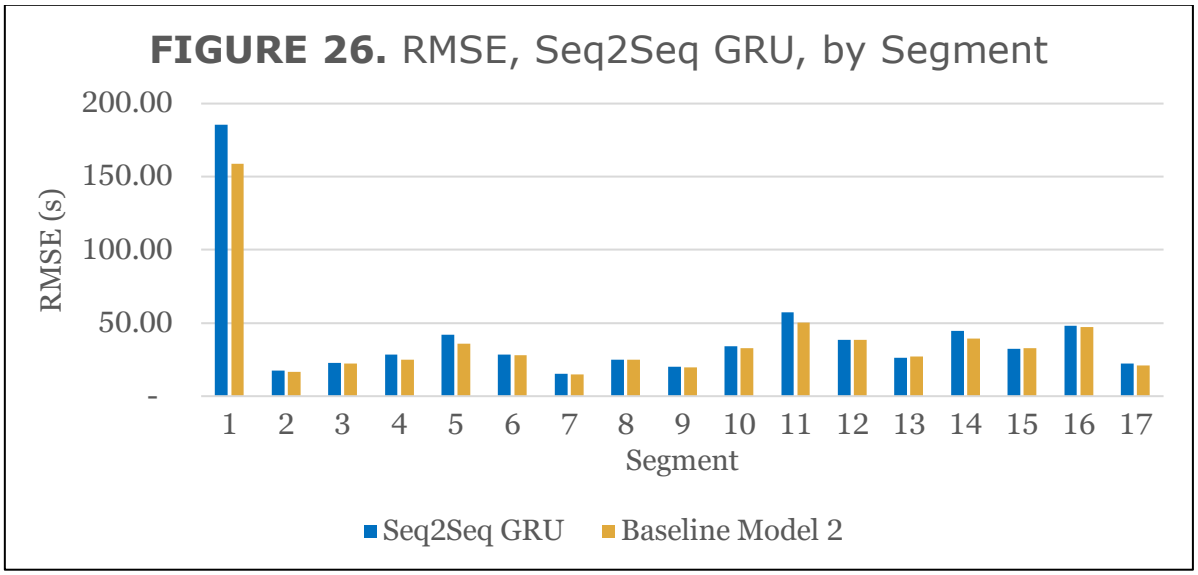
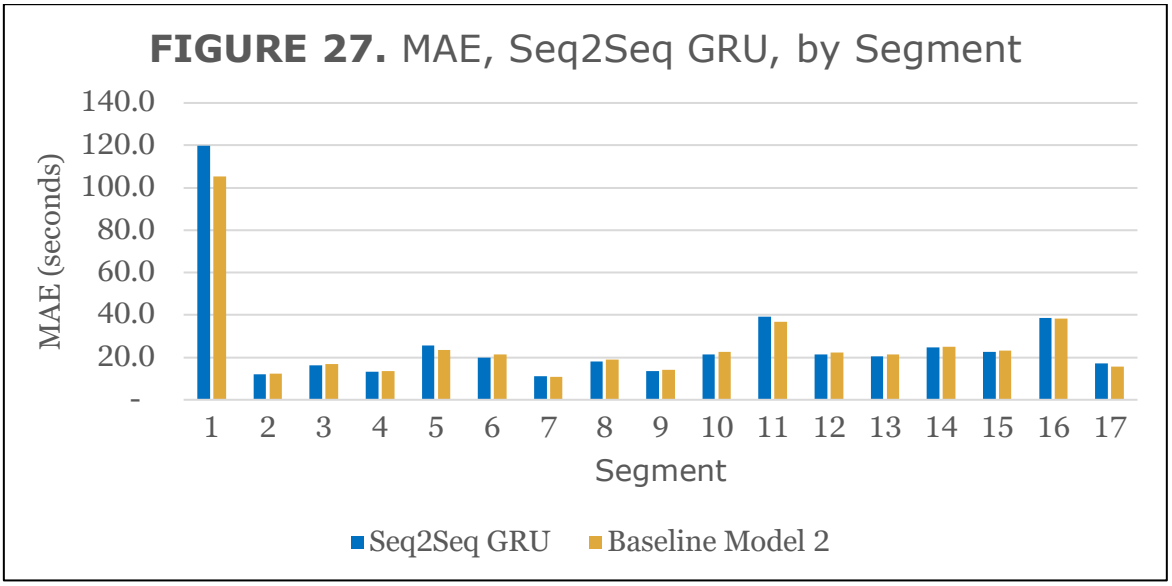


FIGURE 26. RMSE, Seq2Seq GRU, by Segment



For the segment-wise analysis of the Seq2Seq GRU and baseline, we see that the Seq2Seq GRU has rather an opposite situation to the Time Series GRU – it underperforms on segment 1 in particular, while the performance on the rest of the segments compares to the baseline. This excludes the MAPE results, where the Seq2Seq GRU notably performs better than the baseline on most segments.

