The Deep Aion Project:

Exploring How Different Temporal Representations Can Benefit Deep Longitudinal Models

by

Matthew Lenert

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Biomedical Informatics

August 31, 2021

Nashville, Tennessee

Approved:

Colin G. Walsh, MD MA

Asli O. Weitkamp, PhD

Michael E. Matheny, MD MS MPH

Thomas A. Lasko, MD PhD

Jeffrey D. Blume, PhD

ACKNOWLEDGEMENTS

Some time ago at a door I knocked,

Not ’til months later was it unlocked,

Others were open, more still closed

Crossing the threshold, a waft of cigar

The faint smell of whiskey, the strum of guitar

To a brick tower on top of a Moe’s

In a wood cube is where I would start

A village effort, I must disclose

A fellowship of folks who touched my heart

Without Jen and her Romeo, I’d long froze

Stumbled and fallen without Rischelle

And without Cindy, I’d be none too well

Through the forest with so many great guides

These giants of goodness with unrivaled strides

Colin, Kevin, Gretchen, and Michael

Jeffrey, Tom, Asli, and Randy

How can a place have such a modus operandi?

Sherpas all through the NLM’s generous cycle.

The shadows lift, the tree cover thins,

And thus one journey ends where the next begins.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

LIST OF EQUATIONS

Equation Page

Chapter I

INTRODUCTION & BACKGROUND

Current Healthcare Prognostic Modeling Challenges and Hypothesis

Both providers and their patients show great enthusiasm for the potential of models that can forecast future events (prognosis) to improve the efficiency and efficacy of the healthcare system[1,2]. Such prognostic models may one day help providers minimize a patient's exposure to risks for adverse events (e.g., post surgical infection), while optimizing the value (maximizing quality of care, while minimizing cost) delivered to the patient[3]. Prognostic models can improve efficiency by allaying the administrative burden of healthcare through applications such as data-driven staffing adjustments or anticipatory discharge/unit transfer protocols. However, there are many challenges to overcome before realizing this collaborative future between prognostic models and the human actors in healthcare[4,5].

These numerous challenges occur at all stages of the prognostic model lifecycle—in development, implementation, surveillance and maintenance, and at de-implementation. Figure 1 visualizes the prognostic model lifecycle and provides some considerations at each stage[6]. This work will focus on challenges in the model development stage, however; model developers should design their prognostic models with the whole model lifecycle in mind[6].

Some of the challenges of prognostic model development for the clinical setting have to do with the nature of the underlying data[7]. Much of the data used for prognostic model development is observational data (recorded during the course of care) from the clinical setting. Observational data is preferred to ease the burden of implementing prognostic models into the clinical setting. The underlying assumption is that data collection procedures, which can be costly to the patient and/or to the provider, would not require modification for the model to provide forecasts.

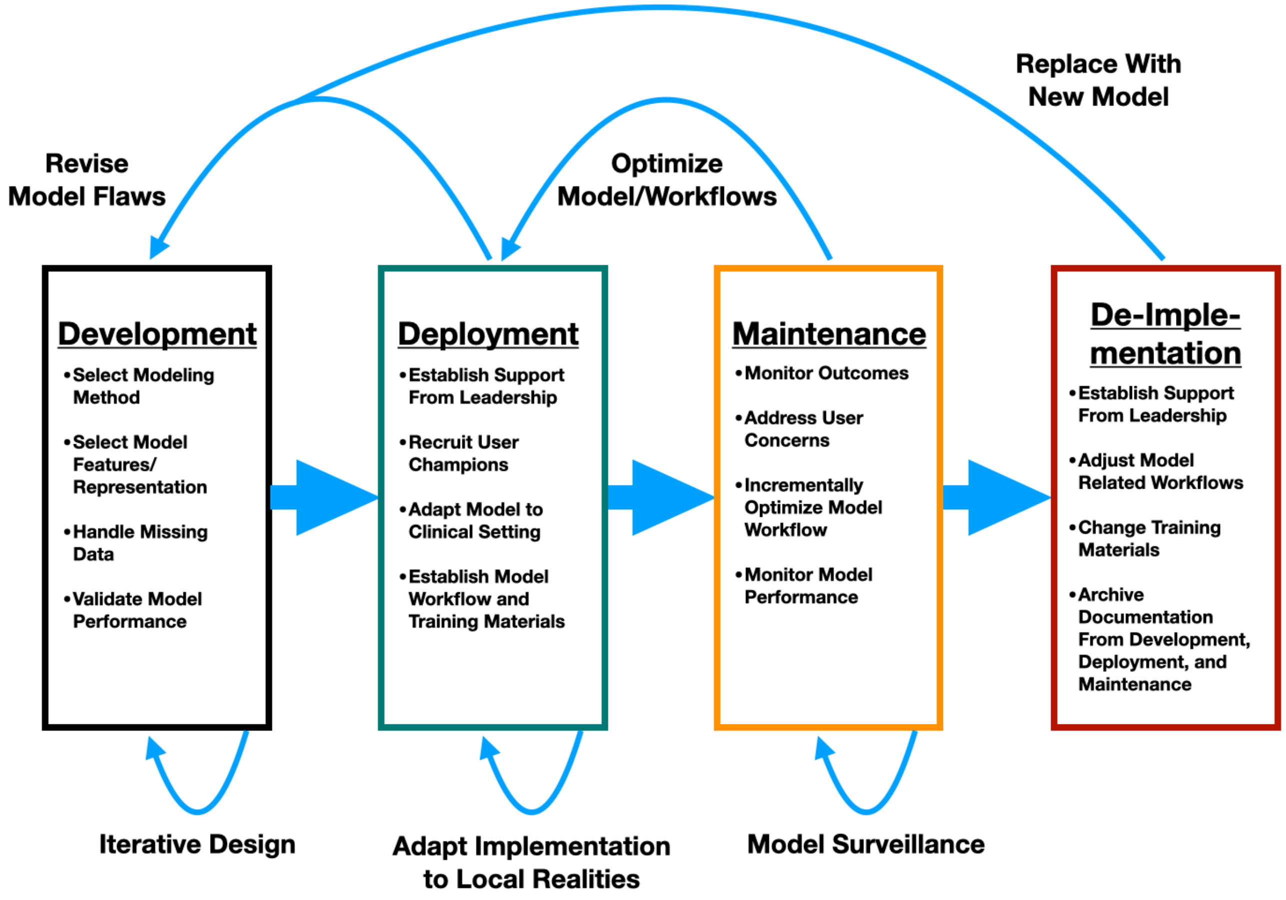

Replace With New Model
Revise Model Flaws
Optimize Model/Workflows
Development
•Select Modeling Method
•Select Model Features/ Representation
•Handle Missing Data
•Validate Model Performance
Deployment
•Establish Support From Leadership
•Recruit User Champions
•Adapt Model to Clinical Setting
•Establish Model Workflow and Training Materials
Maintenance
•Monitor Outcomes
•Address User Concerns
•Incrementally Optimize Model Workflow
•Monitor Model Performance
De-Imple-mentation
•Establish Support From Leadership
•Adjust Model Related Workflows
•Change Training Materials
•Archive Documentation From Development, Deployment, and Maintenance
Iterative Design
Adapt Implementation to Local Realities
Model Surveillance

Figure 1: The Prognostic Model Life Cycle (Adapted from [6])

We define the clinical setting as the set of workflows/socio-technical processes relating to the delivery of healthcare by the employees/contractors of health systems, hospitals, and/or clinics to the patients that sought their care[8]. The processes at any given organization embody an evolving compromise between the goals and incentives of: patients, physicians, nurses, administrators, managers, support staff, insurers, and regulators[9]. Furthermore, the practice of medicine itself is an ever-changing mix of empirical evidence, biochemical theory, experience, tradition, and technology[10-12]. The compromise of conflicting incentives and the practice of medicine influence data collection during healthcare workflows[9]. This influence can be found in the different biases and artifacts found in routinely collected healthcare data[13]. This dissertation will only introduce the data biases and artifacts directly relevant to this study—there are many more which will not be covered.

The collection of different clinical data elements is a non-random process, meaning that data collection usually has a cost to one if not several healthcare actors[14]. For example, for patients, data collection may result in physical suffering and/or financial costs; for providers, data collection may result in time, equipment utilization (capital depreciation), and financial costs; and for payors it generally involves financial costs[9]. Healthcare systems offer a wide variety (tens of thousands) of services in pursuit of diagnosing, treating, and managing a wide variety of conditions and diseases[15]. There are over 65,000 conditions in the International Classification of Disease version 10 (ICD-10)[16]. Given the large number of services and diseases, the data needed to contextualize and describe the process of healthcare delivery can involve hundreds of thousands of discrete data elements[17]. While there is this great range in the number and type of data elements, only a small fraction of data elements are frequently recorded[18,19]. The power-law-like distribution of data element usage reinforces the earlier claim that clinical data collection is a non-random process[19]. The reasons why some elements are observed more frequently than others vary. Some data elements, such as smoking status, are required to be collected due to regulatory incentives/penalties[20]. These data may suffer from biases of providers copying forward past responses or from patients fearful of the perceived stigma of their true status[21-23]. Some data are collected because they are bundled with a group of services (e.g., a laboratory panel such as basic metabolic panel) or administrative process (e.g., a standardized order set for chest pain admissions)[24, 25]. In this situation, the data element may or may not be of interest to clinicians, and its collection is more reflective of a policy or logistical decision rather than specific information seeking behavior from clinicians[26]. Data elements that are rarely collected can be viewed as information-seeking behavior on the first collection, but may be reflective of logistical considerations on subsequent collections (e.g., clinical orders that default to repeated collection for the whole clinical encounter)[24].

Data with this type of mechanism can be biassed by financial considerations, and will typically be minimized in the inpatient (hospital) setting (due to a prospective payment scheme) and maximized in the outpatient (clinic) setting (due to a fee for service payment scheme)[27, 28].

The time dependence of many diseases and conditions means that many of these data elements are also dependent on time[29]. For example, broken bones (if properly treated) heal; the burden of diabetes can grow over time, as unregulated blood glucose levels damage the venous system. Time dependency is also reflected in the episodic nature of healthcare delivery[30]. What this means from a data perspective is that there will be times of dense data collection, but mostly data will be sparse if collected at all. Even within an inpatient admission, data collection is generally episodic following institutional norms such as rounding times, nursing documentation policies, and care setting (eg., intensive care versus medical ward)[31]. In the United States, patients that are or perceive themselves as sicker will generally have more interactions with the healthcare system than those that perceive themselves as healthy[30]. Furthermore, the process of healthcare itself changes over time with the creation of new diagnostics and therapies as well as the accumulation of new biological knowledge and best practices[32]. Laws and payment mechanisms also evolve over time and have their own effects on the healthcare process[33]. These generalities imply that the presence and timing of data is also non-random, which further indicates that the patterns in observational clinical data observed over time might be informative[34]. Prognostic model developers potentially forgo the benefits of these temporal patterns if they choose to ignore the temporal nature of healthcare data[35].

Incorporating time into a prognostic model built with observational clinical data is complicated by some of the data biases previously mentioned. Two grand challenges to the use of time in observational clinical data are how time should be represented (structured) as well as how to account for predictors that are rarely observed together in the same observation period[36]. These two challenges are intimately related. Temporal representation defines the structure of the set of observations, thus affecting how often a predictor is not observed within that structure. This relationship may in turn influence which temporal representation model developers select in an attempt to optimize the number of observation periods with missing data. While temporal representation has a long and detailed literature[36-43], the models that would attempt to harness these different temporal structures have grown more complex and flexible in their assumptions[44,45]. Model developers could use the methods of trial and error to empirically determine the temporal representation that optimizes performance for the developer's application. However, trial and error is time consuming and given a large set of possible solutions, can be intractable. We therefore hypothesize that there are characteristics of clinical observational data which can cause one temporal representation to be advantageous for model performance. The remaining sections of this chapter will provide background on different temporal characteristics in observational data, how time can be represented in data, types of models that learn from temporal data, how those models are tuned and trained, how those models are compared and evaluated, and finally how all those topics inform the specific aims for this research.

