Spring 2019

A Data-Driven Approach for Modeling Agents

Hamdi Kavak
Old Dominion University

A DATA-DRIVEN APPROACH FOR MODELING AGENTS

by

Hamdi Kavak
B.S. July 2008, Karadeniz Technical University
M.E. August 2015, Old Dominion University

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

MODELING AND SIMULATION

OLD DOMINION UNIVERSITY
May 2019

Approved by:

Jose Padilla (Director)

Saikou Diallo (Member)

George Hsieh (Member)

John Sokolowski (Member)

Masha Sosonkina (Member)

ABSTRACT

A DATA-DRIVEN APPROACH FOR MODELING AGENTS

Hamdi Kavak
Old Dominion University, 2019
Director: Dr. Jose J. Padilla

Agents are commonly created on a set of simple rules driven by theories, hypotheses, and assumptions. Such modeling premise has limited use of real-world data and is challenged when modeling real-world systems due to the lack of empirical grounding. Simultaneously, the last decade has witnessed the production and availability of large-scale data from various sensors that carry behavioral signals. These data sources have the potential to change the way we create agent-based models; from simple rules to driven by data. Despite this opportunity, the literature has neglected to offer a modeling approach to generate granular agent behaviors from data, creating a gap in the literature.

This dissertation proposes a novel data-driven approach for modeling agents to bridge the research gap. The approach is composed of four detailed steps including data preparation, attribute model creation, behavior model creation, and integration. The connection between and within each step is established using data flow diagrams.

The practicality of the approach is demonstrated with a human mobility model that uses millions of location footprints collected from social media. In this model, the generation of movement behavior is tested with five machine learning/statistical modeling techniques covering a large number of model/data configurations. Results show that Random Forest-based learning is the most effective for the mobility use case. Furthermore, agent attribute values are obtained/generated with machine learning and translational assignment techniques.

The proposed approach is evaluated in two ways. First, the use case model is compared to another model which is developed using a state-of-the-art data-driven approach. The model's prediction performance is comparable to the state-of-the-art model. The plausibility of behaviors and model structure in the use case model is found to be closer to real-world than the state-of-the-art model. This outcome indicates that the proposed approach produces realistic results. Second, a standard mobility dataset is used for driving the mobility model in place of social media data. Despite its small size, the data and model resembled the results gathered from the primary use case indicating the possibility of using different datasets with the proposed approach.

This dissertation is dedicated to my beloved wife, Ayşe,
for her untiring support and love.

ACKNOWLEDGMENTS

During this compelling doctoral journey, I have received great support and encouragement from many beautiful people. I extend special thanks to my doctoral advisor Dr. Jose Padilla for his guidance and support both for my academic development and career. As a mentor, he helped me make the finish line and become an independent researcher. Many thanks to Dr. Saikou Diallo for his inspiring discussions and support throughout my doctoral research. Special thanks to other committee members Dr. George Hsieh, Dr. John Sokolowski, and Dr. Masha Sosonkina for their help and critical comments.

It is true that doctoral research is a challenging task, but it comes with the benefit of meeting with supportive peers. I would like to especially thank my friends Daniele Vernon-Bido and Chris Lynch for their motivating discussions about work and life in general. It will always be a privilege to have friends like you for the rest of my life. Many thanks to the lovely faculty and staff at the Virginia Modeling Analysis and Simulation Center (VMASC). Specifically, thanks to Dr. Andrew Collins for his encouraging conversations about research in agent-based modeling; Terra Elzie and Hector Garcia for great feedback in our weekly meetings; and David Ralph for technical support in all cluster and data-related needs. I want to thank many colleagues at Old Dominion University for their support in labeling the ground truth dataset of home locations and proofreading this manuscript.

Finally, I sincerely thank my family. Especially, my beloved wife Ayşe who supported me in this process while she was experiencing the challenges more than anybody else. I will always be thankful for her unrequited love and support. With a partner like you, I feel so fortunate. Our children Eda and Arda have been around whenever I need a mental break. Thank you “kerata!”

This material, in part, is based on research sponsored by the Office of the Assistant Secretary of Defense for Research and Engineering (OASD(R&E)) under agreement number FAB750-15-2-0120. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Office of the Assistant Secretary of Defense for Research and Engineering (OASD(R&E)) or the U.S. Government.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Agent-based models (ABMs) have been increasingly used in the past three decades to study complex systems based on their constituent units that interact in an environment [1]. This bottom-up representation promises many advantages such as having a one-to-one ontological correspondence between a simulation model and the real world [2] and the ability to capture parallel processes, non-linear interactions, and heterogeneity of systems [3]. Despite such fruitful features that agent-based modeling promises to a wide audience, agent-based modeling approaches have been limited to exploit such features extensively.[1]

Many modeling approaches for agents embrace the use of theoretical knowledge and idealized assumptions to identify agents' behaviors and attributes [4-7]. In this respect, agent behaviors are often composed of a set of rules that are triggered based on idealized and sometimes arbitrary probability distributions. Similarly, attribute values are initialized arbitrarily using quantities like percentages or probability distributions [8]. Tipping point models such as Schelling's segregation model [9] and Epstein's civil violence model [10] are very well-known examples in this respect. In the segregation model, agents are uniformly distributed and make relocation decisions based on the color of their neighboring agents. In the civil violence model, agents make rioting decisions based on their perceived *hardship* and *legitimacy* values which both are drawn from uniform distributions. In addition to the arbitrary randomness in attribute values, all agents of the same type have the same decision-making mechanism and structure. These practices introduce challenges in establishing a one-to-one correspondence between the

[1] IEEE Transactions and Journals style is used in this thesis for formatting figures, tables, and references.

model and the real-world, which is expected when modeling real-world systems. Such limitations stem from the lack of empirical grounding in the model hindering correspondence between model and reality while limiting predictive capabilities.

Agent-based modeling researchers and practitioners have made efforts to address the "closeness to reality" of ABMs through operational validation [11] and calibration [12]. Operational validation compares simulation outputs with respective data seen in the real world. Calibration process involves iteratively adjusting initial values of a subset of model parameters to align simulation outputs with the empirical data. Calibration becomes challenging when models are based on theoretical frameworks such as BDI [13], Soar [14], and ACT-R [15] due to their complex modeling mechanisms [16]. The granularity of the data used in both validation and calibration processes are mostly at the population level. Such comparisons include checking particular probability distributions, ratios, percentages, and confidence boundaries. Population-level comparisons are the first steps toward improving the empirical grounding of ABMs, but more granular tests are needed. As Klügl [17] states regarding agent-based modeling: *"... not only input-output relations have to be compared for the overall system, but validation procedures have to be performed also for additional sub-ensembles of agents or partial models, down to single agents."*

Despite the need for such empirical grounding, the adoption of data-driven approaches in agent-based modeling has been quite slow and sparse. The lack of data-driven approaches has been criticized especially for models that simulate real-world systems [18]. The sparsity of data-driven modeling approaches has been ascribed to the absence of behavioral data [19] or the lack of quality data at a suitable abstraction level [3, 17]. It is argued here that granular behavioral data has become available even at the individual agent level for particular application domains

such as human mobility and cybersecurity. However, current agent-based modeling approaches are not designed to use such data as shown in the research gap below.

1.1 Research Gap

To date, a limited number of studies attempted to develop modeling approaches to incorporate data directly in agents' design and simulation [6]. These studies can be grouped according to the effect that data makes on the model and the granularity of their agent consideration. In terms of their effect on the model, three cases can be named: **changing initial values**, **modifying model structures**, and **generating model structures**. Changing initial values (or initialization) makes the least empirical grounding among the three. It is applied in such a way that the starting conditions of agents are assigned according to patterns seen in data. Initialization by itself is only a starting point and does not imply that agent behaviors are created according to data. Modification of model structures has more effect and suggests that components in the model are adjusted as per data. For instance, a function describing a utility can be adjusted according to data. Generating new model structures has the highest effect because it creates a new structure entirely based on data which does not previously exist in any form. Data-driven agent behaviors are considered to be in this group generating model structures using data. When it comes to agent granularity, data can be used for agents in two levels: **individual level** and **population level**. Individual-level use considers each agent uniquely with individually identifiable data while population level considers the agent population as a whole and use aggregated quantities (e.g., probability distributions).

When data-driven approaches from the literature are reviewed according to their effect on the model, many of them attempt to improve the empirical grounding by only **changing initial**

values for the model. Sajjad et al. [20]'s approach initializes their family formation model using census data. Ge et al. [21]'s approach generates a synthetic city model that initializes mobility attributes according to empirical mobility studies. Venkatramanan et al. [22]'s approach focuses on calibrating their model for disease spread prediction. Fewer other approaches involve both **initialization** and **structure modification**. Hassan et al. [23]'s approach emphasizes a data-driven design in their civilization model that uses census data for initializing population information and demographic evolution equations. Smajgl et al. [24] present an approach similar to Hassan et al. [23]'s involving the characterization of different data sources and their potential applications in different components of an ABM. Similarly, the use of data is limited as in Hassan et al. [23]'s case. The only data-driven approach identified that involves **structure generation** is from Bell and Mgbemena [25]. In their approach, they suggest using a **population level** CART decision tree to generate agent behaviors which are populated directly from data.

The approach proposed by Bell and Mgbemena [25] achieves the highest-level effect in terms of the use of data. However, their approach still captures the population level behaviors which experiences from the same challenges in terms of one-to-one correspondence and heterogeneity. In short, *the literature has neglected to offer an approach that generates individual level agent behavior from data.*

1.2 Thesis Statement

This dissertation proposes a novel data-driven modeling approach for agents that generates individual level agent behaviors using machine learning and statistical modeling techniques driven by individually identifiable data to address the research gap. It is expected that this type of approach can be very beneficial to reveal hidden behavioral patterns of human

actions, to cover a large percentage of a population, to be collected longitudinally, and is often low-cost [8]. The proposed approach will involve highly detailed steps that focus on the preparation of data, using machine learning and statistical modeling techniques to generate agent behaviors and obtain/generate agent attribute values at the individual level. The applicability of the proposed approach will be shown with a use case from human mobility domain. The evaluation of the approach will be made based on two points. First, the performance of the use case model will be compared against another human mobility model that is developed following a state-of-the-art data-driven approach. Second, the approach will be showcased by driving the use case model with a publicly available dataset which is different from the one used in the main use case model.

1.3 Contribution

The contribution of this dissertation to the Modeling and Simulation (M&S) community and the body of knowledge is *a novel data-driven approach for generating behaviors of individual agents using machine learning and statistical modeling techniques*. This contribution is materialized with a human mobility use case model that combines several data sources including location footprints from Twitter social media and population data from the U.S. Census and the U.S. Social Security Administration. In the process of developing the approach, a secondary contribution is identified – the use of machine learning techniques to elicit individual agent attribute values. The robustness of the proposed approach is illustrated in two cases. First, a state-of-the-art data-driven agent-based modeling approach by Bell and Mgbemena [25] is used to create a model and results are compared against the use case. Second, the use case is driven with a publicly available mobility dataset instead of footprints from social media. The results are

structurally very close to the model that is driven by social media footprints. The potential implications of the proposed approach can trigger fruitful discussions and inspire a new generation of agent development within and outside the M&S community.

1.4 Research Approach

The research approach mirrored in this dissertation is adapted from Diallo [26]'s *theory building* research approach. In Diallo [26], the creation of a theory and its testing are divided into multiple components. Each component feeds into the next component or feeds back into previous components iteratively. As a result, the theory and its test are conducted until sufficient results are gathered. Following that idea, this dissertation builds an approach (instead of theory) and is composed of four main (iteratively improved) components as illustrated in Fig. 1.

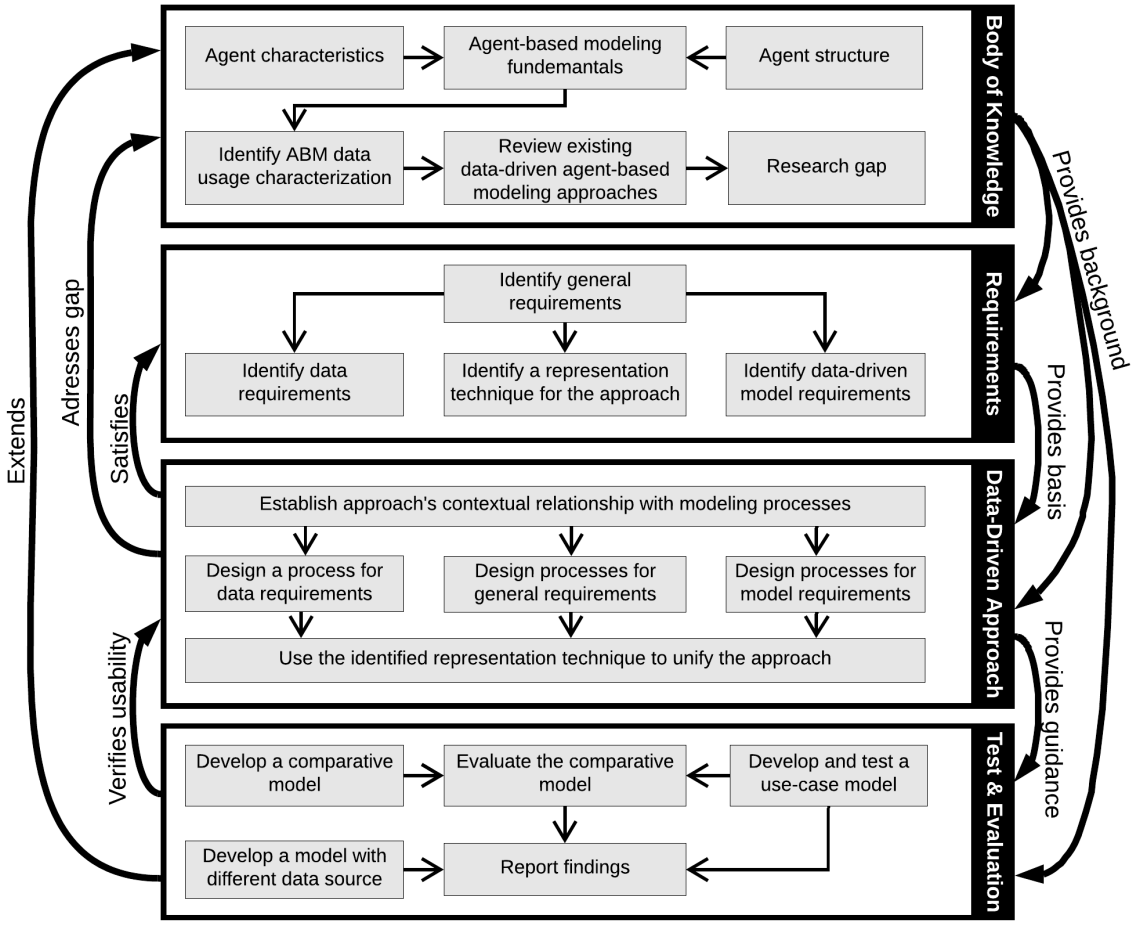


Fig. 1. Research approach used in this work. Adapted from [26].

Body of Knowledge: The first component of the research approach is to establish a body of knowledge. It starts with reviewing what agent-based modeling is by elaborating its common characteristics and structure as well as its historical roots. These constructs are then extended with understanding how agents are modeled in general. Putting it altogether establishes the fundamentals of agent-based modeling. This fundamental knowledge is then used to create a characterization of data usage in ABMs. The characterization has a taxonomical representation that shows different traditions of using data in ABMs. Following that, the data usage

characterization is utilized in reviewing existing data-driven agent-based modeling approaches from the literature. This review establishes the research gap which is the absence of agent-based modeling approaches that support an individual level generation of agent behaviors from data. The result of the body of knowledge provides a background on the next component which is about requirements.

Requirements: The second component aims at identifying a set of requirements for building an approach that addresses the needs of the research gap identified in the body of knowledge. As the first and foremost, there is a need to identify the general frame of the requirements such as identifying common processes in a modeling effort. Specific to the data-driven case, two requirements merit special recognition. *Data requirements* specify the needs of designing a data-intensive process. This requirement leads to a literature search in data handling. *Data-driven model requirements* indicate the needed modeling capabilities to address the research gap. These capabilities include capturing behavior from data and eliciting attribute values. Lastly, a requirement is established to identify a representation technique that unifies the approach.

Data-Driven Approach: The third component, the approach itself, uses the requirements as the basis for development. The first step is to establish how this approach is positioned within the modeling process. For instance, what is the relationship between the approach and the conceptual model? This knowledge is used in designing the processes of the approach. The first type is designing a process which satisfies data requirements including its preparation and tests. Similarly, several processes are designed to address data-driven model requirements such as developing a process that guides the modeler to choose appropriate learning technique or establishing procedures to elicit data for attribute values. Moreover, there is a need to adjust

existing or create a new process to satisfy general requirements. Lastly, the entire process requires to be unified and represented using the selected representation technique.

Test and Evaluation: The last component is about testing and evaluating the proposed approach. A use case model of human mobility is developed to test the usability of the approach. The use case model development and test include several processes such as conceptual model documentation, data identification, learning model candidate identification and the determination of best performing model through testing the performance of candidate models. The use case model uses location footprint data obtained from social media. To evaluate the success of the use case, a comparative model is developed based on a recent data-driven agent-based modeling approach identified in the body of knowledge. The comparison between the two models is made in terms of prediction performance and the plausibility of behaviors generated. Additionally, a completely different and publicly available data source is used to test the model to verify its usability. Overall, this component provides two insights: (1) whether it is possible to use the proposed approach in real-world problems and (2) how useful this approach is compared to other similar approaches.

As a result, the test and evaluation component verify the usability of the approach and extends the body of knowledge. Furthermore, the proposed data-driven approach addresses the research gap identified in the literature and satisfies all the requirements previously stated. While all these components and their relationships are described in a way that they seamlessly work together, practicing them in the real world reveals that there are always iterations that improve the overall production progressively.

1.5 Dissertation Outline

Chapter 1 introduces the main frame of the dissertation and articulates a research gap based on the challenges of agent-based modeling practices. It then states a thesis regarding how the research gap will be addressed and presents the research approach to address the research gap.

Chapter 2 provides more details on the background and establishes a body of knowledge regarding the fundamentals of agent-based models and existing data-driven approaches to modeling agents. This chapter elaborates on the establishment of the research gap.

Chapter 3 deals with enumerating preliminary requirements that need to be addressed in order to develop the approach. Some of these requirements lead to a secondary literature review which is also presented in this chapter. These reviews and addressing the requirement provide building blocks to satisfy the requirements.

Chapter 4 presents the proposed approach which draws from the requirements and building blocks enumerated in the previous chapter. The approach is presented in four main steps and numerous sub-steps that help to create agents that are driven by individual-level behavioral data.

Chapter 5 introduces a use case model of human mobility that follows the proposed approach. The use case model utilizes individual location footprints obtained from the Twitter social media platform. The individual level behavior generation includes the testing of several machine learning and statistical modeling techniques and selecting one with optimal performance. Furthermore, four attribute models are developed in this chapter.

Chapter 6 evaluates the approach in two cases. First, it compares the approach with another data-driven approach through comparing respective use cases that use identical data. Comparison points here are their respective prediction performance and the plausibility of the

generated behaviors. Second, a standard mobility dataset is tested for driving the mobility model in place of the Twitter data, and the results are compared in terms of prediction performance.

Chapter 7 discusses the proposed approach's merits and challenges in simulation replicability, generalization, and re-use. It further discusses the implications of the proposed approach on emerging domains such as urban science, cybersecurity, and homeland security as well as on disciplines such as artificial intelligence (AI) and data science.

Chapter 8 concludes the dissertation with a summary and a set of directions that can be undertaken in the future.

CHAPTER 2

LITERATURE REVIEW

The literature review chapter aims to give the background on ABMs and challenges with the use of data in current agent modeling practice. The starting point is the origins of agent-based modeling to give a sense of why ABMs today lack empirical grounding. Then, the chapter continues with describing common characteristics and structure of ABMs. The goal is to provide a context for the research gap and the proposed solution.

2.1 The Roots of Agent-Based Modeling

While the idea of representing and studying a system as a model in the form of interacting entities goes as early as the seventies [9], it took until the mid-nineties to receive noticeable attention from wide-range disciplines. Agent-based modeling has become a significant simulation paradigm to study complex human and natural systems [27] and preferred over other methods such as *system dynamics* and *microsimulation* as it provides a better mechanism to capture the phenomena being modeled.

System dynamics investigates a system at the macro level using system variables and differential/integral equations [28]. System variables are varied using differential/integral equations and are connected according to causality between them. If the causality and differential/integral equations are possible to identify, that system can be represented as a system dynamics model. However, it is not a minor task to satisfy these requirements [4]. Due to its ability to capture systems at the macro level, system dynamics is often considered to be the opposite of agent-based modeling [29]. Microsimulation, on the other hand, is quite similar to

both system dynamics and agent-based modeling [30]. It is similar to system dynamics in a sense that it uses equations to capture transitions between the states of variables with the exception that no causality is assumed. Also, it is similar to agent-based modeling in the sense that equations are used by individual entities and it is heavily used for policy analysis. Because there is no interaction between entities, microsimulation cannot capture interconnections and feedback loops. Despite this limitation, microsimulation can be considered a useful asset for agent-based modeling practices because it requires data to function [2]. While microsimulation has been around since the late fifties [31], agent-based modeling community neglected using data-driven principles of microsimulation until lately [30].

Different research areas and techniques have influenced agent-based modeling. The roots of agent-based modeling, as a technique, can be traced back to cellular automata in which cells are located on a finite two-dimensional grid. Each cell has states that change based on rules determined according to the states of their immediate neighbors [4]. This neighborhood-based rudimentary interaction gives rise to systemic emergent forms that attract considerable research since the seventies. Game theory-based models (e.g., [32]) bring the aspect of agent interaction that leads to emergent behaviors such as co-operation while distributed artificial intelligence (DAI) brought mechanisms of multiple agents working on achieving a common goal [33]. While the notion of investigating a system through modeling is limited in DAI studies, they provided how agents can be modeled using formal logic and developed robust communication protocols between agents. Interestingly, neither of these three areas that constitutes the roots of agent-based modeling is data intensive.

Despite the lack of data usage, today, ABMs have been developed in many different research fields including sociology [34], ecology [35], criminology [36], economy [37], and

urban studies [38], to name only a few. Considering its extensive roots, a wide range of application areas, and the lack of standards, it is challenging to provide a commonly accepted picture of agent-based modeling that unifies different viewpoints [39]. What follows next is an attempt to depict commonly referred characteristics and structure of agents broadly.

2.2 Common Characteristics and Structure of Agents

The question of 'what is an agent?' often discussed in terms of 'what are the essential and non-essential characteristics of agents?'[40]. The fact is that there is no consensus in the literature on this question except for the autonomy characteristic [41]. That is why the following list is consolidated using several sources [2, 41-43] which summarize a wide range of agent characteristics without imposing any essentiality on them. It can be argued that the essentiality of these characteristics should come from the system or phenomena that is being modeled, not the other way around. For instance, if a phenomenon to be modeled requires agents to adopt changes in the environmental conditions, then, adaptivity becomes an essential characteristic.

- *Autonomy* is the capability that an agent acts on its own without any external direction or central authority. This includes both autonomous decision-making and movement through the environment.
- *Heterogeneity* implies that agents differ in their types, properties or behaviors.
- *Memory* is the mechanism for agents to store and keep track of previous states of the environment and themselves.
- *Perception* is the capability to infer the state of the environment and other agents located in the environment. It can be limited to immediate neighbors, a radius of an area, or closeness in the agent's network.

- *Bounded Rationality* is a characteristic for agents to have limited decision-making capabilities. This idea contrasts with the idea of a perfectly rational view of economic decisions and can be imposed with limited perception or fixed-size memory.
- *Adaptivity* is the ability to adjust its preferences and behaviors according to the situation in the simulation. This adjustment can be made according to fixed rules or learning algorithms such as reinforcement learning.
- *Social ability* implies that an agent can interact with other agents and the environment. Interactions between agents can be established randomly, using space, or using a network.
- *Goal-seeking* is another ability of agents that make an agent monitor its current state and act towards accomplishing the goal. It is possible to have multiple competing goals that continuously present in the agent.

Unlike agent characteristics, common components of ABMs are relatively well established. A typical ABM is composed of two essential components [41]: *agent* and *environment*. Agents make up the main entities of the model. The environment is the space where agents are situated. While agent relations and interaction method are sometimes referred to as two distinct components [41], they are implemented within the agent thus do not warrant a separate component.

2.2.1 Agents

An agent is an entity that represents an individual unit (not necessarily human) of a system or phenomena to be modeled. According to Gilbert [2] *"Agents are either separate computer programs or, more commonly, distinct parts of a program that are used to represent social actors—individual people, organizations such as firms, or bodies such as nation-states."*

A list of agents of a particular type is called agent population. An ABM can have multiple populations each with a potentially diverse set of characteristics. The structure of a typical agent [44] is composed of five distinct components each with a specific role as depicted in Fig. 2. Note that not all these components are required to be used. Instead, the idea is to provide a general picture that one may come across. *Perception* component is tasked to sense the environment and other agents. The perception may be limited by proximity on the environment or network. *Attributes* component describes the properties of agents that have the potential to distinguish them from other agents. Some attributes are static as they do not change while some are dynamic and have the potential to change. *Memory* component is filled with experiences (episodes) gained by an agent. These experiences can be directly given or made by running the simulation. *Decision logic* is the component where an agent holds a set of behavior containing a list of actions and their execution conditions according to inputs gathered through perception, attributes, and memory. *Action* is the atomic component of behavior identified in the decision logic. When an action is executed, it can have an effect on the environment, other agents, or on the agent itself.

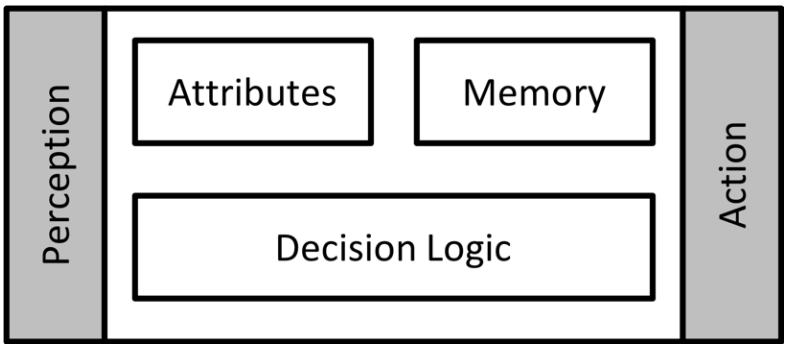


Fig. 2. Structure of an agent. Adapted from [44].

2.2.2 Environment

The environment is the world where agents execute their decision logic and act. Also, it is one of the critical components of an ABM. It facilitates the function of distributing resources/objects on the space, providing the means for agents to situate, move, and interact. Such resources, as in the classical Sugarscape model [45], may impact the location of agents and influence their properties. Similarly, agents can access and change the status of the resources in the environment.

Typically, ABMs contain a two-dimensional grid with continuous or discrete (cell-based) space representation. Notably, more recent urban models include a Geographic Information System (GIS) as the agent environment. Such models are also called "spatially explicit" [2]. A GIS-based environment provides the means for making model closer to the reality as it supports representing various geographical shapes and many urban areas in the world have their digital version of their building and road network openly available. While its effect is often not as discernable (except for pedestrian models), an environment could be constructed using a three-dimensional space representation. It is also common to see multiple layers of environment holding different type of resources making the model more modular [46].

2.3 Characterizing Data Usage in Agent-Based Models

Having described agent-based modeling with its roots and common agent characteristics and structure, this section focuses on the use of data in ABMs. This taxonomical characterization of data usage presented here is an extension of Kavak et al. [6]. The first division of this characterization is made based on its nature: whether *data-related* or *model-related*. These two are the main departing point of creating a data usage taxonomy for agents. In the data-related

concepts, the data type is the first branch representing whether data is qualitative or quantitative. The second one is based on the repetition of data collection that can be one-time or repeated. In the model-related concepts, the effect of data on the model is the first category with the cases of initial value assignment, structure modification, and structure generation. The second one is the model component which could be the environment or the agent as presented in the previous section. The difference here is that the agent component can be targeted at the population level or individually. The third and last one is the modeling and simulation phase (as a step in a simulation model development process) which can be listed as model development/design and simulation run. A high-level view of this characterization is provided in Fig. 3 followed by more detailed descriptions below.

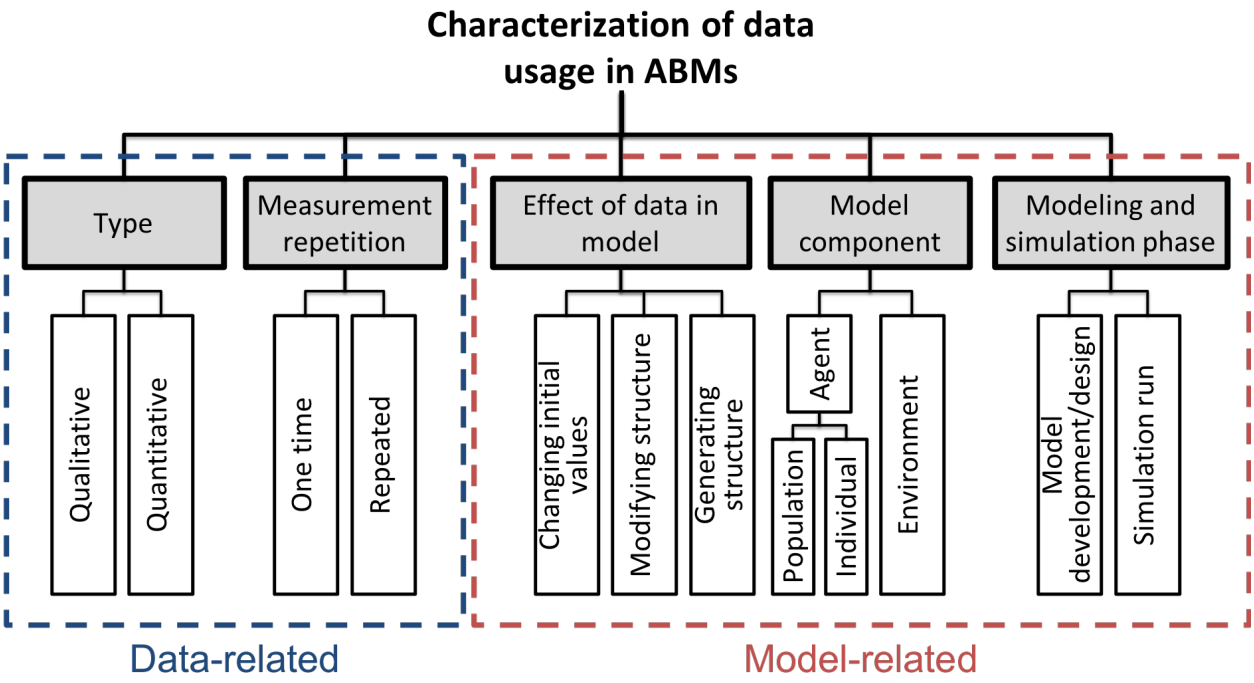


Fig. 3. Characterization of data usage in ABMs.

2.3.1 Data Type

Two main types of data are used in ABMs: *qualitative* and *quantitative*. Both data types are used in a variety of models including theoretical and empirical ones. The choice of the data type is often related to the case to be modeled and the availability and accessibility of data.

Qualitative data contains information that is represented using text or word and does not allow undistorted conversion of information [47]. For example, color or softness of an object is considered to be qualitative. Social science theories often provide qualitative descriptions of social phenomena. When they are used in models, such qualitative descriptions require a form of translation from textual information to categorical (nominal) or quantity information. Interviews [48], focus groups [24], and ethnographic studies [49] are among well-known methodologies to elicit qualitative data.

Quantitative data contains information that can be represented by numbers. For instance, the age of a person is quantitative. Quantitative data can further be classified as *discrete* (e.g., integers) or *continuous* (e.g., π=3.1415...). Official public datasets (e.g., American Community Survey from the US Census) contain a wealth of quantitative data along with some qualitative properties. Surveys, automated data collection mechanisms (e.g., traffic sensors), and web-based sources are among popular data sources that serve quantitative information.

ABMs use quantitative data in design and validation phases while often the use of qualitative data is overlooked [47]. Quantitative data can be used as a number, a set of numbers, percentages, or statistical distributions in the agent or environment. In certain cases, numbers are normalized to balance the variation between properties. Qualitative data is found in two forms [47]. The first form is the survey type of data that has an interpretable scale (e.g., strongly agree or agree) which can be converted into numbers or used as levels. The second and more

challenging form is textual data which often comes from ethnographic studies or interviews. The modeler can use approaches such as content analysis or grounded theory to distill relationships, behaviors, and attributes of an ABM from such qualitative data [50].

2.3.2 Measurement Repetition

Measurement repetition captures what timeframe data represents. This information presents an important role in determining the situations where data can be applied. Two types of repetition measurement can be seen in data: *one time* or *repeated*.

ABMs are suitable to be used for representing non-linear relationships between agents over time [2]. In this respect, data that represents a snapshot (i.e., one time) may have challenges to be used for such a purpose. Such snapshot data points can be suitably used to initialize agent properties. However, the use of the same data for agent behaviors is challenged because behaviors usually change over time or require longitudinal data to capture variation. Data that is collected repeatedly at different times solve such challenges.

When data is collected repeatedly, it can represent the dynamics of the subject at different timeframes. It then becomes possible to capture changes and variation over time. These changes may represent agent behaviors [8] or attributes [23] that vary temporally. The case of Hassan et al. [23], for instance, use three surveys each representing a decade. The first two are used to capture changes over two decades whereas the last one is used for validating the model. Panels are suitable methodologies to elicit such data. Here, repeated measurements could be recorded cross-sectionally or longitudinally. In the former case, different individuals are measured over time whereas the latter case involves the same individuals measured over time.

2.3.3 Effect of Data in Model

There are several cases of data usage in ABMs while only three types of effects are identified. The first case is changing the initial values of environment variables or agent attributes while the model stays the same. The second case is *modifying the structure*, as its name tells, changes the structure of a model component. For instance, a formula that sets yearly agent income can be modified based on empirical data fit a regression model. The third case is *generating a structure*, which creates a complete or partial structure of a model from scratch. The effect of data in the model increases from changing initial values to generating a structure as illustrated in Fig. 4.