6. DISCUSSION

6.1 Current Study

Overall Performance

Overall, the Time Series GRU model clearly performed better than the Seq2Seq model in this application. While the Time Series GRU outperformed its baseline comparison model on all three performance measures, the Seq2Seq GRU model outperformed its baseline comparison model on only MAPE.

The chief advantage of the Seq2Seq GRU is its low MAPE, which was the lowest of all models. MAE is comparable to other models, so the performance on MAPE must be attributable to relatively high accuracy in predicting low ground-truth values. Unfortunately, it appears that this strength is primarily due to this model struggling to anticipate long travel times and its systematic underestimation of travel times. The fact that its chief strength seems to be related to a weakness in the model is troubling. Concretely, it is that tendency to miss high travel times that leads it to have the highest RMSE of all models by almost four seconds.

Did the Time Series GRU outperform both baselines? Comparison between the Baseline Model 2 and the Time Series GRU shows that the baseline model 2 beats the Time Series GRU on two of the three measures. But, the Time Series GRU is considerably better than Baseline Model 2 on RMSE, while performance of the two models is comparable on the other two measures. Therefore, it is difficult to say definitively which model is better head-to-head.

However, it is important to note that the data set for the Time Series GRU may have made a more difficult prediction task, which subjectively makes its performance more impressive. Consider that the Baseline Model 1 performed considerably worse than Baseline Model 2 on all performance measures. As mentioned, the only difference between those models is the input data. Baseline Model 1 took training data from 4 March to 28 May and test data from 29 May to 15 June. Potentially, seasonal or other effects made inference more difficult from the training to the test period, in contrast to the random sampling used for Baseline Model 2. The fact that the Time Series GRU can perform comparably to baseline 2 on a more difficult task reflects favorably on the former.

The performance of the Time Series GRU is also impressive in that it is a very simple model. In fact, the data processing of training and test data meant that it was making predictions only on the basis of scheduled times. After being trained on the historical data, it did not take any real-time data – essentially, the predictions could have been made all before 29 May, with maximum prediction horizon of about 18 days. Now, this fact does not make it a “good” model; certainly it would be better if it were designed to take real-time data into account. It does bode well for the future development of this type of model, since real-time information would likely be able to improve the model’s performance.

Finally, it is important to note that both neural network models had a tendency to underestimate travel times. For the Time Series GRU, this was about five seconds, and it was about eleven seconds for the Seq2Seq GRU. Because of the nature of

Explanations for Performance

The Seq2Seq GRU model notably performed better than competing RNN approaches in the study in which it was introduced (Bhutani et al., 2022). Why did the Seq2Seq model struggle in this example? A few reasons are possible. First, there are slight differences in the model with the original example. In the original example, one-hot coded segment variables in the input were aggregated in a way that reportedly gave better performance. In the current iteration, segment numbers were left in as integers.

In particular, we see two trends regarding Seq2Seq GRU's performance. First, an inability to improve on the historical average approach. Second, the high degree of underestimation, which occurs across the board in segment and time-of-day (which helps to explain the high performance in MAPE given the lower performance in MAE). Third, particular difficulty with segment 1.

It is clear that the Seq2Seq model had some difficulty predicting different segments differently. On segment 1, which takes the longest of any segment by far on average, the Seq2Seq model underestimates travel times by an average of roughly 90 seconds. Other segments are also underestimated, but not nearly to this extent. It seems that the Seq2Seq model is having trouble learning the segment-wise pattern of travel times. A hypothesis is that this relates to the way the GRU is wired. Essentially, the same GRU is trying to predict a very uncommon phenomena when it predicts segment 1, which has more than double the average travel time of the other segments. It may be taking knowledge from other segments and applying it to segment 1, leading to large underestimation of travel time.

There is also another aspect in that the Seq2Seq GRU has the lowest level of direct input of historical knowledge, with only times of two trips taken as input.

The higher variability in errors, as well as the tendency for the neural network models to underestimate the travel time, could possibly be a tendency of neural networks when trained on positively skewed data. In a previous BAT example, Chondrodima and colleagues (2022) found a neural network model that performed well but tended to produce larger errors in the extremes (ground-truth values). It tended to make predictions more towards the center of the predictions rather than the extremes.

It is still surprising that the neural network displays this quality in comparison to a historical average approach, which only makes predictions towards the center of the predictions.

Neural Network Struggles to Learn Time-of-Day Patterns

Another possible explanation for the variability of the errors is that the neural networks struggled to learn the hourly pattern of travel times as indicated in Figures 12-13. In

support of this, we see in the hourly performance metrics that, for the Seq2Seq model, error metrics are greatest from 7-8 and 13-16. We also see this is where travel times are high and also where the mean error of the Seq2Seq model is most negative. To summarize, the Seq2Seq model is underestimating travel times, and getting some predictions very wrong, because it is making predictions that do not consider the larger travel times during peak hours. This does not entirely explain the underestimation – the underestimation is present throughout the day – but a part of it.