Defining Longitudinal Data

We will define clinical data over time as observational longitudinal data. As discussed previously, this data is collected as part of routine clinical operations, and the patients were not subjected to any randomization process or random sampling from the general population. The lack of randomization is the key component to the observational definition. For the longitudinal portion of our definition, there are three other definitions that must first be introduced: the idea of a subject, that of a measurement occasion (observation episode), and that of a sampling period. A subject is the unit of analysis into which observations can be clustered to. In many clinical cases this is a patient, but might also be a clinician or a clinic/unit. A measurement occasion is an interval of time where data is recorded that is specific to a subject. There is no limit on how this time interval is defined; what is encouraged is consistency in the definition between subjects. The measurement occasion symbolizes one complete observation for that subject. Subjects need not have the same number of measurement occasions nor have the timing of those measurements be relatively equal between subjects[46]. There are statistical advantages to engineering subjects to have an equal number of observations taken at relatively similar times, but it is extremely difficult to achieve that level of consistency in clinical research cohorts let alone in retrospective observational cohorts[46]. The last piece to define is the sampling period (observation window). We define the sampling period as the total length of time from the first observation episode to the last. If subjects have different numbers of measurement occasions occurring at different times, then it follows that the sampling period between subjects can also be unequal. Figure 2 provides a visual representation of the subject, measurement occasion, and sampling period (using blood pressure (BP) and heart rate (HR) as the variables of interest). In Figure 2, each measurement occasion is defined as measurements within three days. The subjects have differing starting points that anchor the measurement occasions, but the three-day definition is constant across subjects. One can also see in the figure that Subject 1 not only has fewer measurement occasions, but a shorter sampling period compared to Subject 2.

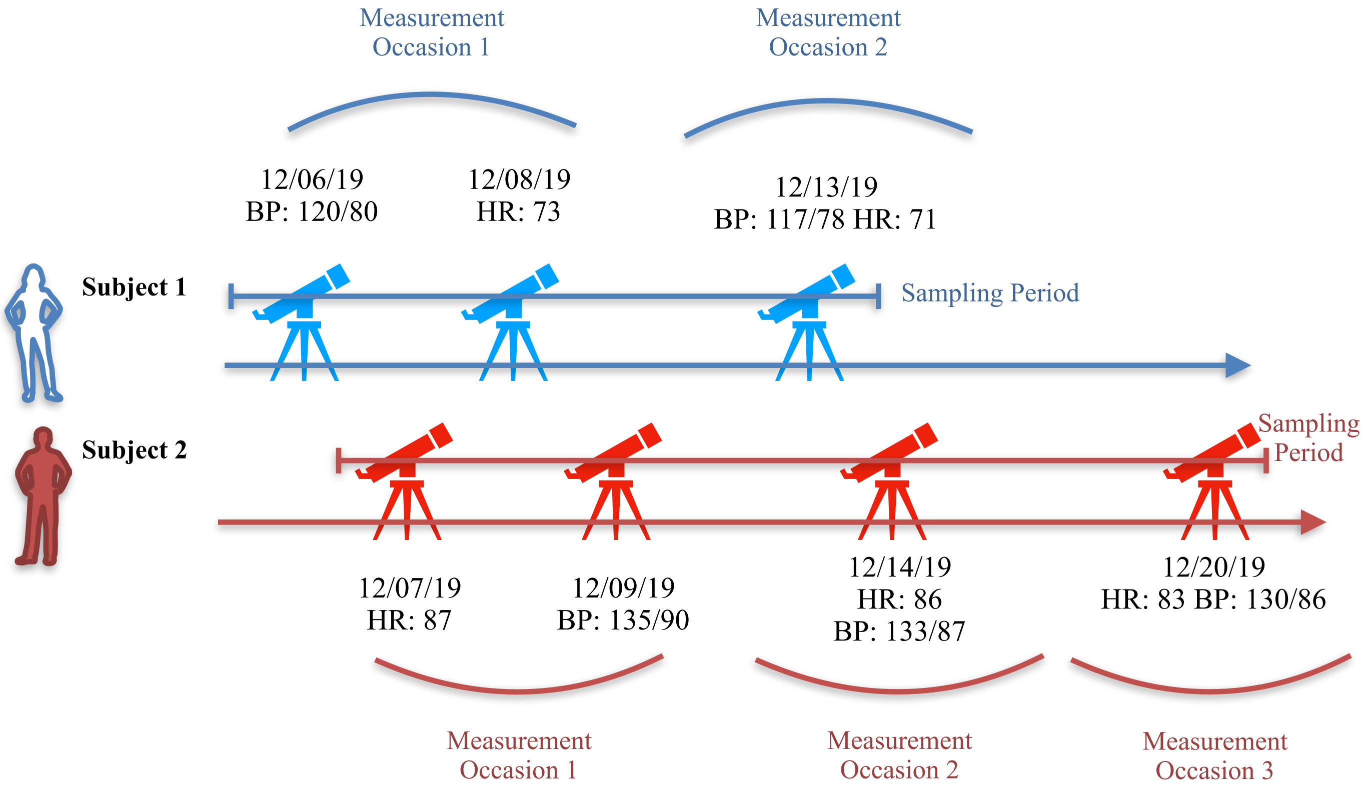


Figure 2: Subjects and Measurement Occasions

This definition is a function of the number of subjects relative to the number of measurement occasions per subject. In many clinical settings, data is sparsely observed over time, yet the number of patients with at least one observation is large[46]. Time series data tends to have a large number of observation periods for a few subjects[46]. Figure 3 provides a visual example to help contrast differences between time series and longitudinal data.

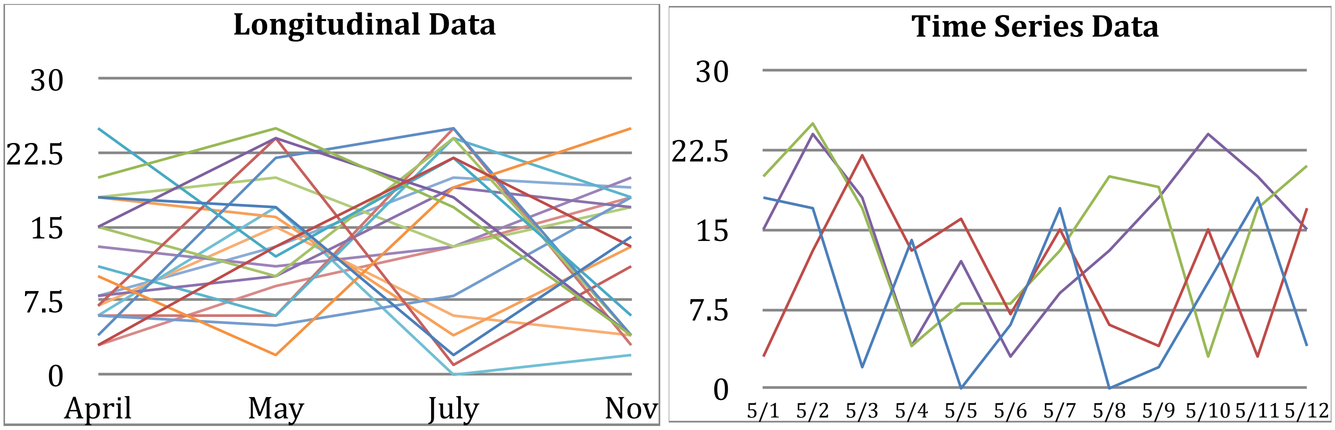


Figure 3: Comparing Longitudinal and Time Series Data

Characteristics of Longitudinal Data

If one is to look for how data characteristics might inform temporal representation, then it is important to specify and describe these characteristics. The characteristics defined here are not a comprehensive set. Again, we detail those most relevant to this study. As with other studies with intractable spaces to explore, we decided to prioritize the characteristics we thought most relevant based on our understanding of the literature[46-52]. Some of our data characteristics of interest are specific to the outcome (the variable one is attempting to predict), and others are specific to the predictors, also known as features, (the variables needed to make a forecast of the outcome)[47].

The first characteristic of interest is autocorrelation. Autocorrelation is a characteristic applicable to both the predictors and the outcome. As the name suggests, autocorrelation is the correlation of a variable with a time-delayed version of itself [53]. The time-delay is not specifically defined, so the autocorrelation may vary depending on how large the delay is. For example, if X is a random variable with high autocorrelation, then the value of X at time t is going to be highly similar to the value of X at time t+1. Given another variable Z made of random draws from a single Gaussian distribution, the value of Z at time t has no influence on what Z is at t+1. Autocorrelation is best measured with observations that are equally spaced over time[49]. Figure 4 gives examples of high and low autocorrelation.

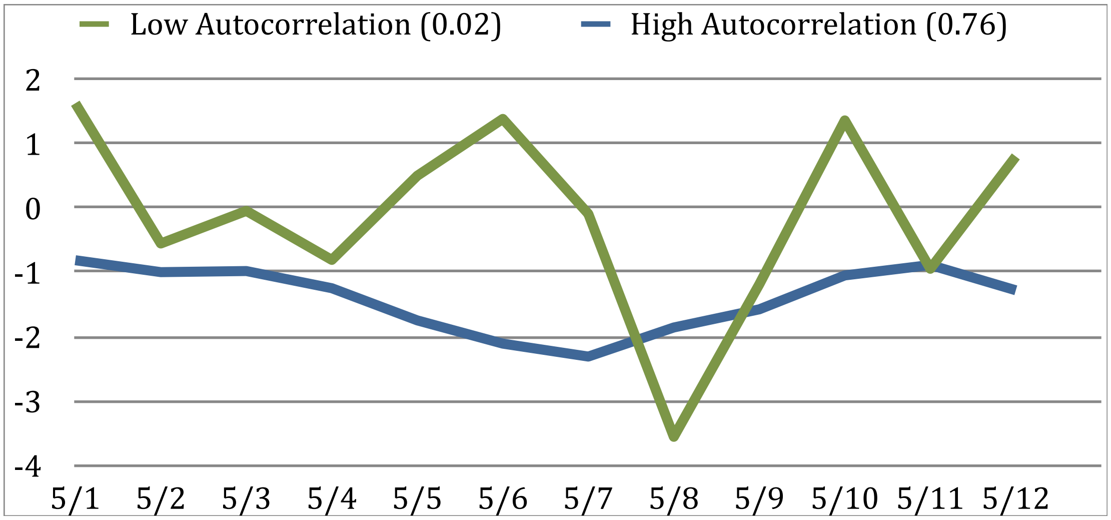


Figure 4: Autocorrelated Data Example

Collinearity is another characteristic of interest. While autocorrelation deals with the correlation of a variable with itself over time, collinearity describes how correlated a variable is with other variables. When building prognostic models, one would like the predictors to be well correlated with the outcome; however, model assumptions can be violated and performance can degrade if the predictors are also highly correlated with each other[54]. In the ideal case each feature included in a model contributes new information related to the outcome. If predictors are highly collinear, then there is likely a large overlap in the information content of the predictors as they relate to the outcome. Many biological variables, such as heart rate and blood pressure, are highly correlated with each other in healthy individuals.

The next characteristic of interest is the distribution of measurement occasions. As previously mentioned, each subject can have a different number of observation episodes. A cohort of subjects will therefore have distribution of the number of observation episodes[55]. The shape of this distribution can have important implications. In healthcare it is common to see exponential or power-law like distributions[19]. These types of distributions imply that extreme outliers frequently occur. For example, in a cohort of patients that underwent coronary artery bypass graft surgery, there were a nontrivial number of patients (5%) with a postoperative length of stay more than 5 times the average[56]. This variability suggests that there are unmeasured external factors with a large effect on the length of stay. Within the healthcare records of this cohort, the patients with long lengths of stay will generally have more clinical measures performed than those with shorter lengths of stay. Thus, the distribution of measurements is informative not only of sampling period, but potentially also of the outcome.

Variability in outcomes can come from two sources: inter-subject variability and intra-subject variability. These data characteristics both describe heterogeneity, but at different units of analysis. Inter-subject variance describes how heterogenous the group of subjects is, while intra-subject variance describes how variable the measurements are within the same subject. Researchers often perform statistical inference at the group level, but these inferences may not readily translate to individuals if intra-subject variance is significantly different from inter-subject variance[57]. These two different levels of variability can occur in both features and the outcome. Being able to accurately model both types of variability leads to statistical efficiency gains (achieving a set level of precision with less data) compared to modeling inter-subject variability alone[46]. Figure 5 provides a visual example of inter-subject and intra-subject variability. The left graph of Figure 5 shows a large variance in the values of the population over time. The right graph of Figure 5 shows the differences in variance each subject experienced over time. If the causes of variability in the outcome at either the inter or intra-subject level are unobserved, then the degree of that variability will set bounds on the reducibility of the model error. For example, increases in outcome measurement error will increase intra-subject variability, while increases in the effect size of unobserved genetic differences between subjects will increase inter-subject variability.