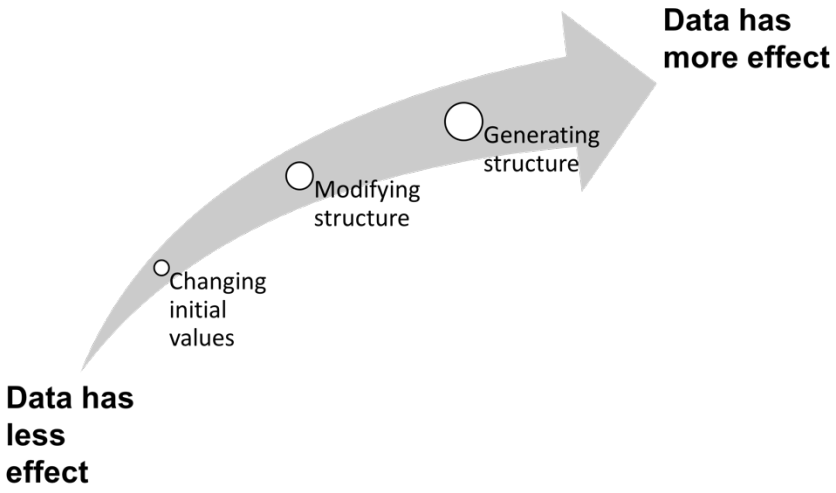


Fig. 4. The effect of data on the model.

The literature shows numerous efforts when it comes to *changing initial values* or also known as *initialization*. These efforts can be summarized concerning the assignment processes. In the simplest case, value initialization involves the direct assignment of a single attribute or a group of individual attributes. Sokolowski et al. [51], for instance, directly assigns health statistics from the U.S. Centers for Disease Control and Prevention's online repositories to their

obesity model during runtime. In a slightly advanced case, value assignment may involve the collection of data from a population and fit that data to a statistical model representing a randomly sampled quantity in the model. Ge et al. [21] use a statistical distribution to distribute transportation-related attributes in the agent population. In recent years, value initialization topic has produced more sophisticated approaches such as the use of data mining. Padilla et al. [52]'s work, for example, population-level eating preferences acquired from social media website feeds. The difference between the two previous examples is that the latter one uses data which is not gathered by official statistics. Instead, such data is mined from people's social media messages which introduces an alternative data source that can be used in agent-based models.

Similar to initialization, the literature shows examples regarding the *modifying structures* of models using data [23, 24]. This process happens when an initial model component is identified in the conceptual model, but later data fine-tunes that model structure to fit better to the reality. This is different from simulation calibration which updates the values of a set of inputs to optimize to fit a particular outcome. In this case, structure modification often happens during simulation development. Examples may include data that modifies behavioral rules or transitioning properties. For instance, Hassan et al. [23] show in their Spanish Civilization model that time-varying variables can be fine-tuned and their structure modified with available data. With this data-driven addition, Hassan et al. [23] show that their model generates output that is closer to reality than the randomly initialized model.

Data usage that causes *generating structures of the model* (a component of a model is generated directly from data) is quite different from the first two. In this case, the conceptual model provides very minimal information similar to expected model output. How that output is generated is the crucial point. There are a few examples in the data literature, especially from

human mobility studies, providing this capability in the model development process. For instance, Schneider et al. [53] generate previously unknown location movement patterns using call detail records data and identify the percentage distribution of prevalent patterns in the population. Then, their model uses these percentages to initialize the population. Such an example shows that prior theory, hypothesis, or assumptions regarding an agent behavior can be avoided if sufficient data is available.

2.3.4 Model Component

This item captures the model component that is affected by data. As discussed, essential model components are the *agent* and the *environment*. The agent component can be further broken down to *individual* or *population*.

When a population of agents is targeted, it is often high-level data that can be used in a model. These data include probability distributions or percentages describing the value of an agent property or the likelihood of events or agent behaviors. The common point here is that such quantities are assigned as same for the entire agent population. Treating each agent identically is one of the main criticisms that was made in this dissertation as it is hindering the correspondence between model and reality. For instance, if an agent income is described using the normal distribution with a specified mean and standard deviation, the same quantities should be used in random variate generation out of that normal distribution for all agents.

When the targeted component is an individual agent, this means that there is a separate data fed into each agent uniquely. Individual-level consideration of agents has recently been possible [6] thanks to the proliferation of large-scale data that has become available.

The last component identified to be used with data is the *agent environment*. Especially empirically grounded environments that are present in models that aim to represent real-world phenomena with the goal of prediction [2]. Data in these cases are used to (1) characterize resources available to agents and (2) represent environment shape. For the former one, data could be used directly as a quantity or as a percentage. It is likely to update this resource quantity based on a growth formula. For the latter case, environment shape is represented as a geographic information system (GIS) with maps describing different layers of information for the environment. Common map types are those that describe environment boundaries and road networks.

2.3.5 Modeling and Simulation Phase

The characterization based on the simulation phase can be summarized as during *model development/design* or *simulation run* (or both). In model design and simulation run phases, data can be used to create any of the three effects (i.e., changing initial value, modifying the model structure, or generating model structure) or target any component of the model (i.e., agent or environment) as explained earlier.

Assignment during the development/design phase means that values or percentages describing a quantity in the model are hard-coded during runtime. It means that the model is ready to be dynamically driven with different data points. The same applies to the cases of structure modification and structure generation. In this case, making data feedable during simulation run creates a generally usable version of the simulation that can be consumed with different types of data and in different use cases.

2.4 State-of-the-Art of Data-Driven Agent-Based Modeling Approaches

A modeling approach is a set of processes, inputs, and outputs that have directional relationships between each other to guide the modeler on designing a simulation model. Each step or process produces an output or uses the output of another step as input. It is usually required to perform steps and processes iteratively because modeling is an evolving process that aims to create more robust models after each iteration. The number of steps, processes, inputs, and outputs can widely vary by the approach's level of detail. In this section, a chronological review is presented on the comparison of general or generalizable data-driven agent-based modeling approaches.

Kennedy et al. [54] developed a simulation assistance system called AIMSS that provides one of the first methodological efforts on data-driven agent-based modeling. In this approach, there is a simultaneous data collection effort from the real world while developing a classical ABM that is driven by theoretical rules. Once model development is completed, it is then simulated, and the simulation output is collected. Next, the real-world data is processed to gather high-level descriptions. These descriptions are then compared with the output of the data collected from the simulation. If these quantities are off by a certain margin, simulation model and parameter values are modified to match the real-world data output. The process described here is very similar to the 'logic of simulation' by Gilbert and Troitzsch [4] where data is used for validation purposes.

The approach proposed in Kennedy et al. [54] is different from the rest of the approaches concerning its specificity. It suggests collecting real-world data without specific details (e.g., data type, collection method). According to this approach, data has an indirect effect on the model in terms of value assignment, structure modification, and structure generation. Note that

these implied effects are not generated based on data but triggered by it. Kennedy et al. [54]'s approach report in their use case application that agent-related components are affected by the data (indirectly, as mentioned) and there is no mention of the environment-related components. Finally, data usage happens after simulation runs in terms of the modeling phase.

Hassan et al. [23]'s work presents one of the first complete data-driven agent-based modeling approaches in the literature. This approach has substantial improvement over the system described in Kennedy et al. [54] and emphasizes on using data in the model that helps improve empirical grounding wherever possible. While a validation process still exists as similar to Kennedy et al. [54]'s case, Hassan et al. [23]'s approach improves the abstraction and initialization processes with the use of data.

Hassan et al. [23]'s approach integrates data from surveys, panels, interviews, and official documents. In other words, data types supported in their approach are mostly limited to qualitative sources. These data sources can influence both model design and simulation phases while the main target of data in the model is considered agent population and their common behaviors. Finally, the effect of data in this approach is on changing of initial values and modifying the structure.

Smajgl et al. [24] presented a more detailed approach than Hassan et al.'s while the goal is still improving the empirical grounding in ABMs. Smajgl et al. [24]'s approach presents three main phases sequentially. The first two phases create types of agents and identify agent attributes and behaviors. This process is similar to what Hassan et al. [23] describe as *abstraction and design* in his approach. However, the difference is that Smajgl et al. [24] provides specific methodologies that can be used in these steps, which makes it more explicit and modeler-friendly. The third phase in this approach focuses on extrapolating a sample to a population. This

case is similar to what Hassan et al. describe in the initialization step. Again, Smajgl et al. [24] provide more specific methods that can be used in the process.

Smajgl et al. [24]'s use of several different empirical data sources includes expert knowledge, participant observation, surveys, interviews, census data, field/lab experiments, and role-playing games. Also, the focus on agent population as the model component (excluding social networks) and leave out the agent environment. One important aspect to note that Smajgl et al. [24]'s approach is designed specifically for ecology models and may need extra steps to be used in general sense.

Ge et al. [21] created an agent-based simulation platform called Virtual City to study urban populations and related problems. The model utilizes statistical and geographic data to drive certain components in the model. In particular, population data is used for agents whereas geographic data is used to shape the environment. The approach takes high-level statistical data and puts into a generation process that identifies basic characteristic values of the population down to the exact number of individuals. This process creates a population that statistically mimics the real world in terms of attributes such as age and gender distribution. It uses only quantitative statistical data that represents one snapshot of the population values. This data is used in the process that initializes agent attribute values. The difference from the previous approaches is that this one incorporates geographical environment data to reflect the real world better. As the authors also admit, their approach lacks realistic agent behaviors.

Sajjad et al. [20] proposed a data-driven agent-based modeling approach for modeling family formation. The approach reads similar to what Hassan et al. [23] propose. Agents' properties are initialized with census data that is collected in 1990. The difference is that this one does not use data in model design. The model is created as a classical rule-based ABM. The

evolution of agent properties over time are tracked in the model and results are compared with the consequent census data. The use of data only occurs at the simulation time and validation. While the approach by Sajjad et al. [20] does not offer novel insights for data-driven modeling, it has the potential to be used for other purposes. For instance, Yohai et al. [55] follow a similar strategy in setting initial values of a simulation model using a survey data and gather additional non-survey data from social media that is used in agent behavior modeling. In the end, the final model generates new data points to predict future agent states that are very close to the real world.

A very similar approach to Ge et al. [21] is also followed by Venkatramanan et al. [22] to construct a virtual city to study the Ebola epidemic in Africa. The difference is that data is used for formal model calibration instead of an arbitrary generation process. However, agent behaviors are not informed using data in this approach either.

Jensen and Chappin [56] propose an algorithmic approach to generate ABMs from data. The approach takes initial agent attribute values, network connections between agents, model properties list, and expected patterns to be developed by model. Then it selects a decision model from a repository of models and checks the suitability of model outputs for expected patterns. The model selection process continues until a degree of the expected outputs is gathered. Even though the goal of this approach is automating the generation of agent-based innovation models, it is similar to Kennedy et al. [54]'s approach from the modeling perspective. The improvement in this approach is that it provides a more detailed and structured way to update models to match the desired output and showcases how this works in the real world. Nevertheless, the approach does not offer a noticeable novelty when it comes to using data in ABMs. What described as data in the approach are parts of the model code, not the data that drives the model. The use of data

type, collection periodicity, its effect on the model are not discussed. Agent consideration is at the population level.

Bell and Mgbemena [25] proposed a data-driven method to model customer behavior for cellular service providers. They propose to use historical customer retention data to generate a 'general' behavior of the customer (i.e., the decision of staying or churning). Here, data collection covers a long period of time and the data is in the form of both qualitative and quantitative features. In the process, the data is fed into a decision tree that creates rules for determining if an agent churns or stays. In this case, this approach generates rules using data which is a novel approach and is not seen in the previous literature. The targeted model component of this model is on agents and their behavior. The rules, however, capture agent behavior at the population level. The model is then used in a simulation scenario that involves agent interactions that affect churning. Note that the approach is among more comprehensive ones while their method lacks in validation describing how well the decision tree represents historical customer behavior. Finally, a critical challenge is the embeddedness of the approach to the use case as no generally applicable form of their approach is presented.

Fig. 5 summarizes the above taxonomy for data-driven approaches concerning the use of data. Majority of these approaches focus on qualitative data such as surveys. This practice limits their ability to be used in newer sources of large-scale behavioral data. Moreover, the majority of them affect the model with the structure modification at the most and focus on model development/design and initialization. Only one approach creates agent behavior from data. However, its focus is at the population level and generalizability and validation are not discussed.

Data type		Measurement repetition	Effect of data (*I*: initialization, *SM*: structure modification, *SG*: structure generation)			Model component (*A*: agent, *P*: population, *I*: individual, *E*: environment)			Modeling and Simulation Phase (*M*: model development/design, *S*: simulation run, *V*: validation)			Notes	Ref.
						A		*E*					
Qual	*Quant*		*I*	*SM*	*SG*	*P*	*I*		*M*	*S*	*V*		
Yes	Yes	One time	No	No	No	Yes	No	N/R	No	Yes	No	Lacks specificity.	[54]
Yes	No	Repeated	Yes	Yes	No	Yes	No	No	Yes	Yes	Yes	First approach embracing data-driven agent design.	[23]
Yes	Yes	N/R	Yes	Yes	No	Yes	No	No	Yes	No	No	Provides one of the first very detailed approaches.	[24]
Yes	No	Repeated	Yes	No	No	Yes	No	N/R	Yes	Yes	No	Very similar to Hassan et al. [23]	[20]
No	Yes	One time	Yes	No	No	Yes	No	Yes	No	Yes	No	The approach supports only high-level statistics to initialize agents and the use of geographic environment data.	[21]
No	Yes	One time	Yes	No	No	Yes	No	Yes	No	Yes	No	The approach supports only calibration and the use of geographic environment data.	[22]
N/R	N/R	N/R	N/R	N/R	N/R	Yes	No	N/R	No	Yes	No	The goal is to automate the ABM generation, not empirical grounding.	[56]
Yes	Yes	Repeated	No	No	Yes	Yes	No	No	Yes	Yes	No	The only approach that generates agent behavior from data. The use of data is at the population level.	[25]

Fig. 5. Comparison of data-driven approaches based on the data usage taxonomy (N/R means not reported; dark cells mean negative value to aid the reader's eye).

As a result, ***there is a need for a data-driven approach that can generate agent behavior from data at the individual level***. Furthermore, such an approach needs to be presented in a generally applicable manner, should support different data sources, and needs to present all steps expected in an agent-based modeling approach. This is the research gap that is targeted to be bridged in this dissertation.

A successful filling of this research gap will produce an approach which can have many implications for the M&S community and the communities of different application domains. For the M&S community, the use of individual-level data to generate individual agent behaviors will be a novel contribution. This contribution has the potential to shift agent-based modeling practice from theory-driven to data-driven, especially for those aimed at representing real-world systems. As a result of following such a data-driven approach, agent-based simulation models will have a better empirical grounding which is a critical property for a computational model developed in the age of big data. Data-driven ABMs have the potential to broaden the impact of agent-based modeling in scientific communities where empiricism is the dominant philosophy.

Numerous application domains which generate large volumes of datasets can benefit from data-driven ABMs. For instance, urban areas are known for their ability to collect near real-time data from traffic sensors, public transportation use, pedestrian movement, location footprints, among others. Such data sources can provide rich input for data-driven ABMs. Furthermore, other data-rich areas such as cybersecurity, homeland security, and intelligence can make use of data-driven ABMs which can provide the means for gathering rapid feedback which can be used for aiding the decision-making process.

CHAPTER 3

PRELIMINARY REQUIREMENTS AND BUILDING BLOCKS

There is a need to determine preliminary requirements to address the research gap of developing a data-driven approach. These requirements will lead us to a set of building blocks that will be used as the basis for the creation of a new approach. These requirements are aimed to be sorted in an order from more general to specific. Requirement numbers are given in parenthesis after the requirement statement.

3.1 Preliminary Requirements (R1-R9)

One of the commonly found elements of a modeling and simulation process is a conceptual model [11, 57, 58]. A conceptual model describes the details of a model in a form such as text or diagrams [59]. In the proposed approach, there should be a basic structure of a conceptual model that identifies agents, attributes, and desired behaviors to be generated (**R1**).

Given its data-intensive nature, the approach is expected to present a comprehensive list of situations of dealing with individual-level data with aspects such as data cleaning and data testing (**R2**). A specific step in the proposed approach can address the needs of the general audience in terms of commonly applied wrangling techniques on data.

Given the agent structure identified in the literature review, one needs to generate a different component of an agent using data. These components are *agent attributes* and *decision logic* that leads to *action* which can be grouped under the term *behavior*. In terms of attributes, there should be an attribute value assignment process which shows the cases one can elicit attribute values from data (**R3**).

There are more requirements regarding behaviors since it is the main contribution of this dissertation. First, there should be a behavior generation process which uses an extensive list of techniques to capture behavioral patterns from data (**R4**). Due to this vast list, the behavior generation process should provide a guide for the modeler to choose appropriate techniques (**R5**). The behavior generation process should facilitate comparison among multiple competing techniques (**R6**).

There is a need to integrate all the mentioned data preparation, attribute, and behavior generation processes consistently (**R7**). It is often the case that integrated models may not work correctly due to errors that might come across during the software coding process. The approach should suggest a verification process to address such errors (**R8**). Since this approach is aimed to be commonly accessible, there needs to be a technique that facilitates a consistent presentation of the entire approach (**R9**). The preliminary requirements are summarized in Table 1.

TABLE 1

PRELIMINARY REQUIREMENTS

R#	Requirement description
R1	The approach should incorporate a basic structure of a conceptual model that identifies agents, attributes, and desired behaviors to be generated.
R2	There should be a data handling process that will help cleaning and to prepare data.
R3	There should be an attribute value assignment process that shows different ways to elicit individual attribute values from data.
R4	There should be a behavior generation process that uses an extensive list of techniques to capture behavior patterns from data.
R5	Behavior generation process should provide a guide for the modeler to choose appropriate techniques.
R6	Behavior generation process should facilitate a comparison between competing techniques.

R7	There should be a process that integrates generated behavior and attribute models.
R8	There should be a process that verifies integrated models.
R9	There should be a commonly accessible technique that facilitates a consistent presentation of the entire approach.

Each of these nine requirements needs to be satisfied to make the building blocks of the new approach. Six of them (R1-2, R5-8) are created as part of the main proposed approach. The other three (R3-4, R9) required a secondary review. A summary is given on how these nine requirements are satisfied in Table 2.

TABLE 2

SATISFACTION OF PRELIMINARY REQUIREMENTS

R#	How is it satisfied?	Section
R1	A desired level of detail is identified within the proposed approach.	4.1
R2	A well-respected data handling process is adapted.	4.2
R3	A literature review is conducted on a sample of related studies to identify the ways to elicit attribute values from data.	3.3
R4	A literature review is conducted on a sample of related studies to identify different techniques to capture behavior patterns from data.	3.2
R5	A guide is identified based on common characteristics of data.	4.4
R6	A procedure that describes how such comparisons can be made is defined.	4.4
R7	An integration process is identified.	4.5
R8	A simple verification process is identified.	4.5
R9	A short review is conducted to identify a suitable technique.	3.4

3.2 Learning Behaviors from Data (R4)

The advancements of communication technologies created new methods of collecting and accessing behavioral data. Mobile sensors such as GPS trackers and phones and web sensors such as location-enabled social media services are capable of providing behavioral traces of individuals. This section presents domain-specific studies from human mobility that will allow synthesizing a broader picture of using data to capture behaviors. At the end of the section, the identified broader picture is explicitly articulated.

González et al. [60] developed a *statistical model* to understand human movements in an area using a large-scale call detail records (CDRs) dataset. The model uses individuals' movement distances between locations as a proxy to characterize people's typical movement distance (i.e., the radius of gyration) and investigate how this distance is distributed among a population. This model provides one of the first comprehensive studies of human mobility using large-scale granular data.

Schneider et al. [53] suggested a more granular model than González et al. [60]. The model investigates both CDRs and mobility surveys to capture daily movement patterns. Unlike González et al. [60], their study captures mobility as the patterns sequential movements between places (nodes) called daily motifs. This information is kept as a directed graph while the movement sequence is explicitly identified. They attempt to create an individual model to represent the movement of a person via *Markov Chains*. Following the motifs idea, Jiang et al. [61] extended this work to create a transportation planner [53].

De Domenico et al. [62] also looked at an individual's movement prediction problem based on CDR data. The difference, in this case, is the use of not only an individual's location history but also the movement of his/her friends and other people. They utilize a multivariate

time series forecasting model to make such predictions which reported to produce 0.19° error in terms of latitude-longitude measures. Etter et al. [63]'s work focused on a more specific problem of predicting a person's next visit given his/her location history. Their work develops several distinct predictive models a *Bayesian network*, a *machine learning classifier, and* a *probabilistic classifier*. Their models reported performing on CDR data better than the state-of-the-art models.

The mobility models mentioned above report that individuals tend to follow a routine, but they also depart from that routine from time to time. McInerney et al. [64]'s study aims to quantify this departure concept and predict when those departures could happen via an entropy-based *Bayesian* framework. Xu et al. [65] take the mobility concept indoors. The idea capitalized here is to quantify how an individual distorts the pattern of the radio signals. Their study develops a *probabilistic classifier* to predict a person's location within a closed place and record over 70% accuracy.

To summarize the studies above, it is reasonable to conclude that there are numerous techniques to capture one's behavior from data. One can categorize them under the umbrella of *Machine Learning Models* (e.g., machine learning classifiers, clustering, and regressors) and *Statistical Learning Models* (e.g., Markov models, probability distributions, time-series forecasting models, regression models, and probabilistic classifiers). These techniques will provide a list of techniques that a modeler can choose.

3.3 Ways to Elicit Attribute Values from Data (R3)

Web sensors such as social media websites, wikis, and blogs may provide text-based content related to an individual's mood, demographics information, and preference, and so on.

Several studies use such data to elicit attribute values. At the end of this section, the identified broader picture is explicitly articulated.

Mislove et al. [66] investigated the specific demographics of Twitter users in the US and were able to capture the hometown, gender, and race of these users by different ways of comparisons. The hometown is identified based on people's self-reported text-based location property in their profiles *converted* to a named real-world location using a commercial geocoding API. The gender and race are identified based on *matching* a name and last name within the databases, respectively.

Culotta et al. [67] developed an approach to predict the demographics of Twitter users based on accounts they follow. The hypothesis here is that people who follow certain websites on Twitter should belong to the general visitor profile of that website. To examine this hypothesis, the study collected visitor demographics data from an audience measurement system regarding 1500 websites and find people on Twitter who follow those websites. Based on regression analysis tests and hand-curated ground truth datasets, they found a high correlation between the two making this a viable approach to use in demographics identification.

Chen et al. [68] examined personality traits from social media text. They focus on two personality traits (openness and neuroticism) and derive them from individual-based social media message history. The method uses the *extraction* of lexicon-based features. While the focus of the study is on an individual's potential interaction with advertisements, the idea of deriving personality traits from a text is shown to be attainable. This work is later extended to capture personality values [69].

Ryoo and Moon [70] investigated Twitter users' locations with 10-kilometer accuracy. The idea they follow is to model the geographic distribution of the words that are used by

geographically explicit social media content. In this respect, they use the center of a word based on its overall shares in different tweets and calculate its dispersion. They then use this model to *infer* a user's location using the words shared in social media text by that user. Other approaches estimate the home location of social media users with accuracies ranging from hundreds of kilometers [71] to 100 meters [72].

In short, in the studies mentioned above, there are two main ways of eliciting attribute values from data. They will form the basis for attribute value obtaining/generation process. The first one is *translational* which involves using an input with one value fed into a function which generates one output (the attribute value). Race, gender, and hometown attributes are obtained this way in studies such as Mislove et al. [66]. The second one is *algorithmic* which involves using an input with several values that are translated through an algorithm (may involve a classifier) to generate a single value. This process is used in the other studies mentioned above [67-70].

3.4 Commonly Accessible Techniques to Represent Processes (R9)

An essential part of the data-driven approach is to describe the lifecycle of data and its use in generating individual agent behaviors and obtaining/generating agent attribute values. The primary requirement here is to show data flow, processes, and systems as well as their relationships between each other in a structured manner. Diagramming techniques are common ways to show how systems work. While diagrams are often not computable, they provide the visual aid for the reader to understand a system and implement it, if needed, in a more flexible form. Flowcharts are useful diagramming constructs to describe the system, especially software

systems [73]. When standard flow diagrams are used for describing systems, they become quite lengthy and challenging to trace with several decision statements.

Data Flow Diagram (DFD) [74] is a particular flow diagram that addresses the requirement of the approach in a more straightforward manner. More specifically, DFD is "a visual tool to depict logic models and expresses data transformation in a system." [75]. Since its focus is on transmission and transformation of data, it provides flexibility for the reader to understand how processes, systems, and data have a relationship with each other. A data flow diagram is made of four types of diagrams [75].

- *Activity/Process* is used for depicting computations during its flow. Activity is generally labeled using a verb and should always have at least one input flow and at least one output flow.
- *Data Flow* is a directed link between diagrams and shows the data flow direction, splits, and joins.
- *Data Store* is used to depict permanent storage for the data.
- Finally, the *External Entity* is the outside entity used as a data source or sink.

As supporting elements, DFDs are accompanied by a set of *Data Dictionary* objects that describes the physical form of the data in detail and *Process Specifications* that describe the activity/processes in detail.

DFD shapes have several different notations that are proposed after its inception in the late seventies [75]. In this dissertation, Gane and Sarson [76]'s notation is preferred to depict DFDs because it is more distinguishable from a regular flow diagram than Ward and Mellor [77]'s notation. Fig. 6 shows four main diagrams according to Gane and Sarson [76]'s notation.

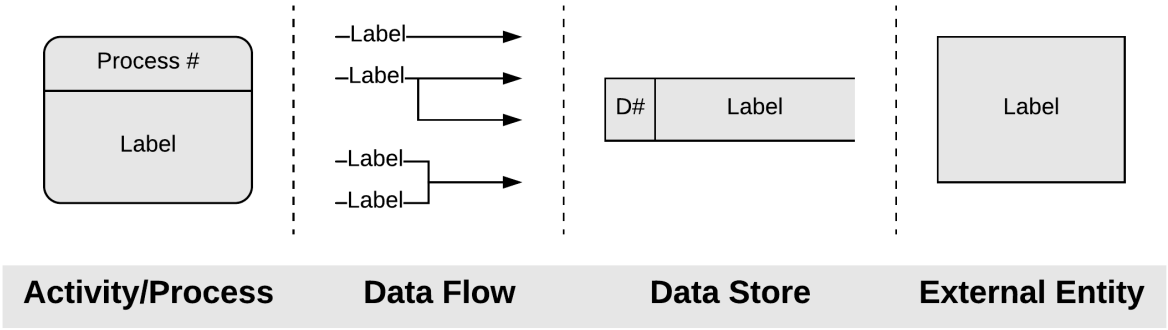


Fig. 6. Data Flow Diagram shapes.

When a system is described using DFDs, it includes a *context diagram*, a *level-0 diagram*, and a *level-1 diagram* which provide an increasing level of details on how data flows [75]. A context diagram depicts system boundaries with the outside world and major data flows in the system. A level-0 diagram describes more details regarding major system processes. A level-1 diagram provides details about individual processes. If a level-1 diagram is not able to provide enough detail, additional levels (2, 3, …, n) can be defined, but practical uses are often limited to level-1 [75]. When necessary, a data dictionary is defined to provide the main properties of data: name, alias, usage, content depiction, and additional information.

CHAPTER 4

PROPOSED DATA-DRIVEN MODELING APPROACH

4.1 Contextual View

At the contextual level, the proposed data-driven approach has a relationship with three external elements/systems illustrated in Fig. 7. *Conceptual model* describes the details of a model in a form such as text or diagrams [59]. *Data source* is a system/repository that generates/serves data to be used for generating agents. *Simulation engine* is a software system that is responsible for the scheduling and execution of data-driven agents. These three elements have a direct relationship with *Data-Driven Agent-Based Modeling Approach (DD-ABMA)* which is the core part of the proposed approach that takes the data and the conceptual model as an input, and creates data-driven agents to be executed by the simulation engine.

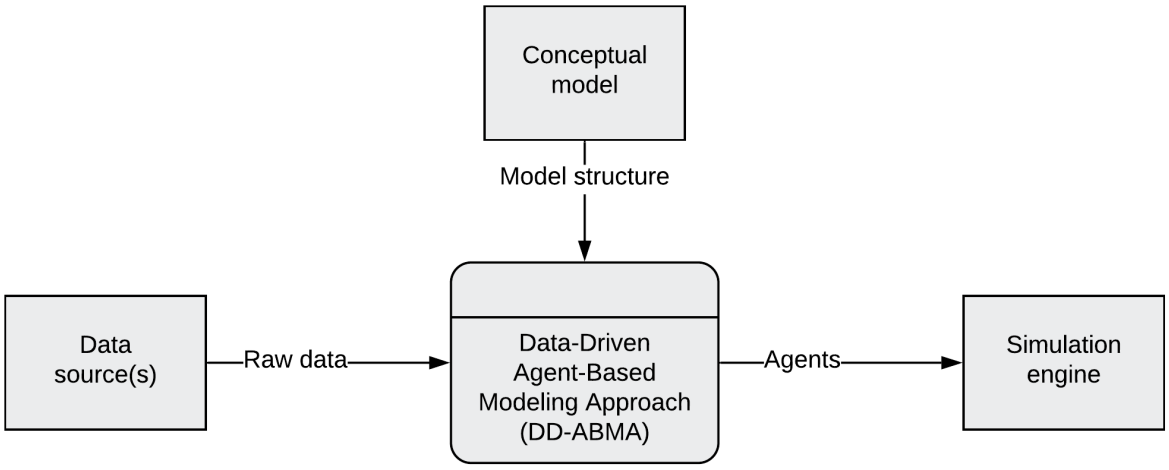


Fig. 7. Contextual DFD of the proposed data-driven approach.

It is assumed that the modeler has a conceptual model of the system to be studied. In other words, the modeler makes the implicit concepts of a model explicit through conceptual modeling [78]. The conceptual modeling process generates a conceptual model that serves as a commonly discussable medium between the stakeholders of an M&S project and assists in reusing simulation models [79]. It is normally desirable in a conceptual model to include detailed descriptions of model elements, their detailed inter-relationships, and assumptions [80]. In the data-driven case, conceptual modeling has an additional purpose: guiding the identification and selection of data sources to be used in model generation. In this case, it is expected that a conceptual model in the data-driven case initially includes only high-level model descriptions and relationships. The conceptual model should include at least: a *purpose statement* describing the goal of the model; *model structure* describing *agent types* with their *attribute* and *behavior signatures*; and the *environment type* and *variables*. The conceptual model here is similar to what one can abstract in a standard ABM [2].

The proposed approach uses the conceptual model (mainly agent attribute and behavior signatures) in the identification and selection of data sources suitable to be used in generating behaviors and eliciting agent attribute values. Example data sources include online repositories, on-demand streams (e.g., APIs), and data generated by physical devices or software. For both behaviors and attributes, the data source has to provide the necessary information at the granularity of an individual agent. Typically, behavior generation requires historical data to include the real-world entity's behavioral actions (e.g., interactions with other entities over time). Attribute data could have direct values of attributes such as the one described in Kavak et al. [81] or may provide the information needed to infer the values through a process such as the highlighted in [82]. Given a conceptual model and data sources, the data-driven agent generation

process uses them to create agents to be executed by a suitable simulation engine. Note that the agent generation process is composed of manually connected processes. A high-level view of this process is provided in the next section.

A more detailed description of the approach is obtained by decomposing the process called “Data-Driven Agent-Based Modeling Approach (DD-ABMA)” into major processes, data stores, and data flows involved in the data-driven agent generation. Fig. 8 describes those relationships using a Level-0 DFD. These processes are namely *data preparation process*, *attribute model creation process*, *behavior model creation process*, and *integration process* which addresses all three requirements respectively. There are two main data stores namely *agent data* and *model database*.

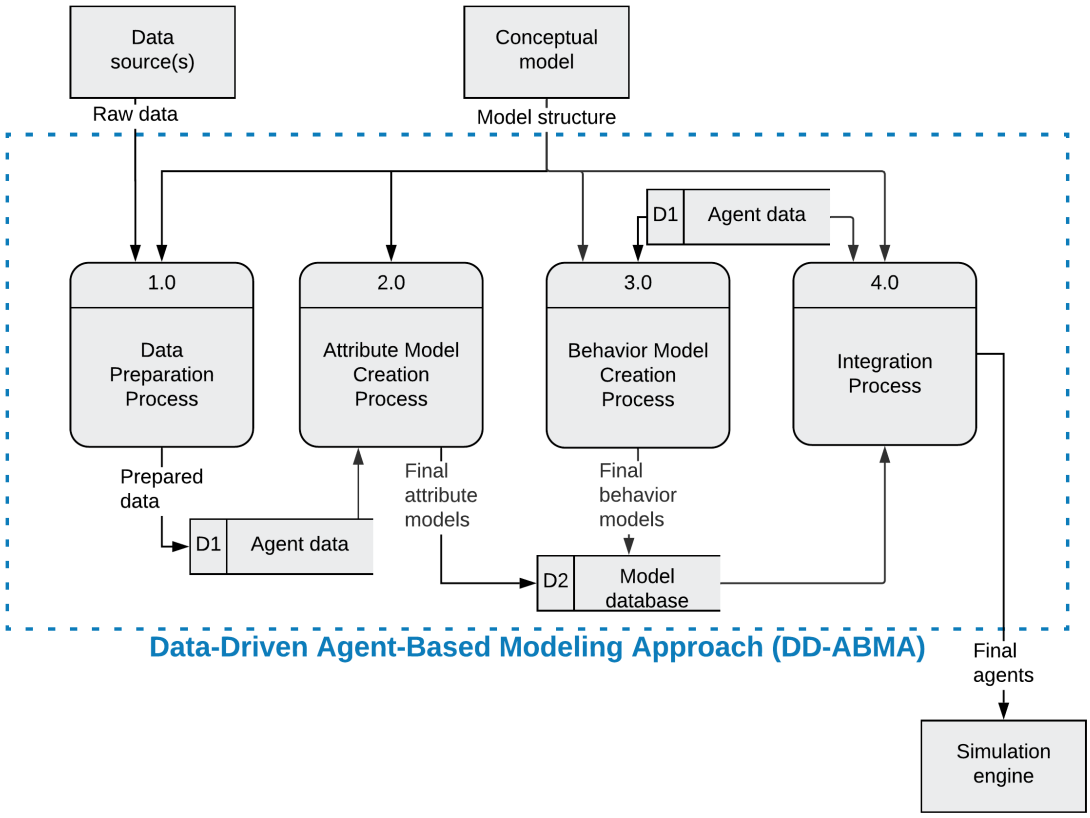


Fig. 8. Level-0 DFD of the proposed data-driven approach.