As for the root cause: why is it that the Seq2Seq model is struggling to reproduce the hourly variability in travel time? We can note, as additional evidence, that travel times are still elevated in hours 17-18, but decreasing. Here, the Seq2Seq does a better job of prediction. This aligns with what we would expect of a moving average approach, where the prediction lags behind changes in the underlying data. We can consider that the Seq2Seq model does take the immediately previous trip as input. It is possible that the predictions ended up being skewed towards repeating the previous travel time, and this resulted in a lag of predictions behind real hourly patterns.

It is worthy to note that the Seq2Seq model does take time of day as input, as well as segment observations from the previous week's trip at the same time. So, information is available that could help the model learn the hourly patterns better. For some reason, this did not occur in this experiment.

6.2 Future Directions

Further Investigations

One area of possibility would be to use synthetic data to investigate the capabilities of the models used in this experiment. Permutation is a way that geographic scientists have attempted to explain the workings of GeoAI models (Liu et al., 2023). Regarding the time-lag effect suggested by Figures 12 and 13, testing using synthetic data could help to explore whether the seq2seq GRU model is capable of modeling certain types of variations. For example, for the question of whether the seq2seq GRU can model variations associated with time of day, it would be possible to create data that only varies associated with the time of day. Then, whether the seq2seq GRU model could effectively model this would show the types of real-life variation that the model can reproduce.

One explanation is that the input data for this study were flawed. It is true that the data were the result of significant processing by the researcher. If errors occurred, it may become more difficult for the model to predict the outcomes. However, it is worth noting that the training and test set are randomly assigned from the same underlying data, and the historical average approach also uses the flawed data – and produces more accurate predictions.

Theoretically, it is possible that noise or non-systematic error in the input data contributes to the difference in performance between the historical average approach and the seq2seq approach. Noise in the input data would be expected to affect the seq2seq

approach more heavily because the seq2seq approach relies on fewer prior examples as input for each prediction than the historical average approach. A way to test this would be to limit the historical average approach to use at max previous two trips as an average and to examine the performance relative to the seq2seq approach.

Note that the MBTA does publish an arrival time dataset that exists for the study period. However, because the trip descriptions for that dataset do not match those found in GTFS or in GTFS-RT, it is not possible to use that dataset for validation of the arrival time calculations in the current study. Hypothetically, if an arrival dataset could be found, it could help validate the arrivals calculated from the GTFS-realtime VehiclePositions that were used in this study.

Areas of Improvement for Current Models

It is possible that the time series' performance is decreased due to it being a non-spatially explicit model. This would mean that the impact of events upstream or downstream are not directly input into the model. The model would have to infer this based only on events within the same segment, which may be impossible. It is possible that this is another possible route to improvement of the Time Series GRU model. Possible ways of increasing the spatial explicitness would be to take times from additional routes as input, or perhaps to take observations from adjacent segments as part of the input data.

The most straightforward explanation of why the seq2seq GRU model does not perform relatively better is that the seq2seq GRU takes less historical information into account than the competitors. In order to fix this, the Seq2Seq GRU could take a historical average as part of its input data. This would allow it to draw on a more representative sample of the past than one or two trips.
It seems clear that a way to improve the seq2seq GRU model would be to expand the amount of historical data that model takes directly as input. For example, instead of only taking one previous week trip as input, it could reach back multiple weeks. Instead of only taking one previous day trip as input, it could look at multiple trips from earlier in the day. Another approach to increase the amount of historical information would be to use historical average as some sort of baseline. For example, deviation from historical average segment time could be used as the dependent variable rather than absolute segment time. Or, historical average times could be input instead of individual trips to give context.

Intuitively, it seems that more data should be able to be input more directly for the predictions. Each trip directly uses only two previous trips of data, but there are thousands of trips in the dataset that could be used. Now, it is true that the machine learning model has some implicit learning potential such that it does learn from the entire dataset. But it is more likely to learn from the data that is directly given to it in the model. One approach would be to use an attention-based mechanism to identify trips that can be used as examples. (Jeong et al., 2024; Ma et al., 2022; Olah, 2015).

Another way to improve the model would be to use more types of input data. As with statistical models, machine learning models perform better when relevant variables are

included in the dataset. In this case, two types of relevant data to include would be traffic data, bus occupancy data, and road network data. Traffic data would help to explain slowdowns in operations that occur due to congestion on the road network. Bus occupancy data could help explain slowdowns related to high passenger loads. The road network would allow the detection of intersections and other road features that may impact bus travel times.