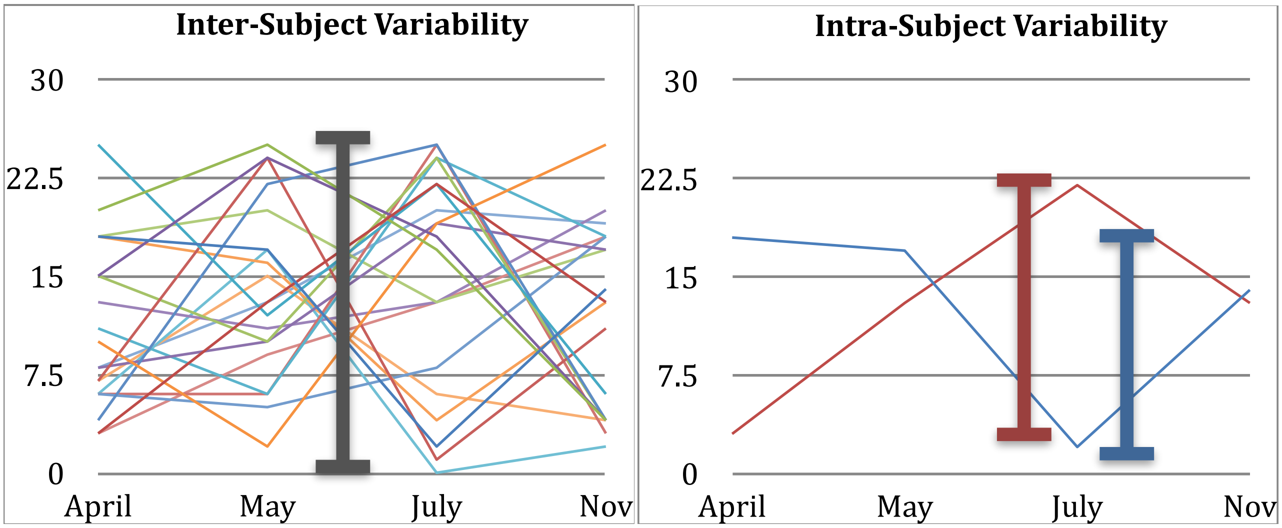


Figure 5: Comparing Inter-Subject Variability and Intra-Subject Variability

The data type of the outcome is another important characteristic to consider even though it is not specific to longitudinal data. Some of the more common outcome data types are unbounded continuous real numbers, un-ordered categories, ordered categories, counts, and bounded continuous real numbers.

For parametric models, the outcome type dictates which known distributions are best suited to minimize the model error. Choosing between distributions is a balancing act between distributional assumptions and model complexity[52]. For non-parametric models, the outcome type can change the amount of data required for precise estimates of model parameters. Models on categorical outcomes (classification) tend to be more data hungry than models for outcomes that are continuous real numbers (regression)[58].

We define the sampling scheme as the mechanism that dictates when measurements are observed. There are often socio-technical processes/policies that dictate when data are observed[59]. In the inpatient healthcare setting, most diagnostic tests and imaging are generally concentrated within the first 1-2 of days of the admission[60]. This research implies that measurement occasions may be more heavily concentrated in the early days of the admission compared to the later days. This type of skew or bias in the timing of observations may be informative to the model.

The last characteristic of direct relevance to this work is synchronicity. We define a data set where all variables (features and outcome) are complete for all measurement occasions as synchronous. A data set where measurement occasions have missing values will be referred to as asynchronous. Observational clinical data is often asynchronous; however, this property is dependent on the temporal definition and representation of the measurement occasions[61]. In observational clinical data, the absence of values within a measurement occasion is generally informative and is usually related to the expected value of that variable by clinicians[62]. Variables with non-random missing values are difficult to address and can be a significant source of bias in prognostic model development depending on the extent to which values are missing[62].

A characteristic that is important to longitudinal modeling, but is out of scope for this study is stationarity. A variable is considered stationary if its distribution (joint, marginal, and conditional) is invariant to time[49]. By the definition the central moments (mean, variance, skew, kurtosis, ect) of a distribution (if they exist) should also be invariant to time. There are multiple types of stationarity. Wide sense stationarity (WSS) is more general. WSS does not impose conditions on the joint, marginal, and conditional distributions[49]. WSS only requires that the mean and the covariance of a random variable remain static over time[49]. Yet another form of stationarity is trend stationarity. A trend stationary process is defined as the composition of a function over time and a stationary stochastic process[49]. To be trend stationary, one should be able to regress out the trend, leaving a WSS or stationary stochastic process[49]. For example, many adult Americans could express their daily weight measurement as trend stationary: stochastic variability with a polynomial upward trend over time. Many variables and models in observational clinical data are non-stationary, because the practices, therapies, diagnostics, workflows, and regulations of healthcare are in constant flux[36].

Data Representation for Prognostic Modeling

Most predictive models require data to be formatted as a matrix format before they are able to train[47]. We will refer to this input matrix as the design matrix. The dimensionality of the design matrix can vary depending on the model. The simplest design matrix in the atemporal setting represents subjects as rows and the variables (predictors and outcome) as columns. This is known as a wide-format-design matrix[46]. While the wide-format-design matrix is fairly straitforward way to represent data, it has some disadvantages in the longitudinal setting[46]. Wide format design matrices can be sparse if the number measurement occasions are not uniformly distributed or if there are subjects with large outliers of measurement occasions. Table 1 provides an example of how the wide-format-design matrix can be inefficient with memory and disk space for longitudinal data. The table has heart rate (HR) as time dependent outcome of a theoretical exercise treatment where time is measured in the number of days since enrollment in the study. Table 1 has to accommodate the outlier of 4 observation episodes by having a column for each time-dependent variable for each observation episode—resulting in a many cells that are empty. This representation also assumes that a subject will have no more than 4 observation episodes.

Subject ID	1st HR	1st Time	2nd HR	2nd Time	3rd HR	3rd Time	4th HR	4th Time	Treat
1	76	0	75	14					0-No
2	86	0	82	12	80	27	81	40	1-Yes
3	65	0	66	15					0-No
4	81	0							1-Yes

Table 1: Wide-Format-Design Matrix Example

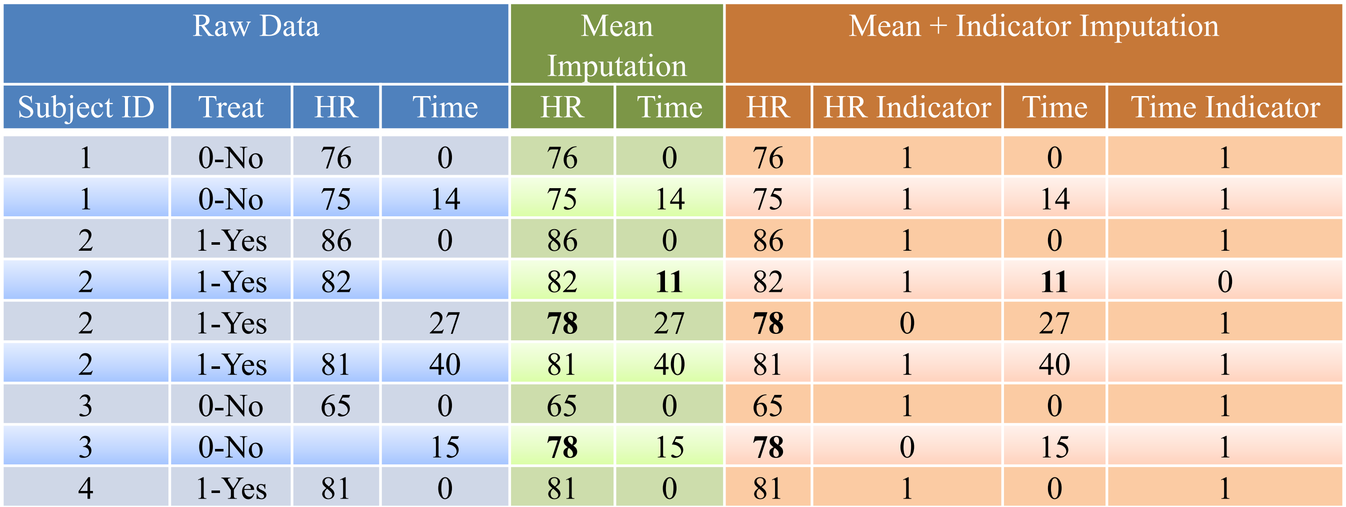
The long-format-design matrix addresses some of the space inefficiencies of the wide format by representing data at the measurement occasion level instead of at the subject level[46]. In this representation, each row signifies one measurement occasion for one subject. The long-format-design matrix is most advantageous in situations where there are few time-independent features, and the number of measurement occasions is highly variable between subjects. The small number of time-independent features minimizes the amount of data that requires repetition for every measurement occasion of that subject, while only storing data that was actually observed. Table 2 shows the same data found in Table 1 in long format. The time independent subject ID and treatment assignment variables are repeated for each measurement occasion. If there were many other time independent features, then one can imagine that this representation might not be particularly efficient. Beyond efficiency concerns, often it is the model and the model assumptions that are the most important considerations in choosing between wide and long format[46]. However, the simple examples in Table 1 and Table 2 demonstrate that data representation can have a significant effects on the amount of missing data and how the structure might codify implicit assumptions.

Subject ID	HR	Time	Treat
1	76	0	0-No
1	75	14	0-No
2	86	0	1-Yes
2	82	12	1-Yes
2	80	27	1-Yes
2	81	40	1-Yes
3	65	0	0-No
3	66	15	0-No
4	81	0	1-Yes

Table 2: Long-Format-Design Matrix Example

Missing data in the previous example was an artifact of data representation due to unequal observation episodes between subjects. Asynchronous data collection can also result in missing values. This type of missing data can be more difficult to handle, because data representation alone may not be able to create a complete design matrix[46]. Whether data is missing because of asynchronous measurements or due to unequal measurement occasions, there are 3 statistical mechanisms for said missing data. When all values within a covariate are equally likely to be missing, statisticians describe the data as Missing Completely At Random (MCAR)[63]. Data that is MCAR usually occurs due to unrelated and random processes, such as lost laboratory samples or equipment malfunctions. Another mechanism of missing data is Missing At Random (MAR)[63]. Data that is MAR can be estimated by using the other variables that are recorded[63]. MAR data points are not associated with the outcome[63]. For example, people without health insurance generally will have fewer measurement occasions for their blood pressure, but this variable is not directly causally related to their blood pressure. Lastly, data can be Not Missing At Random (NMAR)[63]. NMAR data cannot be reliably addressed with observed covariates[63]. Data that is NMAR occur when the value of a variable is directly related to whether the variable was recorded. For example, blood pressure would be NMAR if individuals with high blood pressure purposefully avoided seeking medical care in order to avoid costly therapy. The absence of high blood pressure individuals in the recorded data would preclude attempts to impute the missing values using data-driven methods[62]. The mechanism for missing values in observational clinical data is often NMAR[62]. This poses a challenge for model developers as methods that create sub-models to estimate missing values, such as multiple imputation, may be biased[64, 65]. Simple methods such as mean imputation (using the variable mean to replace missing values) are also likely to be biased, because the absence of data is directly related to certain values of the variable[66].

Many researchers have explored different representations and methodologies for addressing missing data[64-72]. Some methods use statistical models utilizing variables that are complete to build models to impute values[64, 65, 68,, 70, 71]. Other researchers use simple imputation methods that are applied to specific patterns of missing data[66, 69]. Researchers have also turned to different types of data representations such as interval based abstractions or the creation of missing indicator variables that are combined with mean imputation[66, 69, 72]. While missing data is a topic that is closely related to temporal representation, it is not the primary focus of this work. This work will specifically evaluate two related imputation methods commonly used in the machine learning literature: mean imputation by itself and mean imputation paired with an indicator variable for observed values (mean + indicator imputation)[66, 68, 73, 74]. We chose these two imputation methods because they resulted in better comparisons with past work[73, 74]. This point will be elaborated on in future chapters. Mean imputation is a popular method because of its simplicity of implementation and because the method does not alter the observed variable mean[66]. Mean imputation also preserves the degrees of freedom of the model. The mean + indicator imputation adds predictors (increasing the degrees of freedom) that, in terms of explained variance, can only result in a better fit model compared to the original set of predictors. Table 3 provides an example of missing data due to asynchronocity, mean imputation, and mean + indicator imputation.



Raw Data				Mean Imputation		Mean + Indicator Imputation			
Subject ID	Treat	HR	Time	HR	Time	HR	HR Indicator	Time	Time Indicator
1	0-No	76	0	76	0	76	1	0	1
1	0-No	75	14	75	14	75	1	14	1
2	1-Yes	86	0	86	0	86	1	0	1
2	1-Yes	82		82	**11**	82	1	**11**	0
2	1-Yes		27	**78**	27	**78**	0	27	1
2	1-Yes	81	40	81	40	81	1	40	1
3	0-No	65	0	65	0	65	1	0	1
3	0-No		15	**78**	15	**78**	0	15	1
4	1-Yes	81	0	81	0	81	1	0	1

Table 3: Raw vs Mean Imputation vs Mean + Indicator Imputation

Temporal Data Representation

Temporal representation/abstraction attempts to apply mathematical or knowledge based transformations on timestamped data to aid in the discovery and use of patterns over time[39]. Researchers have evaluated many different strategies for both longitudinal and time series type data[36-43]. Below is a quick survey of methods with some of their advantages and disadvantages.