The *data preparation* process is responsible for transforming the raw data into a new form that is suitable to be stored in the *agent data* store. It involves several iterative *data checks* to make sure that there are no data issues. The prepared data is then stored in the Agent data repository and used in the remaining three primary processes named *attribute model creation process*, *behavior model creation process*, and *integration process*. Respectively, these processes are responsible for creating attribute models if needed or feasible, generating behaviors from data and creating the agents by combining attribute models, behavior models, and agent data according to the model structure provided in the conceptual model.

4.2 Data Preparation Process

Given that there are potentially multiple data sources that could be used in collecting data and several steps to be followed, it is important to capture how data flows from one source to its destination and what processes it follows to reach its final form. This data preparation step is adapted from the data preprocessing steps of the Knowledge Discovery in Databases or KDD Process [83] – a well-respected approach that is used to infer novel patterns from data and thus well align with our data-driven modeling goals. The entire data preparation and checking processes are given in Fig. 9.

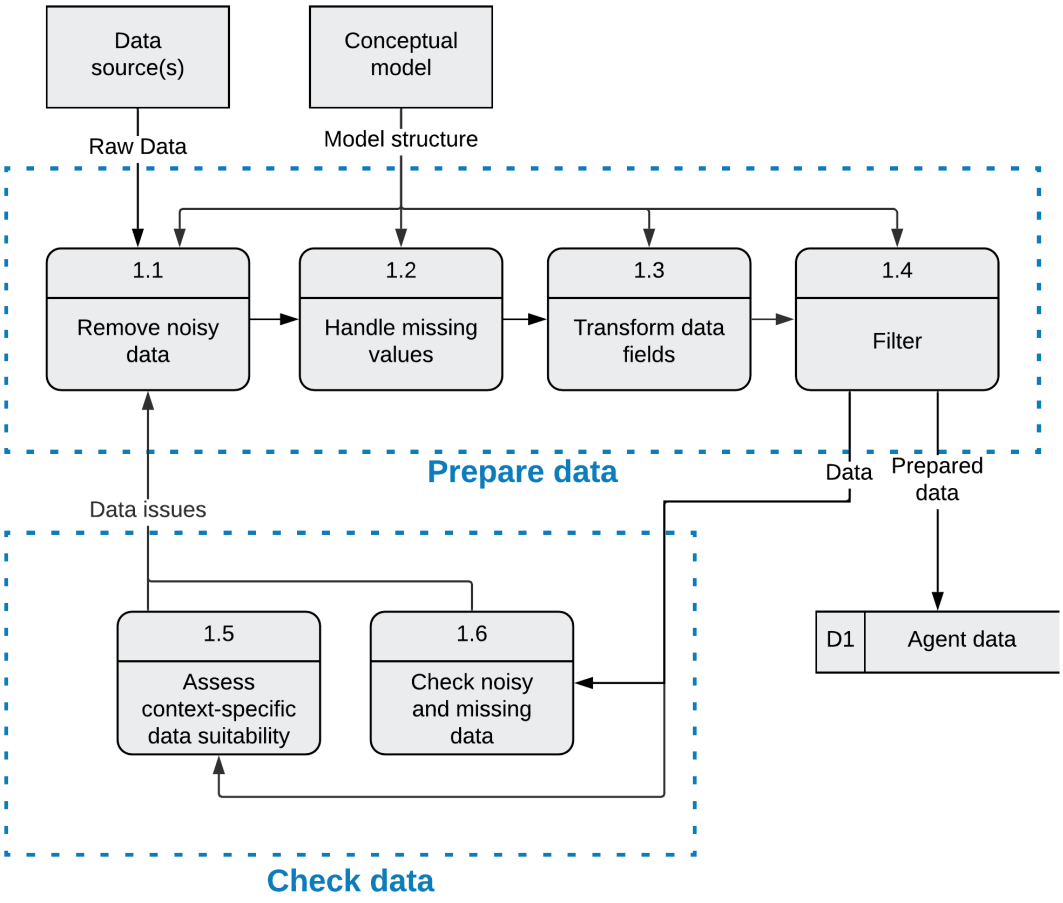


Fig. 9. Level-1 DFD of the data preparation processes.

Data sources, which are external entities to the system, are similar to those seen in a typical data science project including online repositories, on-demand streams (e.g., APIs), and data generated by physical devices or software [84]. Depending on the model and data types, data sources can be used for obtaining/generating agent attribute values and generating individual-level behaviors. It is likely to have multiple data sources aiming to generate behavior for the same model component. In such cases, the modeler should choose data sources that maximize trust to the model according to the goal of the simulation project and the feasibility of accessing the data. Data acquisition is needed to gather data from a data source to the data

preparation process. The acquisition step involves designing and running computer software or using an existing one to obtain data from the data sources. It can become quite complex based on the diversity and the nature of data sources which may require a simple function such as downloading files from an online location or develop/utilize a function that authenticates an online service, and continuously downloads live data. The format of the acquired data does not necessarily make the acquisition process challenging, but it may pose challenges in the preparation step.

After the raw data is acquired, the very first process is to **remove noisy data**. Noisy data entries are those items that skew or bias data which may mislead results. Noisy entries may exist due to a technical challenge on the data collector or just due to an unexpected action from a participant. There are three main techniques to remove noise from data.

- *Removing outliers* is based on statistical properties of the data or based on the knowledge about the phenomena that data represents. One needs to use caution when relying on statistically identified outliers as it assumes that data is normally distributed [85]. It is a fact that many patterns of human behavior are reported to be heavy-tailed [86] and in definition, they have a significant number of outliers but they shouldn't be treated as noise. When noisy entries are identified based on the knowledge of the phenomena, it is expected to use subject matter expertise or relevant scientific literature as reliable sources.
- *Removing redundant entries* is especially critical in crowd-sourced data. Redundant entries could be an artifact on the data source that generates duplicate entries, or there could be data generated by the same individuals multiple times or within a short amount of time. The removal of redundant entries may require developing special (e.g., checksum) algorithms.

- *Removing entries that are generated by unwanted parties* who are not of interest to the developed model is also important. This situation could be present in cases where data is generated by, for instance, bots or individuals that are providing data randomly. Identification of such data varies by case. For bots, one may need to create a machine learning or statistical classifier that is trained with a human-curated set of entries that are labeled as usable or unusable. For data generated by individuals in a crowd-sourcing environment, a specific test needs to be made such as providing already known data points and testing their responses, checking their completion time or response distribution.

The process that comes after noisy data removal is **handling missing values**, which is an inevitable step when dealing with data gathered from public sources. One of the main reasons for having missing values in data is that data collection is a tedious process that may involve humans in the loop (as data generator or curator) and their participation is more prone to error. Another important reason for missing values is that most of the time data collection is not designed according to our modeling goal thus it can miss values or attributes that are significant for our research while that field may not necessarily be significant for the original data collection purpose. An often unaccounted factor in missing value is the damage that occurs while transmitting the data. The challenge is that missing data may impose bias and can make parsing and processing challenging [87]. The data-driven approach suggests four missing data handling methods based on Larose and Larose [88].

- *replacing with a constant suggested by the modeler,*
- *replacing with the mean or mode values of the observed values of the field*,
- *replacing with a random value generated according to the statistical distribution of the observed values of the field*, and

- *replacing with imputed values*.

The selection of a suitable missing data handling method largely depends on the characteristics of data, missing field type, and severity of the missing value. If none of these methods seems to be viable for the data and if data has a large enough sample, it is also possible to remove those records from the dataset.

The third main data preparation process is to **transform data fields**, which is applied when there is a need for updating existing field values or generating new ones to make data compatible or suitable for a given data-driven modeling task. Notably, certain machine learning algorithms such as K-means clustering or neural networks are influenced by the magnitude of data field ranges [88]. Data normalization needs to take place, and normalized field values are scaled to the same range of values to reduce the bias caused by fields with broad ranges. Typically, the normalization process keeps the data field values between [0, 1] or [-1, 1] intervals. Similarly, changing categorical values to numerical values or the opposite is among common data transformation approaches. The generation of new fields based on existing ones is suggested by Fayyad et al. [83] as one of the main tasks to prepare data for a data mining task. For instance, joining tables based on a common data field and generating new values is one way to create a new field. The data transformation that produces a new field may also entail the acquisition of additional data. These first three processes should make the data clean in general cases, but it is possible that one needs to **filter** out certain agent data based on the context of the model.

The cleaned agent data may go through a process named "**assess context-specific data suitability test**." Data suitability assessment identifies what constitutes suitable data to be used in individual-level agent design. This process will become essential in instances when individual

level data size is found to be inconsistent across the dataset. For example, the number of points per agent could vary. In these cases, there are two broad ranges of tests.

- *Individual-level tests* involve selecting data entries that belong to individuals that have sufficient data and express this as data suitability ranges. In this context, "sufficient" means the frequency, length, and the number of entries is satisfactory enough to capture behavior and attribute values.
- *Population-level tests* involve checking the high-level quantities of data. These quantities include different forms of distributions, quantiles, frequencies, and ranges.

Once all data preparation steps are conducted and data suitability checks are performed, there are three potential options. (1) If context-specific data tests fail, then the modeler should stop here and look for alternative data sources. (2) There is still a possibility to have issues such as noisy and missing data due to the subsequent steps applied to data. There is a need for additional **noisy and missing data check** which reports the problems back to the data preparation process. Note that these data preparation and check processes described here are not meant to be completed sequentially or all at once. Similar to the cases seen in other modeling approaches, it requires the modeler to go through these processes and revise them as needed iteratively. (3) Data is suitable in all aspects and ready to be used in the subsequent processes.

4.3 Attribute Model Creation Process

Attributes such as age and gender are essential for a model especially when they influence agent decision making. Thus, assigning realistic attribute values will increase correspondence of the model with the real world. The attribute model creation is based on the

findings from the literature presented in section 3.3 and is illustrated within the proposed approach in Fig. 10.

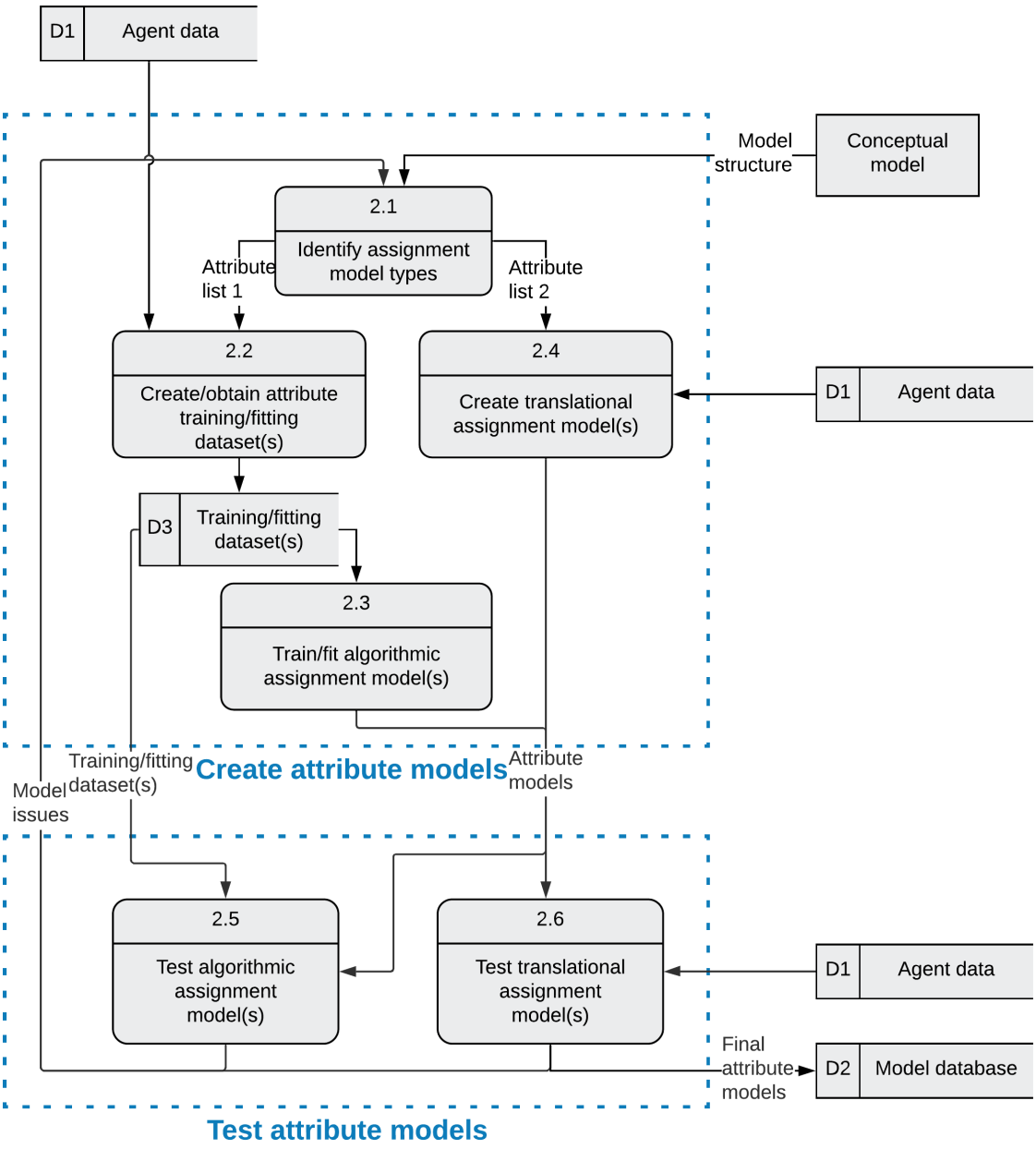


Fig. 10. Level-1 DFD of the attribute model creation process.

The first step within this process is to **identify the attributes that can be assigned empirical values**. This process comprises checking the agent data along with the attribute list and identifies two sets of attributes. The first set of attributes are the ones that require simple translation to gather the attribute values. The second set of attributes are the ones that require more complex and extensive steps of model development such as machine learning models. For simplicity, the former models are called translational assignment models whereas the latter is called algorithmic assignment models.

The creation of the translational assignment model involves cases of having a single record per agent for an attribute. In this case, the assignment is created as a translation function (f_t) which is defined based on the value type required by the targeted attribute and available data. The translation function takes a single value and transforms it into an appropriate form for the attribute, meaning that the function f_t is a one-to-one function that can have one of the following two cases.

- The direct translation assigns the same single value for the attribute (e.g., agent age).

$$f_t(x) = x$$

- The conditional translation assigns a value according to the condition satisfied for the data record (e.g., agent *age range* based on age).

$$f_t(x) = \begin{cases} value_1 & if\ x\ satisfies\ condition_1 \\ \dots & \dots \\ value_n & if\ x\ satisfies\ condition_n \end{cases}$$

The creation of the algorithmic assignment model involves cases of having a set of records per agent for identifying the value of an attribute algorithmically. Unlike the translational assignment, the algorithmic assignment model may have several intermediate steps and advanced algorithms to produce a value. Typical algorithms mentioned here are machine learning

algorithms or a series of mathematical operations. When machine learning algorithms are used, they require training data to be provided for supervised classification/regression approaches. Such training data or **Ground-truth** datasets are obtained by processing agent data over **an attribute dataset creation process**. This process typically transforms the agent data to a format that can be used in machine learning algorithms. When **a ground truth dataset** requires having data to be labeled, one can use specific automated rules that label data or crowdsource. Ground truth dataset is then fed into **algorithmic assignment model creation and training/fit** process. In other cases when it is not required to create a machine learning model, the ground-truth dataset creation process can be ignored and the model could just merely be created as a simple algorithm. A generic form of an algorithmic model can be described according to three main components shown in Fig. 11: input, output, and statement [89]. Statements may involve mathematical operations such as counting, division, summation and other operations such as assignment, comparison, and iteration.

$Algorithm < algorithm\ name >$
$\boldsymbol{INPUT}$: $< input\ specification >$
$\boldsymbol{OUTPUT}$: $< output\ specification >$
$< \boldsymbol{statement}_1 >$
$< \boldsymbol{statement}_2 >$
.
.
.
$< \boldsymbol{statement}_n >$
end

Fig. 11. A generic form of an algorithm description.

Once translational and algorithmic assignment models are created, they go through a test process. For **testing algorithmic assignment models**, one can test the goodness of training/fit of the model. In this case, it often involves evaluating the training/fitting dataset through repeated

experiments. The idea here is to identify how good the model results are concerning the entire data. In other words, this testing process shows how generalizable the attribute model is. If there is a low success, then, the process goes back to the beginning of the attribute model creation step for a revision. The same idea of generalizability applies to **test translational assignment models**. In such models, this testing process is mostly devoted to making sure that the translation works for all the data available.

Note that using algorithmic assignment model for agent attributes is not a novel concept by its own as already suggested by [52]. However, both assignment types here are different from the approach proposed in [52] as their focus is on high-level model variables whereas here it is on the individual level agent attributes which are not addressed in the literature based on the knowledge of the author.

4.4 Behavior Model Creation Process

The behavior model creation and testing processes are the core components of this approach with several highly detailed sub-processes along with three inputs. In the end, this process generates behavior models that can be used in executing agent behaviors. A visual depiction of these processes is displayed in Fig. 12.

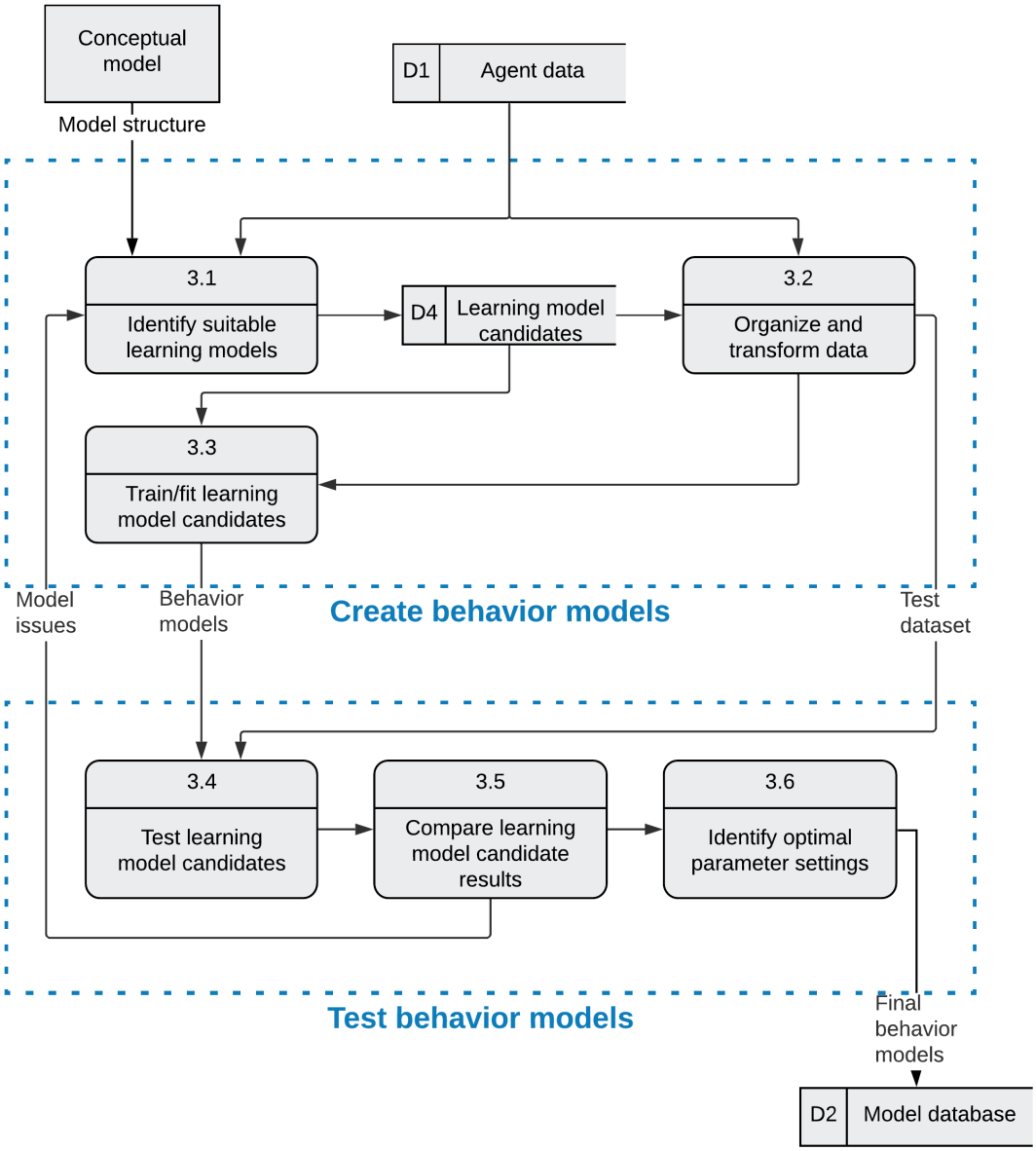


Fig. 12. Level-1 DFD of the behavior model creation process.

The behavior model creation process starts with **identifying suitable learning models** to capture behavior from agent data based on the guidance of the conceptual model. For each behavior to be modeled, one needs to (1) explore corresponding agent data and (2) identify learning models which are appropriate to capture behavior from such data. This duo of data and

learning models need a common ground to match with each other. One method is to review learning models in terms of data requirements posed by them [90]. Generally, two types of models allow learning from data [91]: *machine learning models* and *statistical learning models*. Machine learning models "make less restrictive assumptions and deal with more general models than in statistics" according to Abu-Mostafa et al. [91]. These assumptions posed by both models need to be used in characterizing the agent data.

A data characteristics structure presented as data summary in Table 3. Learning models are used for predicting two types of variables namely ***numeric*** and ***categorical*** [92]. Numeric variables can be represented as real or integer-valued while categorical variables take values from a finite set of possibilities which can be listed as nominal, ordinal, and interval ratio [92]. These two types are the possible decision variables for capturing behavior. Learning models are affected by the sample size in their predictions [93]. Thus, **data size per agent** should be a part of the characterization of behavioral data. This value is especially critical to determine suitable learning models as a low or high number of points may not be suitable to be used in specific models. Next, **the number of variables** (i.e., features) should be identified (excluding decision variable(s)) because high dimensional spaces may introduce the curse of dimensionality [94]. Lastly, several properties of the data and underlying behavior needs to be reported. Especially, statistical learning models make assumptions on the data [95-97]. For instance, time series forecasting models assume time dependency and stationarity whereas random variables that represent probability distributions are assumed to be independent and identically distributed. Uncertainty is the only data property that is not directly related to learning model assumptions and is the measure of irregularity in data [98]. Information-theoretic measures such as Shannon's entropy can be used to quantify uncertainty in data [98]. Low uncertainty means that there is a

noticeable pattern that can be captured from data with learning techniques. If there is high uncertainty (e.g., uniform class distribution), classical machine learning techniques might not be able to capture data adequately as they usually do not account for uncertainty, unlike statistical learning techniques. Finally, agent data might have multiple properties at the same time (e.g., time-dependent and stationary).

TABLE 3

BEHAVIOR DATA SUMMARY TEMPLATE

	Potential values
D = Decision Variable(s) Type	Numeric Categorical
N = Number of Data Points per Agent	$[1, D_N]$ where $D_N < \infty$
V = Number of Variables	$[1, D_V]$ where $D_V < \infty$
Data Properties	Time-dependent vs. Time-independent Stationary vs. Non-stationary Independent and Identically Distributed (I.I.D.) vs. non-I.I.D Low uncertainty vs. High uncertainty

Once the summary table is populated, it can be used to identify suitable learning models. Lantz [93] groups learning models based on the learning task namely classification or numeric prediction. scikit-learn [99] extends it with the addition of data size into the guide. Here, the extension goes even further with the addition of the number of variables and data properties as shown in Fig. 13. This guide walks through the modeler to identify the characteristics of data and suggest suitable learning models according to those characteristics. The guide is designed in a way that it minimizes repetition and covers highly probable paths and mainly guides the modeler on the suggestion of machine learning or statistical learning models which are broadly identified

from the literature and presented in section 3.2. The tables shown on this guide have notes that provide further details on the assumptions or best practices of learning models.

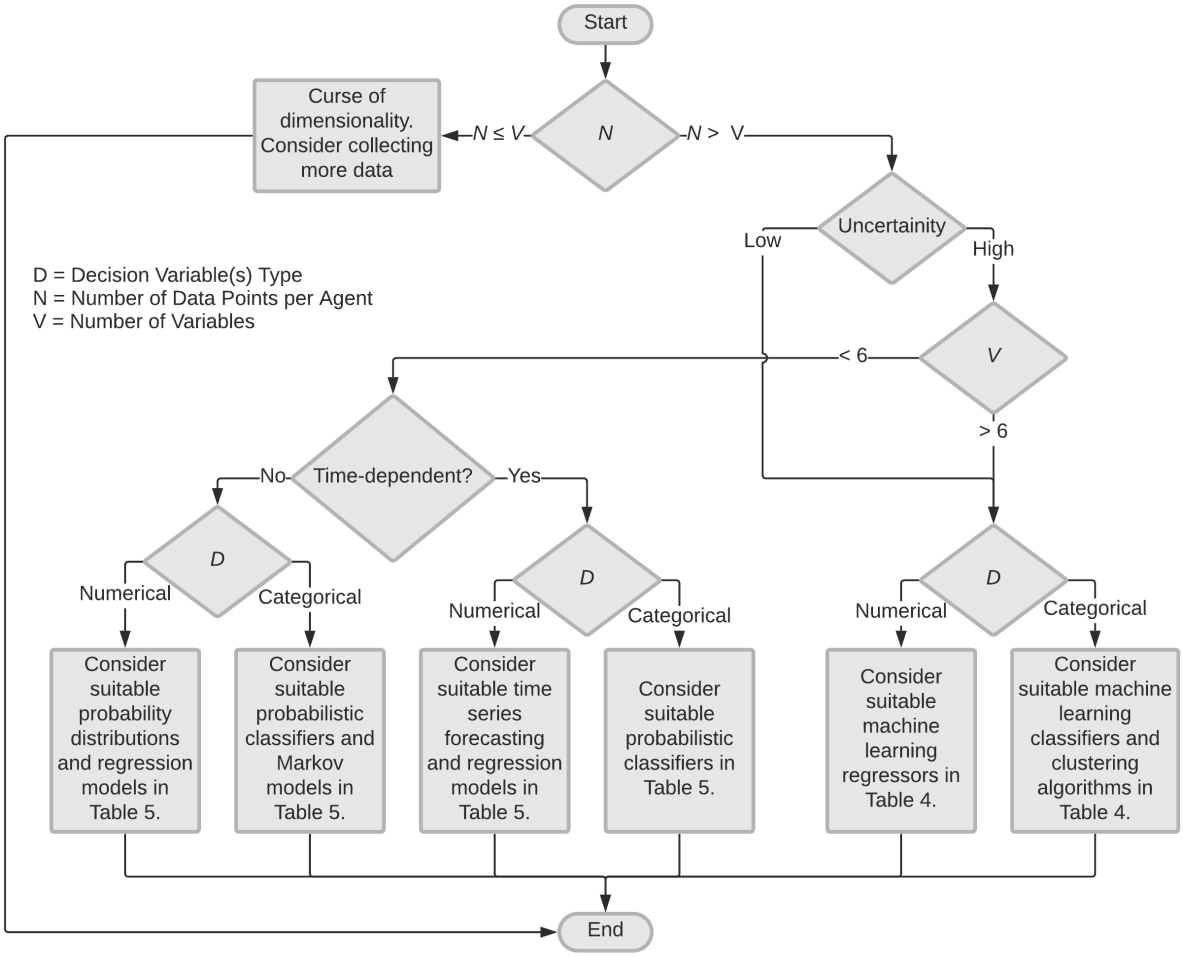


Fig. 13. A guide for selecting suitable learning models.

To sum up, as long as the data size is higher than the number of data points and uncertainty is not high (in the sense of entropy measures), the guide suggests using machine learning models listed in Table 4 according to their decision variable types. It is because statistical learning models make assumptions that make it challenging to satisfy and time-

consuming to test. If there is high uncertainty in data, the guide favors statistical learning approaches. In this case, the guidance becomes more specific based on data properties including linearity, time-dependency, stationarity, and so on. For instance, regression models assume normality, independence, and homoscedasticity while machine learning models such as neural network regressors do make such assumptions [100]. These assumptions are not easy to satisfy. As Thomson [101] indicates "experience with real-world data; however, soon convinces one that both stationarity and Gaussianity are fairy tales invented for the amusement of undergraduates."

Nevertheless, at the end of this step, there will be a list of **learning model candidates** that are suitable to capture the behavior in data. It is important to note here that learning models suggested in Table 4 and Table 5 are added here to provide a starting point for the modelers and are not meant to cover all the learning models available in the literature exhaustively.

TABLE 4

MACHINE LEARNING MODELS

Model Family	Example Models	Notes
Classifiers	Support Vector Machines, Ensemble Classifiers, Multilayer Perceptron, LSTM, K-Neighbors Classifier, 1R	Support Vector Machine's Linear vs. Non-linear kernel may differ significantly on computational time.
Clustering Algorithms	K-Means Clustering, Association Rule Learning, DBSCAN, DENCLUE, Partitional, Density-Based, Hierarchical Clustering, subspace clustering, ensemble clustering	Some of these models require identifying the number of clusters.
Regressors	Support Vector Regressor, Random Forest Regressors, Multilayer Perceptron Regressor, LSTM Regressor	Regressors are a variation of the classifiers without applying activation function to the output and evaluating the output according to a different error measure such as the mean squared error.

TABLE 5

STATISTICAL LEARNING MODELS

Model Family	Example Models	Notes
Markov Models	Markov Chain, Hidden Markov Model	Suitable for capturing state transitions. A future state is only dependent on the current state and not earlier ones.
Probability Distributions	*Discrete:* Bernoulli, Beta-binomial, Binomial, Discrete Uniform, Geometric, Poisson *Continuous:* Beta, Cauchy, Exponential, Gamma, Logistic, Lognormal, Normal, Pareto, Uniform, Weibull	I.I.D. is assumed. Requires 30+ instances when estimated using frequentist approaches. Bayesian inferences can work with fewer data points.
Time-Series Forecasting	Autoregression (AR), Autoregressive Integrated Moving Average (ARIMA), Vector Autoregression Moving-Average (VARMA), Simple Exponential Smoothing (SES)	Stationarity and uncorrelated random error are assumed. Suitable for modeling time-dependent data. Uneven spacing treatment is not commonly found.
Regression Models	Simple Linear Regression, Multiple Regression, Non-linear Regression	Normality, independence, and homoscedasticity are assumed.
Probabilistic Classifiers	Naive Bayes, Logistic Regression, Decision Tree Classifiers	All variables have the same influence (Naïve Bayes)

Then, the next step is to **organize and transform agent data** in the form that it will be suitable to be used to **train/fit the learning model candidates** mentioned above. Individual agent level modeling consideration can become a challenging effort for computational requirements and machine learning model performance. For computational requirements, one needs to ensure that training times of machine learning models are reasonable depending on data

size and parameters' values. When it comes to model fitness and robustness, issues arise on the selection of machine learning techniques, the choice of an appropriate set of features, and the identification of machine learning model parameters. Failing to address such challenges may result in unfeasible training times, low-performance prediction performance, and overfitting. Such tests are specific to learning model families. Machine learning models are usually tested by their prediction accuracy and related scores for classification tasks and mean error measures for numerical prediction tasks. Such tests involve separating agent data by individuals and **organizing and transforming** in the form of training and testing data. The former is used for training a learning model whereas the latter one is used for evaluating the score which is called "**test learning model candidates**" and "**compare learning model candidate results**" in Fig. 12. Such tests do not only look at the mentioned scores, but also identify a suitable set of parameter settings for the learning task that maximizes the performance and minimize overfit. Also, an important aspect of this testing process is on the computational time required for the training. This process is called 'hyperparameter tuning' in the machine learning literature [102]. Making an extensive set of parameter tuning process may take long computational time given that there will potentially be multiple machine learning models to test, hence it is more practical to limit the variation in parameter values. Also, since these tests are conducted at the individual agent level, it is noteworthy to evaluate the distribution of scores across the agent population.

Statistical learning models, on the other hand, require model family-specific evaluation tests to be conducted. Regression models can be tested using different metrics such as R squared value or mean square error that tests the difference between the actual and predicted values. Statistical models based on probability distribution fitting can be evaluated using goodness-of-fit tests such as Chi-Squared test or Kolmogorov–Smirnov test. Time series model forecasts can be

evaluated by comparing the predicted value against the actual values using metrics such as Mean Absolute Percentage Error. Probabilistic classifiers and Markov model performances can be tested in the same way the machine learning classifiers are tested. It is critical to note that statistical learning models can also provide confidence intervals that give the range of values that can be generated. One of the main challenges of evaluating such models is the lack of common comparison ground. On the flip side, statistical learning models can capture the causality of the behavior whereas machine learning models act as black-box models.

Once an appropriate learning model is selected, then the actual behavior pattern learning process can take place. That is, the selected learning model is further evaluated for **identifying optimal parameter settings for** the model performance, and this involves extensive hyperparameter tuning and evaluation of different training/test data ratios. Also, repeated cross-validation will help to eliminate bias originating from data. Later, the most optimal model parameters are selected to be used for generating agent behaviors. Finally, machine learning of behavior takes place through training of the model with data. This process generates a model configuration. For instance, if the learning model is a neural network, then the model configuration will involve nodes and connections (i.e., weights) of the network.