7. CONCLUSION

7.1 Spatial Explicitness Considerations

In Chapter 2, it was reported that geoscientists believe that spatially explicit models tend to both represent theory better and to make better predictions. This was because transportation networks naturally have a spatial structure, and these networks describe how random events like nonrecurring congestion spreads along the network (Julio et al., 2016; Petersen et al., 2019). In terms of bus movements, models that can leverage knowledge of the bus route to identify when a bus is going to reach congestion or other disruptions would seem to have an advantage over models that cannot (Bhutani et al., 2024; Cats & Loutos, 2016).

In this experiment, the spatially explicit model (Seq2Seq GRU) was designed to take these factors into account. The way that successive bus route segments fed data to one another suggested that they could note spatial relationships. The bi-directional layer was specifically designed to be able to take into account downstream propagating and upstream propagating congestion, which is a sophisticated use of traffic theory to inform the AI model (Bhutani et al., 2022). This is the type of physically-grounded model that has been proposed as a promising future direction for the BAT prediction literature (Kumar et al., 2025). However, it performed worse than its non-spatially explicit competitors. None of the other three models had any mechanism to allow information to flow between adjacent route segments.

One possibility is that the model makes certain assumptions about the nature of the problem that were not held in this instance. In the initial paper proposing the model, the segments to be predicted were interpreted as a standardized length. In this case, there is substantial variance in the travel times of different segments.

Judging from theory, it seems unlikely that the spatial explicitness per se reduced the performance of the model. There are perhaps two broad explanations. One is simply that, other choices were made in the construction of the model, and these were to blame for the performance deficit of the spatially explicit model. For example, the reduced historical information taken by this model compared to the others is likely to be a weakness. For a more detailed understanding of how spatial explicitness affects BAT prediction models using GRU, more research is needed.

7.2 Final Notes

At this point, AI researchers are preparing for AI agents to beat "Humanity's Last Exam," a 3000-item multiple choice test with graduate-level questions that is likely outside the abilities of any one human to answer (New York Times, 2025). If we understand the narrative around this test, it is that the passing of this test, expected later this year, will mark the sunset of human expertise and its replacement by AI. However, from undergoing this experiment, I am convinced of the enduring importance of human creativity for problem solving.

First of all, it was clear that, for the BAT prediction problem alone, there was a tremendous amount of variation in the methods employed. It was human creativity that brought those models about. The results of the experiment showed how decisions must be made when building AI models, and these decisions have consequences. The most important figure in this paper is probably Figure 4, which shows the information flow in the three models. This summarizes the key differences in how information is used in the models. From looking at this figure alone, you can already understand why it is that the Seq2Seq GRU struggled to detect historical averages, for example. It also shows clearly what could be built from here: for example, a model that combines the spatial integration with the temporal integration of the two GRU models deployed here.

The differences in the models really come down to how predictions ought to be made. In other words, they are differences based in theoretical understandings of how the world works. For the time being, humans are still better than LLMs at consistent logical and deductive reasoning. Our theories of the world will still be used to build things, whether they are built with the help of AI or not.

In past industrial revolutions – when general purpose technologies spread throughout society – it took decades for their effects to be fully realized (Crafts, 2021). This time will be available for scientists to continue to look for ways to understand AI. As we begin to understand a technology in more depth, we gain a greater ability to understand it mechanistically and reductively. The same technology then begins to seem less intelligent to us, and less forbidding. This might be a worthy goal for anyone who believes in the enduring power of human intelligence to pursue.

Reference List

Abduljabbar, R., Dia, H., Liyanage, S., Bagloee, S. A. (2019). Applications of artificial intelligence in transport: An overview. *Sustainability, 11,* 189. http://dx.doi.org/10.3390/su11010189

Adorno, J. (2023). *Refining the Machine Learning Pipeline For US-based Public Transit Systems*. [Doctoral dissertation, University of South Florida]. USF Digital Commons. https://digitalcommons.usf.edu/etd/10015/

Aemmer, Z., Ranjbari, A., et al. (2022). Measurement and classification of transit delays using GTFS-RT data.

Aemmer, Z., Sorbo, S., Clemente, A., & Ruocco, M. (2024). Generalization strategies for improving bus travel time prediction across networks. *Journal of Urban Management, 13,* 372-385. https://doi.org/10.1016/j.jum.2024.05.002

Achar, A., Bharathi, D., Kumar, B. A., & Vanajakshi, L. (2020). Bus arrival time prediction: A spatial Kalman filter approach. *IEEE Transactions On Intelligent Transport Systems, 21*(3), 1298-1307. https://doi.org/10.1109/TITS.2019.2909314

Alexandre, T., Bernardini, F., Viterbo, J., & Pantoja, C. E. (2022). Machine learning applied to public transportation by bus: A systematic literature review. *Transportation Research Record, 2677*(7), 639-660. https://doi.org/10.1177/03611981231155189

Altinkaya, M., & Zontul, M. (2013). Urban bus arrival time predictions: A review of computational models. *International Journal of Recent Technology and Engineering, 2*(4), 164-169.