The simplest temporal representations treat measurements as fixed points and instead manipulate the representation of the timeline where these points lie. One common representation, which we will call absolute-time, has an anchoring event from which time is measured as a count variable from that anchor[75]. The anchoring event can be shared between subjects or it can be relative to each subject. Figure 6 provides an example of how a shared versus a relative anchor point can lead to a different representation.

Using the same point mass idea for measurements, one could frame time as the distance from one point to the previous point. This relative-time framing uses the previous measurement for reference instead of a fixed point[68]. There are applications where it makes more sense to weight decay/growth from the last measurement than from a fixed anchor point. For example, the relevance of a political poll may be more closely related to the poll's timing relative to the previous poll than to the moment when the candidate began their campaign. Figure 6 also shows how relative-time compares to other point mass time representations. This representation assumes stationarity in the process as it would be difficult to distinguish earlier observations from later ones.

Another type of point mass representation seeks to smooth out differences in measurement occasions. What we will name, sequence-time, provides either a relative count of the number of measurements preceding the current measurement. The sequence-time representation provides an ordinal smoothing that results in potential information loss, but may result in more robust stationarity in the the temporal patterns found[36]. Figure 6 also shows how this representation might compare to the other point mass representations mentioned.

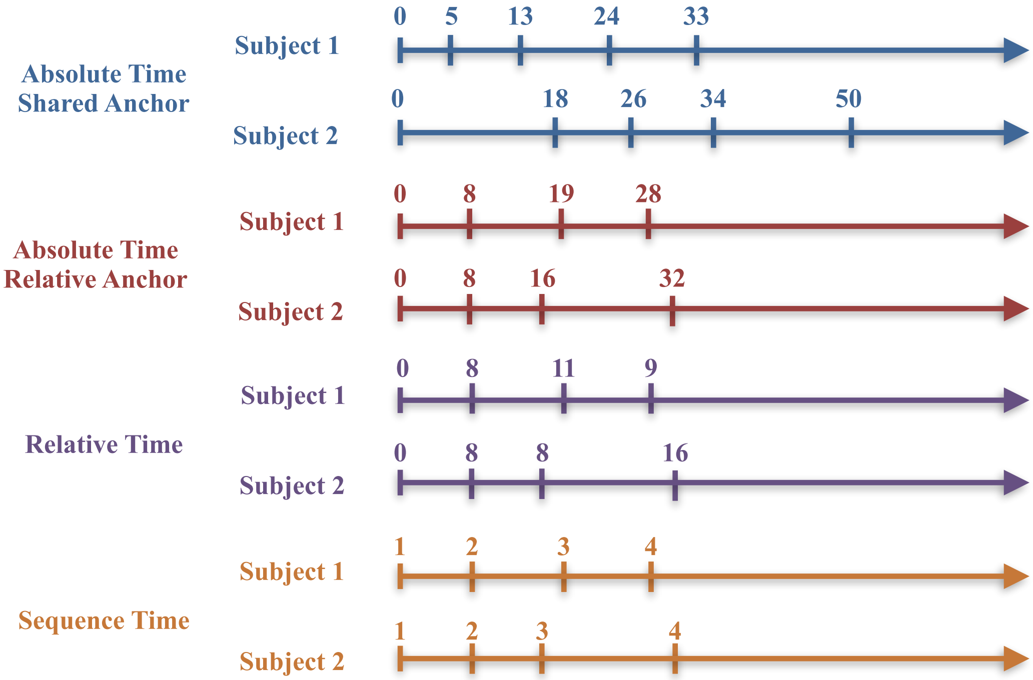
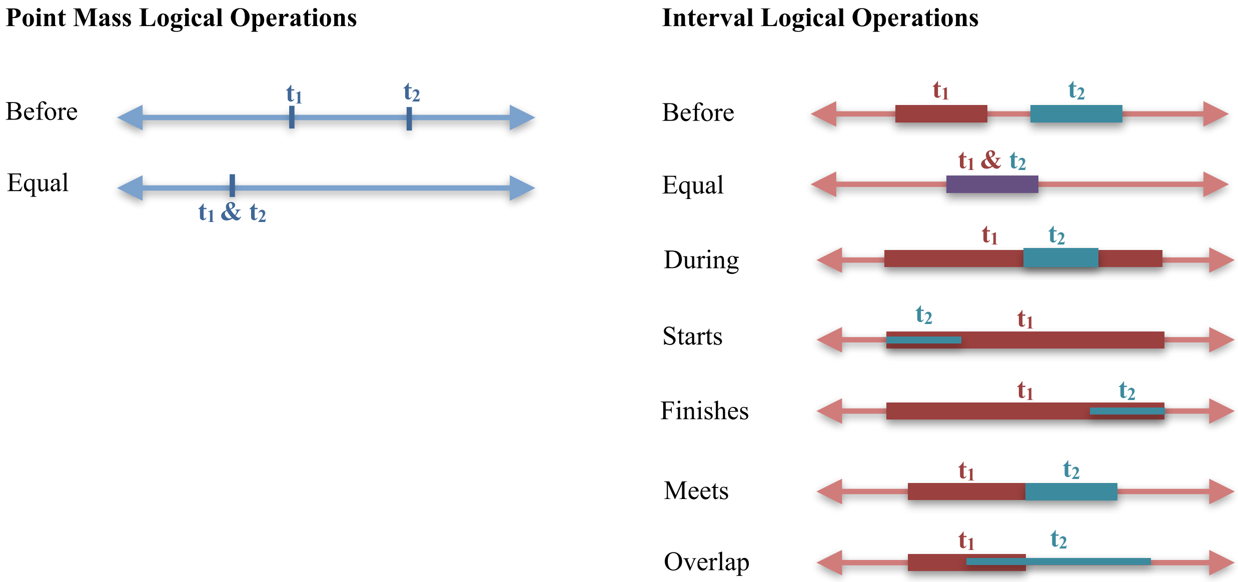


Figure 6: Different Representations of Time for Point Events

Somewhat related to sequence-time are graphical and string sequence representations. In a graphical temporal representation the sequence of observations are represented using nodes and edges[76]. One observation node is connected to the next through an edge. The observation nodes may be abstractions of higher level concepts or categorical variables[76]. All of the subjects are represented on the same graph structure, opening the door for graph theory based analyses and algorithms[76]. String sequence representation on the other hand, looks to represent the observations as an ordered string of events or categories[77-79]. This type of representation draws parallels to Deoxyribonucleic acid (DNA) analyses and uses the algorithms of bioinformatics for temporal pattern discovery[77]. The graphical and string sequence representations work best with categorical data and have a difficult time representing continuous measures. Categorizing data can be difficult and can lead to a loss of statistical efficiency.

Beyond a point mass representation of measurements, one can abstract observations into intervals or trends[39-42]. These intervals or trends can be created through knowledge based means or through interpolation methods. Logically operating on intervals is more challenging than on points; intervals have more operands[43]. Figure 7 depicts different algebraic operations for points and intervals.



Point Mass Logical Operations

Interval Logical Operations

Figure 7: Algebraic Operations for Points and Intervals

The goal of knowledge-based temporal abstraction is to go from raw data (often continuous variables) to higher level qualitative concepts that that form an interval[40, 42]. This abstraction accomplishes a smoothing of intra-subject variability (by abstracting to a category) and a smoothing over time that includes information on duration (by abstracting to an interval). Knowledge based temporal abstraction gets the gains of smoothing, but also enriches the data by going from time stamped points to intervals[40, 42]. However, knowledge based temporal abstraction is not without costs. The knowledge bases necessary to abstract diverse sets of data are expensive to curate and maintain in terms of time and expertise[80]. To abstract data into higher level concepts requires detailed knowledge of each data element, such as how rapidly an observation may be able to shift from its current category to another category[80]. Defining meaningful categories for each data element is not trivial to begin with. Furthermore, working with intervals can be more computationally challenging than working with points, and there are fewer algorithms to take advantage of[81, 82].

In what we are referring to as interpolated temporal abstraction the goal is to smooth the intra-subject variability using statistical methods[83-87]. This type of temporal abstraction is easier to implement than knowledge-based temporal abstraction, as it is not attempting to abstract data into expert-derived higher level concepts. This type of abstraction can take a few different forms. Some researchers split the observation period of each subject into adjacent windows and use an aggregate statistic to summarize the measurements that occur in each window (e.g., mean, median, mode, count, variance, etc)[86, 87]. Windows can be defined as being equally long or can grow or shrink in length due to variability in the measurement values. We will call this representation window-time. These types of techniques are fairly common to signal processing that tends to have a high density of data. Another approach seeks to use observed values to statistically estimate a continuous process[83-85]. This modeling based approach interpolates the values between observations, allowing the users to sample any time point they wish. This statistical interpolation can have varying degrees of sophistication from using linear interpolation to fitting a subject specific Gaussian process model for each variable[83-85]. Figure 8 shows some examples of window-based and model-based interpolation.

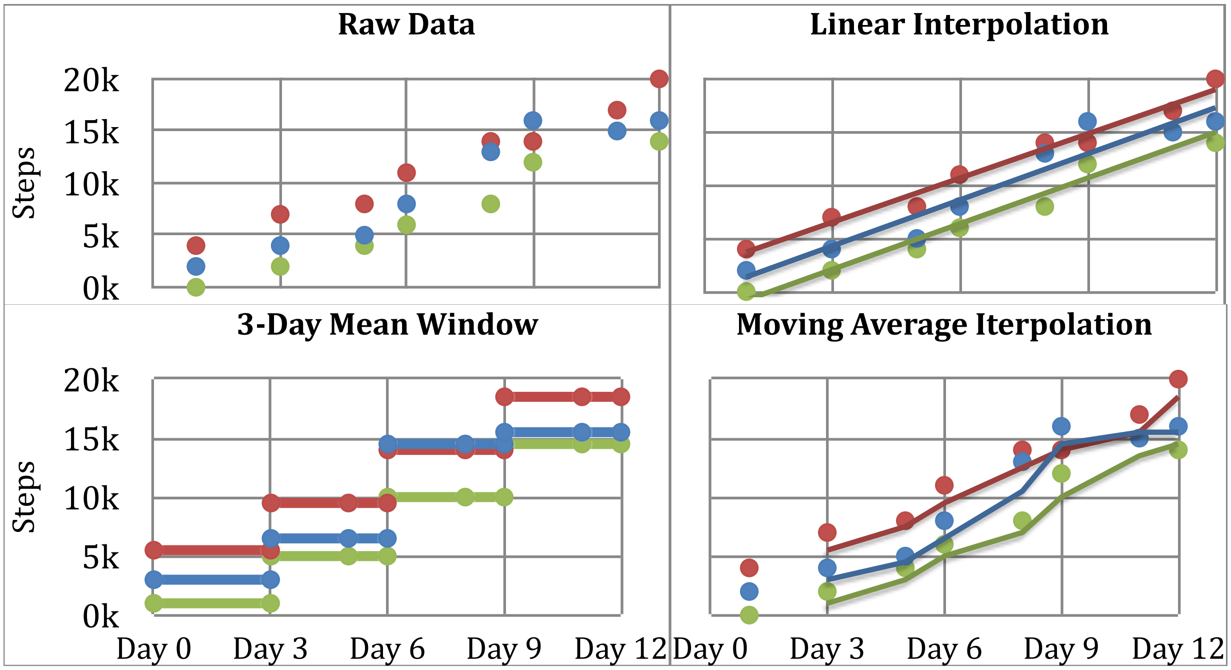


Figure 8: Examples of Interpolation Based Temporal Abstraction

Statistical Longitudinal Models

Statistical longitudinal models have a longer history than their deep counterparts and are the tools of choice for causal inference in longitudinal or clustered data[46, 88]. In this section we will briefly touch on how two of the most popular longitudinal hierarchical models work and their advantages and disadvantages for prediction and inference. We used these simpler and more interpretable models as our baseline comparison and a means to learn more about the behavior of deep sequence models with longitudinal data.

Marginal models are powerful inferential models with only a few assumptions. Marginal models can handle data in the long format and and estimate parameters through a flexible generalized estimating equations (GEE) approach[46]. The marginal model has three major assumptions: first, the mean response depends on additive features through a known link function[46]. As with other generalized linear models, the link function enables the marginal model to fit different types of outcomes such as counts or unordered categories[46]. The inverse of the link function maps the unbounded sum of the weighted predictors to what can be a strictly bounded outcome space[46]. Said another way, one is assuming that the mean of the outcome of interest is linearly related to the predictors after applying a known function. The second assumption is that variance of the outcome given the features is time-invariant (stationary)[46]. This type of model cannot handle fundamental changes to the process it is trying to describe. The last assumption is that the correlation of repeated measures for the same subject is a function of the mean[46]. The implication of this assumption is that correlations of individual responses are some variation of population wide effects. In situations where subjects are very different from each other, this assumption may not be valid. However the marginal model with GEE parameters is robust to this assumption being violated, though at reduced efficiency[46]. The marginal model primarily works with point mass absolute time with a subject specific anchor. This model is a natural tool for population level inference, but can be a poor choice for subject specific predictions or inferences, because the model does not have mechanisms to handle data with high degrees of intra subject variability. The marginal model is designed to produce a robust measure of the average response[46].