4.5 Integration Process

The last process deals with integrating and verifying the previously created attribute and behavior models and generating agents to be executed by a simulation engine. A visual representation of this process is displayed in Fig. 14.

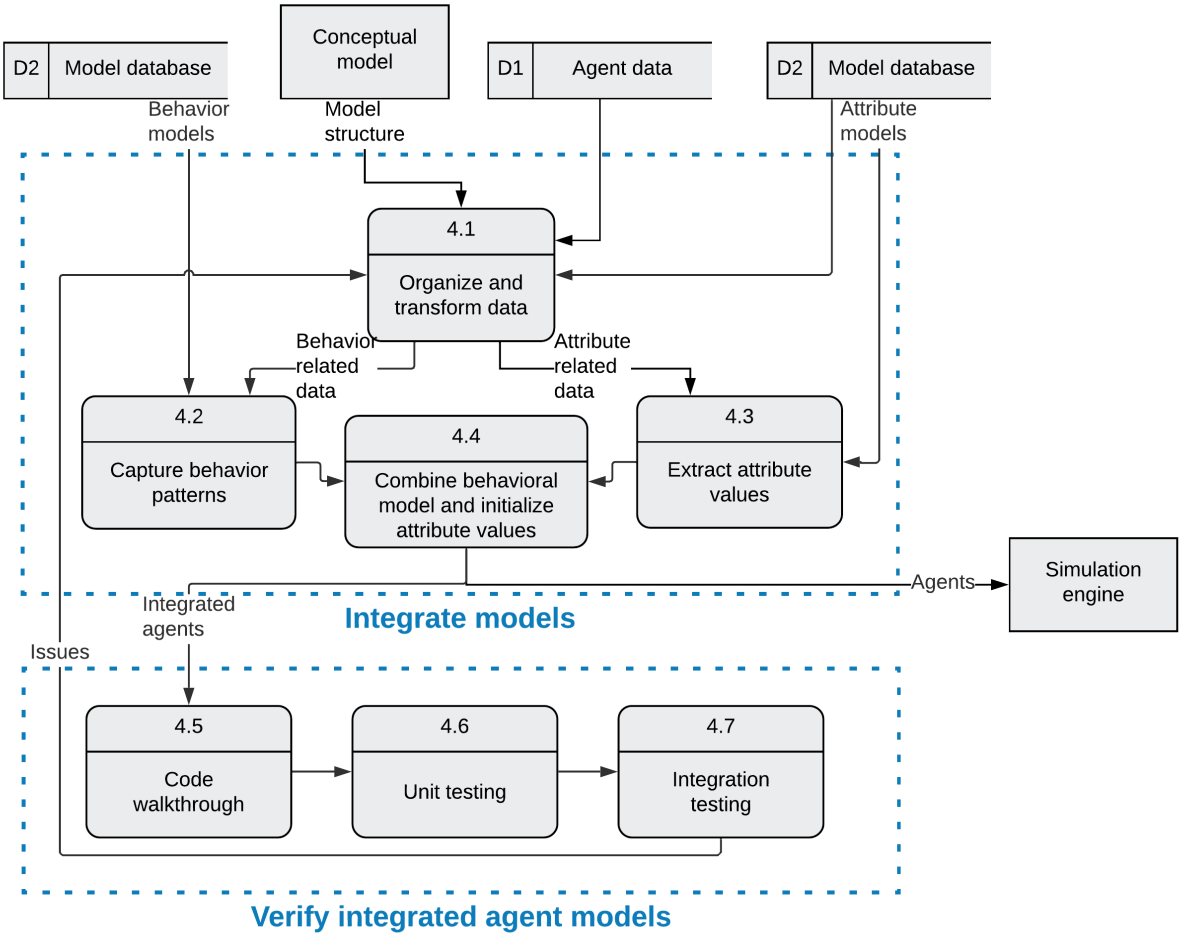


Fig. 14. Level-1 DFD of the integration process.

The integration is a very straightforward process that starts with **organizing and transforming agent data** for behavior/attributes and model settings kept in the model database. Simultaneously, behavior and attribute models are used in **capturing behavior patterns** and **extracting attribute values**, respectively. The organized data is then separated at the individual level and applied to attribute models that are trained and stored in the model database. The output of this process is usually a value. After all attribute values are extracted, one needs to represent them as key-value pairs that are passed to the next and final step. In the final step of the integration process, behavior generation models and attribute generation models are **combined**

at the agent level. All components should be organized so that once the simulation engine executes them, they should be able to work without outside intervention. If a particular simulation engine is used, then agent implementation should follow that specific agent design.

It is often the case that integrated models may not work properly due to various errors that might come across during the software coding process. One needs to go through a series of verification processes which can use numerous techniques as summarized in [103] to address such challenges. Three different techniques are suggested here. **Code walkthrough** is the process of peer reviewing code by inspecting it together with a peer in order to uncover possible software errors. **Unit testing** is the procedure of testing individual pieces in the code and making sure that such individual pieces such as classes or functions perform as expected. **Integration testing** process deals with testing larger code pieces that are made of multiple unit-tested pieces with the idea of uncovering potential problems that might occur due to integrating several pieces of this code. If any issues are spotted, the process may go back to the organization and transforming of the data step to check other unnoticed issues. Otherwise, the integrated agents will be sent to the simulation engine for execution.

CHAPTER 5

USE CASE: HUMAN MOBILITY SIMULATION

This chapter presents a use case on human mobility that follows the proposed data-driven approach. Mobility is an inherent part of our lives capturing our movements from place to place. Within the scope of this use case model, human mobility is conceptualized as short distance daily trips made on any regular day with considerations including trip origin, destination, duration, purposes, and associated activities [21, 104, 105]. Human mobility significantly affects daily life on topics ranging from urban planning to the spread of diseases [106]. It is also an important factor in traffic jams, fuel consumption, and air pollution among other issues. Thus, undertaking mobility challenges can help increase people's quality of life. Within the rest of this chapter, the use case is presented with increasing details following the proposed approach. It starts with a contextual view and ends with results gathered from multiple types of simulations.

5.1 Contextual View

Contextual view of the approach provides a high-level perspective on the main inputs and outputs. A depiction of the contextual view of the use case is shown in Fig. 15. Here the contextual view starts with the description of the human mobility conceptual model and proceeds with the description of data sources to be used for the model. Here, the simulation engine is a simple custom-built Python program that orchestrates the execution of data-driven agents.

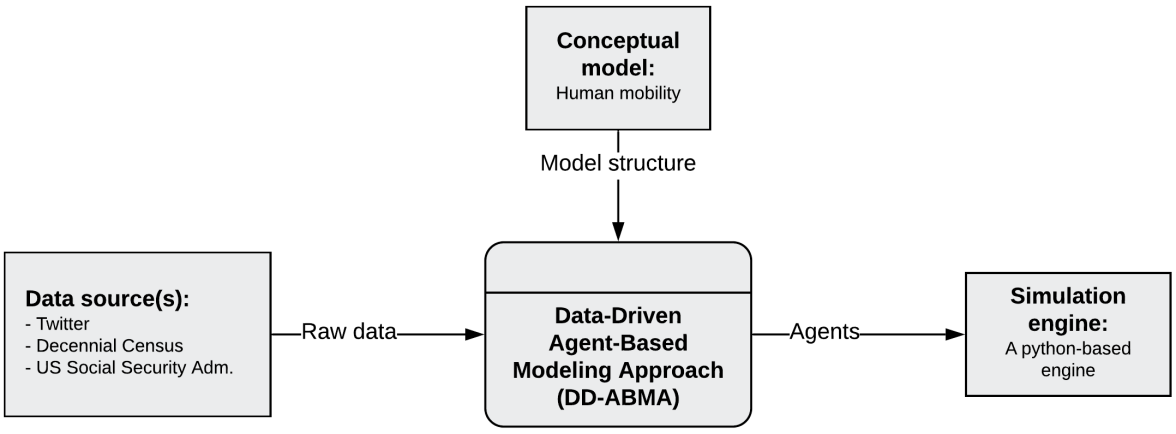


Fig. 15. Contextual DFD of the use case model.

5.1.1 Conceptual Model

The goal with this model is to simulate the movement of people in urban areas and use it as a baseline to address urban issues affecting humans or issues affected by them. The agent type to be developed in this simulation is a human agent (*HumanAgent*). Fig. 16 depicts a simple conceptual model representing the agent structure as a class diagram and a visualization of how an agent could move. *HumanAgent* is an entity situated in a geographical environment and has static and dynamic attributes. Static attributes are *name*, *gender*, *race*, and *home location*. Dynamic attributes are *current location* and *visited locations*. All these attributes are initialized at the beginning of a simulation run. The agent is set to an initial *current location* and it moves to other locations based on the movement patterns to be identified in mobility data. Research in human mobility reveals that human movements are influenced by the time [107] and periodic visit repetitions [60]. The mobility model in this dissertation will represent time information at different granularities to capture the periodicity and circadian rhythm in human movements, as well as the connection between consequent locations [53].

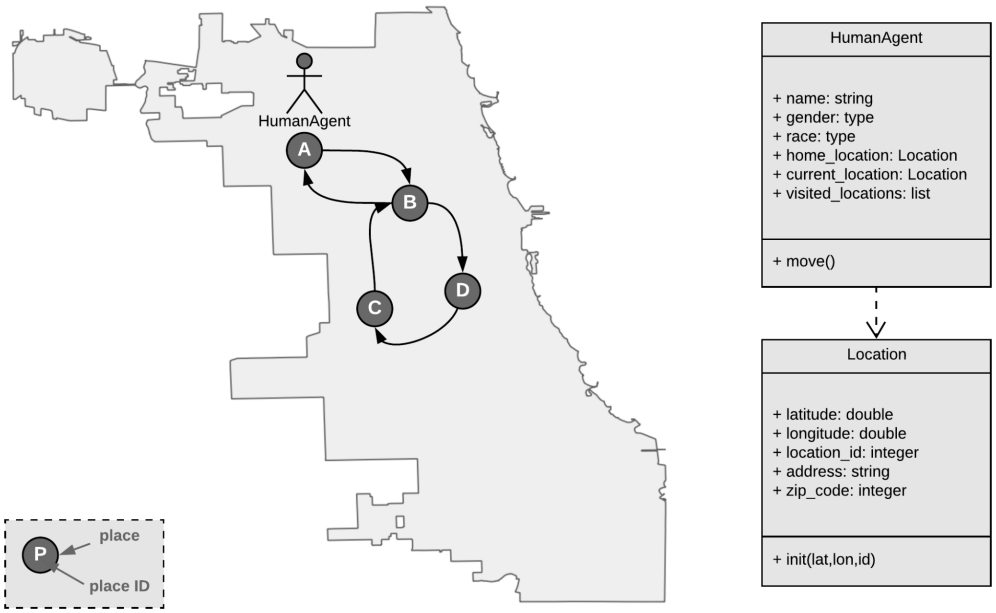


Fig. 16. The conceptual model. The map on the left side illustrates the movement of an agent. The diagram on the right side shows attribute and behavior structure of the *HumanAgent*.

Agents are located on an environment that represents a city or a similar sized urban area. The environment may have several layers depending on the research problem to be addressed. Default layers considered here are *location boundary*, *roads*, and *buildings* of the area. Agents can access any of these layers and make changes on their properties. Additional layers can be added if the research problem requires. The layered agent representation has been found useful in other spatial agent-based simulations [46].

5.1.2 Data Sources

There is a need for a mechanism to capture human movements within a given area, longitudinally, to understand individuals' movement behavior across space over time. A Twitter

dataset is used for this purpose. Twitter (twitter.com) is a popular social media platform used by over 330 million monthly active users [108]. It allows the users to share their status updates (tweets) with up to 280-character text in addition to optional images and videos. If a user enables location share, his/her tweets are tagged with geographical information at the coordinate level. Geo-tagged Twitter messages have been a source of empirical evidence regarding human behavior in disasters [109], public health [110], physical activities [111], and tourist movements [112], to name a few. Fig. 17 shows an example of the structure of a Twitter message. Details about the Twitter data can be found in Appendix B.

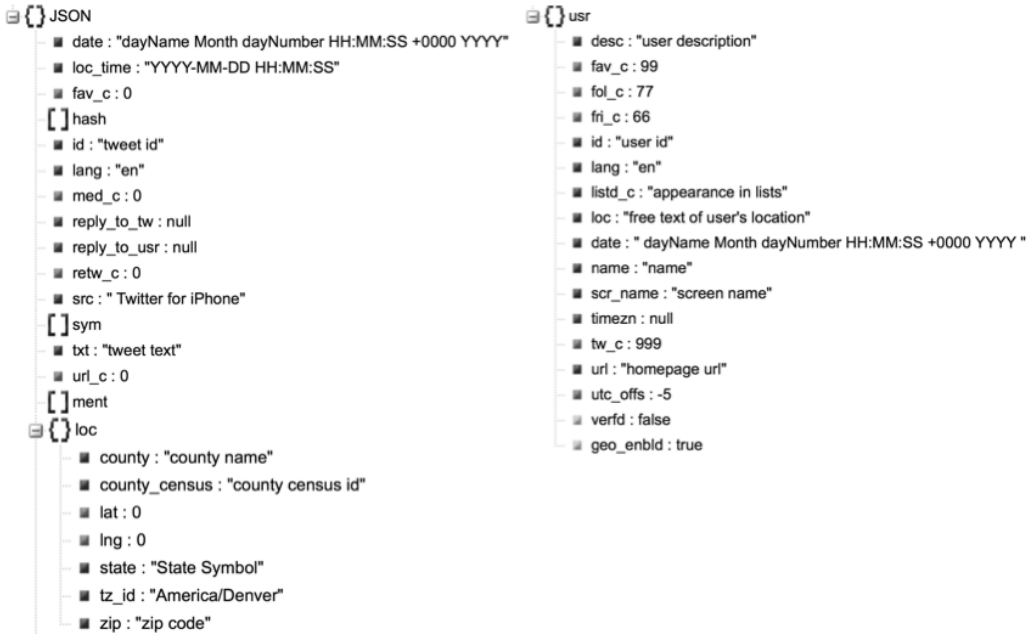


Fig. 17. An example Twitter message structure.

Geo-tagged Twitter messages have been used in spatiotemporal movement analysis of people. The literature reveals examples from mobility at the global scale [113] and local-scale [114]. A seminal work by Jurdak et al. [115] uncover that individual-level patterns of movement

can also be studied using geo-tagged Twitter data similar to those found in call detail records from cell phones. Given its relatively straightforward collection interface and public access, this dissertation uses geo-tagged Twitter messages as one of the two main data sources.

The Twitter data is collected from public geo-located tweets using Twitter's Streaming API by identifying a bounding box (as latitude-longitude pairs) that covers the conterminous U.S. Instead of focusing on a specific use case area, this large area is covered for two reasons: (1) determining people traveling across US states and (2) using the same data in several experiments. Twitter4J [116], a Java library to facilitate authentication and data acquisition from the API, is used.

The other two data sources are aimed to be used for gathering race and gender values. The first one is the US Decennial Census dataset that serves the last names of the population and their corresponding race distributions within the US. Such dataset is useful matching last names with certain races. The dataset includes several columns: last name, rank, count, percentage white, percentage black, etc. The second one is the US Social Security Administration's baby names dataset, which shows gender distribution of each name given in the US dating back to 1880. Such dataset can be used to infer gender based on a person's first name. There are only three columns name, gender, and count organized by yearly files. Both datasets are freely available. In this work, they are kept in flat files due to their small size.

5.2 Data Preparation Process

The data preparation process for the use case starts with the first three steps described in the approach. These steps consist of removing noisy data, handling missing, values, and transforming data fields and applies to the Twitter data. Due to its nature, Twitter data also

requires context-specific checks to examine its suitability. When it comes to the US Decennial Census dataset and US Social Security Administration's baby names dataset, the process was shorter because these two datasets are already clean. The only step that is conducted is the selection of certain columns, calculating their percentages, and filtering based on date. A summary of the operations conducted within the data preparation process is listed in Table 6. Elaboration on how these operations are conducted follows.

TABLE 6

DATA PROCESSING OPERATIONS

Operation	Step	Data Source
Calculate race percentages	Transformation	US Decennial Census
Date specific filtering	Filter	US Social Security Administration
Merge same name/gender pairs	Transformation	US Social Security Administration
Remove data with unusable location information	Remove noisy data	Twitter
Remove data outside the U.S.	Remove noisy data	Twitter
Remove data from non-human accounts	Remove noisy data	Twitter
Remove redundant entries	Remove noisy data	Twitter
Gather complementing location information	Transform data fields	Twitter
Match with a time zone	Transform data fields	Twitter
Calculate local time	Transform data fields	Twitter
Create partial time fields	Transform data fields	Twitter
Capture sentiment	Transform data fields	Twitter
Cluster visits and create previous place field	Transform data fields	Twitter
Temporal coverage	Context-specific assessment	Twitter
Regularity in location visit frequency	Context-specific assessment	Twitter
Regularity in location visits temporality	Context-specific assessment	Twitter

Removal of noisy data starts with eliminating records that have unusable location information. According to Twitter's API Documentation [117], geo-located twitter messages (i.e., tweets) provide location information in at least one of two properties: *coordinates* and *place*. The *coordinates* property is a longitude-latitude pair acquired from the user's device. The *place* property, on the other hand, provides "named locations with corresponding geo-coordinates" but a tweet with a place property is "not necessarily issued from that location but could also potentially be about that location" according to the Twitter API Documentation. In many instances, the place property provides some fields (e.g., country code, attributes, place type, full name) that include a bounding box area that potentially covers the tweet location. Among these, the country field provides a consistent value while other fields do not. For instance, country field values are spelled in the same manner for the same country; however, the full name of a place is seen differently (e.g., Manhattan, NY; New York, US; New York, USA). In addition to this, it is observed that the bounding boxes, in some instances, cover a large geographic area – over 400,000 square miles. For these inconsistencies and low resolution, tweets that have precise coordinates, through coordinates property, are used and the rest are removed. Since the use case model focuses on the mobility of people who live in the United States, tweets that are shared from outside the conterminous US is removed.

Once the aforementioned unusable tweets are removed, the next operation is to clean tweets that do not belong to humans. These non-human accounts are, based on manual inspection, found to be disseminating automated job postings and post messages about important events, etc. It is found that some of these accounts' tweeting activities are beyond human limits (e.g., over 1000 message posting/day). One common aspect in these non-human accounts is that

they post frequent messages where their message text include links – similar behavior is seen in spam Twitter accounts [118, 119]. This link-based approach is used in combination with Guo and Chen [120]'s approach of exceptional applications (e.g., Foursquare) that automatically create location records for humans and location displacement velocity test [115]. Thus, the following rules are identified for finding accounts that do not belong to humans.

- Accounts that have an average of 100 tweets/day ≥.
- Accounts that have an average of 1+ tweet/day ≥, link percentage of these tweets are 90% ≥, and none of tweets are sent from *flickr*, *endomondo*, *path*, *instagram*, *foursquare*, and *untappd* applications.
- Accounts that have maximum location displacement velocity of 240+ m/s for the consecutive visits of 1500+ m.

The last item regarding removing noisy data is to remove redundant entries. In this respect, data points are organized as longitudinal traces of individuals. For each person, consequent tweets, in other words, check-ins, are compared with each other. If two consecutive tweets are shared within 60 minutes and located within 100 meters, they are considered redundant. For such data points, the second item is removed. This elimination cleared approximately 24% of location traces.

Missing value handling and *data field transformation* are combined within a single step. First, by using the latitude and longitude information, it is aimed to find corresponding location information such as city and state names. This process is also known as reverse geocoding – a technique to find the closest address information using location coordinates (latitude and longitude). A typical reverse geocoding algorithm requires a detailed map of the area of interest, so that latitude and longitude is mapped to the closest physical address. However, this geocoding

process is time-consuming when there is a large number of data points. Instead, a partial geocoding process is conducted using 'Counties (and equivalent)' dataset to map each latitude and longitude point to a state and county. Zip code of each tweet is then identified using the Zip Code Tabulation Areas boundary database. Note here that tweets that do not match with a zip code are cleaned. This especially applies to tweets that are shared close to the US borders. After that, time zone information for each tweet location is obtained from Efele[2], which is constructed using separate county boundaries and time zones layer of the US National Atlas. Lastly, by using the time zone information, the local time of each tweet is calculated. Local time is important to properly understand certain behavior of a population such as hourly activity volumes. The time zone information replaces the Twitter messages' time zone information which is not always accurate. Furthermore, due to temporal features of human mobility, three partial time objects (*hour of the day*, *day of the week*, and *weekday/weekend*) are created for each location footprint. Following that, sentiment (mood) of each footprint is calculated using SentiStrength [121] – a Java-based sentiment analysis library specialized in short and informal text.

The last two data transformation steps within the data preparation step is location clustering and the addition of the previous location. The clustering here identifies tweets sent from same places because GPS data usually has a relative inaccuracy even when shared from the same location. In that, such close points are clustered by giving them the same *location ID* label using the DBSCAN algorithm [122]. The DBSCAN algorithm is an effective clustering technique for geo-tagged twitter data [115] based on the *spatial distance between tweets* and the *minimum number of points at a cluster*. In this step, DBSCAN parameters are identified as follows: 100-meter as the maximum distance parameter and one as the minimum number of

[2] efele.net/maps/tz/us/

points. 100-meter is a better resolution compared to related literature [115]. Lastly, previous tweet location (as a coordinate and location ID) is added to the trace data to provide the means to capture the consecutive visits. With the clustering of location traces and the addition of the previous place, *the initial agent data* is created.

While the initial agent data is cleaned, there is a need to assess the suitability of the Twitter data at the user level. It is because location traces of different Twitter users are not equal in terms of the data size and temporal coverage in addition to being sparsely shared and having no labels. Fig. 18 shows such characteristics on an example Twitter use user. The data is grouped into *hour of the day* and *day of the week* to understand the sharing habits of users. When superimposed to appear on the same week, one can capture the weekly coverage visits. In this respect, the idea is to have different weekly data points to capture weekly temporal coverage. In this work, it is expected that at least 12 unique hours of each day of the week is captured (making 84 unique hours total). With that, users with fewer data points are eliminated.

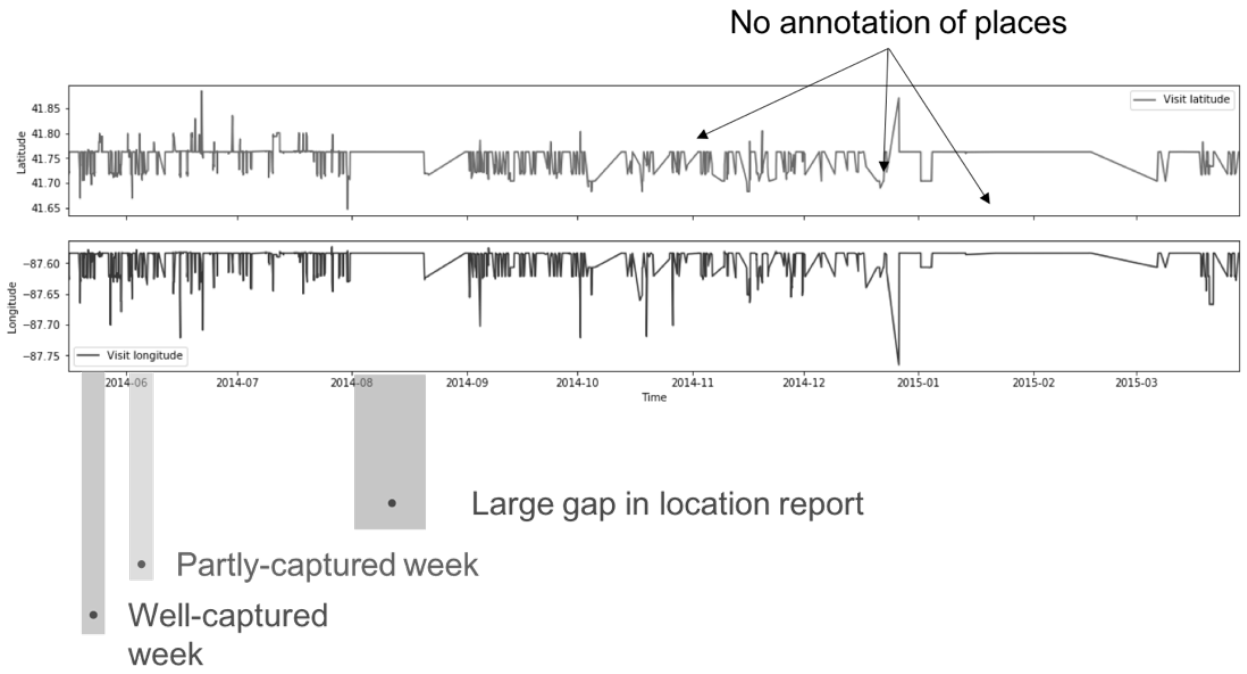


Fig. 18. Coordinate-based location traces of an example Twitter user. It shows the sparsity and inconsistent location capture of Twitter users as well as the lack of annotation.

The last suitability check is needed on the high-level characteristics of data according to [60]. It is reported in empirical human mobility studies in which individuals tend to visit a few locations (e.g., home, work) a lot more frequently than the others. It is also noted that individuals tend to visit previously visited locations regularly. The former is captured with a Zipf plot that displays the frequency of visits to unique locations. The rank of the plot shows the visit frequency in Fig. 19 (left). The latter is captured with temporality that shows the return probability of an individual to a previously visited location estimated from location footprints. As shown in Fig. 19 (right), 24-hour cycles are present in the Twitter data. Having these mobility characteristics present in the data gives additional assurance on using it. If these characteristics were not present in data, further use in modeling might be questionable.

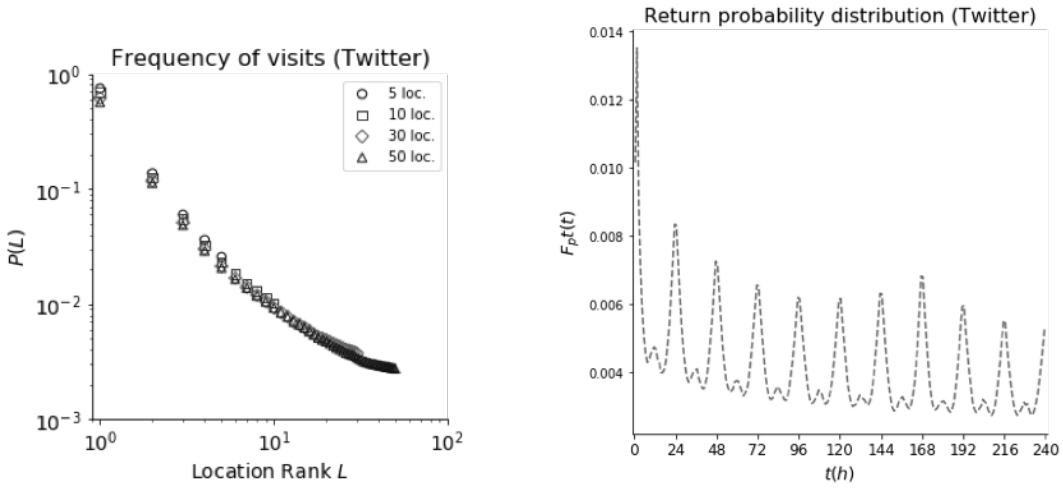


Fig. 19. High-level mobility characteristics of Twitter data. The figure on the left shows a Zipf's plot showing the location visit frequencies of all users based on place rank. The figure on the right shows the return probability to a previously visited place after certain hours passed.

5.3 Attribute Model Creation Process

In the process of attribute value assignment, the first step is to identify attributes to assign values using agent data. In this model, several attributes are found to be suitable for assigning values from data. These attributes include *name*, *race*, *gender*, and *home location*. The first three attributes are suitable for translational assignment while the last one requires a complex model creation as summarized in Table 7.

TABLE 7

LIST OF ATTRIBUTES AND THEIR VALUE ASSIGNMENT TYPES

Attribute	Value Assignment Model Type
name	Translational assignment
race	Translational assignment
gender	Translational assignment
home location	Algorithmic assignment

5.3.1 Translational Assignment Models

The *name* attribute is the simplest translational assignment model, which is simply one to one value assignment. The cleaned agent data (Twitter) has the attribute *usr.name* which represents the Twitter user's display name on the website. While the data is free of bots, which usually constitutes 15% of the twitter population [123], Twitter does not enforce a strict account naming policy. Since Twitter has been a popular platform for anonymous users to express their opinions freely, it is expected that the percentage of the names provided in the data is not entirely truthful. As most Twitter user profile studies focus on bots and spam accounts, the limitation is that it is a challenging task to provide such percentages robustly.

The *race* attribute provides important information about an agent's characteristics because it is a known fact that people with the same or similar races tend to live nearby to each other making cities segregated [9]. Thus, this attribute helps to capture such a value empirically. Following Mislove et al. [66] and Luo et al. [124], the idea is to generate a look-up list of common last names and their corresponding race distribution. The US Census Bureau's public datasets serve such information for last names that are used at least by 100 people [125] which makes the total number of lines 151,671. In this study, census data for the year 2000 is used because it has a better potential to cover a wider range of Twitter users. Races used in this dataset are *Black*, *White*, *Asian*, and *Hispanic*. The one with the highest percentage is selected. When the last name is not able to bring any results, the user is randomly appointed into a race proportional to US Census race values based on the home location of the user.

The *gender* attribute is also important especially when gender-specific differences play a role in mobility scenarios. Here potential gender values are *male* and *female* because official

statistics only consider these two genders. To identify gender, the first name of people has been reported to be useful [66, 124]. In this case, the first name is obtained from the name attribute and compared with a list of names that were given to US citizens between 2002 and 1965 covering the birth names of people who were between 13 and 50 ages old during the data collection. The name list is compiled from the US Social Security Administration baby names database [126]. Simply, all unique names and their gender counts are combined. While many names are highly associated with one gender, the database contains unisex names – names given to both males and females. To break the tie, more commonly used gender is selected. When the first name is not able to bring any results, the user is randomly appointed into a gender proportional to US Census race values based on the home location of the user.

Testing of the translational models was quite straightforward. Since both datasets were already structured and small, checking noisy or missing data returned no results. In terms of context-specific checks, some names were tested, and the results appeared plausible.

5.3.2 Algorithmic Assignment Model

The *home location* attribute is the most complicated attribute among all because it is not readily available in the agent data. The process of generating home location from data is published by the author in Kavak et al. [82]. The home is one of the most important hubs for people when transitioning from one daily activity to another [53]. Moreover, home is one of the most frequently visited and stayed locations in one's life [60]. By inferring a person's home located close enough, it is possible to gather significant information such as person's financial situation, access to public and private services, and even exposure to worse environmental

conditions [127]. However, it is a challenging task to predict home location from social media data due to the sparsity of location footprints.

There is a wide range of methods to infer home location from social media. These methods can be categorized based on the granularity of their inference. Mahmud et al. [71], for instance, develop a statistical classifier trained with the words seen in geo-tagged tweets. While their approach aims to work on all Twitter users, the accuracy of their classifier reaches 0.4 score with up to 10 miles of error. Ryoo and Moon [70] predict the home location of Twitter users based on the projection of the words compared with geo-tagged user's words. Their approach reaches 0.56 accuracy with up to 10-kilometer of error distance. Pontes et al. [128] develop a classifier which achieves an accuracy score of 0.6 with up to 6 kilometers of error. While insightful, the spatial resolution of these studies is too low to make estimations about the user's other inferable properties (e.g., accessibility). Hu et al. [72], by contrast, performs home-location prediction of Twitter users with a resolution of 100 meters. While their approach reaches 0.75 accuracy, its applicable user scope is only 50-55%. The common point of all these models is on their aim to predict home location based on the given input set which is past location footprints.

The goal here is to develop a similar home location prediction mechanism or classifier which outperforms the state-of-the-art. The process starts building the classifier with six human mobility features identified to be important in Hu et al. [72]. Furthermore, the classifier advances Hu et al. [72]'s work by (1) adding two additional mobility features and (2) exploring the effect of data collection length, tweeting rate, and the number of tweets on the classifier accuracy. All these are done within the scope of the algorithmic attribute value assignment process involving the *creation of the ground truth dataset* and *training of the classifier*.

The ground truth dataset is a subset of the same Twitter dataset but with people whose home location is known. This subset is constructed with a portion of Twitter users who are actively tweeting from Chicago, Illinois involving ≈7.78 million location footprints from 92,296 Twitter users. The aim is to identify tweets similar to 'I'm finally home' which likely indicates that the user is sending the tweet from home. However, just having the word 'home' in the tweet does not ensure that the user is actually at home. Thus, a secondary keyword list identified that might appear in home-related tweets. This secondary keyword list[3] is made with help from a word cloud of home tweets. As a result, 13,279 tweets are filtered containing the word home and at least one of the secondary home presence-related keywords and stored in a local database.

Next task is to label these filtered tweets to categorize them into two: *from home* and *not from home*. For this purpose, a web application for labeling home-related tweets was developed. The web application simply displays a tweet from the dataset and asks the user to choose a label: *from home*, *not from home*, or *unsure*. Each question is displayed up to three times randomly while precedence is given to the ones that already have an answer. In total, 14,076 responses were received for 4,679 questions filled by colleagues at Old Dominion University. For tweets with three responses agree that the user is sending the message from home is taken as the evidence. Approximately 38% of tweets from 1,268 users satisfy this criterion. Now that there are over one thousand users with ground truth home location labels, it is possible to create a training/test set for the home location classifier. One last step is to use the entire location footprints of the users in the ground truth dataset and label other tweets based on their previously identified *Place ID* and create eight features including the class label (*home* or *not home*). This entire ground truth generation process is summarized in Fig. 20.