Amaris, M., Morais, M. A., & de Camargo, R. Y. (2021, September 19-22). *Efficient prediction of region-wide traffic states in public bus networks using LSTMs*. 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), Indianapolis, IN, USA, 2021. https://doi.org/*10.1109/ITSC48978.2021.9564881*.

Anselin, L. (1989). What is special about spatial data? Alternative perspectives on spatial data analysis.

Bharathi, D., Vanajakshi, L., & Subramanian, S. C. (2022). Spatio-temporal modelling and prediction of bus travel time using a higher-order traffic flow model. *Physica A,* 596, 127086. https://doi.org/10.1016/j.physa.2022.127086

Bhutani, N., Pachal, S., & Achar, A. (2024). Encoder-decoder RNNs for bus arrival time prediction. *Knowledge Management and Acquisition for Intelligent Systems: 20th Principle and Practice of Data and Knowledge Acquisition Workshop, PKAW 2024, Kyoto, Japan, November 18-19, 2024, Proceedings,* 266-275. https://doi.org/10.1007/978-981-96-0026-7_22

Brakewood, C., Macfarlane, G. S., & Watkins, K. (2015). The impact of real-time information on bus ridership in New York City. *Transportation Research Part C, 53,* 59-75. http://dx.doi.org/10.1016/j.trc.2015.01.021

Brakewood, C., & Watkins, K. (2019). A literature review of the passenger benefits of real-time transit information. *Transport Reviews, 39*(3), 327-356. https://doi.org/10.1080/01441647.2018.1472147

Cats, O., & Loutos, G. (2015). Real-time bus arrival information system: An empirical evaluation. *Journal of Intelligent Transportation Systems, 20*(2), 138-151. https://doi.org/10.1080/15472450.2015.1011638

Cathey, F. W., & Dailey, D. J. (2003). *Transportation Research Part C, 11,* 241-264. A prescription for transit arrival/departure prediction using automatic vehicle location data. doi:10.1016/S0968-090X(03)00023-8

Cats, O., & Loutos, G. (2016). Evaluating the added-value of online bus arrival prediction schemes. *Transportation Research Part A, 86*, 35-55. http://dx.doi.org/10.1016/j.tra.2016.02.004

Čelan, M., & Lep, M. (2017). Bus arrival time prediction based on network model. *Procedia Computer Science, 113*, 138-145. https://doi.org/10.1016/j.procs.2017.08.331

Chen, M., Liu, X., & Xia, J. (2004). A dynamic bus-arrival time prediction model based on APC data. *Computer-Aided Civil and Infrastructure Engineering, 19*(5), 364-376. https://doi.org/10.1111/j.1467-8667.2004.00363.x

Chinmayee, P., Patil, A. H., Bhavika, V. K., Ashwini, B. P., & Sumathi, R. (2022, December 16-17). *A comprehensive survey on parametric and non-parametric machine learning approaches for bus arrival time prediction*. 2022 International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), Greater Noida, India, 2022. https://doi.org/10.1109/ICAC3N56670.2022.10074390

Chondrodima, E., Georgiou, H., Pelekis, N. & Theodoridis, Y. (2022). Public transport arrival time prediction based on GTFS data. In: Nicosia, G., et al. *Machine Learning, Optimization, and Data Science. LOD 2021*. Lecture Notes in Computer Science, vol. 13164. Springer, Cham. https://doi.org/10.1007/978-3-030-95470-3_36

Chow, W., Block-Schachter, D., & Hickey, S. (2014). Impacts of real-time passenger information signs in rail-stations at the Massachusetts Bay transportation authority. In: Proceedings of the 93rd Transportation Research Board Annual Meeting, Washington DC.

Crafts, N. (2021). Artificial intelligence as a general-purpose technology: An historical perspective. *Oxford Review of Economic Policy, 37*(3), 521-536. https://doi.org/10.1093/oxrep/grab012

Crudden, S. Ó., & Berrebi, S. (2023). An open-source framework to implement kalman filter bus arrival predictions. *Networks and Spatial Economics, 23,* 429-443. https://doi.org/10.1007/s11067-021-09541-w

De Sabbata, S., & Liu, P. (2023). A graph neural network framework for spatial geodemographic classification. *International Journal of Geographical Information Science, 37*(12), 2464-2486. https://doi.org/10.1080/13658816.2023.2254382

Deng, Y., & Chen, M. (2020). Impacts of real-time transit information on bus passengers' travel choices based on travel behavior survey. *Promet – Traffic & Transportation, 33*(4), https://doi.org/10.7307/ptt.v33i4.3637

Do, L. N. N., Taherifar, N., & Vu, H. L. (2019). Survey of neural network-based models for short-term traffic state prediction. *WIREs Data Mining and Knowledge Discovery, 9,* e1285. https://doi.org/10.1002/widm.1285