Generalized linear mixed effect model (GLME) offers a different approach to modeling longitudinal data than the marginal model. In a GLME the developer not only specifies the unit of analysis (how data are clustered), the features, and the outcome, but also which of those features vary at the inter-subject level (fixed effects) and which features have significant intra-subject variance (mixed/ random effects)[46, 88]. GLME models are also more personalized to the subjects the model is fit on. The GLME model will estimate subject specific parameters for all of the random effects specified in the model. This personalization does come at the cost of additional assumptions not made by the marginal model. Assumptions by the GLME model include: all random effects have a mean of zero and a multivariate normal distribution, the measurement error of a given observation is uncorrelated with other observations for that subject, the random effects are statistically independent for the measurement error, and that the outcome depends on additive fixed and mixed effects through a known link function[46, 88]. This model also only works with an absolute temporal representation with a subject specific relative anchor, however the use of time related random effects sets the covariance structure for observations over time[46, 88]. For example, the specification of a time related random intercept is equivalent to assuming a constant correlation between observations over time[46, 88]. The random effects are fitted through a two step process. First subject specific models that only include random effects are fit. Since there are many subjects, this creates a distribution of random effects coefficients. The mean and variance of the subject specific coefficients from stage one, inform the inter-subject parameters that are fit in stage 2 using the mean and variance of the random effects coefficients. The model coefficients go through an iterative optimization process that maximizes some version of the model likelihood given the data. This mathematical optimization makes use of parameter estimates to iteratively converge to a stable solution of model coefficients. The GLME model makes many more structural assumptions than the marginal model. These assumptions can be difficult to validate and much more care is needed when designing a GLME model. All that said, the ability of GLME models to create personalized predictions through random effects makes them well suited to replicating the high levels of inter-subject and intra-subject variability in clinical data.

We will introduce some GLME specific notation that we refer to later on when using the GLME to generate data. The measurement occasion values for a specific subject are generated using Equation 1. We adopt a positional notation where the first index refers to global parameters, the second index refers to the subject (indexed with the variable i), and the third index refers to the measurement occasion (indexed with the variable j). Capital letters denote a matrix, while lower case letters with a directional bar represent vectors. Beta symbolizes a fixed effect model coefficient, b represents a model random effect coefficient, and epsilon is the random error. The previously mentioned link function is represented with Phi. As an example we will define a model that has three features x_1, x_2, and time. The model will have four fixed effects: an intercept, effects for x_1, x_2, and an effect for time. The model will also have two random effects: a random intercept and a random time effect. The random effects allow each patient to have their own intercept term and their own time slope. We demonstrate what we mean by subject specific intercepts and time slopes in Figure 9. Figure 9 also contrasts how a marginal model would have one prediction line for all subjects, while the mixed effects model is more personalized.

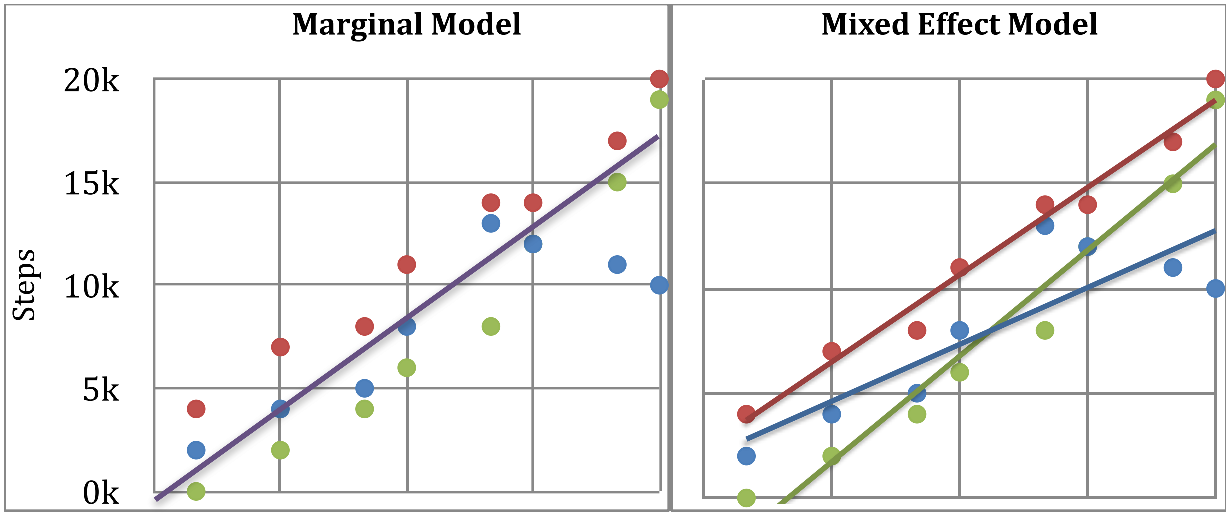


Figure 9: Marginal Model vs Mixed Effects Model

Putting all of this notation together, we have that the outcome for a specific subject for a specific measurement occasion is equal to the inverse link function applied to the global intercept plus the features times the fixed-effect coefficient plus a subject specific intercept term plus the time multiplied by a subject specific time coefficient. The quantity after applying the inverse link function is then summed with a random error that is specific to both this particular measurement occasion and subject. Equation 1 provides a notational example of the model of x_1, x_2, and time specified above.

$$y_{-,i,j} = \phi^{-1}(\beta_0 + \beta_1 x_{1,i,j} + \beta_2 x_{2,i,j} + \beta_3 time_{-,i,j} + b_{0,i} + b_{1,i} time_{-,i,j}) + \epsilon_{-,i,j}$$

Equation 1: Model Definition for a Single Measurement Occasion

If one were to aggregate all of the outcomes for a specific subject, then one would see Equation 1 would become Equation 2. The main difference between the two is that where we once had single values for the single measurement occasions, we now have vectors and matrices that are comprised of these different components. In Equation 2, y symbolizes all the outcomes for a subject, x symbolizes all of the features for a subject, and z represents only the features with random effects. A subtle difference in this summarized representation is that the X and Z matrices both begin with a column of ones to accommodate the intercept terms.

$$y_{-,i} = \phi^{-1}(\vec{\beta} X_{-,i} + \vec{b}_{-,i} Z_{-,i}) + \vec{\epsilon}_{-,i}$$

Equation 2: Model Definition for a Single Subject

Deep Learning and Deep Sequence Models

Deep models are a type of of machine learning model loosely based on the structure of neurons[89]. A machine learning model does not require the exact relationships of the predictors to the outcome to be specified by the model developer and instead uses the training data to estimate the form of these relationships[89]. The lack of pre-specification is an advantage of machine learning methods compared to statistical models; however this flexibility often comes at the price of efficiency (more data is required to achieve a set level of precision)[90]. As data bases and sources have grown more plentiful there is a growing enthusiasm about the capabilities of these deep models to positively impact the system of healthcare[5, 91-95]. Deep learning methods have achieved state of the art performance results on many different healthcare related benchmarking prediction problems[73, 74, 91]. The application of deep sequence models to healthcare problems still involves many decisions by model developers that are arbitrary or data driven through a trial and error process[76]. Before looking to improve the decision making process, it is important to have a basic understanding of deep sequence models and how they work. Deep sequence models build off of the fundamental concepts of the basic feed forward neural network. We will start by providing background on this basic type of deep model before proceeding to the different kinds of deep sequence models.

The basic feedforward neural network is atemporal and exists as a directed graph that feeds inputs into neurons that either activate or remain dormant. The states of the neurons are then passed to the next layer as inputs, which continue the cycle until the last layer of the network produces one or more outputs. Figure 10 provides a pictorial example of a fully connected architecture where each input (data element or neuron state) of the previous layer feeds into each neuron at the current layer that produces two predictions (outputs). There are many ways to configure the graph of neurons, inputs, and outputs. Figure 10 is but one example. A neural network is known as a deep neural network when it has more than one layer in between the layer that consumes the data and the layer that produces outputs.

Subject 1	
Feature Name	Feature Value
Heart Rate	85
Temperature	98.7
Body Mass Index	28

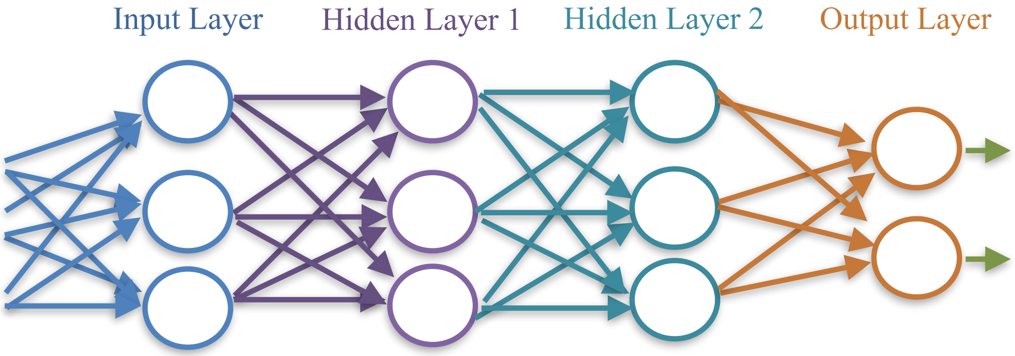


Figure 10: Fully Connected Feedforward Neural Network

Each neuron uses an activation function to determine whether it activates. Each neuron has two types of parameters weights and biases. Inputs to the neuron are multiplied by the weighting parameters before being summed with the bias parameter[51]. The weights are the learned effects of the inputs, and the bias adjusts the firing frequency. Originally neural networks used sigmoid or hyperbolic tangent (tanh) functions as the activation function, but this resulted in networks that did not learn from data very well if the network had more than one hidden layer[51]. This training problem became known as the vanishing gradient problem and can be solved by using an asymmetric activation function known as the Rectified Linear Activation Unit (ReLU). While both the sigmoid and tanh functions have curvature that bounds them between 0 and 1, the ReLU activation function is a ramp function defined as f(x) = max(x,0). The ReLU function does not have an upper bound, meaning that perturbations of x while x > 0 will always have some effect on f(x).

To understand why ReLU solves the vanishing gradient problem, one has to look at the inner workings of the neural network. The neural network optimizes those previously mentioned weight and bias parameters using the gradient descent algorithm[51]. Gradient descent is an algorithm for minimizing a function of one or more variables[91]. The function the network is minimizing is the loss of the model (e.g., the mean squared error) as a function of the model weight and bias parameters. The algorithm searches around the local region for the largest gradient by perturbing the weight and bias parameters from some initial value. Once the largest gradient is found, the algorithm moves opposite of that gradient and adopts those new weight and bias values as its current position. This process is repeated until a stopping condition, such as reaching a point where the result of the loss is stable around its local region. The neural network calculates the gradient of the model through all of its neurons and layers using the back propagation algorithm[51]. The network can be thought of a function of functions, where the nodes at later layers are dependent on all of the nodes in previous hidden layers. This algorithm takes advantage of the Chain Rule to calculate the gradient at each layer starting from the last hidden layer and moving backward to the first. Back propagation uses the Chain Rule to compose the derivatives of each layer together (with multiplication) to reflect the cascade of effects from earlier layers to later layers for each hidden node in the network[51]. The nested structure of the back propagation algorithm is at the crux of the vanishing gradient problem. As the number of hidden layers grow, the number of terms composed and multiplied together in the first layer also grows. Traditional sigmoid activation functions (e.g. sigmoid or tanh) have a derivative bound from above by 0.25. The derivatives of a single node are a product of weights and partial derivatives of the activation function. When all the nodes of all later hidden layers are composed together, multiplying an increasing number of values that are less than 1 forces the gradient toward 0. In this sense the gradient, which for the last hidden layer is generally adequate for optimization, is said to vanish in previous layers. One may see the opposite problem of an exploding gradient if the gradients from later layers tend to be greater than 1. In either case extreme gradient values (low or high) from later layers are propagated back to earlier layers. The overall result is that earlier hidden layers will be optimized by the Gradient Descent algorithm very slowly (vanishing) or erratically (exploding), if at all. This dynamic signified a delicate sweet spot of derivatives just large or small enough for back propagation to enable optimization of all model parameters. Since the ReLU function is a ramp, its derivative when activated is 1. A derivative of 1 prevents the multiplicative issues encountered by the sigmoid and tanh functions during back propagation.