[3] Secondary keywords: shower, sofa, TV, sleep, nap, bed, alone, watch, night, sweet, stay, finally, tonight, arrived

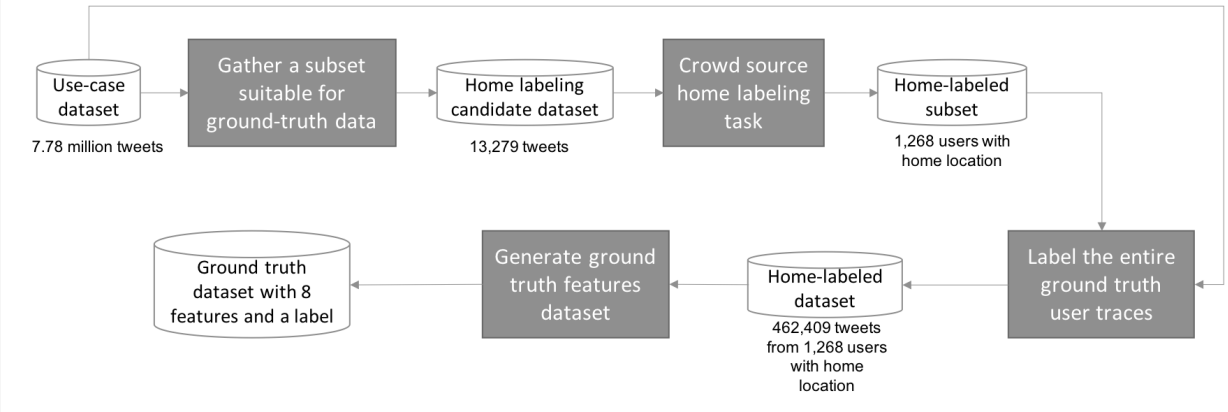


Fig. 20. Ground-truth data generation process.

The eight features are calculated for each unique location (based on place label ID) for each user. Here, check-in ratio is used instead of the actual number of check-ins to keep all features normalized between 0 and 1. These eight features are summarized below.

- **Check-in Ratio (CR):** Check-in Ratio is the measure of the number of check-ins of a user at a location against total check-ins in all locations. A common assumption made about the home location is that it is the most visited location [128-130] This certainly holds for continuously captured data [131] and is found to be an important feature for sparsely shared social media data [72].
- **Check-in Ratio during Midnight (MR):** Midnight check-in ratio looks at all midnight check-ins (12:00 am - 07:00 am) of a user and calculates the ratio of midnight check-ins per visited location. While the home is usually the last place before a person becomes stationary [132], social media users share their locations during midnight while they are outside their home as well [72].

- **Check-in Ratio of the Last Destination of a Day (EDR):** This feature captures the last destination of the day which is found to be important to predict home location [132]. All last check-ins of days are identified, and the ratio per location using tweets shared between 05:00 pm and 03:00 am are calculated.
- **Check-in Ratio of the Last Destination of a Day with Inactive Midnight (EIDR):** This feature is very similar to the EDR feature but ignores days when a user shares tweets during midnight. This feature captures the assumption that the user ends the day at home and do not spend time at night outside [72].
- **PageRank (PR, RPR):** PageRank [133] is a well-known graph measure to show the importance of nodes based on the number of influential edges to them and is a decent predictor for home location. Unique places are represented as nodes and transition between places as edges based on consequent check-ins on the same day until 3 am. Both weighted PageRank and reverse weighted PageRank scores are calculated. Weights are based on the number of transitions between nodes, and reverse PageRank is captured by swapping source-destination pair.
- **Land Use Pattern (LU):** Land use patterns are designed by local governments to regulate the consumption of space by inhabitants. The assumption here is that the home location of a person has to be at a residential area. There are many codes describing different uses of the land. For this study, only *residential* and *non-residential* are used.
- **Kilometer Distance from Most Checked-in Location (KM):** Previous studies [128-130] showed the importance of most checked-in place when it comes to home location prediction. However, previous studies report that most checked-in location is not as effective for social

media users. It can be argued that the distance to the most checked-in location might be worth investigating [53].

Next, the home location classifier is trained with the mobility feature set generated in the previous step; it can predict the home location for a given test set. According to [72], the check-in ratio-related features contribute to the prediction of home location. Therefore, check-in ratio feature is plotted in Fig. 21 with respect to other seven features and mark home and non-home locations. A visual inspection reveals a linearly separable home and non-home features in all plots, especially when they are combined, which is consistent with [72].

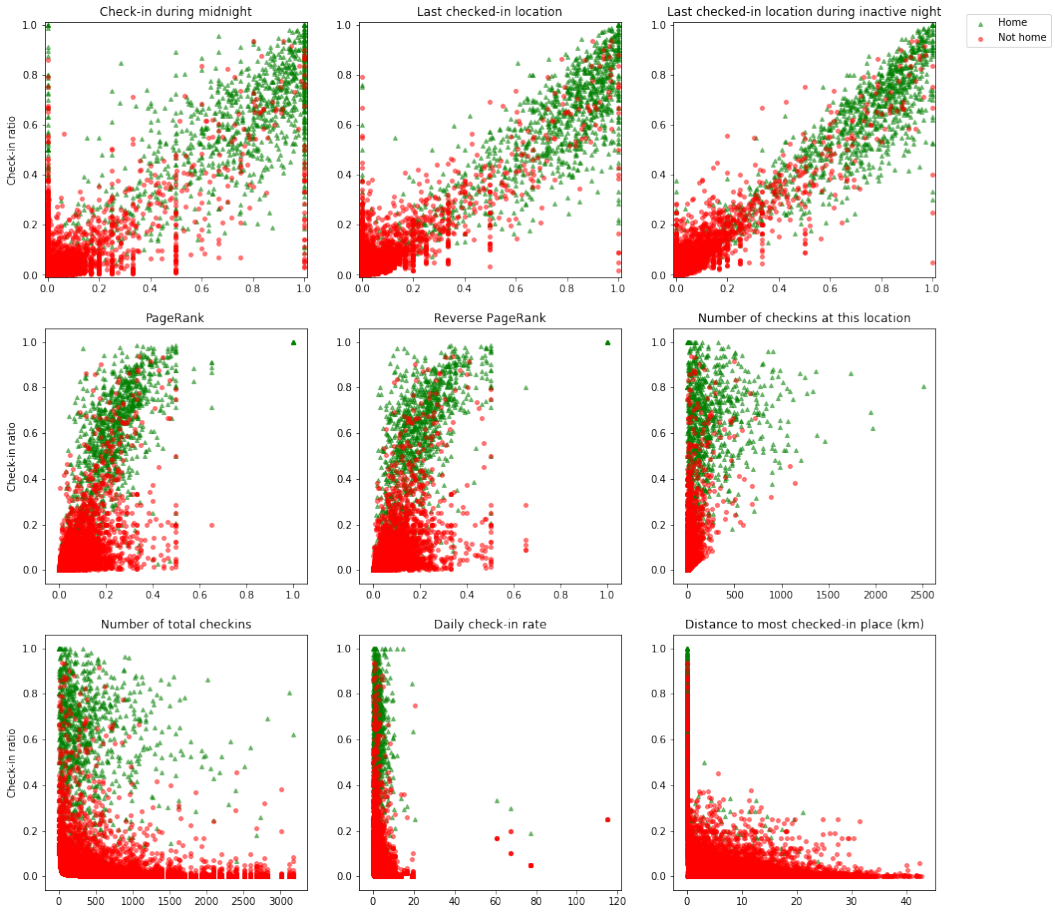


Fig. 21. Scatter plot of features against check-in ratio.

SVM (Support Vector Machines) is used as the classifier. It is a robust technique for two-group classification problems and is reported to be effective in the home location prediction [72]. SVM works by creating an optimally placed decision boundary to separate elements of two classes with maximum margin [134]. When the SVM uses a linear kernel, the decision boundary is a hyperplane. It can be computationally costly to train an SVM classifier due to the involvement of numerical optimizations, but it becomes very intuitive to use a classifier. For instance, when an SVM classifier is trained with a linear kernel, the classification problem is turned into a simple calculation of $c = W \cdot X + b$. When c is non-negative, then it indicates one class, and when it is negative, it indicates the other class. Here, W is the weight vector for the hyperplane, b is the intercept parameter, and X is the input whose class is investigated.

Finally, the testing and evaluation of the classifier take place which involves creating several different scenarios to test the performance of the classifier. For each scenario, a repeated 5-fold cross validation is applied. The ground-truth users are split to five equally distributed groups and the classifier is trained with four groups and tested with the remaining one. This continues until all groups are used in training four times and in testing one time. This minimizes the bias by ensuring all points are used in training and testing equally. This procedure is repeated five times, and the user list is randomly shuffled in each occurrence. In total, each evaluation takes 25 runs. In each fold, home location is predicted at the user level by calculating the SVM score for each unique location label ID and pick the one with the highest score. Then, the average accuracy of predicting home location is captured.

In scoring the performance, the accuracy of each mobility feature is evaluated separately and paired with each other. As a single feature, *End of Day Ratio* (EDR) has the highest accuracy with 0.791 while general *Check-in Ratio* (CR) and *End of Inactive Day Ratio* (EIDR) are just

marginally lower. *Midnight Ratio* (MR) is slightly lower with 0.756 followed by *PageRank* (PR) and *Reverse PageRank* (RPR) scores around 0.714 and 0.638 respectively. Finally, *Land Use* (LU) feature has an accuracy score of 0.464 and *Kilometer Distance to Most Visited Location* (KM) feature just performs the worst with the accuracy score of 0.151.

When feature combinations are considered, the accuracy of best ranking three features (EDR, CR, EIDR) only marginally change and have SVM weights (W) that correlate when combined with other features. It is also found that the next two graph-based features' (PR, RPR) accuracy increase by ≈8-14% when combined with the worst individually performing feature (KM) that provides negative weight to the classifier. Furthermore, the combined features' (two or more combinations) accuracy scores were around 0.795 which is very close to the highest performing single feature (EDR). This score performs better than the literature including [72] who reports the same accuracy with the applicable scope of 30-40% whereas against the applicable instance which is 100%. To check the robustness of this result, the change of accuracy based on data collection length is reported.

The question '*how does data collection length affect the accuracy'* is important because data collection is a lengthy and costly process. Knowing the optimum data collection time will help develop smart data collection strategies. The first subsets of the data with different lengths are created (see Appendix C) to answer the question. In these subsets, the start dates are kept the same as with the original dataset. Fig. 22 shows prediction accuracy score concerning data collection length for top five single features and combined features.

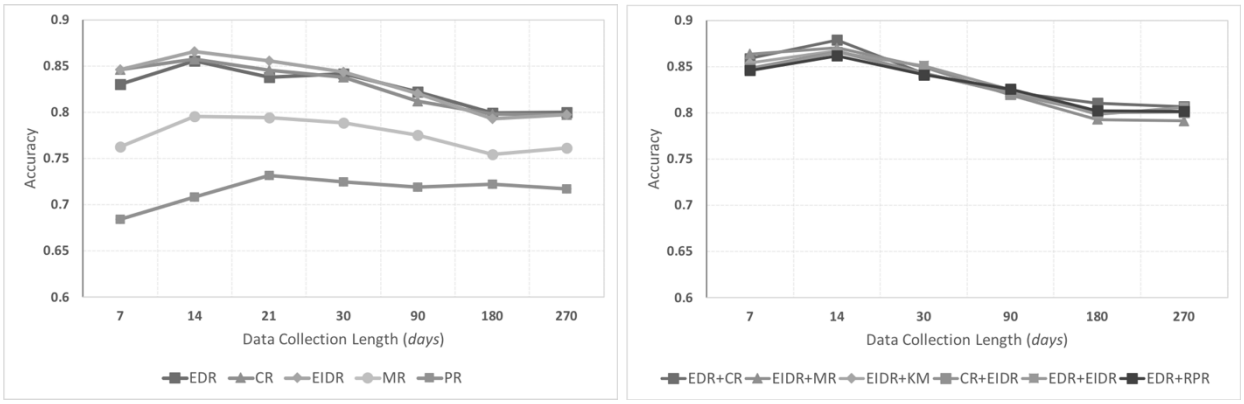


Fig. 22. Accuracy scores based on data collection length. Single best performing features are on the left and combined best performing features are on the right.

According to these results, the length of data collection almost inversely affects the classifier accuracy. It is noticed that top performing single and combined features reach their best accuracy in just fourteen days (except for PR feature that reaches its peak in 21 days). For single features, there is a slight difference in the rankings of the first three features although they are still very close. For combined features, their accuracy is slightly better than the single features especially when data collection length is 30 days or lower. Subsets in Fig. 22 start on the same day making a potential bias. New subsets with different data collection dates are created to investigate it. A variation between different instances of the same length of data is created when the starting date is different. In this case, the early timestamped data instances have better accuracy; in fact, up to 10% drop in best accuracy in some certain instances are observed. When the number of location footprints used in each instance, it seems to follow the accuracy change trends which is discussed below.

In order to understand the influence of the number of footprints, two measures: *number of tweets per user* (G_n) and *user tweeting rate* (G_r) were investigated. The number of tweets per user will capture the direct relationship between footprint size and classifier accuracy. However,

it is also the case that Twitter users do not tweet homogeneously so there is a need for a common denominator: tweeting rate – the average number of tweets per day captured as $\frac{number\ of\ tweets}{number\ of\ days\ between\ first\ and\ last\ tweet}$. Four groups of users were created for each measure with distributing each group similar number of users (see Appendix C).

Fig. 23 shows the average accuracy for the top five performing single and combined features. It is noticed that the users with a low number of tweets or the low tweeting activity perform up to 0.7 average accuracy while single features perform 0.6 on average. Also, graph-based features alone (PR, RPR) perform very poorly for the first group of users because the number of check-ins needed to capture a proper graph is not present for those users.

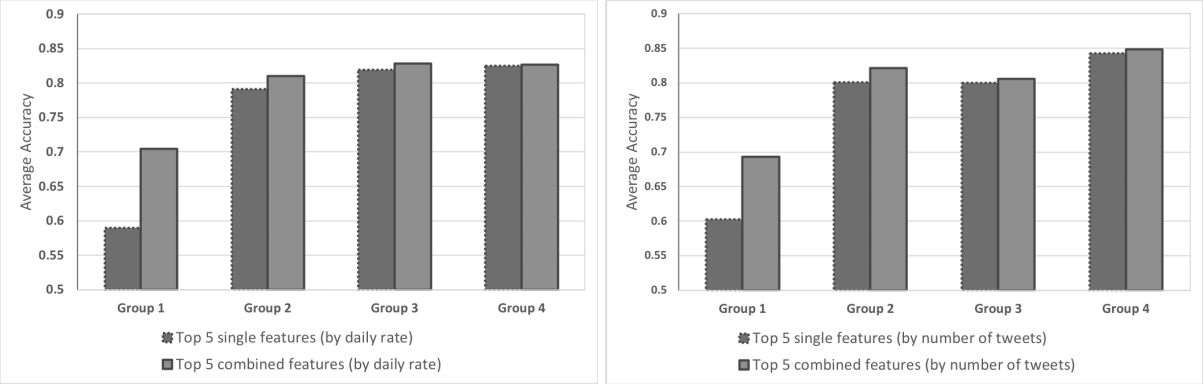


Fig. 23. The change of accuracy based on number of tweets and their tweeting rate.

The accuracy reached by Group 2 is the optimum point in this study; the higher numbers of tweets or higher tweeting rates do not increase the accuracy significantly. In this case, it can be reported that 0.6 to 1.4 daily geo-tagged tweeting activity or 75 to 225 number of geo-tagged tweets per individual are enough to predict one's home location with over 0.8 accuracy. With

these results, the mechanism that generates features using given footprints and the trained SVM classifier combined to be the home location value generation model.

5.4 Behavior Model Creation Process

5.4.1 Identifying Suitable Learning Models

Recall that identifying suitable learning models is contingent on the characteristics of the behavioral data to be used. In this case, geo-located Twitter data is evaluated to summarize its characteristics as seen in Table 8. In this table, two equivalent decision variables are identified: one representing location as a *latitude-longitude pair* and the other one representing location as a *categoric* value. The idea is to evaluate both options to pick the one with more useful representation. The number of data points per agent range between 84 (comes from 84-hour slot) and more than 10 thousand. The number of variables is seven while the contribution of the variables to behavior learning is yet to be seen. In terms of data properties, mobility data is proved to be time-dependent [60, 107]. The uncertainty of mobility data was once reported as a time-dependent measure [64] which should be confirmed.

TABLE 8

MOVEMENT BEHAVIOR DATA SUMMARY

	Values
D = Decision Variable(s) Type	Numeric (Continuous): Location (latitude, longitude) Categorical (Nominal): Place Id
N = Number of Data Points per Agent	[84, 10k+]
V = Number of Variables	hour of the day, day of the week, weekday/weekend, previous location latitude, previous location longitude, previous location id, and mood.
Data Properties	Time-dependent, Mixed Uncertainty

Since uncertainty of mobility behavior has the potential to change over time, it needs to be tested through an entropy measure that also changes over time. In this respect, a measure called *sliding normalized entropy* is defined as follows. $V = \{v_1, v_2, ..., v_n\}$ is location visit history of a user where v_i represents a single visit with a timestamp and unique location identifier ($1 \leq i \leq n$). Let $h: \{h \in Z \mid 0 \leq h < 24\}$ be the hour of the day and $V^h = (v_1^h, v_2^h, ..., v_k^h)$ be a k-tuple visits seen in the hour h within the location visit history V where v_j^h only contains a unique location identifier ($1 \leq j \leq k$). Let $V^{hu} = (v_1^{hu}, v_2^{hu}, ..., v_m^{hu})$ is a m-tuple of unique locations visited in the hour h. Hourly entropy is calculated using Shannon's Entropy measure [135] $H(X_i) = -\sum_{j=1}^{n} p_j \log_b p_j$ where probability of visiting each unique location (p_j) are calculated using V^h. The calculated entropy value is then divided by the maximum possible entropy value for normalization. The maximum possible entropy value is $\log(m)$ recalling that m is the number of unique locations visited in the hour h. This calculation gives us a value between zero and one where zero represents the lowest entropy (i.e., lowest uncertainty) and one represents the highest entropy (i.e., highest uncertainty). This entropy calculation is applied to all 24 hours for weekdays and weekends by sliding the hour value h from 0 to 23. If there is no location visit found in an hour, entropy is then given as -1.

Fig. 24 shows the sliding entropy measure applied to the users for calculating their weekday and weekend entropies. One noticeable trend is the absence of location visit data (results closer to -1) from 1 a.m. to 6 a.m. for weekdays and 1 a.m. to 8 p.m. for weekends. This is the timeframe in which people usually sleep, so it is reasonable. During the day, the entropy reaches and stays around 0.70 on weekdays and 0.55 on weekends. These results indicate that the movement behavior of people has *mixed uncertainty* by having different entropy values at

different times and weekdays/weekends. In other words, there is a need to test learning models that can capture behaviors with low and high uncertainty.

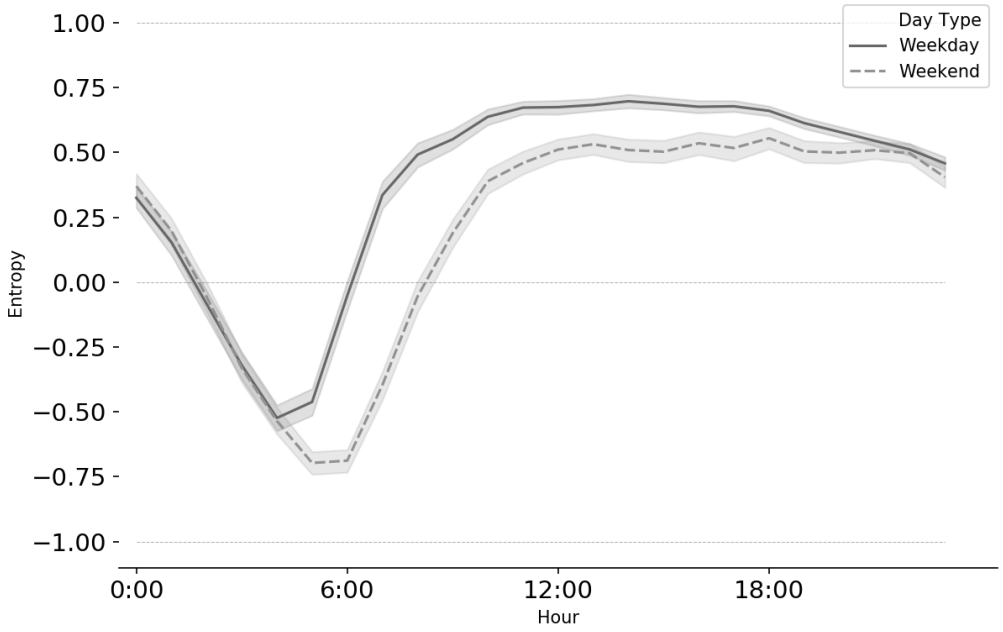


Fig. 24. Entropy measure over time for Twitter users. The lines show the mean value for the weekday and weekend entropy measures with 95% confidence band.

Following the guide for selecting suitable learning models (see Fig. 13), one can pinpoint the right bottom two ends of the table including machine learning classifiers when the nominal decision variable is used and machine learning regressors when continuous decision variables are used. Moreover, assuming mixed uncertainty and time dependency, probabilistic classifiers, and time series models seem to be useful. In this respect, five models are identified to be suitable to test. These models are *Multi-Layer Perceptron* [136], *Support Vector Machines* [134], *Random Forests* [137], *Rudimentary Rule Learning* [138], and a *Time Series Forecasting Model* [139].

Multi-layer perceptron (MLP) is based on the concept of artificial neural networks [136]. That is, multiple layers of artificial neurons (nodes) are organized so that each layer takes input and forwards it to the next layer through connections called *weights*. In each layer, incoming connection signals are summed and fed into an activation function that generates the output of that neuron. The weights of an MLP model is updated based on a training step where the expected model outputs are compared to current model outputs. Once trained, an MLP model can be used to predict outcomes.

Support Vector Machines (SVM) is based on the concept of finding optimally placed hyperplanes that maximally separates given two classes of objects [134]. While the original application was intended for separating linearly separable classes using linear kernels, later non-linear kernels (e.g., polynomial) are introduced to separate more complex classes.

Random Forest (RF) is an ensemble-based machine learning technique that uses multiple decision trees to predict outcomes [137]. Each tree in the forest is selected based on a voting mechanism. The idea is that different parts of data is learned with different trees in the forest. Rudimentary Rule Learning algorithm [138] is selected as a suitable model because it is parameter-less and extremely simple for training and testing.

The Rudimentary Rule Learning (1R) is normally applied to classification problems by making a simple if-else rule on one feature that maximizes the prediction accuracy (e.g., similar to a single decision tree with depth=1). In this case, it is applied to a continuous value prediction which involves choosing one value according to minimum, maximum, and average values in the training data.

A *Time Series Forecasting Model* [139] is also selected because it can uniquely capture uncertainty, time-dependency, and cyclic behavior in the mobility data. To understand temporal

and periodic movement behavior, transformed fields (*hour of the day*, *day of the week*, and *weekday/weekend*), and *previous place* are considered.

Since it is a costly process to test both categorical and continuous decision variables, a decision needs to be made regarding the choice. Continuous (i.e., latitude-longitude coordinate-based) decision variable allows one to extrapolate the space-time relationship whereas the discrete representation of places (i.e., place ID) is only limited with places seen in the check-in history. For instance, if one is usually at home at 8:00 am and at work at 09:00 am, space representation should capture that the person is somewhere in between the two places at 8:30 am; however, discrete representation does not have the capability to capture such extrapolation. That is why the discrete representation of places is not captured within the main body of the dissertation; instead, it is given in Appendix A.

To reduce the cost of running an extensive number of experiments on the entire data, a two-level evaluation approach is adopted to test the learning model candidates. First, computationally costly techniques (MLP, SVM, and RF) are separately evaluated on synthetic mobility data that captures movements of a person in two weeks with perfect detail. This data is used to test these three techniques using the following parameter and data variable combinations shown in Table 9. Multi-Layer Perceptron regressor is evaluated with different *alpha* numbers to prevent model overfit, different *learning rates* to test the convergence to the expected outputs, and the different *number of layers and neurons* to test the success of the capturing of patterns in the data. Support vector regressor is tested with different regularization term (*C*) values to prevent model overfit, different *tolerance numbers* to test the convergence to the expected outputs, and *epsilon* parameter that specifies the tolerance of how much error in the output is considered as a penalty. These settings are tested for both linear and non-linear (RBF) kernels

where the non-linear kernel has an additional *gamma* parameter that tests the influence of a particular training instance. Finally, Random Forest Regressor is tested with a different *number of trees* to capture the patterns in the data, different *depth* values to limit overfit. Six different data feature combinations are tested based on different time and previous place considerations.

TABLE 9

PARAMETER AND DATA VARIABLE COMBINATIONS FOR FIRST LEVEL TEST

	Learning Model Parameters	Variables (features)
MLP Regressor	• **Alpha:** 10^{-5}, 10^{-3}, 10^{-1}, 10^{1} • **Learning rate:** 10^{-3}, 10^{-2}, 10^{-1} • **1 hidden layer:** 5, 10, …, 100 • **2 hidden layers:** - *Layer 1:* 5, 15, …, 75 - *Layer 2:* 2, 6, …, 22	• Hour of the day • Hour of the day + weekday/weekend • Hour of the day + day of the week • Hour of the day + previous place coordinates • Hour of the day + weekday/weekend + previous place coordinates • Hour of the day + day of the week + previous place coordinates
Support Vector Regressor (SVR)	• **Linear kernel:** - *C:* 10^{-5}, 10^{-3}, 10^{-1}, 10^{1} - *Tolerance:* 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2} - *Epsilon:* 0.001, 0.01, 0.1, 0.2, 0.3 • **Non-linear (RBF) kernel:** - *C:* 10^{-5}, 10^{-3}, 10^{-1}, 10^{1}, 10^{3} - *Tolerance:* 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2} - *Epsilon:* 0.001, 0.01, 0.1, 0.2, 0.3 - *Gamma:* $\frac{1}{2n}, \frac{1}{n}, \frac{2}{n}$ given n is num. of features	
RF Regressor	• **Number of trees:** 10, 40, 60 • **Maximum depth:** unlimited, 4	

The best performer of the above settings is the RF regressor model. Test details are elaborated in the next section. As the second level of the learning model candidate tests, RF, 1R, and Time Series forecasting models are compared. Unlike the first level of the test, this time the goal is to train and test these models on the real mobility data from Twitter. Table 10 shows the model settings used in this test. Each unique setting is evaluated with 60%, 75%, and 90% of the data.

TABLE 10

PARAMETER AND DATA VARIABLE COMBINATIONS FOR SECOND LEVEL TEST

	Learning Model Parameters	Variables (features)
RF Regressor	• **Number of trees:** 20, 50, 100 • **Maximum depth:** 4, 8, 16	• Hour • Hour, Day • Hour, Weekday/Weekend • Hour, Day, Weekday/Weekend • Hour, Day, Weekday/Weekend, Previous Coordinates • Hour, Day, Weekday/Weekend, Mood • Hour, Day, Mood • Hour, Weekday/Weekend, Mood • Hour, Mood
Time Series Forecast Model	• **Trend flexibility:** 0.005, 0.05, 0.01, 0.5, 5 • **Prior scale for holidays:** 0.05, 5 • **Number of Fourier estimators:** - *Daily:* 3, 7 - *Weekly:* 1, 5, 10	
1R Model		• Hour, Day, Weekday/Weekend, Mood

5.4.2 Testing Learning Model Candidates

When evaluating machine learning models, one can assess the success in capturing the behavioral patterns in the agent data with accuracy or error amount from the actual values. However, it is just one aspect; learning models need to be evaluated against overfitting/underfitting and having a reasonable model training time [93]. To do that, several experiments are created to test the prediction performance of learning models using different parameter value combinations as described in the previous section. This process is called hyper-

parameter optimization [102]. The goals are to find the optimum parameter combination that maximizes the prediction performance and eliminate overfit/underfit.

First level tests:

All the combinations provided in Table 9 are tested with the synthetic mobility data using 5-fold cross-validation. Place representation is provided as a continuous coordinate pair (latitude and longitude) in regressor models. Model performance is evaluated based on the mean distance error (MDE) calculated using the mean of the Haversine formula [140] as below.

$$\frac{1}{n}\sum_{i=1}^{n} 2r \sin^{-1}\left(\sqrt{\sin^2\left(\frac{\hat{y}_{i,lat} - y_{i,lat}}{2}\right) + \cos\left(\hat{y}_{i,lat}\right)\cos\left(y_{i,lat}\right)\ \sin^2\left(\frac{\hat{y}_{i,lon} - y_{i,lon}}{2}\right)}\right)$$

The above formula simply calculates the real-world distance between expected and predicted outputs with the spherical world assumption given that n is the number of samples, y is latitude-longitude coordinate prediction result set from the trained ML algorithm, $\hat{y}$ is expected latitude-longitude coordinate result set, and $r = 6{,}371\ km$ is the radius of the earth. Fig. 25 shows the best MDE of different ML regressor models' results with respect to several features that influence mobility.

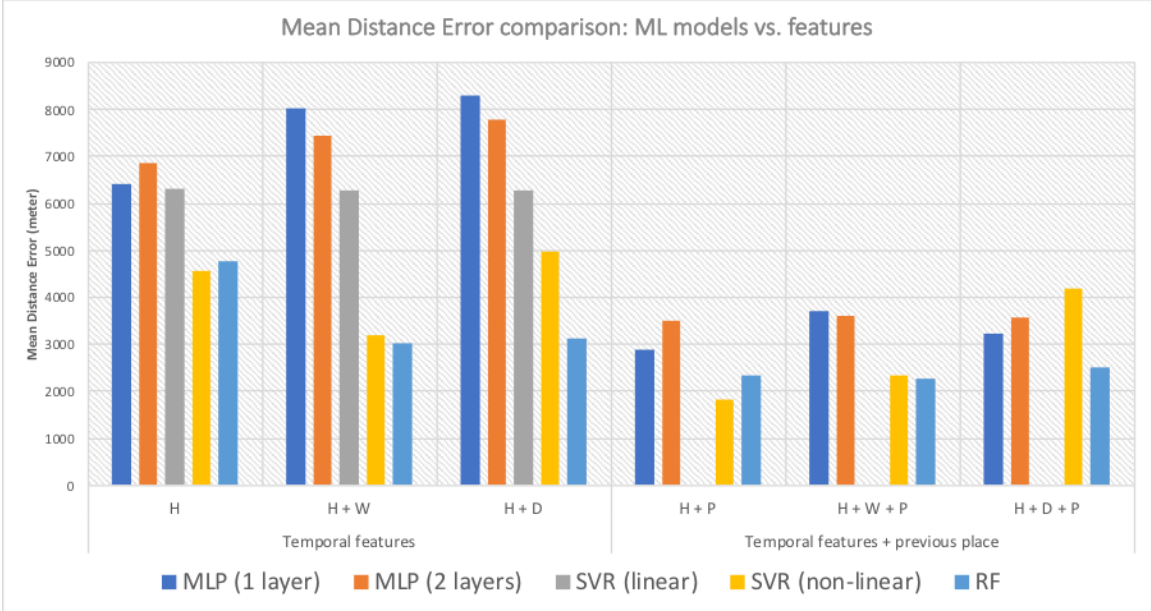


Fig. 25. Best prediction error results based on different feature considerations (H: Hour of the day, W: Weekday/Weekend, D: Day of the week, and P: Previous place).

Temporal-only features in regressors perform poorer than the rest. In temporal features, MDE ranges from 3,021 meters up to 8,310 meters. MLP models perform the worst among all. SVR with a linear kernel performs slightly better than the MLP models. The best performances are seen in SVR with a non-linear kernel and RF models.

When *previous place coordinates* are included in addition to the temporal features, regressor performance is improved with the MDE drops to the range of 1,814 meters to 4,174 meters. MLP models perform mostly poorer than the rest and one or two hidden layers do not make any discernable difference. It is, however, important to note that no prevalent overfit is observed in the MLP models. The SVR model with the linear kernel could not be trained with the mobility data because the Support Vector Machines are very sensitive about the scale of location coordinates (latitude: [-90, 90] and longitude: [-180, 180]). When scale for all features are normalized to the 0-1 interval, the SVR model was able to be trained but this time the

precision of the coordinates is lost and the model no longer able to give any close predictions. Thus, no result of linear kernel SVR is reported here. The best result is taken by a non-linear SVR model with the MDE of 1,814 meters when *hour of the day* and *previous location coordinates* are considered. Interestingly, the addition of other temporal features increases the MDE in non-linear SVR. RF model, however, performs consistently (MDE of ≈2,300 meters) with different features involved. It is observed that the non-linear SVR model overfits the data more while RF overfit is limited. Insights gathered from learning model analyses including the effects of different parameter values on the prediction performance, overfit, and training time are provided as follows.

MLP Regressor performed the worst among all in terms of prediction performance, overfit and training time. SVR performed better in terms of prediction; however, the linear kernel was not able to train for a larger number of inputs while non-linear kernel performed very good in some instances, it was due to the overfit seen in the data. Finally, RF Regressor performed well in a consistent manner despite having only two parameters to tune. Furthermore, training time was minimal, and model overfit was easy to avoid without sacrificing performance when maximum depth is specified with a low number.

Second level tests:

Within the scope of second level tests, several measures are reported. Similar to the first level tests, the main parameter used was *MDE*. Also, *underfit* is captured by considering MDE higher than 4000 meters. This shows that the model performs very poor and not able to generalize patterns from data. *Overfit* is defined as the case where the result does not underfit and training data MDE is at least 50% better (i.e., percentage deviation) than the testing data MDE given that MDE difference between the two is larger than 500 meters. Then, MDE is

evaluated according to the influence of variables (features), the influence of training data, computational overhead, and best performing model parameter that minimizes MDE. Overall, 182,250 runs for the Random Forest model, 135,000 runs for the time series model, and 2,250 runs for the One R model are performed.

According to the above definition, underfit and overfit percentages of each learning model is calculated considering all combinations of test runs. Overall, all three models have at least 40% cases that overfit/underfit as illustrated in Fig. 26. Time series model is the worst one with ≈70% total overfit/underfit. Random Forest model comes second with ≈22% overfit and ≈32% underfit. One R model is the best in terms of overfitting (17%) and underfitting (26%).