Elliott, T., & Lumley, T. (2020). Modelling the travel time of transit vehicles in real-time through a GTFS-based road network using GPS vehicle locations. *Australian & New Zealand Journal of Statistics, 62*(2), 153-167. https://doi.org/10.1111/anzs.12294

Gao, S. Geospatial Aritifical

Gebru, T., Krause, J., Wang, Y., Chen, D., Deng, J., Aiden, E. L., & Li, F.-F. (2017). Using deep learning and google street view to estimate the demographic makeup of neighborhoods across the United States. *PNAS, 114*(50), 13108-13113. https://doi.org/10.1073/pnas.1700035114

Gillain, E. (Ed). (2024). Demystifying artificial intelligence. De Gruyter. ISBN 978-3-11-142567-2

Gillam, W. J., & Wright, D. A. (2000). An innovative approach to real-time bus information and signal priority. *Tenth International Conference on Road Transport Information and Control, 2000. (Conf. Publ. No. 472),* London, UK, 2000. 205-208. https://doi.org/10.1049/cp:20000133.

Goldfarb, A. (2024). Pause artificial intelligence research? Understanding AI policy challenges. *Canadian Journal of Economics*, *57*(2), 363-377. **https://doi.org/10.1111/caje.12705**

Goodchild, M. F. (2004). The validity and usefulness of laws in geographic information science and geography. *Annals of the Association of American Geographers, 94*(2), 300-303. https://doi.org/10.1111/j.1467-8306.2004.09402008.x

Gooze, A., Watkins, K. E., & Borning, A. (2013). Benefits of real-time transit information and impacts of data accuracy on rider experience. *Transportation Research Record: Journal of the Transportation Research Board, 2351,* 95-103. https://doi.org/10.3141/2351-11

Horbury, A. X. (1999). Guidelines for specifying automatic vehicle location and real-time passenger information systems using current best practice. *Transport Reviews, 19*(4), 331-351. https://doi.org/10.1080/014416499295439

Hu, Y., Mai, G., Cundy, C., Choi, K., Lao, N., Liu, W., Lakhanpal, G., Zhou, R. Z., & Joseph, K. (2023). Geo-knowledge-guided GPT models improve the extraction of location descriptions from disaster-related social media messages. *International Journal of Geographical Information Science, 37*(11), https://doi.org/10.1080/13658816.2023.2266495.

Huang, Y. P., Chen, C., Su, Z. C., Chen, T. S., Sumalee, A., Pan, T. L., & Zhong, R. X. (2021). Bus arrival time prediction and reliability analysis: An experimental comparison of functional data analysis and Bayesian support vector regression. *Applied Soft Computing, 111,* 107663. https://doi.org/10.1016/j.asoc.2021.107663

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521,* 434-444. https://doi.org/10.1038/nature14539

Lin, W.-H., Zeng, J. (1999). An experimental study on real time bus arrival time prediction with GPS data. *Transportation Research Record: Journal of the Transportation Research Board, 1666*(1)*,* 101-109. https://doi.org/10.3141/1666-12

Janowicz, K., Gao, S., McKenzie, G., Hu, Y., & Bhaduri, B. (2020). GeoAI: Spatially explicit artificial intelligence techniques for geographic knowledge discovery and beyond. *International Journal of Geographical Information Science, 34*(4), 625-636. https://doi.org/10.1080/13658816.2019.1684500

Jeong, S., Oh, C., & Jeong, J. (2024). BAT-transformer: Prediction of bus arrival time with transformer encoder for smart public transportation system. *Applied Sciences, 14,* 9488. https://doi.org/10.3390/app14209488

Jiang, W., & Luo, J. (2022). Graph neural network for traffic forecasting: A survey. *Expert Systems With Applications, 207,* 117921. https://doi.org/10.1016/j.eswa.2022.117921

Julio, N., Giesen, R., & Lizana, P. (2016). Real-time prediction of bus travel speeds using traffic shockwaves and machine learning algorithms. *Research in Transportation Economics, 59*, 250-257. http://dx.doi.org/10.1016/j.retrec.2016.07.019 0739-8859

Koushik, A. N. P., Manoj, M., Nezamuddin, N. (2020). Machine learning applications in activity-travel behavior research: A review. *Transport Reviews, 40*(3), 288-311. https://doi.org/10.1080/01441647.2019.1704307

Kumar, B. A., Singh, R., Shaji, H. E., & Vanajakshi, L. (2025). Bus arrival time prediction: A comprehensive review. *IEEE Transactions On Intelligent Transportation Systems, 26*(6), 7362-7379. https://doi.org/10.1109/TITS.2025.3545695

Li, M., Lu, F., Zhang, H., Chen, J. (2020). Predicting future locations of moving objects with deep fuzzy-LSTM networks. *Transportmetrica A Transport Science, 16*(1), 119-136. https://doi.org/10.1080/23249935.2018.1552334