A recurrent neural network (RNN) is a more complex version of the feedforward network specifically designed for sequences. An RNN has 1 layer for each element of the longest sequence. The RNN does not require each sequence have the same number of elements, but the data representation is similar to a wide format. Domains with long term dependencies require long input sequences (e.g. language translation). An RNN for this type of sequence unravels into a very deep feedforward network, leading to the vanishing/exploding gradient problem[96]. A key difference with RNNs is that in feedforward representation all the hidden layers have the same input weights. The equal weighting constraint across all the hidden layers is similar to repeatedly applying the same function onto itself. The weights in a feedforward network can have many more degrees of freedom than the RNN. The result is that the gradient is more unstable in an RNN than in the regular feedforward architecture[96]. The ReLU can solve the vanishing gradient problem for RNNs, but causes other problems because ReLU nonlinearities are unbounded. When the RNN is unraveled, the activation output can explode to values not representable as floating point numbers due to the very large number of hidden layers.

The long-short-term memory model effectively solved the RNN exploding gradient problem by changing the fundamental unrolled unit from a fully connected hidden layer to a block. Each block is made of five nodes[97]. The current state node holds the current activation state, and feeds into itself with a unit weight. There is a forget node with sigmoid activation that weights the past output and current input to decide which elements of the current state to forget (multiply by 0). The forget gate updates the current state through an element wise multiplication. There is an input gate which has two nodes. One node weights the elements of the inputs of the current time-step and the outputs of the previous time-step to decide which elements of the current state to update (sigmoid activation). The other input node decides what the updated values should be (using tanh activation) based on weighted inputs from the current time-step and weighted outputs from the previous time-step. The results from both input nodes are multiplied and then added to the current state. Lastly there is an output gate that determines which of the elements are output to the next node (sigmoid activation). The output to the next memory cell is equal to the current state put through a tanh activation function and then multiplied by the output node. The effect of this structure is that the gradient becomes a sum across time-steps, not a product. Summation leads to much more stability in the gradient, resolving the vanishing and exploding gradient problems[97]. Figure 11 illustrates the described structure of the LSTM model. There are other configurations of blocks with the different gating structures, but the transformation of the gradient calculation from being multiplicative to additive is the same[68, 98].

The last deep sequence model relevant to this study is the feedforward network with a sequential

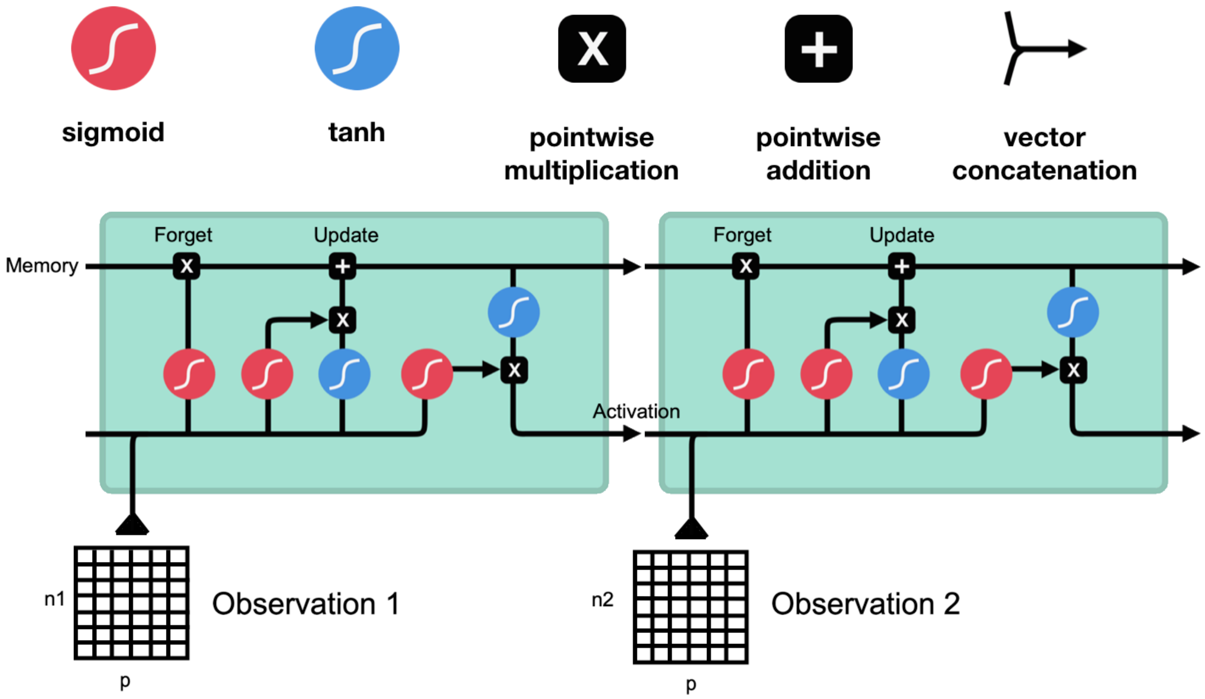


Figure 11: LSTM Block Diagram (Adapted from 99)

multi-head attention mechanism. The attention mechanism can be combined with other architectures such as the LSTM; however, a simple feedforward network can achieve parity with LSTM performance when combined with a multi-head self-attention mechanism[45, 100]. The self-attention based transformer model (Attention model for short) is slightly different than other sequence models. The predictors are concatenated with a positional encoding that the self attention mechanism can interpret as the order of the inputs. The self-attention mechanism itself is a means to to prioritize information from past parts of the sequence that are specific to the current value[45]. This prioritization happens through the optimization of query, key, and value parameter matrices[101]. These matrices weight how relevant each of the past inputs are to the current value and position in the sequence. The current piece of the sequence and the resulting weighted context vector are then fed into the feedforward network to produce a prediction[101]. The multi-head self attention mechanism extends this concept by have multiple query, key, and value parameter matrices[101]. Each part of the sequence produces its own context vector for each attention mechanism. The parameter matrices for each attention mechanism are randomly initialized, which can result in different prioritizations for the same input and position[101]. Yet another weighting matrix helps to aggregate the multiple context vectors produced by the multiple attention mechanisms into a single context vector[101]. The additional weighting matrix needed to aggregate the results of the different attention mechanisms suggests that the number of parameters grows super-linearly with the number of attention heads.

Training and Tuning Deep Longitudinal Models

Deep sequence models have many parameters that are optimized to fit the training data. Beyond those parameters that are directly tuned through gradient descent or other algorithms are hyper-parameters that must be decided on before the model starts learning from data[51]. These hyper-parameters can have a significant impact on model performance and can relate to the architecture of the deep model or settings controlling how the model is trained[51]. Deep models are made up of different modular components that can be ordered and combined in many different configurations[51]. For example, one might have data that is fed through an LSTM using a self-attention mechanism, which then flows into a feedforward network that produces a forecast.

As with data representation, there are implementation and maintenance considerations for hyper-parameters[102, 51]. Table 4 defines some common hyper-parameters for LSTM, Attention, and feedforward models. Each parameter generally has a Goldilocks zone that balances the tradeoffs of hyper-parameters that are tuned to be too small or too large.

Hyper-parameter	Type	Related Model	Definition
Batch size	Training	Attention, Feedforward, LSTM	The number of subjects propagated through the network to update model parameters. All training subjects eventually are used, but they are sent through one batch at a time.
Depth	Architecture	Attention	The size of the query, key, and value matrices. More depth results in more weighting parameters.
Drop out rate	Training	Attention, Feedforward, LSTM	The rate at which weight parameters are randomly omitted during model training
Hidden dimension	Architecture	Attention, Feedforward, LSTM	The number of input parameters expected by a hidden layer in a neural network
Learning rate	Training	Attention, Feedforward, LSTM	How much parameter weights should be changed in response to the model error
Loss funciton	Training	Attention, Feedforward, LSTM	How the neural network should measure model error. The subject of optimization as a function of the model parameters
Number of layers	Architecture	Feedforward	The number of stacks of neurons to propagate data through
Number of heads	Architecture	Attention	The number of attention mechanisms to train
Optimizer	Training	Attention, Feedforward, LSTM	The algorithm used to adjust the model parameters to minimize the loss function.
Weight decay	Training	Attention, Feedforward, LSTM	A penalty parameter that shrinks weights toward 0: shrinkage takes place within the optimizer

Table 4: Hyper-parameter Definitions for Deep Models

The goal of hyper-parameter tuning is to find a combination of hyper-parameters that helps to produce the best fit model[103]. One should exclusively tune hyper-parameters on training data (versus evaluation data) to prevent overly optimistic performance assessments of the model[103]. There are different strategies for choosing hyper-parameter values beyond evaluating all possible hyper-parameter combinations. The grid search scans fixed ranges of hyper-parameter values resulting in a grid of possible hyper-parameter combinations. The grid search tends to be most tractable for models with only a few hyper-parameters to tune[104]. A randomized search, where combinations of hyper-parameters are drawn uniformly at random, can be more efficient and effective than a grid search, when there are many possible hyper-parameter combinations[105, 106]. Alternatively, one could use optimization algorithms or even models to tune hyper-parameters[107, 108]. These approaches may introduce meta-parameters for the tuning algorithms and models, but these approaches have shown to be even more efficient still at finding optimal hyper-parameter combinations[107, 108]. This gain in efficiency does come at the cost of additional complexity and the implementation of yet another model.

Comparing and Evaluating Prognostic Models

To evaluate a prognostic model one must decide on the metrics for evaluation and the strategy for producing an estimate using those metrics. The goal of model evaluation is to get an estimate of how a model might perform if deployed into a real-world setting as well as a measure of certainty about said estimate[47, 52]. The type of outcome modeled along with the application should inform the choice of evaluation metrics[47]. Evaluation strategies tend to be outcome agnostic and are often dependent on practical constraints such as data size, computational power, and complexity of implementation[109].

There are many different metrics for evaluating predictive models. We will focus on the metrics most relevant to this study. In this work we evaluated a regression model, a prediction problem with a continuous real world outcome, and a classification model, a prediction problem where the model is attempting to sort patients into one of two unordered categories. In the realm of regression, model developers can attempt to evaluate the model through two different views: how much model predictions deviate from the true values (error or deviance) and/or how much of the variance in the data can be explained by the model. These two views have their own sets of associated metrics. When attempting to quantify error, the mean squared error (MSE) is a popular choice. Given n different predictions (y-hat) with true value y the MSE can be mathematically expressed as Equation 3. The MSE can be split into an error variance and error bias component[63]. This implies that one may be able to reduce error through a bias-variance tradeoff (adding bias to reduce variance), which is the underpinning of regularization also known as shrinkage[63]. The MSE metric can be difficult to interpret, because MSE is not on the same scale as the data. This difference in scaling between MSE and the data makes it difficult to communicate to users and decision makers.

$$\sum_{n}^{i} \frac{(\hat{y}_i - y_i)^2}{n}$$

Equation 3: Mean Squared Error

The mean absolute deviation (MAD) on the other hand is much more interpretable[110]. This metric does not express the variance or bias of the error like the MSE, but does keep the deviance on the same scale as the data. The MAD gives users a sense how much the deviance they can expect between the predictions and reality. Like MSE, the smaller the MAD the better the model. The MAD metric is calculated as shown in Equation 4.

$$\sum_{n}^{i} \frac{|\hat{y}_i - y_i|}{n}$$

Equation 4: Mean Absolute Deviation

Looking at the other view on regression fit, the explained variance score (EVS) is the ratio of the variance of the error over the variance of the true outcomes. The EVS is a more generalized version of the R^2 metric, because it can be calculated for machine learning models in addition to statistical models. The closer the EVS is to 1 the more of the variance of the outcome the model explains. This interpretation is similar to that of the R^2. The EVS metric can be calculated using Equation 5, where the terms with a bar over them represent a mean. There are other metrics for assessing regression model fit such as the log likelihood or the Akaike/Bayesian Information Criterion, but these methods do not generalize well to neural networks[111].

$$1 - \frac{\frac{1}{n}\sum_n^i (y_i - \hat{y}_i - (\overline{y - \hat{y}}))^2}{\frac{1}{n}\sum_n^i (y_i - \bar{y})^2}$$

Equation 5: Explained Variance Score

Metrics for binary classification differ from those for regression. The lack of a continuous outcome makes the use of a error or deviance measures less useful. As in regression, classification has two primary views for judging model fit discrimination and calibration. Discrimination is the measure of a model's ability to distinguish subjects between two or more classes[47]. Said another way, discrimination attempts to measure how often the model correctly labels the subjects. Calibration is the measure of a model's ability to accurately assign probabilities[47,]. For example, a model could be perfectly calibrated if of all the subjects a model predicted had a 40% probability of belonging to a particular class, 40% of that group actually belong to that class. In order for the model to be perfectly calibrated, this alignment between predicted and true probabilities would have to be true for all probabilities.