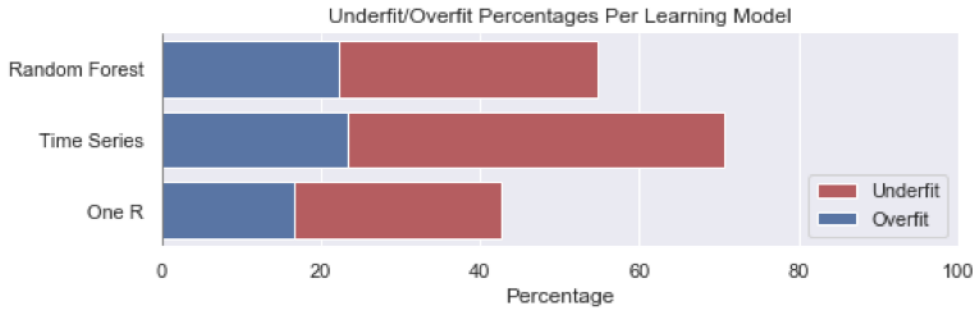


Fig. 26. Percentage of overfit/underfit per learning model.

When investigating the percentage deviation (percentage difference between testing and training MDE values), the results seem very similar to the overfit/underfit values. As depicted in Fig. 27, RF and 1R models perform quite similar deviations peaking at 5-10% while time series model peaks at 30-40% and have more deviation than the other two. This suggests that the time series model is indeed not very helpful in learning mobility patterns from data.

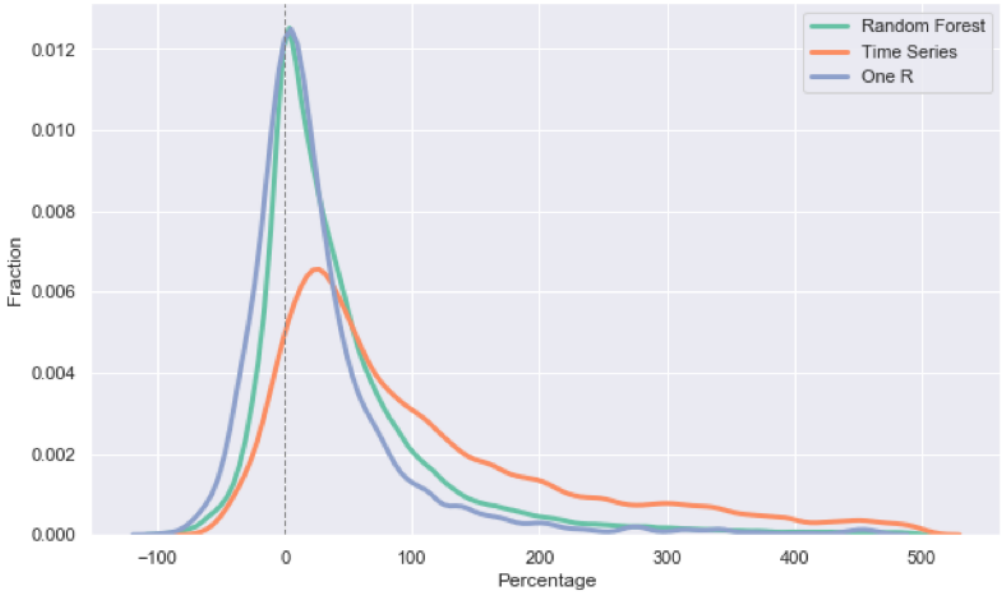


Fig. 27. Percentage deviation from the training set MDE.

There are several learning model-specific reasons why overfit or underfit happens. In the RF model, smaller data size per agent, increased number of features (variables), and maximum depth lead to overfitting and underfitting. In the time series model, increased data decreases overfit/underfit while hours covered, or Fourier order parameters do not make a significant difference. For the 1R model, data size does not affect overfit/underfit. It is likely that it is the model itself why overfits/underfits occurs.

After eliminating overfit and underfit trained models, the next goal is to identify the model with the least training error and best coverage. Fig. 28 shows that RF performs slightly better than the other two in terms of test set MDE. More importantly, RF also performs better coverage – the percentage of users covered after eliminating overfit/underfit models. In this setting, 1R ranks second in terms of test set MDE and coverage while the time series model performs the worst.

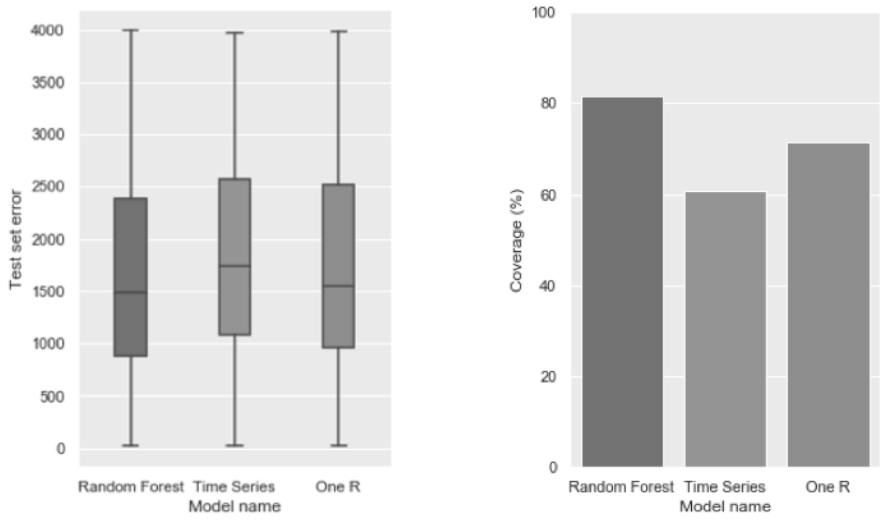


Fig. 28. Test set error (left) and coverage (right) of each learning model.

The final portion of the comparisons between learning models is dedicated to understanding the computational overhead. The above tests are performed on a desktop computer with macOS operating system (Version 10.13.6), 3.8 GHz quad-core Intel Core i5 (Turbo Boost up to 4.2GHz) central processing unit (CPU), and 16 GB of 2400 MHz DDR4 memory. Python programming language version 3.6 [141] is used as the general underlying software system. Tests were run separately but in parallel on the same computer system using a parallelization library called Ray (v. 0.5.3) [142]. When it comes to the implementation of learning models, *sk-learn* [143] is used for implementing Random Forest model, *prophet* [139] is used for time-series forecasting model, and R1 is implemented by the author of this dissertation. Execution times are captured within the code by comparing the time before and after certain tasks. In the evolution of making the use case work, several other technologies are used. Apache Hive [144] is employed to capture the entire mobility data which is over 700 million records. For this study, Apache Hive is built on top of an Apache Hadoop [145] cluster with 14 physical nodes, 224 GB of memory, 112 CPU cores, and 23 TB total disk size. Apache Hive is a big data warehousing solution that

supports SQL-like queries to access data instead of writing map-reduce programs. For use case cities, PostgreSQL is used to maintain the data in locally available servers.

Regarding the tests conducted on different learning models, it is first found that training time for training each agent varies based on the learning model and data size. Fig. 29 (left) shows the distribution of training time using box plots. On average, the Random Forest model performed the best with 0.06 second training time, the One R model performed worse with 0.11 second training time while the Time Series model performed the worst with 0.84 training time. In other words, the Random Forest model training time is 14 times lower than the Time Series model. This detail very important when training, especially millions of agents. Test time results, which shows how fast models can run, provided a similar picture as shown in Fig. 29 (right). Time series model runs the slowest with 0.05 seconds on average. Random Forest model performed tests within 0.00054 seconds whereas the One R model performed very closely with 0.000053 seconds on average. That is, Time Series model runs almost 1000 slower than the other two. Training and testing time comparisons provide significant input to the modeler on the requirements for computational resources.

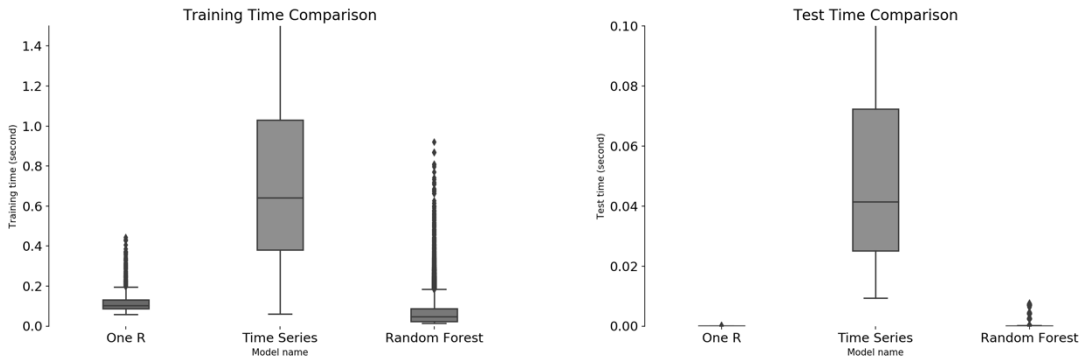


Fig. 29. Training and testing time comparison of learning models. The left figure shows the comparison of training times among three candidate learning models whereas the figure on the

right shows the test time for the same learning models. Both figures are cropped from the top to make them more readable.

According to the above evaluation, the RF model is selected for capturing mobility behavior of agents as it performed the best in terms of performance and computational overhead. Now, one needs to test optimal model feature values and learning model parameters. After extensive checks, it is found that the model performs best when *hour*, *day of the week*, and the *sentiment score* is used for features. When it comes to model parameters, the best performance came from the setting of Number of Trees=50 and Maximum Depth=8.

5.5 Integration Process

This study uses Python programming language, version 3.6 [141], which integrates all components of the model and agents. Python is an object-oriented programming language interpreted during run-time. Some important advantages of Python over other object-oriented languages are the clarity in syntax, flexibility supporting different programming paradigms (e.g., functional programming), and extensibility. Today, there are over 166,000 open source Python packages covering a wide range of needs. Specifically, this dissertation used several data-science packages including *sk-learn* [143], *pandas* [146], *numpy* [147], *matplotlib* [148], and *prophet* [139], to name a few. All the steps including tests are conducted using *sk-learn* toolkit which is an open source machine and statistical modeling library. The combination of all components is summarized in Fig. 30.

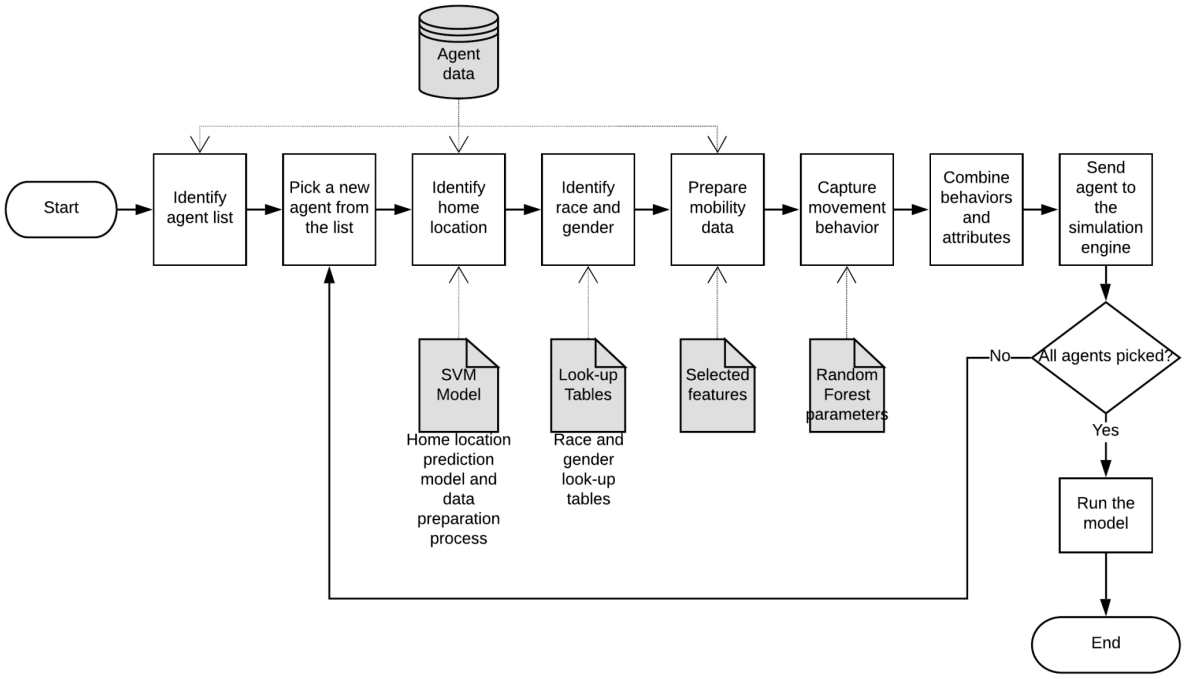


Fig. 30. Summary of the overall process after learning models are tested and identified.

The combined model is simulated using the custom-built Python engine to produce movement patterns of agents. Fig. 31 shows the progression of travels gathered from a sample of 100 agents. A straight line in the figure represents travel from an origin to a destination. Continuously connected lines of the same color represent a movement of a specific agent. Each of the four panels in the figure represents a different time in the simulation. It is inferable from this figure that travels seen in early hours in the simulation are sparse whereas later travels appear to be more convoluted. This visual indicates that the data-driven simulation can produce patterns that show the temporal evolution of new place visits which conforms with empirical evidence from mobility studies [107, 149].

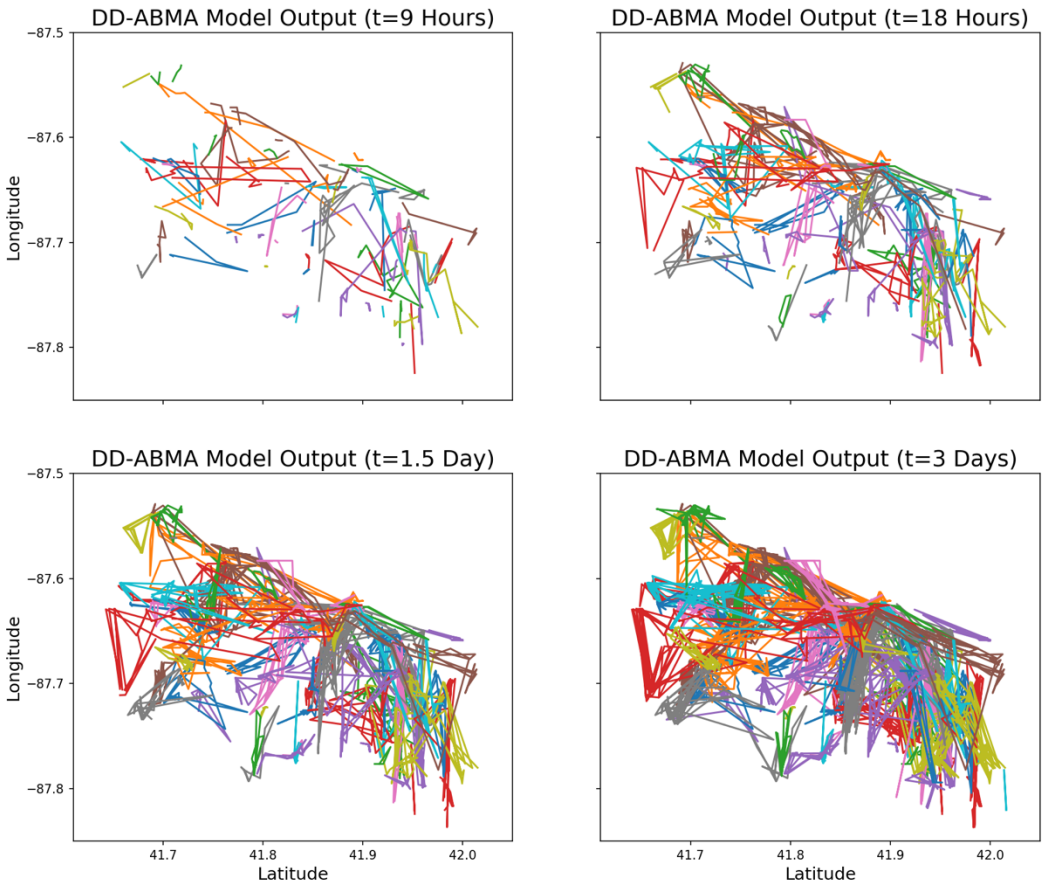


Fig. 31. A visualization of a sample of 100 agent travels. A straight line represents a travel from an origin to a destination with each continuously connected line with the same color indicate a unique individual.

As mentioned in the approach, the integration process is prone to error which requires further tests. Verification is the process in which the simulation code is checked to determine whether it correctly implements the conceptual model [11]. In the process of verification, UML class diagrams and flow diagrams are created. These diagrams help describe the logic of the

model in a visual manner helping find potential bugs. After creating the diagrams and making sure they follow the code description, a complete self-code walkthrough is conducted [103]. After the code walkthrough, each function in the code is unit-tested with a small dataset. This process helped to find software bugs and to correct them.

CHAPTER 6

EVALUATION OF THE APPROACH

This chapter summarizes two efforts regarding the evaluation of the proposed approach. These efforts complement each other in a way that they cover the test of the use case model and data. First, a comparative model is developed following the state-of-the-art data-driven modeling approach by Bell and Mgbemena [25]. The performance of the use case model developed in Chapter 5 is compared with the Bell and Mgbemena [25] model using identical mobility data. This comparative study is presented in section 6.1.

The second effort is aimed at testing the performance of the approach using a publicly available dataset for the same use case model. In this respect, GeoLife [150] dataset is selected which provides location trajectories for a limited number of individuals. The difference from Twitter data is that GeoLife data was captured passively using GPS loggers. Section 6.2 elaborates on this comparison.

6.1 Against a State-of-the-Art Approach

6.1.1 Comparative Model

The Bell and Mgbemena [25] approach propose that agent behaviors at the population level can be captured using a single CART decision-tree. The decision tree is created with agent attribute values and temporal values captured from the real world. Bell and Mgbemena [25] show in their work how they applied this decision-tree-based approach into a problem of customer satisfaction, involving customer churn data from a service provider. In this case, a global decision tree is established to capture entire customer behavior globally.

By following the literal description in their use case model [25], it is not directly suitable to use their approach because the mobility use case is more complex than the binary customer churn/stay behavior. In particular, representing the place as coordinates makes it problematic to generalize as each place has its unique latitude-longitude coordinates. As a potential remedy, an individual's unique locations can be labeled with a value. However, arbitrary labeling of coordinates would make the prediction still impossible. In this respect, more structured labeling is made according to the number of visits made to each unique place. In other words, places with more visits received smaller ranking number starting with one. If two places have an equal number of visits, then they are ranked with the same value. In addition to ranking and labeling places; visit hour, day, and weekday/weekend features are captured per visit along with a person's gender and race. A central single decision tree classifier is established where the decision variable is *rank* and features used in training are *visit hour*, *day*, *weekday/weekend*, *race*, and *gender*. When a decision is made from this classifier, it returns the rank value of the place while corresponding coordinates are taken from the user's check-in history data.

6.1.2 Evaluation

The evaluation is performed between the Bell and Mgbemena [25] approach and the DD-ABMA approach on two dimensions: prediction and plausibility. Both approaches used identical training (70%) and testing (30%) data which is also used in the use case model presented in Chapter 5. For the prediction aspect, both models are trained with the same training data, and future location prediction is performed on the test data. For the plausibility aspect, each agent is given with the task of generating a week-long mobility data. This data is used to measure how plausible the generated location traces are. Both cases are elaborated as follows.

Prediction Performance: Prediction performance is evaluated for future location prediction based on test data. When average distance error is calculated per user, the results span across a broad spectrum. One of the suitable ways of communicating such results is statistical measures and displaying value distributions. In this case, both models performed quite similar in terms of location prediction error (i.e., mean distance error between predicted and actual locations). The proposed model (i.e., DD-ABMA or Data-Driven Agent-Based Modeling Approach) scores 2940.18 meters error on average whereas the Bell and Mgbemena [25] model performed 2540.67 meters error on average. That makes the Bell and Mgbemena [25] model 13% better than the proposed DD-ABMA. A similar result is observed in the distribution comparisons shown in Fig. 32. The Bell and Mgbemena [25] model score smaller error in terms of the overall error distribution. Most errors in their model peak around 1200 meters while most errors on the DD-ABMA peak around 1,800 meters. This result indicates that Bell and Mgbemena [25] performs slightly better than the proposed DD-ABMA model.

Fig. 32. Average prediction error distribution between the two models. Distribution curves are smoothed to aid the reader's eye.

Examining the Bell and Mgbemena [25] simulation results reveal that location predictions almost always belong to the two top-ranked places. Most movement predictions are from the highest ranked place. In other words, the model most of the time suggests that the agent is at the most ranked place such as the home. The bias towards highly visited places needs to be further explored through the plausibility test.

Plausibility: Plausibility in this section refers to the soundness of behaviors to the real-world ones. In the plausibility test, both models are executed to generate a week-long trajectory which is then investigated against mobility patterns seen in the real world. One of the well-known measures of populations is the distribution of travel distances. In short, all people's travel distances from place to place are captured at the population level, and its distribution is investigated. Several studies [60, 115, 149, 151] report that population level travel distances follow a Power Law distribution. In this case, the results originating from both models should produce similar patterns. Fig. 33 shows patterns obtained from the DD-ABMA and the Bell and Mgbemena [25] model as well as a Power Law distribution expected from human mobility trace data.

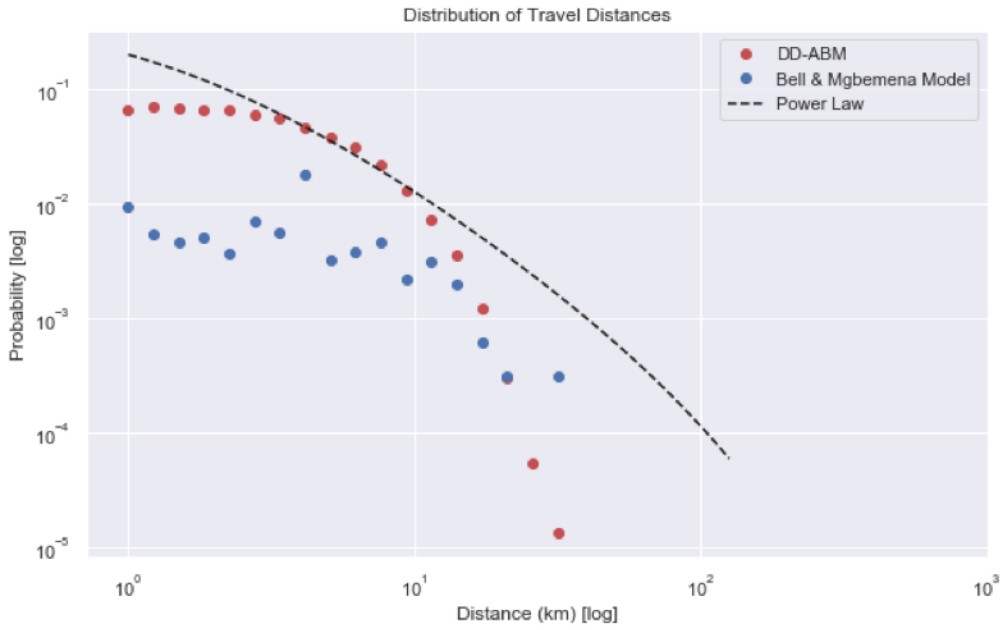


Fig. 33. Distribution of travel distances between the two models.

The result from the DD-ABMA provides a picture which is closer to the ideal distribution whereas the Bell and Mgbemena [25] model appear to be not following the trend. A visualization of the decision tree (which is too large to visualize here) showed the overestimation of high ranked places. That is, the decision tree points to the highest ranked place in most of the leaves and second highest ranked place in some leaves. That is the reason why the Bell and Mgbemena [25] model performed with smaller mean distance error which assumed that a person is always at the highest ranked place (e.g., home). With the idea of confirming the overestimation of the Bell and Mgbemena [25] model, two simulations' output is compared at different simulation times for the same sample of users. Fig. 34 displays travel patterns obtained from simulation outputs at time=9 hours, time=18 hours, and time=3 days. A straight line in the figure represents travel from an origin to a destination whereas continuously connected lines of the same color represent a movement of a specific agent. It is clear from simulation outputs is that the majority of agents

in the Bell and Mgbemena model are contained in the same positions regardless of simulation time. Unlike, the Bell and Mgbemena model, the DD-ABMA model output produces more unique patterns per agent which imply that Random Forest tree structures that are generated from DD-ABMA model capture more detailed movement behavior than those in the Bell and Mgbemena model.

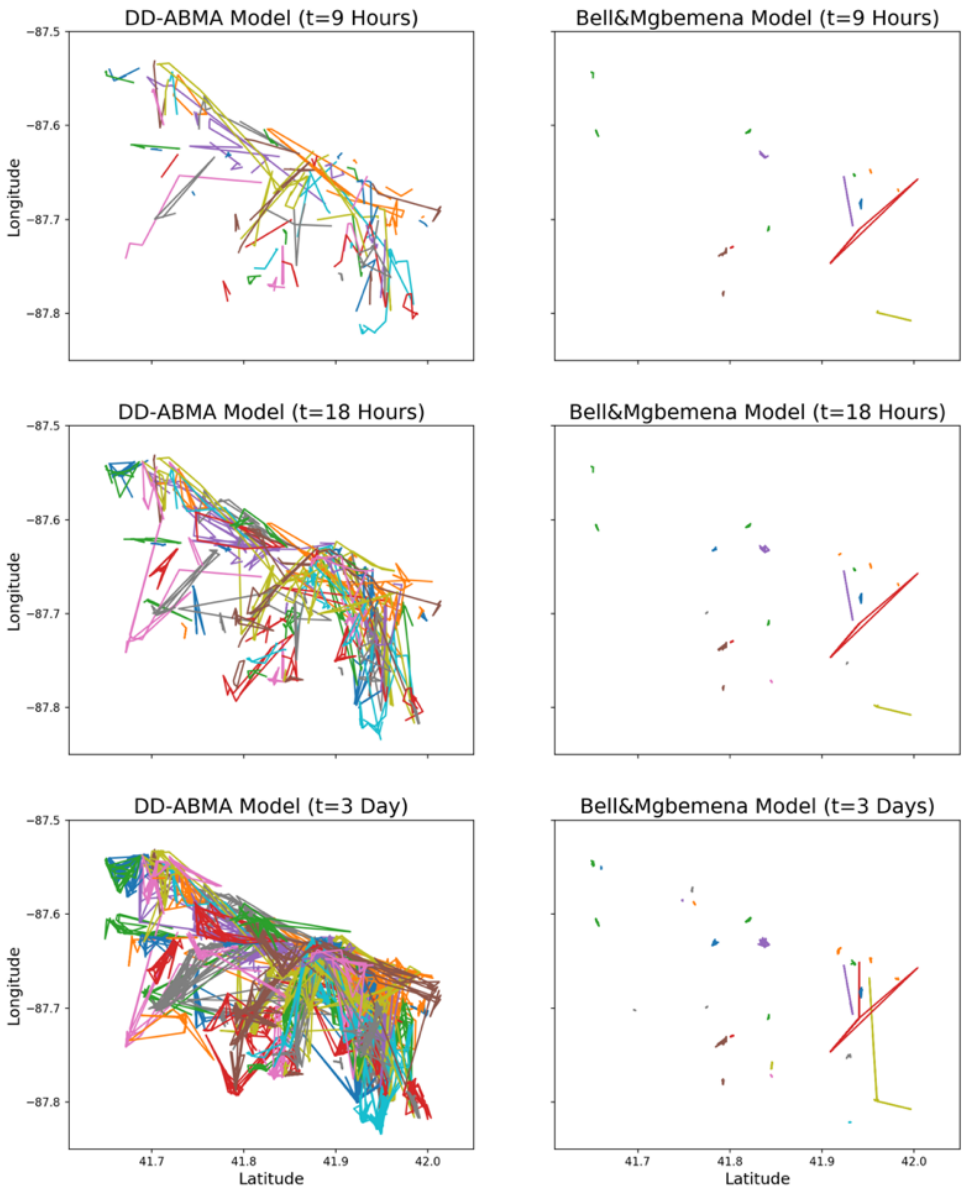


Fig. 34. A visualization of the travel patterns of the same sample of 100 agents.

The deviations that are seen in the DD-ABMA model travel distances in Fig. 33 is explainable. The upper left part of the graph shows many short distance movements captured with lower probabilities. The studies [60, 115, 149, 151] which report power law for travel distances are reported using cell phone data which is lower resolution (up to a few kilometers). That means, it is very likely that short distance travels (smaller than one kilometer) are not captured in those studies and aggregated at a higher probability. For the tail of the power law, the DD-ABMA only captures within-city movements in the use case model which is mostly smaller than 50 kilometers whereas the cell phone-based studies report country-wide movements which can extend the tail.

To sum up, the DD-ABMA mobility model outputs are compared with another model developed by following Bell and Mgbemena [25]'s state-of-the-art data-driven modeling approach. The results show that Bell and Mgbemena [25]'s model performed slightly lower mean distance error on movement predictions compared to those seen in the output of DD-ABMA model. It is shown that the Bell and Mgbemena [25] model overestimated highly-ranked places in its central decision tree. The results are confirmed by comparing the output of both models visually. The idea with the Bell and Mgbemena [25] approach is to develop a single decision tree, populated using data, that allows making different decisions based on agent properties. This concept is very close to the classical rule-based agents which also makes decisions based on attribute values and environmental conditions. In this respect, the advantage of developing heterogeneous agents with DD-ABMA extends beyond producing more heterogeneous agents than Bell and Mgbemena [25]'s approach. It can be argued that the DD-

ABMA approach can potentially produce agent populations that mirror real world with greater detail. Nevertheless, its comparison with the classical ABMs is remained to be studied.

6.2 Using Publicly Available Dataset

6.2.1 Dataset and Data Processing

The GeoLife [150] dataset is made available through a project that passively collected GPS logged trajectories from 182 people for over three years. Notwithstanding the data collection was passive, data coverage and density are different indicating that not everybody used recording devices in all of their trips. Despite this fact, the GeoLife dataset has been extensively used by the spatial data community for mining trajectories [152].

Processing of the GeoLife dataset is different from the geolocated Twitter data. In GeoLife data, user trajectories are densely captured (i.e., in seconds) while the subject was moving. However, the only limited number of trajectories per user is found. From these trajectories, the goal is to understand when and what unique locations that each person visited. That is accomplished through a staypoint detection algorithm proposed by [152]. Staypoints are locations that a person wanders around its vicinity longer than a passing by. In this case, 50 meters distance and 20 minutes are identified as distance threshold and minimum stay time, respectively. These values are reasonable in terms of GPS inaccuracies. Fig. 35 shows an example user trajectory from GeoLife dataset with staypoints identified.

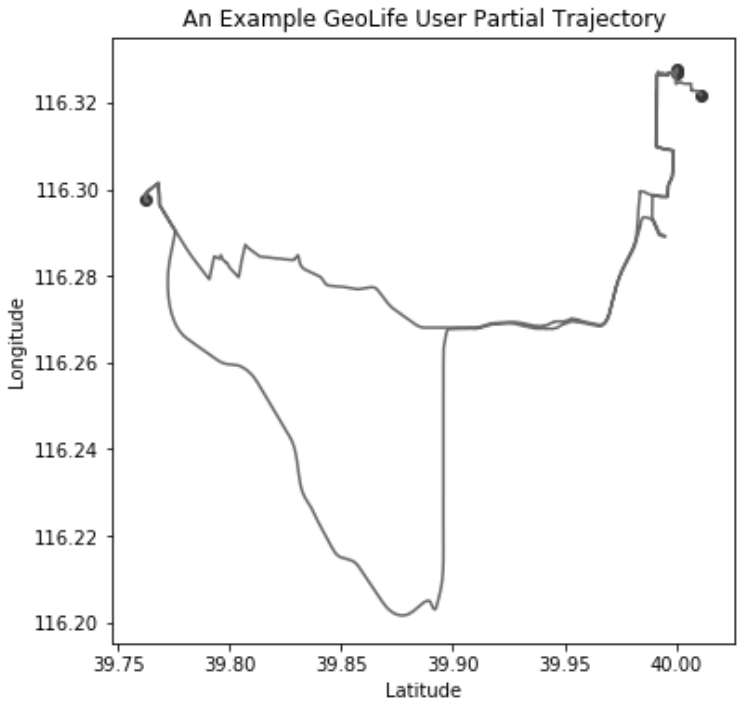


Fig. 35. An example GeoLife user trajectory with three identified staypoints.

One of the other preprocessing steps applied is using the DBSCAN algorithm [122] to identify unique location labels and cleaning outliers. DBSCAN parameters used here are 100 meters for the maximum distance and three for the minimum point parameter. The next is filtering stay points within the bounding box of Shanghai, China. This step ensures urban areas are used in GeoLife data as well. Furthermore, a low number of staypoints in a user's history is an indication that it will be challenging to capture their mobility. In this respect, people with less than 84-hour slots of data are eliminated as in the Twitter case. In the end, only 43 people are left with a decent number of stay points.

6.2.2 Evaluation

The idea here is to compare how the proposed data-driven approach (DD-ABMA) performs in a dataset that is not used in the original use case. In this respect, all 43 users are used in generating the mobility behavior of 43 agents. Their prediction performance (i.e., mean distance error) are recorded. To evaluate these results, an additional 43 users are selected from the Twitter dataset that shows a close resemblance to the 43 GeoLife users in terms of the number of data points and hour coverage. This resemblance will minimize the biases that might occur due to sampling. Fig. 36 shows the distribution comparison of the two datasets and their overlap in terms of distribution.