Liu, P., Biljecki, F. (2022). A review of spatially-explicit GeoAI applications in urban geography. *International Journal of Applied Earth Observation and Geoinformation, 112,* 102936. https://doi.org/10.1016/j.jag.2022.102936

Liu, P., Zhang, Y., & Biljecki, F. (2023). Explainable spatially explicit geospatial artificial intelligence in urban analytics. *Environment and Planning B: Urban Analytics and City Science, 51*(5), 1-20. https://doi.org/10.1177/23998083231204689

Lock, O., Bednarz, T., & Pettit, C. (2021). The visual analytics of big, open public transport data – a framework and pipeline for monitoring system performance in Greater Sydney. *Big Earth Data, 5*(1), 134-159. https://doi.org/10.1080/20964471.2020.1758537

Ma, J., Chan, J., Rajasegarar, S., & Leckie, C. (2022). Multi-attention graph neural networks for city-wide bus travel time estimation using limited data. *Expert Systems With Applications, 202*, 117057. https://doi.org/10.1016/j.eswa.2022.117057

Ma, J., Chan, J., Rostanoski, G., Rajasegarar, S., & Leckie, C. (2019). Bus travel time prediction with real-time traffic information. *Transportation Research Part C, 105,* 536-549. https://doi.org/10.1016/j.trc.2019.06.008

Mishalani, R. G., Lee, S., & McCord, M. R. (2000). Evaluating real-time bus arrival information systems. *Transportation Research Record, 1731*(1), 81-87. https://doi.org/10.3141/1731-10

Mishalani, R. G., McCord, M. M., & Wirtz, J. (2006). Passenger wait time perceptions at bus stops: Empirical results and impact on evaluating real-time bus arrival information. *Journal of Public Transportation, 9*(2), 89-106. https://doi.org/10.5038/2375-0901.9.2.5

Newmark, G. L. (2024). *Assessing GTFS Accuracy.* Mineta Transportation Institute. https://transweb.sjsu.edu/research/2017-Public-Transit-Statistical-Analysis

Nigam, A. (2025). Spatiotemporal bus arrival prediction using ConvLSTM and CTGANs-augmented data. *International Journal of Intelligent Transportation Systems Research, 23,* 372-384. https://doi.org/10.1007/s13177-024-00454-9

Olah, C. Understanding LSTM Networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Papers With Code. (nd.) *Benchmarking*. https://paperswithcode.com/task/benchmarking

de M. Pereira, H., Bessa Junior, J. E., & de A. Nobrega, R. A. (2024). Geospatial-based decision support system for prioritizing road segments for maintenance and

rehabilitation. *Case Studies in Transportation Policy, 16*, https://doi.org/10.1016/j.cstp.2024.101170

Petersen, N. C., Rodrigues, F., & Pereira, F. C. (2019). Multi-output bus travel time prediction with convolutional LSTM neural network. *Expert Systems With Applications, 120,* 426-435. https://doi.org/10.1016/j.eswa.2018.11.028

Pourebrahim, N., Sultana, S., Thill, J.-C., Mohanty, S. (2018). Enhancing trip distribution prediction with twitter data: Comparison of neural network and gravity models. *GeoAI '18: Proceedings of the 2nd ACM SIGSPATIAL International Workshop on AI for Geographic Knowledge Discovery,* 5-8, https://doi.org/10.1145/3281548.3281555.

Qiu, T., Lam, C.-T., Liu, B., Ng, B. K., Yuan, X., & Im, S. K. (2025). FEN-MRMGCN: A frontend-enhanced network based on multi-relational modeling GCN for bus arrival time prediction. *IEEE Access, 13*, 5296 – 5307. https://doi.org/10.1109/ACCESS.2024.3525357

Roose, K. (Jan. 23, 2025). When A.I. Passes This Test, Watch Out. *The New York Times*. *https://www.nytimes.com/2025/01/23/technology/ai-test-humanitys-last-exam.html*

Shanthi, N., Sathishkumar, V. E., Babu, K. U., Karthikeyan, P., Rajendran, S., & Allayear, S. M. (2022). Analysis on the bus arrival time prediction model for human-centric services using data mining techniques. *Computational Intelligence and Neuroscience, 2022*, 7094654. https://doi.org/10.1155/2022/7094654

Shelton, T. (2017). The urban geographic imagination in the age of big data. *Big Data & Society,* 1-14. https://doi.org/10.1177/2053951716665129

Shen, J., Liu, Q., Zhang, Y., & Yu, M. (2025). A novel model incorporating deep learning and Kalman filter augmentation for route-level bus arrival time prediction with error accumulation mitigation. *Expert Systems With Applications, 281,* 127622. https://doi.org/10.1016/j.eswa.2025.127622

Singh, N., & Kumar, K. (2022). A review of bus arrival time prediction using artificial intelligence. *WIREs Data Mining and Knowledge Discovery, 12*(4), e1457. https://doi.org/10.1002/widm.1457

Stackpole, B. (2024). The impact of generative AI as a general-purpose technology.