Sensitivity and specificity are two popular discrimination metrics for comparing models. If we consider one of the two classes positive and the other negative, then sensitivity expresses the true positive rate, while specificity expresses the true negative rate. Figure 12 visually defines sensitivity and specificity. Model developers value sensitivity and specificity because they are invariant to the prevalence of the positive class compared to the negative class. Said another way, in a model that is unbiased and evaluated in an unbiased fashion, the sensitivity and specificity should not change if tested against data with different mixtures of positives and negatives. However, the sensitivity and specificity do depend on the probability threshold used to split the classes. For example, labeling all subjects with a probability of 80% and above as positives and subjects with probabilities less than 80% as negatives will have different sensitivities and specificities than a threshold of 40%.

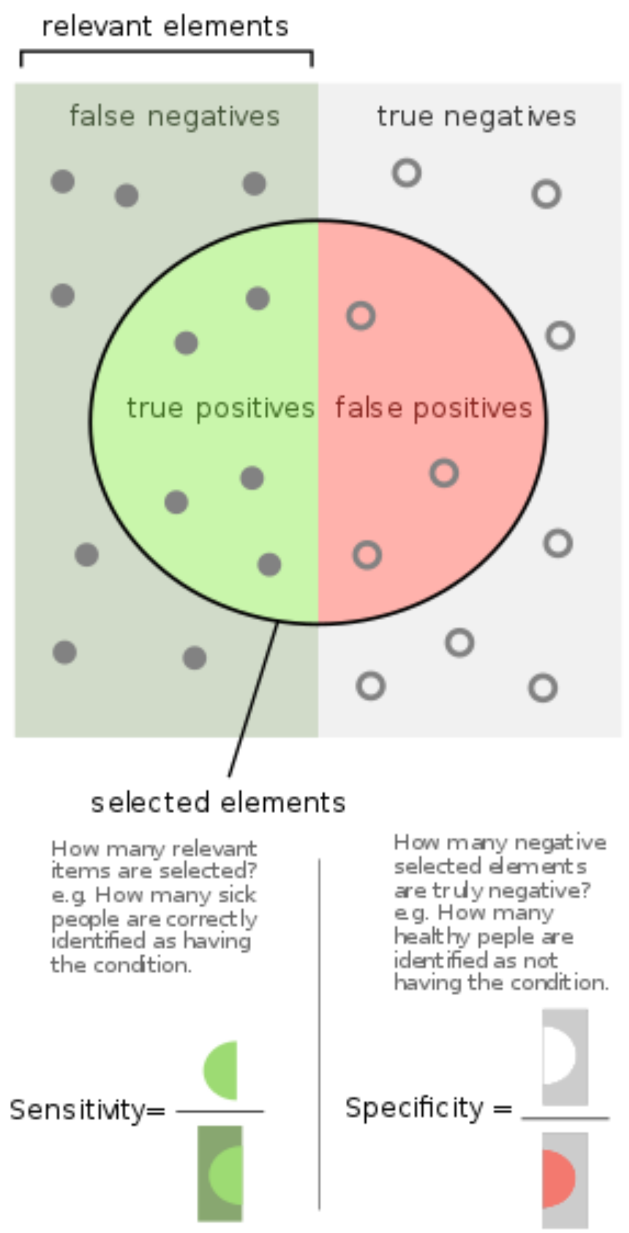

relevant elements
false negatives
true negatives
true positives
false positives
selected elements
How many relevant items are selected? e.g. How many sick people are correctly identified as having the condition.
How many negative selected elements are truly negative? e.g. How many healthy peple are identified as not having the condition.
Sensitivity=
Specificity =

Figure 12: Sensitivity vs Specificity (Adapted from [112])

The area under the receiver operator characteristic curve (AUROCC) takes the sensitivity and specificity metrics and summarizes them across all possible decision thresholds. Since sensitivity and specificity are class prevalence invariant, the AUROCC metric is invariant to both the class prevalence and the decision threshold. The AUROCC plots the model sensitivity as a function of inverse specificity (1-specificity) for different probability thresholds across the [0, 1] range for splitting one class from the other and then calculates the area under that curve. AUROCCs closer to one are considered superior. An AUROCC of 0.5 is viewed as the worst performance because that is the standard that a coin flip model would achieve. An AUROCC less than 0.5 is viewed to have a flipped label problem, as in one could achieve an AUROCC greater than 0.5 if they flipped the predicted labels. Figure 13 shows a Receiver Operator Characteristic curve plot; however the utility of this plot to decision makers compared to the aggregated AUROCC is a matter of current debate.

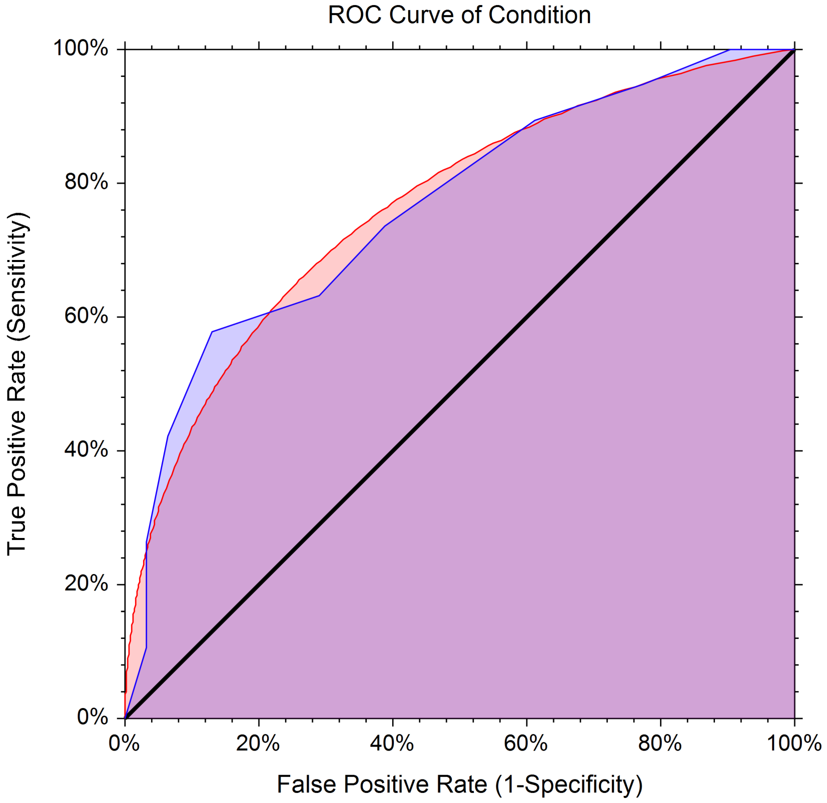


Figure 13: A Receiver Operator Characteristic Curve Plot

Some other metrics for discrimination include positive predictive value (PPV), the F1 score, and the are under the precision recall curve (AUPRC). The PPV is one of the most practically useful metrics, especially in the clinical domain, but it is dependent on class prevalence and the decision threshold[113, 114]. The F1 score is a geometric average of the sensitivity and the PPV, while the AUPRC is a graph of the PPV as a function of the sensitivity. All of these metrics are dependent on the class prevalence because they incorporate the PPV. These alternative metrics are practically useful for local validation of a model, but are not good at generalizing model performance across different data sets or settings[47, 52].

Calibration is an often overlooked model evaluation perspective[115-117]. Commonly seen metrics for assessing model calibration are the Brier score, the mean observed to expected ratio (MOER), and the cox calibration curve intercept and slope[116]. The Brier score measures the mean squared difference between the probability of the predicted class and the actual class. For example if the model predicted that a subject had a 40% probability of belonging to the positive class, but the subject actually belonged to the negative class, then the Brier score of this prediction would be $(0.4\text{-}0)^2 = 0.16$. Brier scores closer to 0 are superior, and a Brier score of 0.25 would be the equivalent of predicting all outcomes to have a probability of 50%. The Brier score can be decomposed into a mean measure of the observed versus expected ratio and a measure of discrimination related to AUROCC[116].

The MOER metric divides up predictions into equally wide quantiles of predicted probability. For each quantile the method averages the observed class probability for a quantile and divides that average by the average predicted probabilities for that quantile. MOER metrics closer to one are superior. MOER measures greater than one imply that the model is under-predicting the positive class membership, while MOER measures less than one suggest the opposite. The MOER methodology can be difficult to implement in practice because there are often quantile that do not have many if any associated predictions[118].

The Cox Calibration slope and intercept measures do not require binning of predictions into quantile. Instead the developer uses a logistic regression to regress the observed binary outcomes as a function of the predicted probabilities. A slope of 1 combined with an intercept of 0 indicates perfect calibration. Deviations from these values can signify over or under prediction of the outcome if the intercept is greater than or less than one respectively[119]. If the slope is greater than one, then the predicted probabilities are overly uniform[119]. If the slope is less than one, then the predicted probabilities are overly dispersed[119].

The metrics discussed give insight into different aspects of how a prognostic model fits the data. As mentioned at the beginning of this section, there are strategies that model developers can employ that can help produce estimates of these metrics (and their uncertainty) that may be externally valid to other plausibly related populations[109]. Three commonly used strategies are hold-out validation, cross-validation, and optimism adjusted bootstrap validation[109]. Holdout validation is the simplest strategy. It involves randomly dividing the data set into a training set and a test set. The model is trained, tuned, and potentially calibrated using data in the training set and the evaluation takes place on data the model has not seen before. For classification type problems, developers can bias the random sampling to preserve the class prevalence of the overall data set. This may be necessary when working with problems with low prevalence rates, as a purely random approach might create a test set without all of the classes. This evaluation strategy has the highest variability of of the three and can be biased. These characteristics are properties of the randomness involved in creating a single split. Model developers may need to use a resampling approach on the test set or repeat the entire hold-out validation procedure multiple times to get variance estimates for performance metrics.

Cross-validation has less variance and bias as a validation strategy than holdout validation, but requires more model fitting and complexity[109]. In cross-validation, the data is divided into equally sized folds; often the number of folds is greater than two and less than eleven. One fold is selected as the holdout for testing and the remaining folds are used for training. The process continues until each fold has had a turn of being the testing holdout. Each fold will have validation metrics associated with the run where that fold was considered the testing holdout. The developer can then report the average of all the runs along with an estimate of variance. The more folds one creates at the start of the process, the more models will need to be trained, tuned, and finally evaluated. Folds can be randomly created through sampling without replacement and as in the holdout validation this sampling can be biased to preserve class prevalences. Figure 14 visualizes a simple version of cross validation.

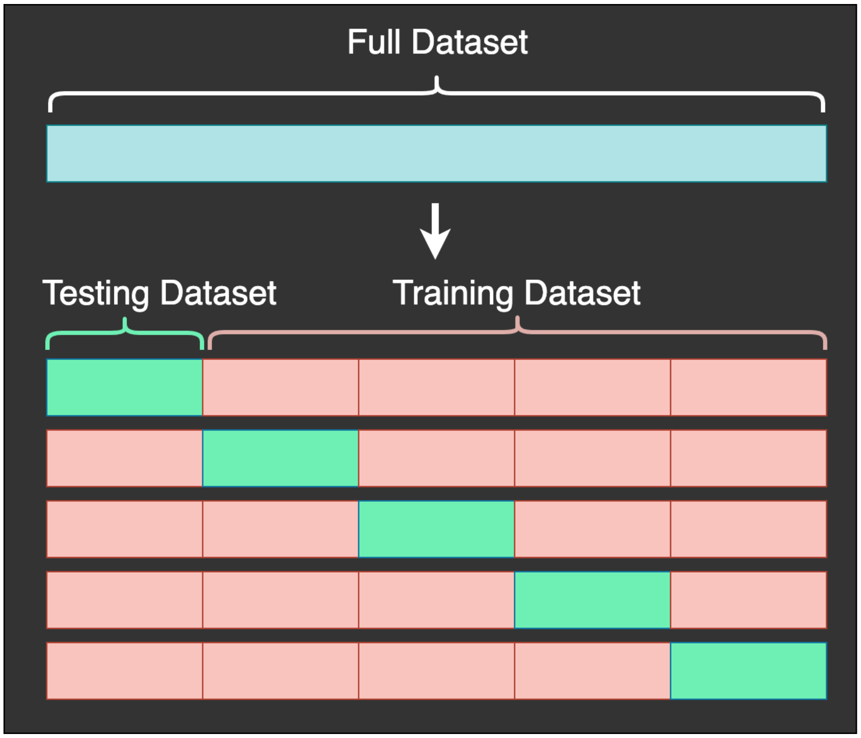


Figure 14: Cross Validation Example (Adapted from [120])

The most complex validation of the three strategies is the optimism adjusted bootstrap validation (described below)[109]. This strategy produces the smallest variance and least biased estimate of model performance, but requires the most model fits. If model training and tuning require a large numbers of computational resources and time, then the other validation strategies are more practical alternatives. The first step of optimism adjusted bootstrap validation is to fit a model on the entire data set and record the performance of the model on the training data. We shall call this the apparent model fit. Next, take samples with replacement from the original data until each replicate data set has the same number of subjects as the original data set. One should create at least 100 or more of these replicate data sets. Now, fit a model on one of the replicate data sets and calculate the training performance of that model. We shall refer to this performance as the bootstrap fit. Next take the bootstrap trained model and evaluate the model on the whole data set. We will refer to this performance as the original fit. Calculate the bootstrap fit and original fit for all of the replicate data sets and then average their difference (see Equation 6). After calculating the optimism quantity from Equation 6, subtract the optimism from the apparent fit to produce the estimated optimism adjusted bootstrap performance[121].

$$optimism = \frac{\sum_{i=1}^{n} fit_{bootstrap} - fit_{original}}{n}$$

Equation 6: Optimism Adjustment

To tune hyper-parameters one can nest any of these three strategies within each other. For example one could split using 5-fold cross validation. Then, within the training folds one could use a random or grid search for hyper parameters that used a nested 5-fold cross-validation to determine the best hyper-parameter combination. Figure 15 visualizes what this cross-validation within a cross-validation example would look like.