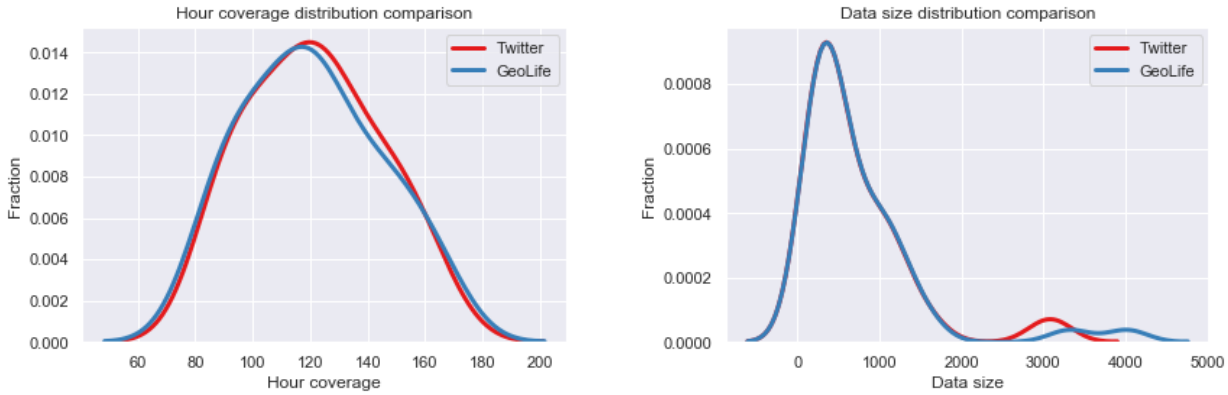


Fig. 36. User characteristic distribution comparison between Twitter and GeoLife datasets (hour coverage: left figure; data size: right figure).

Next, two mobility data tests are conducted on the GeoLife data. These tests were applied to the Twitter data in section 5.2. As mentioned, the first one checks the relationship between the frequency of visits to locations and their rank. Despite the low number of users, visit frequency related patterns were present in data as shown in Fig. 37 (left). The second one is captured with temporality that shows the return probability of an individual to a previously visited location estimated from location footprints. This information is also present in Fig. 37 (right). By having

these characteristics in mobility data, one can expect that it will produce results comparable to the original use case model.

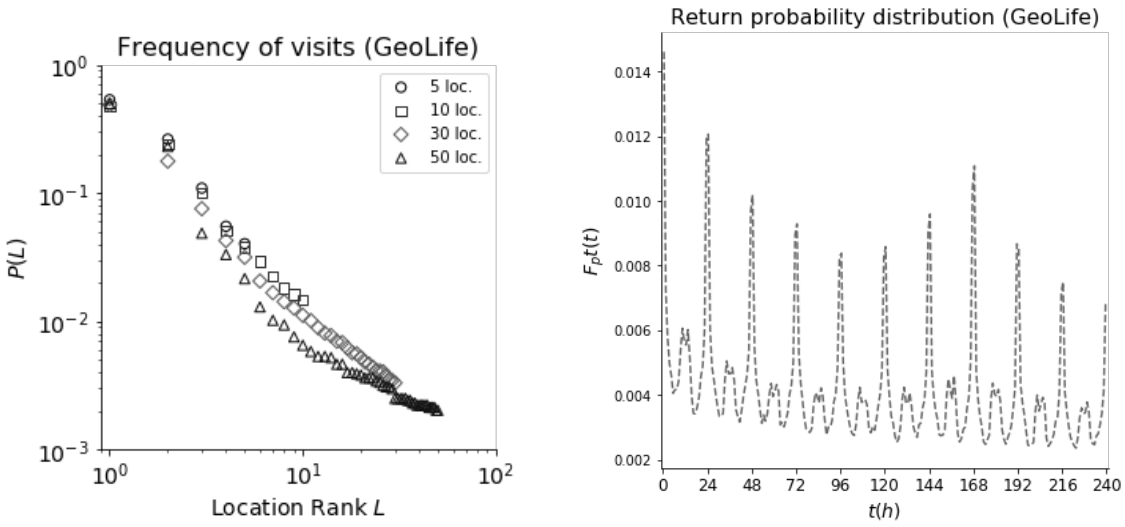


Fig. 37. High-level mobility characteristics of GeoLife dataset. The figure on the left shows a Zipf's plot showing the location visit frequencies of all users based on place rank. The figure on the right shows the return probability to a previously visited place after certain hours passed.

Next, the prediction performance of both models is compared with each other. This is done by running each model 25 times (5-fold repeated cross-validation). Fig. 38 illustrates the error distributions obtained from each model illustrated using a probability-probability (PP) plot. Based on this plot, it can be argued that there is a high possibility for two samples to be generated from the probability distribution. The presence of deviations can very well be related to the sample being used. In other words, the GeoLife data performed as good as the twitter data showing the usability of multiple sources with this proposed approach.

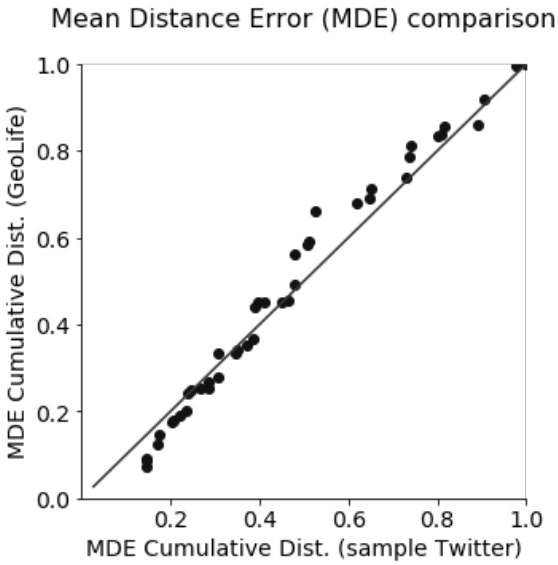


Fig. 38. Prediction performance comparison of models driven by Twitter and GeoLife datasets.

CHAPTER 7

DISCUSSION

This research aimed at filling the gap of a data-driven approach which focuses on individual-level agent behavior generation from data. The process of making such an approach is presented in several chapters. Chapter 3 identified preliminary requirements and building blocks. Chapter 4 presented the proposed approach with an increasing level of details. Chapter 5 illustrated the approach with a human mobility use case model. Finally, chapter 6 evaluated the approach by comparing it with another data-driven approach and using a publicly available dataset. When considered together, this dissertation contributes to the M&S body of knowledge methodologically with the design and modeling of agents from data and practically with introducing data-driven agent-based human mobility simulations.

The body of knowledge produced in this research has implications beyond the contributions mentioned above and needs to address particular challenges to gain a wider acceptance within and outside the M&S community. This chapter discusses DD-ABMA's merits and challenges in simulation replicability, generalization, and re-use and implications of DD-ABMA on emerging domains such as urban science, cybersecurity, and homeland security as well as on disciplines such as AI and data science.

7.1 Replicability and Generalization of DD-ABMA

Scientific endeavors have a broader impact when they are able to reach and adopted by a wider audience. Replication and generalization are two key concepts to achieve such success.

Replication refers to “the confirmation of results and conclusions from one study obtained independently in another,” and it is the scientific gold standard [153]. Thus, the lack of replicability affects confidence in scientific studies. The same applies to simulation models. The M&S community makes efforts to overcome challenges in model replication [154] through standardized text-based model documentation efforts [155] or using more formal standards such as Unified Modeling Language [156], Business Process Modeling Language [157], and the W3C Web Ontology Language [158]. These efforts aim at generating an intermediate meta-model or document which describes the system being modeled. There are other efforts on developing platforms [159, 160] to disseminate simulation code and model details.

Regarding the documentation efforts for replicability, DD-ABMA makes it easier to document the description of the model because the approach requires a minimal conceptual model which can serve as the basis for replicability. On the other hand, confirming simulation results independently from the original study is not as straightforward. One needs to obtain data that produces the reported results. Finding different data does not guarantee to obtain the same results generated from the original study. In this respect, there is a need for developing safe data usage protocols that protect data privacy but at the same time allow replicability. Even if data is shared, one may have challenges in replicability because of using different modeling platforms and the use of different pseudo-random generators.

Generalizability refers to “a theory in a setting different from the one where it was empirically tested and confirmed” [161]. In the scope of this study, generalizability refers to the usage of DD-ABMA by the community outside the presented use case. Considering the four detailed steps of the approach, descriptions of each step are generalized to cover possible different use cases. Nevertheless, there are limited cases where DD-ABMA might not directly be

usable. These limitations originate from *the choice of decision variable types in data* and *the characteristics of the behavior to be modeled*.

DD-ABMA supports *numeric* and *categorical* decision variable data. While many behaviors can be captured using these two, there are still decision variable data in text form. Notably, social media can be seen as a repository of free-text of behavioral data. One option is to process such data and convert it to numeric or categorical values which is a research topic that can be addressed in future studies.

Another limitation is on capturing behavior that is not habitual, meaning behavior that does not repeat often. DD-ABMA can be used to capture behaviors whose pattern can be observed at the individual level. In this respect, rarely observed events such as moving from one place to another might not be captured properly at the individual level. In that respect, capturing behavior at meta-population level might be a potential solution.

7.2 Model Re-Use and the Need for an Integrated Platform

Model re-use is one of the grand challenges of the M&S community in general [162] and agent-based modeling community in particular. Hales et al. [163] argue that agent-based modeling researchers "tend to work in isolation, designing all their models from scratch." This problem can be attributed to the challenge of understanding someone else's code which can be even more challenging if such an open-access model has not gone through verification and validation tests.

In this respect, DD-ABMA provides an advantage to the community because agent behaviors are not explicitly modeled but driven by data which makes it easier to inspect. Furthermore, the modeler can assess the data suitability and prediction performance at the

individual and population level which is usually performed at the population-level only. Ideally, the same model can be used in different cases by using different data. As one might notice in chapter 7, different data sources might require a different preparation process.

Nevertheless, re-using a model built following DD-ABMA can become a challenging task in the absence of integrated platforms that facilitate access and control. Such an integrated platform should provide certain properties to eliminate errors that may originate from handling code and data in a non-automated way. These properties include the following items.

- *Automated data ingestion and flow:* This feature should facilitate a standardized connection between data and other components of the approach. This may require developing data models and harmonized application programming interfaces to communicate with the outside systems.
- *Candidate learning model discovery and standardized learning model interfaces:* The platform should semi-automate the process of matching agent data with candidate learning models. Such a feature would also imply standardized interfaces to a vast number of learning models.
- *Attribute model creation:* There is a need for creating attribute models using data routing and transformation systems. In this way, incoming data can be directly assigned to generate attribute values.
- *Scalability:* The platform should scale in terms of resources in three specific circumstances: 1) testing candidate models which may require millions of runs, 2) training/fitting the actual agents, and 3) running agents for a long time.
- *Accessibility:* An integrated platform is most useful when it is accessible by different communities. In this respect, the platform should be commonly accessible and available through the web and mobile platforms.

7.3 Emerging Application Domains for DD-ABMA

DD-ABMA is illustrated using human mobility use case model. It is true that human mobility is a suitable topic for DD-ABMA because of the opportunity to obtain large behavioral data. However, it is one of the many potential concentrations that DD-ABMA can contribute. Some of the other domains/disciplines are as follows.

- *Urban science:* Urban science is a discipline that uses techniques and methodologies from various disciplines to address challenges of urban areas and populations. Human mobility is one of the many concentrations that the urban science community tackles. Other concentrations also generate vast datasets which can be used in DD-ABMA. Transportation is one of those topics that can make use of DD-ABMA. More specifically, the concept of smart cities, which are known for their vast network of sensor-based data collection, land itself to the DD-ABMA. It is possible to create new smart city tools using the proposed approach to study the multifaceted nature of urban areas in terms of safety, transportation efficiency, tourism, and urban planning.
- *Cybersecurity and homeland security:* Cybersecurity is a critical domain with its economic and social aspects and implications that go beyond individuals and organizations. DD-ABMA can be used in cybersecurity in particular and homeland security in general. One potential use could be on representing cyber attackers and modeling their attacking behavior. This can be achieved with leveraging intrusion detection system logs or traces belonging to attacks. Furthermore, the instruments that attackers use to conceal themselves can be simulated with DD-ABMA. For instance, the internet of things devices such as routers is popular targets for attackers to use as zombie devices [81]. These devices can be simulated in both attacker mode and regular mode as already partly presented in Kavak et al. [81]. Finally, the emerging topic

of the spread of information over networks of people can make use of DD-ABMA. Prevalence of social media and online news sources make it possible to capture large samples of information (both legitimate and fake ones) dissemination on networks. The data-driven approach can model the transition of information between nodes and also capturing impacts of the information on the receiving node.

- *International relations:* Agent-based modeling has been used in studying international relations in the form of interactions/negotiations between countries [164]. In recent decades such relationships have moved to cyberspace in addition to the spatial one. In this respect, DD-ABMA can allow capturing behaviors or reactions of countries in certain events like conflicts. The web contains multifaceted information about global events (e.g., news websites, columns) and their impact on the social fabric (e.g., blogs, social media). If these data sources can be organized in a way that they capture different event types, triggering conditions, and their consequences, it can be used in DD-ABMA to represent countries as data-driven agents. Global event databases such as GDELT [165] can provide a good starting point to capture events and actors.

7.4 Implications on AI, Data Science and Beyond

While the DD-ABMA approach's building blocks come from AI and data science, there are many implications it brings to these two areas and beyond.

The recent success of AI has been possible thanks to the advancements in machine learning techniques such as deep learning [166] and the availability of commonly accessible tools to use them. Despite this success, the practice to train and test machine learning models for a specific purpose remain unchanged. In fact, the focus is given on training single huge models

that can learn a specific task such as image labeling [166]. The challenge that DD-ABMA brings to the AI community is the consideration of a considerable number of learning models that are significantly simpler than the current focus of the community. Furthermore, the challenge stems from using different and potentially diverse datasets for each agent. In such cases, the comparison between potential learning models poses the challenge of finding the optimal performing models while performance values vary by agent. In other words, having multiple dimensions of performance measures that vary by agent warrants new ways to evaluate learning models. In the use case model, this potential challenge was not experienced due to the multi-level test setup and the fact that the difference between candidate learning models was discernable.

While the AI community deals with the use of data in learning models, the data science community focuses on the lifecycle of data and related issues. DD-ABMA and its requirement of individual-level data have implications on the lifecycle of data in terms of data collection and ethical considerations. There is a need to develop secure protocols for obtaining data that belongs to individuals. Crowdsourcing is an emerging area that makes tailored data collection possible and has the potential to implement such protocols. Still, one may have challenges with data collection lengths and the truthfulness of crowdsourcing participants. Furthermore, ethical issues may arise from using personally identifiable data. The data science community can help to address these challenges.

Finally, DD-ABMA can have implications on complex adaptive systems. DD-ABMA can capture the interactions between individuals which may rise to emergent behaviors. An area to address in this respect is on the coordination of multiple agents that learned interactions individually. Currently, such interactions between agents can be implicitly captured based on the

proximity between agents. This limited form of capturing interaction might be a starting point for introducing the DD-ABMS approach to the complex adaptive systems community.

CHAPTER 8

SUMMARY AND FUTURE WORK

A review on agent-based modeling approaches showed that agent behaviors are often made of a set of simple rules that are created according to theoretical knowledge, hypotheses, and idealized assumptions. Moreover, all member of an agent population is mostly considered identical as they often use the same decision rules that are triggered according to agents' attributes and the environment states. Furthermore, it is rather a common practice to use idealized or arbitrary probability distributions in triggering agents' decision rules and initializing attribute values. It is argued in this dissertation that such modeling of agents limits the simulation modeler's ability to exploit the premises of agent-based modeling about representing real-world systems by *establishing a close correspondence between the model and the real-world*.

Operational validation and calibration practices are discussed as a potential remedy to this limitation. It is concluded that both practices are commonly conducted at the population level which does not correctly address the individual correspondence. As another potential solution, current data-driven approaches that allow using empirical data are reviewed. This review showed that none of the current approaches support generating individual agent behaviors using individual-level data.

This dissertation proposed and presented a data-driven agent-based modeling approach that uses individual-level data to generate not only individual agent behaviors but also to obtain/generate individual agent attribute values. This approach is different from the literature with its individual level representation that can facilitate a one-to-one correspondence and have the potential to capture the heterogeneity of the real-world systems.

The proposed approach is composed of four main processes that draw from related literature capturing the general needs of a simulation modeler. *The data preparation* process is composed of several steps and is responsible for cleaning and transforming data and making it available to be used in other processes. This involves many iterative *data checking* processes to make sure that the data is sanitary. *Attribute model creation* process is responsible for developing *translational* and *algorithmic models*. These attribute models are used to obtain or generate attribute values using related data. *Behavior model creation* process deals with guiding the modeler on choosing appropriate *machine learning* and *statistical modeling* techniques to generate agent behaviors from data. It also involves checking which technique is the most suitable one. Lastly, *the integration* process handles the organizing and transforming of data and integrating attribute models and behavior models. A specific verification process is introduced to address the coding issues occurring during the integration process. Once the model is verified, this process generates the population of agents to be used in a simulation engine.

The application of the proposed approach is illustrated with a mobility use case model that aims to predict individual level movements in an urban area. The model utilizes millions of individual-level location footprints collected from Twitter social media platform for about ten months continuously. In this use case model, attribute value obtaining/generation process involved both translational and algorithmic assignments that capture agent *name*, *gender*, *race*, and *home location*. The behavior generation process is established by testing a large number of configurations using several statistical and machine learning techniques. In the end, the model that uses the Random Forest model to generate behavior can successfully simulate the movement of individuals.

The proposed approach is further evaluated by comparing it with state-of-the-art data-driven agent-based modeling approach from the literature. This comparative approach also addressed behavior learning from data, but the granularity they were interested in was at the population level CART decision trees. A use case model, following this comparative approach, is developed with the same mobility data that is used by the proposed approach. The results were compared with two aspects. In terms of prediction accuracy at the individual level, the proposed model performed comparable but slightly higher error from the comparative model. However, when it comes to the plausibility of the output to represent human mobility, the proposed model outperformed the comparative one by generating movement patterns that are closer to those seen in real urban areas. This result suggests that the proposed approach produces results that are close to their real-world counterpart.

This dissertation makes two types of contribution to the M&S body of knowledge: methodology and practice. The approach itself presents a methodological contribution with its presentation which is generalizable to different use cases. More specifically, the individual level behavior generation for ABMs is novel. Furthermore, the use of algorithmic assignment models (e.g., machine learning techniques) for attribute value generation at the individual level is a novel way of initializing attributes as well. Besides, in the evaluation of the approach, both individual and population level validations are conducted. This multi-level validation practice is previously suggested by multiple researchers with a caveat of its feasibility. This dissertation shows that multi-level validation is feasible.

In terms of practice, this dissertation makes a direct contribution to the practice of urban science. It shows the feasibility of developing empirically-grounded mobility behavior rules of agents driven by data. The concept of smart cities, which are known for their vast network of

sensor-based data collection, land itself to the data-driven agent-based modeling approach proposed in this dissertation. It is possible to create new smart city tools using the proposed approach to study the multifaceted nature of urban areas in terms of safety, transportation efficiency, tourism, urban planning, and so on. Urban science is one of the many domains that can benefit from the proposed approach. For instance, the cybersecurity domain can use the proposed approach to create attacker behavior using intrusion detection system logs.

Furthermore, such data-driven attacker agents can be used in cybersecurity simulation scenarios to represent the cascading effects of attacks on networked computer systems. Another potential use is on the information dissemination domain. Prevalence of social media and online news sources make it possible to capture large samples of information (both legitimate and fake ones) dissemination on networks. The data-driven approach can model the transition of information between nodes and also capturing impacts of the information on the receiving node.

While there are many aspects of this study that can start fruitful discussions within and outside the M&S community, there are still several directions to be undertaken for future research. The mobility use case model developed with the proposed approach shows a promising first step towards constructing a fully supported *life simulation* that captures the lifecycle and wide range of preferences of individuals. If different aspects of an individual's life are decomposed into layers, mobility would be a base layer. Capturing a person's mobility gives hints on the potential characteristic values (e.g., socialness, wealth) that belong to other layers. For instance, one's frequent visits to places such as recreational facilities or public meeting locations can indicate the person's socialness value. However, it would require new data sources or different data types to understand such concepts. In this respect, several research questions can be posed to work on layers other than mobility. Example research questions are as follows.

- *"How can we capture using data-driven approach other layers of life including cultural, social, and political?"*

  This is not a trivial research question, nor many studies aim to address such concepts within the M&S community from a data-driven perspective. This dissertation was able to capture people's gender and race which have an effect on social and cultural life. However, there is a need to explore how such attributes shape one's political views and, for instance, their reaction to events happening in the real-world. Since such information and reactions are available on social media platforms, one can use text-based data to model people's, for instance, political views. The challenge, in this case, stems from the unbounded and sometimes informal nature of text-based data captured from the real world. Such a task may require advanced Artificial Intelligence (AI) techniques including natural language understanding and ontologies.

- *"How can we design a new agent architecture that would allow us to have data-driven behaviors and theory-driven rules co-exist in the same model?"*

  Such a research question would be precious for researchers who want to run experiments on the data-driven simulation. In other words, once the layers of the data-driven model are established as posed in the previous research question, one may want to experiment with the model on a context different than regular life patterns. In cases such as emergencies, people depart from their routine behavior. Such non-routine behaviors can be captured using theoretical rules that still uses data-driven attributes. For instance, a person who is at a shopping center during an event of a natural disaster would immediately leave or seek shelter which is different from the routine behavior. Another example would be simulating disease spread scenarios on a data-driven model. Simulation is a suitable approach to study such experiments which are dangerous, unlawful, and unethical to conduct in the real-world.

- *"What are the advantages, challenges, and pitfalls of fully automating the data-driven process?"*

  The proposed approach of this dissertation presented each step separately including, data suitability checks, inspection of generated behaviors, checking of attribute values, and verifying the computer code. Now having a baseline data that is sanitized, it is possible to automate the process. However, what if a new data source is introduced? Since many data sources have biases, it will be a challenge to comfortably put a new data source and assume that the model will work properly. Furthermore, there are limitations to the interpretation of machine learning models' results. Based on just accuracy, for instance, one might be deceived by high scores and become a victim of Type I and Type II errors. Finally, non-causal relationships established through machine learning models may pose challenges. This is a challenge for a broader research community. Currently, agencies including the Defense Advanced Research Projects Agency (DARPA) is actively seeking solutions for such problems under the term "Explainable AI."

The above questions indicate new synergetic research directions for the M&S community. In order to address these questions, the M&S community needs to focus on the intersection of AI and simulations. Since M&S studies mostly involve subject matter experts from application domains, the new AI-Simulation synergy may blossom multi-disciplinary collaborations.

REFERENCES

[1] E. Bonabeau, "Agent-based modeling: Methods and techniques for simulating human systems," *Proceedings of the National Academy of Sciences,* vol. 99, no. suppl 3, pp. 7280-7287, 2002.

[2] N. Gilbert, *Agent-Based Models*. Thousand Oaks, CA, USA: SAGE Publications, 2008.

[3] E. Bruch and J. Atwell, "Agent-Based Models in Empirical Social Research," *Sociological methods & research,* vol. 44, no. 2, pp. 186-221, 2015.

[4] N. Gilbert and K. G. Troitzsch, *Simulation for the Social Scientist*, Second Edition ed. Open University Press, 2005, pp. 308-308.

[5] A. T. Crooks and A. J. Heppenstall, "Introduction to Agent-Based Modelling," A. J. Heppenstall, A. T. Crooks, L. M. See, and M. Batty, Eds.: Springer, Dordrecht, 2012, pp. 1047-1053.

[6] H. Kavak, J. J. Padilla, C. Lynch, and S. Y. Diallo, "Big Data, Agents, and Machine Learning: Towards A Data-Driven Agent-based Modeling Approach," in *Spring Simulation Multi-Conference*, Baltimore, MD, USA, 2018: ACM.

[7] N. Gilbert and T. Balke, "How Do Agents Make Decisions? A Survey Introduction: Purpose & Goals Dimensions of Comparison," *Journal of Artificial Societies and Social Simulation,* vol. 17, 2014.

[8] J. J. Padilla, S. Y. Diallo, H. Kavak, O. Sahin, and B. Nicholson, "Leveraging Social Media Data in Agent-based Simulations," Tampa, FL, 2014, vol. 46, pp. 121-128.

[9] T. C. Schelling, "Dynamic models of segregation," *The Journal of Mathematical Sociology,* vol. 1, no. 2, pp. 143-186, 1971.

[10] J. M. Epstein, "Modeling civil violence: An agent-based computational approach," *Proceedings of the National Academy of Sciences,* vol. 99, no. suppl 3, p. 7243, 2002.

[11] R. G. Sargent, "Verification and validation of simulation models," *Journal of Simulation,* vol. 7, pp. 12-24, 2013.

[12] P. Windrum, G. Fagiolo, and A. Moneta, "Empirical Validation of Agent-Based Models: Alternatives and Prospects," *Journal of Artificial Societies and Social Simulation,* vol. 10, no. 2, 2007.

[13] A. S. Rao and M. P. Georgeff, "BDI Agents : From Theory to Practice," *Practice,* vol. 95, no. Technical Note 56, pp. 312-319, 1995.

[14] J. F. Lehman, J. E. Laird, A. Arbor, and P. S. Rosenbloom, "A gentle introduction to Soar, an architecture for human cognition," *Invitation to cognitive science,* vol. 4, pp. 212-249, 1996.

[15] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin, "An integrated theory of the mind," *Psychological review,* vol. 111, no. 4, p. 1036, 2004.

[16] R. Conte and M. Paolucci, "On agent-based modeling and computational social science," *Frontiers in Psychology,* vol. 5, pp. 668-676, 2014.

[17] F. Klügl, "A validation methodology for agent-based simulations," *Proceedings of the 2008 ACM Symposium on Applied Computing,* pp. 39-43, 2008.

[18] G. Manzo, "The Potential and Limitations of Agent-based Simulations. An introduction," (in En), *Revue française de sociologie,* vol. 55, no. 4, pp. 653-688, 2014.

[19] W. G. Kennedy, "Modelling Human Behaviour in Agent-Based Models," A. J. Heppenstall, A. T. Crooks, L. M. See, and M. Batty, Eds.: Springer, 2012, pp. 167-179.

[20] M. Sajjad, K. Singh, E. Paik, and C.-W. Ahn, "A Data-Driven Approach for Agent-Based Modeling: Simulating the Dynamics of Family Formation," *Journal of Artificial Societies and Social Simulation,* vol. 19, no. 1, p. 9, 2016.

[21] Y. Ge, R. Meng, Z. Cao, X. Qiu, and K. Huang, "Virtual city: An individual-based digital environment for human mobility and interactive behavior," *Simulation,* vol. 90, no. 8, pp. 917-935, 2014.

[22] S. Venkatramanan, B. Lewis, J. Chen, D. Higdon, A. Vullikanti, and M. Marathe, "Using data-driven agent-based models for forecasting emerging infectious diseases," *Epidemics,* 2017/02/22/ 2017.

[23] S. Hassan, J. Pavón, L. Antunes, and N. Gilbert, "Injecting data into agent-based simulation," *Simulating Interacting Agents and Social Phenomena,* pp. 177-191, 2010.

[24] A. Smajgl, D. G. Brown, D. Valbuena, and M. G. A. Huigen, "Empirical characterisation of agent behaviours in socio-ecological systems," *Environmental Modelling & Software,* vol. 26, no. 7, pp. 837-844, 2011.

[25] D. Bell and C. Mgbemena, "Data-driven agent-based exploration of customer behavior," *Simulation,* vol. 94, no. 3, pp. 195-212, 2018.

[26] S. Y. Diallo, "Towards a Formal Theory of Interoperability," Modeling and Simulation, Old Dominion University, 2010.

[27] L. An, "Modeling human decisions in coupled human and natural systems: Review of agent-based models," *Ecological Modelling,* vol. 229, pp. 25-36, 2012.

[28] J. Sterman, *Business Dynamics: Systems Thinking and Modeling for a Complex World*. Irwin/McGraw-Hill, 2000.

[29] E. Bonabeau, "Agent-based modeling: Methods and techniques for simulating human systems," *Proceedings of the National Academy of Sciences,* vol. 99, pp. 7280-7287, 2002.

[30] S. Hassan, J. Pavon, and N. Gilbert, "Injecting Data into Simulation: Can Agent-Based Modelling Learn from Microsimulation," pp. 1-9, 2008.

[31] G. H. Orcutt, "A New Type of Socio-Economic System," *The Review of Economics and Statistics,* vol. 39, no. 2, pp. 116-123, 1957.

[32] R. M. Axelrod, *The Complexity of Cooperation: Agent-based Models of Competition and Collaboration*. Princeton University Press, 1997.

[33] M. J. Wooldridge and N. R. Jennings, "Intelligent agents Theory and practice.pdf," *The knowledge engineering review,* vol. 10, no. 2, pp. 115-152, 1995.

[34] S. Moretti, "Computer Simulation in Sociology: What Contribution?," *Social Science Computer Review,* vol. 20, no. 1, pp. 43-57, 2002.

[35] R. B. Matthews, N. G. Gilbert, A. Roach, J. G. Polhill, and N. M. Gotts, "Agent-based land-use models: a review of applications," *Landscape Ecology,* vol. 22, no. 10, pp. 1447-1459, 2007/12/01 2007.

[36] C. V. M. Cornelius, C. J. Lynch, and R. Gore, "Aging out of crime: exploring the relationship between age and crime with agent based modeling," in *Spring Simulation Conference*, Virginia Beach, Virginia, 2017, no. p.3: Society for Computer Simulation International.

[37] L. Tesfatsion, *Agent-based computational economics : a constructive approach to economic theory*. 2006, pp. 831-880.

[38] I. Benenson, "Agent-Based Modeling: From Individual Residential Choice to Urban Residential Dynamics," in *Spatially integrated social science: Examples in best practice*: Oxford University Press, 2004, pp. 67-95.

[39] A. Collins, M. Petty, D. Vernon-Bido, and S. Sherfey, "A Call to Arms: Standards for Agent-Based Modeling and Simulation," *Journal of Artificial Societies and Social Simulation,* vol. 18, no. 3, p. 12, 2015.

[40] S. Franklin, S. Franklin, A. Graesser, and A. Graesser, "Is it an Agent, or Just a Program?: A Taxonomy for Autonomous Agents," 1997, pp. 21-35.

[41] C. M. Macal and M. J. North, "Tutorial on agent-based modelling and simulation," *Journal of Simulation,* vol. 4, no. 3, pp. 151-162, 2010.

[42] N. R. Jennings, "Agent-based computing: Promise and perils," *IJCAI International Joint Conference on Artificial Intelligence,* vol. 2, pp. 1429-1436, 1999.

[43] J. M. Epstein, "Agent-Based Computational Models And Generative Social Science," *Complexity,* vol. 4, no. 5, pp. 41-60, 1999.

[44] L. J. Moya and A. Tolk, "Towards a taxonomy of agents and multi-agent systems," *Proceedings of the 2007 Spring Simulation Multiconference,* pp. 11-18, 2007.

[45] J. M. Epstein and R. L. Axtell, *Growing artificial societies: social science from the bottom up*. Washington D.C., USA: Brookings Institution Press, 1996.

[46] C. Hüning, J. Wilmans, N. Feyerabend, and T. Thiel-clemen, "MARS-A next gen simulation framework," J. Wittmann and D. Maretis, Eds.: Shaker, Aachen, 2014.

[47] L. U. Yang and N. Gilbert, "Getting Away From Numbers: Using Qualitative Observation for Agent-Based Modeling," *Advances in Complex Systems,* vol. 11, no. 02, pp. 175-185, 2008.

[48] A. Geller and S. Moss, "Growing Qawm: An evidence-driven declarative mofel of Afghan Power Structures," *Advances in Complex Systems,* vol. 11, no. 2, pp. 321-335, 2008.

[49] A. Ghorbani, G. Dijkema, and N. Schrauwen, "Structuring Qualitative Data for Agent-Based Modelling," *Journal of Artificial Societies and Social Simulation,* vol. 18, no. 1, p. 2, 2015.

[50] B. Edmonds, "Using Qualitative Evidence to Inform the Specification of Agent-Based Models," *Journal of Artificial Societies and Social Simulation,* vol. 18, no. 1, p. 18, 2015.

[51] J. A. Sokolowski, C. M. Banks, S. Y. Diallo, J. J. Padilla, and C. J. Lynch, "A Simulation Analysis to Weigh the Impact of Obesity: Corresponding Patient Need with Medical Capacity," Vista, CA, 2013: Society for Modeling and Simulation International, 2013.

[52] J. J. Padilla, S. Y. Diallo, H. Kavak, O. Sahin, J. S. Sokolowski, and R. J. Gore, "Semi-automated initialization of simulations: an application to healthcare," *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology,* vol. 13, no. 2, pp. 171-182, 2016.

[53] C. M. Schneider, V. Belik, T. Couronné, Z. Smoreda, and M. C. González, "Unravelling daily human mobility motifs," *Journal of The Royal Society Interface,* vol. 10, no. 84, 2013.

[54] C. Kennedy, G. Theodoropoulos, V. Sorge, E. Ferrari, P. Lee, and C. Skelcher, "AIMSS: An architecture for data driven simulations in the social sciences," 2007: Springer Berlin Heidelberg.

[55] I. Yohai, B. Skarin, R. McCormack, and J. Hsu, "Deriving Population Assessment through Opinion Polls, Text Analytics, and Agent-Based Modeling," in *Proceedings of the Social Computing, Behavioral-Cultural Modeling and Prediction 2014*, W. G. Kennedy, N. Agarwal, and S. J. Yang, Eds. Washington, DC, USA: Springer International Publishing, 2014, pp. 187-194.

[56] T. Jensen and É. J. L. Chappin, "Automating agent-based modeling: Data-driven generation and application of innovation diffusion models," *Environmental Modelling and Software,* vol. 92, pp. 261-268, 2017.

[57] A. Tolk, S. Y. Diallo, J. J. Padilla, and H. Herencia-Zapana, "Reference modelling in support of M&S—foundations and applications," *Journal of Simulation,* vol. 7, no. 2, pp. 69-82, 2013.

[58] S. Robinson, *Simulation: The Practice of Model Development and Use*, First ed. West Sussex, England: John Wiley & Sons, Ltd, 2004, p. 336.

[59] F. D. McKenzie, "Systems modeling: analysis and operations research," John Wiley & Sons, Inc., 2010, pp. 147-180.

[60] M. C. González, C. A. Hidalgo, and A.-L. Barabási, "Understanding individual human mobility patterns," *Nature,* vol. 453, no. 7196, pp. 779-782, 2008.

[61] S. Jiang, J. Ferreira, and M. C. González, "Activity-Based Human Mobility Patterns Inferred from Mobile Phone Data : A Case Study of Singapore," *ACM KDD UrbComp'15,* 2015.