Tang, L., Ross, H., & Han, X. (2012). Substitution or complementarity: Examination of ridership effects of real-time bus information on transit rail in Chicago, Illinois. *Transportation Research Record, 2276,* 156-163. https://doi.org/10.3141/2276-19

Tedjopurnomo, D. A., Bao, Z., Zheng, B., Choudhury, F. M., & Qin, A. K. (2022). A survey on modern deep neural network for traffic prediction: Trends, methods, and challenges. *IEEE Transactions On Knowledge and Data Engineering, 34*(4), 1544-1561. https://doi.org/10.1109/TKDE.2020.3001195

Xian, T., Chin, T. K., Marks, B., Nelson, J. D., & Moylan, E. (2024). Bus arrival and departure time updates in the Greater Sydney Area. *Nature Scientific Data, 11,* 1034. https://doi.org/10.1038/s41597-024-03873-1

Xie, Z.-Y., He, Y.-R., Chen, C.-C., Li, Q.-Q., & Wu, C.-C. (2021). Multistep prediction of bus arrival time with the recurrent neural network. *Mathematical Problems in Engineering, 2021,* 6636367. https://doi.org/10.1155/2021/6636367

Wang, S., Huang, X., Liu, P., Zhang, M., Biljecki, F., Hu, T., Fu, X., Liu, L., Liu, X., Wang, R., Huang, Y., Yan, J., Jiang, J., Chukwu, M., Naghedi, S. R., Hemmati, M., Shao, Y., Jia, N., Xiao, Z., . . . Bao, S. (2024). Mapping the landscape and roadmap of geospatial artificial intelligence (GeoAI) in quantitative human geography: An extensive systematic review. *International Journal of Applied Earth Observation and Geoinformation, 128*, 103734. https://doi.org/10.1016/j.jag.2024.103734

Watkins, K. E., Ferris, B., Borning, A., Rutherford, G. S., & Layton, D. (2011). Impact of mobile real-time information on the perceived and actual wait time of transit riders. *Transportation Research Part A, 45,* 839-848. http://dx.doi.org/10.1016/j.tra.2011.06.010

Wegman, E. J. (1988). Computational statistics: A new agenda for statistical theory and practice. *Journal of the Washington Academy of Sciences, 78*(4), 310-322. https://www.jstor.org/stable/24536995

Yamaguchi, T., Mansur, A. S., & Mine, T. (2018, December 17-20). Prediction of bus delay over intervals on various kinds of routes using bus probe data. 2018 IEEE/ACM 5th International Conference on Big Data Computing Applications and Technologies (BDCAT), Zurich, Switzerland. https://doi.org/10.1109/BDCAT.2018.00020

Yu, B., Lam, W. H. K., & Tam, M. L. (2011). Bus arrival time prediction at bus stop with multiple routes. *Transportation Research Part C, 19,* 1157-1170. https://doi.org/10.1016/j.trc.2011.01.003

Zervaas, Q. (2015). *The Definitive Guide to GTFS-realtime: How to Consume and Produce Real-Time Public Transportation Data With the GTFS-RT Specification* (1st ed.). Creative Commons Attribution 4.0 International License. *https://github.com/MobilityData/GTFS-books*

Zhang, F., Wu, L., Zhu, D., & Liu, Y. (2019). Social sensing from street-level imagery: A case study in learning spatio-temporal urban mobility patterns. *ISPRS Journal of Photogrammetry and Remote Sensing, 153,* 48-58. https://doi.org/10.1016/j.isprsjprs.2019.04.017

Zhang, X., Lauber, L., Liu, H., Shi, J., Xie, M., & Pan, Y. (2022). Travel time prediction of urban public transportation based on detection of single routes. *PLoS ONE, 17*(1), *e0262535. https://doi.org/10.1371/journal.pone.0262535*

Zhang, Y., & Cheng, T. (2020). Graph deep learning model for network-based predictive hotspot mapping of sparse spatio-temporal events. *Computers, Environment and Urban Systems, 79,* 101403. https://doi.org/10.1016/j.compenvurbsys.2019.101403

Zhou, J. Q., Jackson, J., Stravitz, P., Barlow, G. J., & Koonce, P. (2024). Addressing data latency in GTFS (General Transit Feed Specification) Realtime to improve transit signal priority.

Zhou, L., Song, Y., Zhang, C., Liu, Y., Wang, P., & Lin, T. (2019). T-GCN: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems, 21*(9), 3848-3858. https://doi.org/10.1109/TITS.2019.2935152

Zhu, D., Cheng, X., Zhang, F., Yao, X., Gao, Y., & Liu, Y. (2018). Spatial interpolation using conditional generative adversarial neural networks. *International Journal of Geographical Information Science, 34*(4), 735-758.