[62] M. De Domenico, A. Lima, and M. Musolesi, "Interdependence and predictability of human mobility and social interactions," *Pervasive and Mobile Computing,* vol. 9, no. 6, pp. 798-807, 2013.

[63] V. Etter, M. Kafsi, E. Kazemi, M. Grossglauser, and P. Thiran, "Where to go from here? Mobility prediction from instantaneous information," *Pervasive and Mobile Computing,* vol. 9, no. 6, pp. 784-797, 2013/12/01/ 2013.

[64] J. McInerney, S. Stein, A. Rogers, and N. R. Jennings, "Breaking the habit: Measuring and predicting departures from routine in individual human mobility," *Pervasive and Mobile Computing,* vol. 9, no. 6, pp. 808-822, 2013.

[65] C. Xu, B. Firner, Y. Zhang, R. Howard, and J. Li, "Exploiting human mobility trajectory information in indoor device-free passive tracking," presented at the Proceedings of the 11th international conference on Information Processing in Sensor Networks, Beijing, China, 2012.

[66] A. Mislove, S. Lehmann, Y.-y. Ahn, J.-p. Onnela, and J. N. Rosenquist, "Understanding the Demographics of Twitter Users," in *Fifth International AAAI Conference on Weblogs and Social Media*, 2011, pp. 554-557.

[67] A. Culotta, N. K. Ravi, and J. Cutler, "Predicting the Demographics of Twitter Users from Website Traffic Data," *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence,* pp. 72-78, 2015.

[68] J. Chen, E. Haber, R. Kang, G. Hsieh, and J. Mahmud, "Making Use of Derived Personality: The Case of Social Media Ad Targeting," in *Ninth International AAAI Conference on Web and Social Media*, 2015.

[69] G. Hsieh, J. Chen, J. Mahmud, and J. Nichols, "You Read What You Value: Understanding Personal Values and Reading Interests," in *Proceedings of the 32nd annual ACM conference on human factors in computing systems*, 2014, pp. 983-986.

[70] K. Ryoo and S. Moon, "Inferring Twitter user locations with 10 km accuracy," *Proceedings of the companion publication of the 23rd international conference on World Wide Web,* pp. 643-648, 2014.

[71] J. Mahmud, J. Nichols, and C. Drews, "Home Location Identification of Twitter Users," *ACM Transactions on Intelligent Systems and Technology (TIST),* vol. 5, no. 3, 2014.

[72] T. Hu, J. Luo, H. Kautz, and A. Sadilek, "Home Location Inference from Sparse and Noisy Data: Models and Applications," *Proceedings - 15th IEEE International Conference on Data Mining Workshop, ICDMW 2015,* pp. 1382-1387, 2016.

[73] I. Nassi and B. Shneiderman, "Flowchart techniques for structured programming," *ACM SIGPLAN Notices,* vol. 8, no. 8, pp. 12-26, 1973.

[74] T. DeMarco, *Structured Analysis and System Specification*. Prentice-Hall, 1979.

[75] Q. Li and Y.-L. Chen, "Data Flow Diagram," in *Modeling and Analysis of Enterprise and Information Systems: From Requirements to Realization*, Q. Li and Y.-L. Chen, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 85-97.

[76] C. Gane and T. Sarson, *Structured systems analysis: tools and techniques*. Prentice-Hall, 1979.

[77] P. T. Ward and S. J. Mellor, *Structured Development for Real-Time Systems, Vol. II: Essential Modeling Techniques*. Pearson Education, 1986.

[78] S. Robinson, G. Arbez, L. G. Birta, A. Tolk, and G. Wagner, "Conceptual Modeling: Definition, Purpose and Benefits," 2015, pp. 2812-2826.

[79] O. Balci and W. F. Ormsby, "Conceptual modelling for designing large-scale simulations," *Journal of Simulation,* vol. 1, no. 3, pp. 175-186, 2007.

[80] S. Robinson, "Conceptual Modeling for Simulation: Issues and Research Requirements," in *Winter Simulation Conference*, 2006, pp. 792-800.

[81] H. Kavak, J. J. Padilla, D. Vernon-Bido, R. J. Gore, and S. Y. Diallo, "The Spread of Wi-Fi Router Malware Revisited," in *Spring Simulation Multi-Conference*, Virginia Beach, VA, USA, 2017: ACM.

[82] H. Kavak, D. Vernon-Bido, and J. Padilla, "Fine-Scale Prediction of People's Home Location using Social Media Footprints," in *SBP-BRIMS*, Washington, D.C., USA, 2018: Springer.

[83] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases," *AI magazine,* pp. 37-54, 1996.

[84] P. J. Guo, "Software tools to facilitate research programming," Stanford University, May, 2012.

[85] F. E. Grubbs, "Procedures for Detecting Outlying Observations in Samples," *Technometrics,* vol. 11, no. 1, pp. 1-21, 1969.

[86] A.-L. Barabási, "The origin of bursts and heavy tails in human dynamics," *Nature,* vol. 435, no. 7039, pp. 207-11, 2005.

[87] J. Barnard and X.-L. Meng, "Applications of multiple imputation in medical studies: from AIDS to NHANES," *Statistical methods in medical research,* vol. 8, no. 1, pp. 17-36, 1999.

[88] D. T. Larose and C. D. Larose, *Discovering Knowledge in Data: An Introduction to Data Mining*. Wiley, 2014.

[89] L. Soicher and F. Vivaldi, Algorithmic Mathematics: The University of London, 2004. [Online]. Available: http://www.maths.qmul.ac.uk/~lsoicher/ambook.pdf.

[90] A. Smola and S. V. N. Vishwanathan, *Introduction to Machine Learning*. Cambridge University Press, 2010.

[91] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H. T. Lin, *Learning from Data: A Short Course*. AMLBook.com, 2012.

[92] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition*. Elsevier Science, 2005.

[93] B. Lantz, *Machine Learning with R*. Packt Publishing, 2015.

[94] M. Verleysen and D. François, "The Curse of Dimensionality in Data Mining and Time Series Prediction," in *Computational Intelligence and Bioinspired Systems*, Berlin, Heidelberg, 2005, pp. 758-770: Springer Berlin Heidelberg.

[95] T. Hastie, R. Tibshirani, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, Second ed. (Springer Series in Statistics). Springer, 2009.

[96] C. R. Shalizi, "Advanced Data Analysis from an Elementary Point of View," 2012.

[97] A. Blum, J. Hopcroft, and R. Kannan, *Foundations of Data Science*. 2017.

[98] M. Bramer, *Principles of Data Mining*. Springer London, 2016.

[99] scikit-learn. (2019). *Choosing the right estimator*. Available: https://scikit-learn.org/stable/tutorial/machine_learning_map/

[100] J. Gaudart, B. Giusiano, and L. Huiart, "Comparison of the performance of multi-layer perceptron and linear regression for epidemiological data," *Computational Statistics and Data Analysis,* vol. 44, no. 4, pp. 547-570, 2004.

[101] D. J. Thomson, "Jackknifing multiple-window spectra," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing 1994*, Adelaide, SA, Australia, 1994, vol. 6, pp. 73-76.

[102] S. B. James, R. Bardenet, Y. Bengio, and K. Balazs, "Algorithms for Hyper-Parameter Optimization," pp. 2546-2554, 2011.

[103] O. Balci, "Principles and techniques of simulation validation, verification, and testing," *Winter Simulation Conference Proceedings, 1995.,* pp. 1048-1053, 1995.

[104] V. Kaufmann, *Rethinking the City: Urban Dynamics and Motility*. EPFL Press, 2011, pp. 157-157.

[105] B. Vilhelmson, "Daily mobility and the use of time for different activities. The case of Sweden," *GeoJournal,* vol. 48, pp. 177-185, 1999.

[106] F. Asgari, V. Gauthier, and M. Becker, "A survey on human mobility and its applications," 2013.

[107] C. Song, Z. Qu, N. Blumm, and A.-l. Barabási, "Limits of Predictability in Human Mobility," *Science,* vol. 327, no. 5968, pp. 1018-1021, 2010.

[108] Statista. (2018). *Twitter - Statistics and Facts*. Available: https://www.statista.com/topics/737/twitter/

[109] Y. Kryvasheyeu *et al.*, "Rapid assessment of disaster damage using social media activity," *Science Advances,* vol. 2, no. 3, p. e1500779, 2016.

[110] A. Sadilek, S. Brennan, and H. Kautz, "nEmesis: Which Restaurants Should You Avoid Today?," *First AAAI Conference on Human Computation and Crowdsourcing,* pp. 138-146, 2013.

[111] R. J. Gore, S. Diallo, and J. Padilla, "You Are What You Tweet: Connecting the Geographic Variation in America's Obesity Rate to Twitter Content," *PLOS ONE,* vol. 10, no. 9, p. e0133505, 2015.

[112] J. J. Padilla, H. Kavak, C. J. Lynch, R. J. Gore, and S. Y. Diallo, "Temporal and spatiotemporal investigation of tourist attraction visit sentiment on Twitter," *PLOS ONE,* vol. 13, no. 6, p. e0198857, 2018.

[113] B. Hawelka, I. Sitko, E. Beinat, S. Sobolevsky, P. Kazakopoulos, and C. Ratti, "Geo-located Twitter as proxy for global mobility patterns," *Cartography and Geographic Information Science,* vol. 41, no. 3, pp. 260-271, 2014.

[114] B. Hofer, T. J. Lampoltshammer, and M. Belgiu, "Demography of Twitter Users in the City of London: An Exploratory Spatial Data Analysis Approach," in *Modern Trends in Cartography: Selected Papers of CARTOCON 2014*, J. Brus, A. Vondrakova, and V. Vozenilek, Eds.: Springer International Publishing, 2015, pp. 199-211.

[115] R. Jurdak, K. Zhao, J. Liu, M. AbouJaoude, M. Cameron, and D. Newth, "Understanding Human Mobility from Twitter," *PLOS ONE,* vol. 10, no. 7, p. e0131469, 2015.

[116] Y. Yamamoto. *Twitter4J - A Java library for the Twitter API*. Available: http://twitter4j.org/en/index.html

[117] Twitter. *Docs - Twitter Developers*. Available: https://dev.twitter.com/overview/api/

[118] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, "Detecting spammers on twitter," *Collaboration, electronic messaging, anti-abuse and spam conference (CEAS),* vol. 6, pp. 12-12, 2010.

[119] C. Grier, K. Thomas, V. Paxson, and M. Zhang, "@ spam : The Underground on 140 Characters or Less," *Proceedings of the 17th ACM conference on Computer and communications security,* pp. 27-37, 2010.

[120] D. Guo and C. Chen, "Detecting Non-personal and Spam Users on Geo-tagged Twitter Network," *Transactions in GIS,* vol. 18, no. 3, pp. 370-384, 2014.

[121] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas, "Sentiment Strength Detection in Short Informal Text," *Journal of The American Society for Information Science and Technology,* vol. 61, no. 12, pp. 2544-2558, 2010.

[122] M. Ester, H.-P. Krieger, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," Portland, Oregon, 1996, pp. 226-231: AAAI.

[123] O. Varol, E. Ferrara, C. A. Davis, F. Menczer, and A. Flammini, "Online human-bot interactions: Detection, estimation, and characterization," *arXiv preprint arXiv:1703.03107,* 2017.

[124] F. Luo, G. Cao, K. Mulligan, and X. Li, "Explore Spatiotemporal and Demographic Characteristics of Human Mobility via Twitter: A Case Study of Chicago," *Applied Geography,* vol. 70, pp. 11-25, 2015.

[125] U. C. Bureau. (2000). *Decennial Census Surname Files*. Available: https://www.census.gov/data/developers/data-sets/surnames.2000.html

[126] S. S. Administration. (2019). *Popular Baby Names*. Available: https://www.ssa.gov/oact/babynames/

[127] A. Sadilek and H. Kautz, "Modeling the Impact of Lifestyle on Health at Scale," *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining,* pp. 637-646, 2013.

[128] T. Pontes *et al.*, "Beware of what you share: Inferring home location in social networks," *Proceedings - 12th IEEE International Conference on Data Mining Workshops, ICDMW 2012,* pp. 571-578, 2012.

[129] E. Cho, S. Myers, and J. Leskovec, "Friendship and mobility: user movement in location-based social networks," in *Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining*, San Diego, California, USA, 2011, pp. 1082-1090: ACM.

[130] S. Scellato, A. Noulas, R. Lambiotte, and C. Mascolo, "Socio-spatial properties of online location-based social networks," pp. 329-336, 2011.

[131] J. P. Bagrow and Y.-R. Lin, "Mesoscopic structure and social aspects of human mobility," *PLOS ONE,* vol. 7, no. 5, p. e37676, 2012.

[132] J. Krumm, "Inference Attacks on Location Tracks," *Pervasive Computing,* vol. 10, no. Pervasive, pp. 127-143, 2007.

[133] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web," Stanford InfoLab1999.

[134] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning,* vol. 20, no. 3, pp. 273-297, 1995/09/01 1995.

[135] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal,* vol. 27, no. 3, pp. 379-423, 1948.

[136] F. Rosenblatt, "Principles of neurodynamics perceptrons and the theory of brain mechanisms," Cornell Aeronautical Lab Inc.1961.

[137] H. Tin Kam, "Random Decision Forests," in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, 1995, vol. 1, pp. 278-282.

[138] R. C. Holte, "Very Simple Classification Rules Perform Well on Most Commonly Used Datasets," *Machine Learning,* vol. 11, no. 1, pp. 63-91, 1993.

[139] S. J. Taylor and B. Letham, "Forecasting at Scale," *The American Statistician,* vol. 72, no. 1, pp. 37-45, 2018.

[140] G. A. Korn and T. M. Korn, *Mathematical Handbook for Scientists and Engineers: Definitions, Theorems, and Formulas for Reference and Review*. Dover Publications, 2000.

[141] P. S. Foundation. (2019). *Welcome to python.org*. Available: https://www.python.org

[142] "Ray: A system for parallel and distributed Python that unifies the ML ecosystem," ed, 2019.

[143] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of machine learning research,* vol. 12, pp. 2825-2830, 2011.

[144] A. Thusoo *et al.*, "Hive: a warehousing solution over a map-reduce framework," *Proceedings of the VLDB Endowment,* vol. 2, no. 2, pp. 1626-1629, 2009.

[145] V. K. Vavilapalli *et al.*, "Apache hadoop yarn: Yet another resource negotiator," in *Proceedings of the 4th annual Symposium on Cloud Computing*, 2013, p. 5: ACM.

[146] W. McKinney, "pandas: a foundational Python library for data analysis and statistics," *Python for High Performance and Scientific Computing,* pp. 1-9, 2011.

[147] T. E. Oliphant, *A guide to NumPy*. Trelgol Publishing USA, 2006.

[148] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in science & engineering,* vol. 9, no. 3, pp. 90-95, 2007.

[149] C. Song, T. Koren, P. Wang, and A.-l. Barabási, "Modelling the scaling properties of human mobility," *Nat Phys,* vol. 6, no. 10, pp. 818-823, 2010.

[150] Y. Zheng, X. Xie, and W.-Y. Ma, "Geolife: A collaborative social networking service among user, location and trajectory," *IEEE Data(base) Engineering Bulletin,* vol. 33, no. 2, pp. 32-39, 2010.

[151] D. Brockmann, L. Hufnagel, and T. Geisel, "The scaling laws of human travel," *Nature,* vol. 439, no. 7075, pp. 462-465, 2006.

[152] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from GPS trajectories," in *Proceedings of the 18th international conference on World Wide Web*, Madrid, Spain, 2009, pp. 791-800, 1526816: ACM.

[153] B. R. Jasny, G. Chin, L. Chong, and S. Vignieri, "Again, and Again, and Again …," *Science,* vol. 334, no. 6060, p. 1225, 2011.

[154] U. Wilensky and W. Rand, "Making models match: Replicating an agent-based model," *Journal of Artificial Societies and Social Simulation,* vol. 10, no. 4, 2007.

[155] V. Grimm, U. Berger, D. L. DeAngelis, J. G. Polhill, J. Giske, and S. F. Railsback, "The ODD protocol: A review and first update," *Ecological Modelling,* vol. 221, no. 23, pp. 2760-2768, 2010/11/24/ 2010.

[156] H. Bersini, "UML for ABM," *Journal of Artificial Societies and Social Simulation,* vol. 15, no. 1, p. 9, 2012.

[157] B. S. S. Onggo and O. Karpat, "Agent-based conceptual model representation using BPMN," presented at the Proceedings of the Winter Simulation Conference, Phoenix, Arizona, 2011.

[158] P. Livet, J. P. Müller, D. Phan, L. Sanders, and T. Auatabu, "Ontology, a mediator for agent-based modeling in social science," (in eng), *Journal of Artificial Societies and Social Simulation,* Article de revue à facteur d'impact vol. 13, no. 1, p. 14 p., 2010.

[159] (2019). *OSF*. Available: https://osf.io

[160] (2019). *CoMSES Net*. Available: https://www.comses.net

[161] A. S. Lee and R. L. Baskerville, "Generalizing Generalizability in Information Systems Research," *Information Systems Research,* vol. 14, no. 3, pp. 221-243, 2003.

[162] J. J. Padilla, S. Y. Diallo, H. Kavak, A. Barraco, C. J. Lynch, and H. Kavak, "Cloud-based simulators: Making simulations accessible to non-experts and experts alike," 2014, vol. 2015-Janua, pp. 3630-3639: IEEE.

[163] D. Hales, J. Rouchier, and B. Edmonds, "Model-to-model analysis," *Journal of Artificial Societies and Social Simulation,* vol. 6, no. 4, 2003.

[164] D. C. Earnest, "Coordination in Large Numbers: An Agent-Based Model of International Negotiations," *International Studies Quarterly,* vol. 52, no. 2, pp. 363-382, 2008.

[165] K. Leetaru and P. Schrodt, "GDELT: Global data on events, location, and tone," presented at the ISA Annual Convention, 2013.

[166] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature,* vol. 521, p. 436, 05/27/online 2015.

APPENDICES

APPENDIX A. CLASSIFIER PERFORMANCE

Table Appendix 1 summarizes the hyperparameter optimization test setup. Multi-Layer Perceptron classifier is evaluated with different *alpha* numbers to prevent model overfit, different *learning rates* to test the convergence to the expected outputs, and different *number of layers and artificial neurons* to capture the patterns in the data. Support vector classifier is tested with different regularization term (*C*) values to prevent model overfit, different *tolerance numbers* to test the convergence to the expected outputs, and balanced/unbalanced *class weights* to evaluate tolerance against class imbalance. These settings are tested for both linear and non-linear (RBF) kernels where the non-linear kernel has an additional *gamma* parameter that tests the influence of a particular training instance. Finally, Random Forest Classifier is tested with different *number of trees* to capture the patterns in the data, different *depth* values to limit overfit, different *split criteria*, *number of features*, and *class weights* in classifier performance test.

TABLE APPENDIX 1

PARAMETER AND DATA VARIABLE COMBINATIONS FOR ML CLASSIFIERS

	ML Parameters	Variables (features)
MLP Classifier	• **Alpha:** 10^{-5}, 10^{-3}, 10^{-1}, 10^{1} • **Learning rate:** 10^{-3}, 10^{-2}, 10^{-1} • **1 hidden layer:** 5, 10, …, 100 • **2 hidden layers:** - *Layer 1:* 5, 15, …, 75 - *Layer 2:* 2, 6, …, 22	• Hour of the day • Hour of the day + weekday/weekend • Hour of the day + day of the week • Hour of the day + previous place ID • Hour of the day + weekday/weekend + previous place ID • Hour of the day + day of the week + previous place ID
Support Vector Classifier (SVC)	• **Linear kernel:** - *C:* 10^{-5}, 10^{-3}, 10^{-1}, 10^{1} - *Tolerance:* 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2} - *Class weight: balanced, unbalanced* • **Non-linear (RBF) kernel:** - *C:* 10^{-5}, 10^{-3}, 10^{-1}, 10^{1}, 10^{3} - *Tolerance:* 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2} - *Class weight: balanced, unbalanced* - *Gamma:* $\frac{1}{2n}, \frac{1}{n}, \frac{2}{n}$ given n is number of features	
Random Forests Classifier	• **Number of trees:** 10, 40, 60 • **Maximum depth:** unlimited, 4 • **Split criteria:** Gini index, information gain • **Class weight:** balanced, unbalanced	

All the above ML combinations are tested with a week-long synthetic mobility data using a 5-fold cross validation. Six different data feature combinations are tested based on different time and previous place considerations. Place representation is provided as a categorical variable in classifier models. Classifier performance is evaluated based on the prediction accuracy – the ratio of correctly predicted places $accuracy(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n} f_a(y_i, \hat{y_i})$ whereas n is number of samples, y is the prediction of the ML model, $\hat{y}$ is the expected result, and $f_a(k, m) = \begin{cases} 1, k = m \\ 0, otherwise \end{cases}$.

Fig. Appendix 1. shows the best accuracy of different ML classifier results with respect to several features influencing mobility. Since it is a common feature on all model, accuracy with respect to the hour of the day is compared.

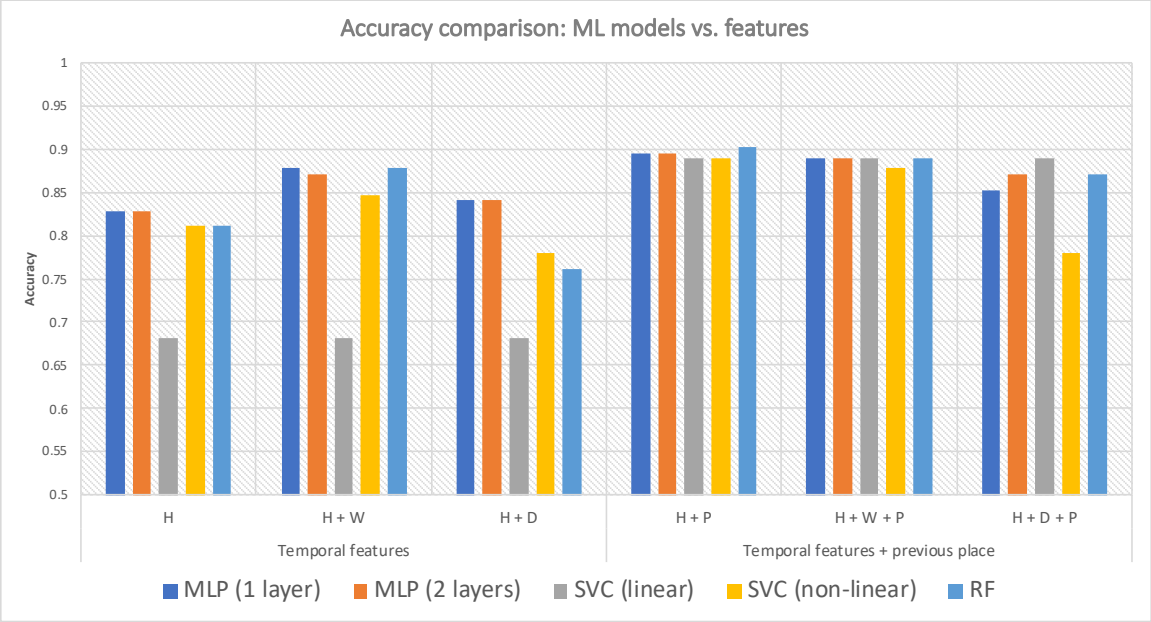


Fig. Appendix 1. Best classifier accuracies based on different feature considerations (H: Hour of the day, W: Weekday/Weekend, D: Day of the week, and P: Previous place).

When only temporal features are used, ML classifier accuracies range between 0.68 and 0.87. The consideration of Weekday/Weekend increases the accuracy for all ML classifiers while the Day of the Week only makes a slightly positive change in MLP models. Single hidden layer MLP and RF performs the best among temporal features while linear SVC performs the worst. It is because temporal features are very likely not to be properly separable with a linear function. When the previous place is introduced as an additional feature, the accuracy results of all ML models have increased, and their range is found to be between 0.78 and 0.90. Even the linear SVC performs comparable to other models. The best performing model in this setting is RF

(0.901) with Hour of the day and Previous place features, followed very closely by the 1-2 hidden layer MLP models (0.895). An interesting result here is that the single hidden layer MLP outperforms the 2-hidden layer MLP in almost all the conditions and linear SVC outperforms the non-linear one when previous place is introduced as an additional feature. These results tell us that more complex ML mechanisms, for human mobility, do not always perform better than the simpler ones. ML hyper-parameter optimization results reveal insights into the influence of parameters on prediction, overfit, and training time which are summarized in Table Appendix 2.

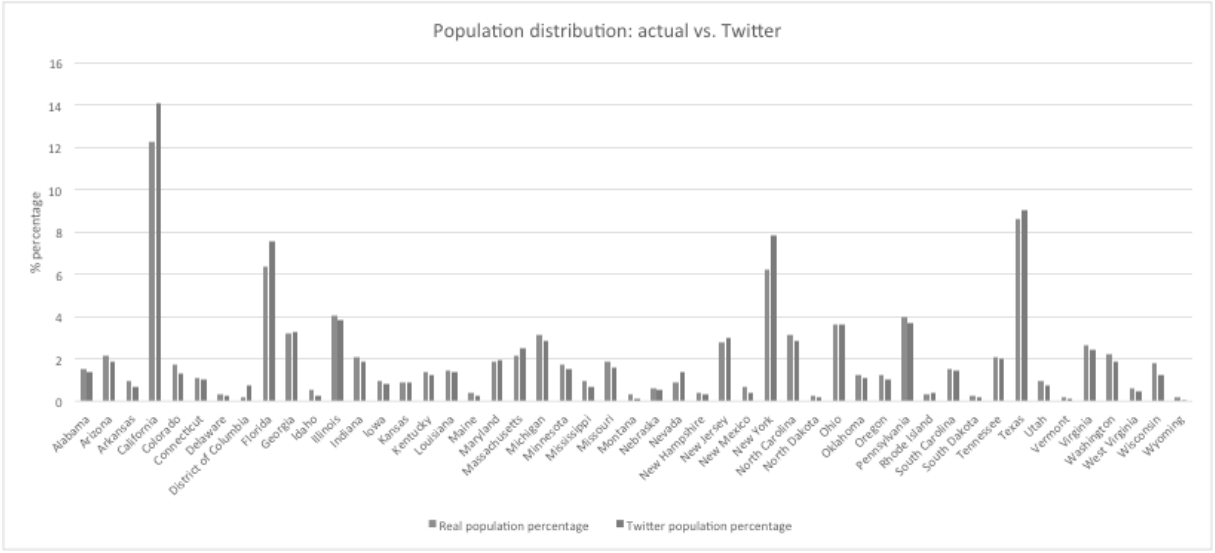
TABLE APPENDIX 2

INSIGHTS GATHERED FROM ML HYPER-PARAMETER OPTIMIZATION RESULTS

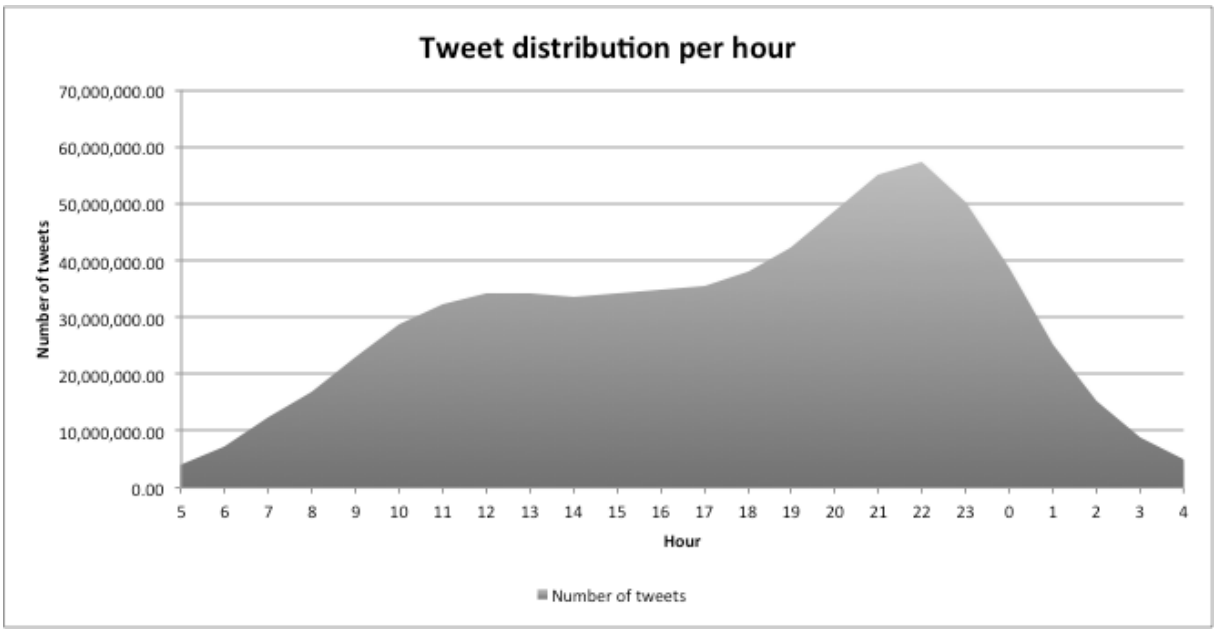
	Parameter	Insights
MLP Classifier	*Number of neurons*	In a single hidden layer MLP, model predicts ≈2% better when having neurons between 25 to 100. Having more neurons leads to overfit and increased training time. Thus, 25-30 neurons are optimal. In a 2-hidden layer MLP, model prediction is slightly better when having 15-25 neurons in each layer. When the number of neurons is closer to the upper side, the model takes ≈30-50% longer to train and overfit more.
	Learning rate	For all three value combinations, learning rate has no influence on prediction accuracy, overfit, or training time.
	Alpha	*Single hidden layer MLP:* prediction accuracy is better when number of neurons are too low and bigger alpha is used; when number of neurons increase to 40, smaller alpha performs better. When number of neurons are even higher, no apparent pattern is seen. Greater alpha number slightly decreases overfit but has impact on training time. *2-hidden layer MLP:* smaller alpha numbers seem to perform slightly better in accuracy while this pattern is not so strong. Overfit and training time is similar to the single hidden layer case.
SVC	*C*	*SVR linear kernel:* when set to 0.1, it works the most optimal for best prediction accuracy. When set to 10 or higher, the classifier overfits ≈3% more takes significantly longer time to train. *SVR non-linear kernel:* When set to 10 or higher, the classifier accuracy performs best but it is due to overfit. When set to 0.1, the overfit disappears. Unlike the linear case, it does not affect accuracy.
	Tolerance	For all three value combinations, tolerance value has no influence on the accuracy, overfit, or training time for linear and non-linear kernel.
	Class weight	When not balanced, the classifier predicts ≈5-25% better for both linear and non-linear kernel.
	Gamma	Accuracy is best when gamma value is set to $1/n$ given that n is the number of features.
RF Classifier	*Number of trees*	Does not have a prominent effect on the accuracy or overfit. Increased number of trees also increases training time, but linearly.
	Maximum Depth	When set to 4 or unlimited, does not have a prominent effect on the accuracy or training time while overfit decreases by 1% when set to 4.
	Split criteria	Gini or Entropy perform quite the same when it comes to prediction accuracy or overfit. Entropy takes ≈17% more time to train.
	Class weight	When it is balanced, it has a very minor negative impact on the prediction accuracy but no apparent impact on overfit or training time.

APPENDIX B. TWITTER DATA DETAILS

The mobility dataset collected for this dissertation is from Twitter.com using their Streaming API between May 16, 2014 and April 27, 2015 (343 days – 4 days missing due to interruptions). This dataset contains geo-located Twitter messages shared within the conterminous US totaling 826,021,868 records and consuming 2.39 terabyte raw data size. Data size is reduced to 716,553,502 records from 6,375,210 users after cleaning non-US, low resolution, and non-human Twitter messages. Fig. Appendix 2. (a) and (b) show a comparison between Twitter user population in this dataset vs. actual US population at the state level gathered from US Census. A subset of this dataset is used for the use case city of Chicago which is composed of ≈7.78 million Twitter messages from 92,296 active users.



(a)



(b)

Fig. Appendix 2. Some characteristics of the Twitter data used in this dissertation (a) Twitter population vs. actual population percentages. (b) Hourly Twitter message distribution based on local time of the posting person.

Shapefile datasets used in the models are collected from the use case city's data repository – Chicago GIS data (https://data.cityofchicago.org).

APPENDIX C. HOME LOCATION PREDICTION EXPERIMENTATION GROUPS

TABLE APPENDIX 3

DATA DESCRIPTION FOR FIG. 22

Days	Start date	End date	Data size	Number of training/test instance generated	Number of users represented
7	2014-05-16	2014-05-23	26,132	2,886	383
14	2014-05-16	2014-05-30	56,762	5,589	470
21	2014-05-16	2014-06-06	90,422	8,681	535
30	2014-05-16	2014-06-15	128,322	11,815	681
90	2014-05-16	2014-08-14	419,950	31,669	850
180	2014-05-16	2014-11-12	759,163	53,795	1058
270	2014-05-16	2015-02-10	1,041,359	68,805	1,195

TABLE APPENDIX 4

DATA DESCRIPTION FOR FIG. 23

Condition	Number of users (G_n)	Condition	Number of users (G_r)
$0 \leq G_{n1} < 75$	316	$0 \leq G_{r1} < 0.6$	321
$75 \leq G_{n2} < 225$	331	$0.6 \leq G_{r2} < 1.4$	330
$225 \leq G_{n3} < 475$	298	$1.4 \leq G_{r3} < 2.75$	301
$475 \leq G_{n4}$	323	$2.75 \leq G_{r4}$	316

VITA

Hamdi Kavak is currently a Research Associate at George Mason University's Geography and Geoinformation Science Department. He received his M.E. degree in Modeling and Simulation (2015) from Old Dominion University (ODU). He received his B.S. degree in Computer Engineering (2008) from Karadeniz Technical University (Turkey). His research focuses on designing and developing data-driven computational models to address domain-specific problems. He has served as a graduate research assistant for the Virginia Modeling Analysis and Simulation Center at ODU.