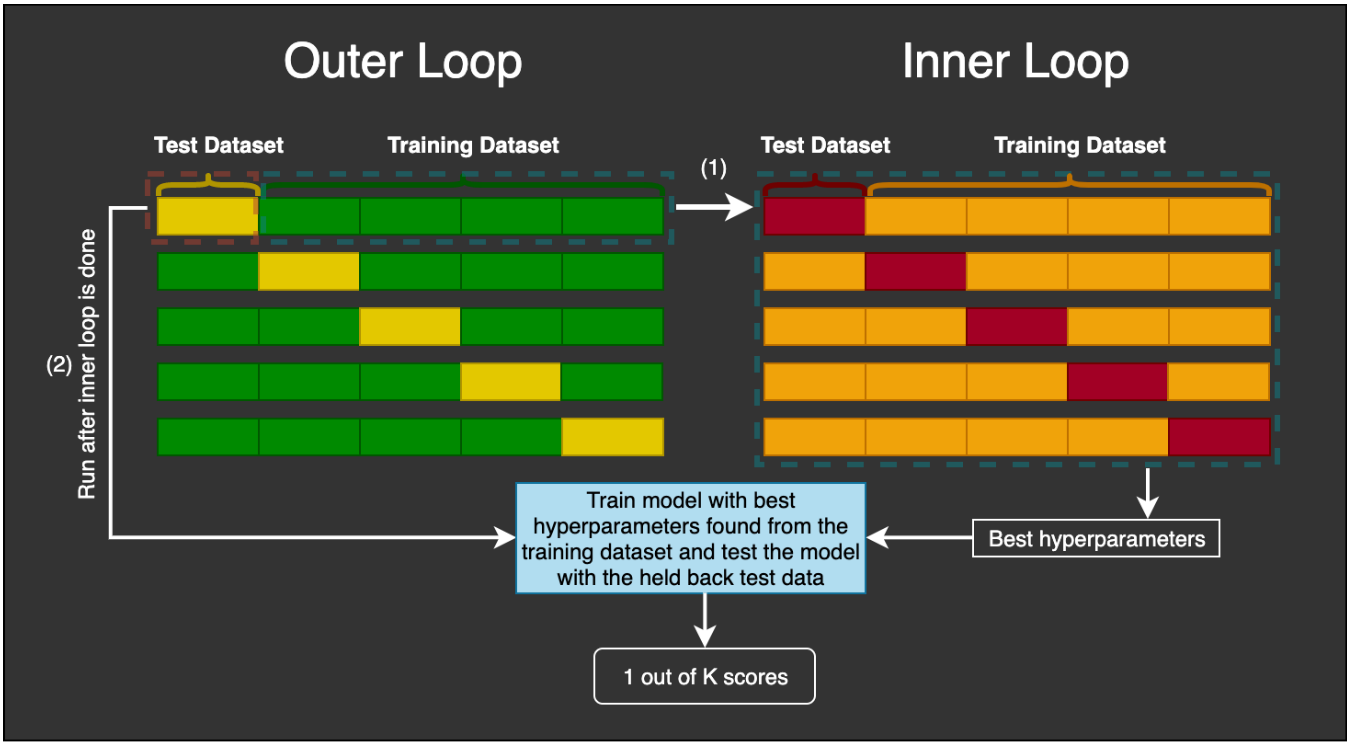

Outer Loop
Inner Loop
Test Dataset
Training Dataset
Test Dataset
Training Dataset
(1)
(2)
Run after inner loop is done
Train model with best hyperparameters found from the training dataset and test the model with the held back test data
Best hyperparameters
1 out of K scores

Figure 15: Nested Cross-Validation for Hyper-parameter Tuning within a Cross Validation (Adapted from [120])

Any of these strategies can be nested within the others. It is critical to use at least one of the mentioned validation strategies as a means of comparing different hyper-parameter combinations, lest one select hyper-parameters specific to the test data, leading to a model that does not generalize well[51]. It is difficult to claim that one has selected the best hyper-parameters if the metrics of comparison were not designed to provide unbiased and precise estimates of model performance on new data.

Experiment Outline

In this work we seek to evaluate how different temporal representations affect the prediction performance of LSTM models and Attention models. Specifically, we are seeking to identify associations between characteristics of longitudinal data and the dominant temporal representation (if one exists). To accomplish this goal we developed and validated a longitudinal data generator based on GLME models. Using this data generator we performed a simulation study inspired by two real world problems: predicting remaining length of intensive care unit (ICU) stay and predicting 24-hour ICU mortality. We simulated data mimicking the said prediction problems. In each simulation we perturbed different data characteristics and evaluated the model performance of all temporal representations of interest on each unique perturbation. In a stepwise manner, we added different elements of realism to our synthetic data sets to see if/how the associations of data characteristics with temporal representation changed. We sought to formulate generalizable knowledge for model developers by laying out best practices for the temporal representation of longitudinal data based on the measurable characteristics of that data.

We repeated the analysis of temporal representation on a well bench marked publicly available ICU cohort (MIMIC III)[122]. We hoped to verify and extend the results from the synthetic data by evaluating whether similar results held in real clinical data. The modeling problems selected, the cohort and the performance measures are based on a benchmarking study which should allow comparison of results for the same prediction task[73]. The benchmark provides a frame of reference to assess the quality of this study's evaluation of different temporal representations.

Chapter II

CONTROLLED GENERATION OF SYNTHETIC LONGITUDINAL DATA

Study Design

We built and evaluated a software package that can generate correlated observations over time based on the GLME statistical model. The GMLE statistical model is a popular tool for statistical inference in longitudinal observational studies and clustered clinical trials[46]. This model is well understood and we hoped that by using the GLME as a basis for generating data, we might be able to glean insights about the working of deep models with respect to temporal representation. The objective in developing this software package was to be able to create data sets where we could precisely tune the characteristics of interest: autocorrelation, collinearity, distribution of measurement occasions, inter/intra-subject variability, measurement error, outcome type, sampling scheme, and synchronicity. We also built mechanisms and data quality checks into the data generation pipeline that would allow the software package to produce data with different mechanisms of simulated realism based on our analysis of data observed in the MIMIC III cohort. To our knowledge, there were no published packages that fit these requirements, necessitating custom development. In this chapter we will detail how we developed this software package, how the package generates data in its different configurations, and the experiments done to validate that the package behaves as expected. Our package (long-gen) is publicly available through the Python package index[123] and its code can be viewed on Github[124].

Materials

To develop and evaluate the long-gen package, we used a 2015 MacBook Pro with four 2.9 GHz

Intel processors and eight GB of RAM. We used the Python programming language version 3.6.3 to develop the long-gen package and Stata version 29 Jan 2018[125] to evaluate the package. We used IPython's[126] Jupyter[127] notebooks and the Sublime Text editor[128] (version 2) as the editors to do the actual development in Python. Table 5 details the software dependencies we had for Python. We used data from the MIMICIII database[122] (version 1.4) to inform the design of the long-gen package.

Package	Language	Version
IPython[125]	Python	7.12.0
Jupyter[126]	Python	1.0.0
Matplotlib[129]	Python	3.1.3
Numpy[130]	Python	1.18.1
Pandas[131]	Python	1.0.1
Scipy[132]	Python	1.4.1
Scitkit-learn[133]	Python	0.22.1

Table 5: Software Dependencies for Long-Gen Package

Inner Workings of the Long-Gen Package

The long-gen package has an object oriented design and uses three nested classes: a longitudinal data set, a patient, and a patient model. The data set is made of multiple patients and a patient's measurement occasions are generated by a patient model. If the user desires stationary data, then each patient will have only one model that generates data. This interlocking hierarchy is visualized in Figure 16.

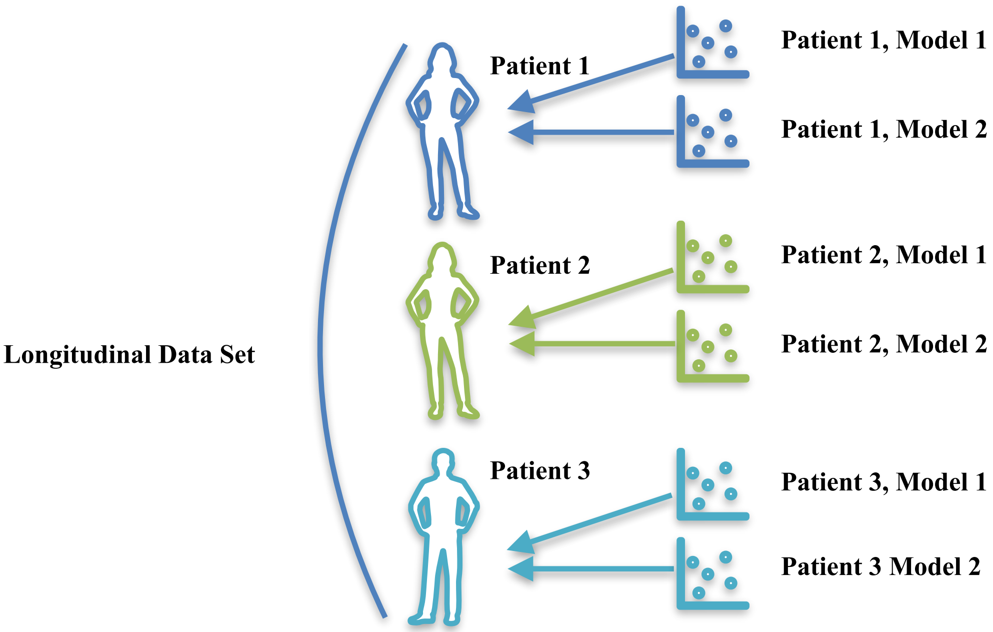


Figure 16: Class Hierarchy for Long-Gen Package

We will begin by describing the attributes of the Longitudinal Data Set class before diving into Patient and Patient Model class attributes. The Longitudinal Data Set class attributes are set first because it is at the top of the hierarchy. Table 6 introduces attributes of the Longitudinal Data Set class.

Attribute	Definition	Data Type
Coefficient_Values	The values of the fixed effect coefficients	Dictionary
Collinearity/Autocorrelation	The amount of collinearity & autocorrelation desired for the features	Categorical
Link_Function	The type of outcome and the distribution of the measurement error	Categorical
Measurement_Distribution	The distribution of the number measurement occasions across all subjects	Categorical
Measurement_Parameters	The location and shape parameters for the measurement distribution	Dictionary
Num_Extraneous_Variables	How many non-causal variables should be created	Integer
Number_of_Features	How many features should be created	Integer
Number_of_Model_Changes	Defines how stationary the process is (0 for stationary, 1 or more for non-stationary)	Interger
Number_of_Subjects	The number of subjects to create data for	Integer
Probability_Threshold	For categorical outcomes, the probability threshold separates cases from controls	Float
Random_Effects	The list of variables that have random effects	List

Attribute	Definition	Data Type
Random_Effect_Collinearity	Sets the level of correlation between random effects coefficients	Float
Random_Effect_Cut_Point	In the distribution of the absolute value of random effects, the quantile of that distribution where values above the quantile should be positionally reshuffled	Float
Random_Effect_Insert_Point	For random effects above the quantile threshold, where those values should be approximately shuffled into	Float
Realism_Functions	Functions that add different elements of realism to the longitudinal data	Categorical
Sampling_Scheme	The mechanism that determines the timing of measurement occasions for each subject	Categorical
Temporal_Trend	The type of relationship between the outcome and time	Categorical
Time_Breaks	A list of time points where the model changes	List
Variance_of_Betas	If Beta coefficients are randomly produced from a zero centered normal distribution, what should be the variance of that distribution	Float
Variance_of_Error	Controls how large unobserved error terms can be	Float
Variance_of_Random_Effects	This tunes how much inter-subject variability there is. Note that random effects are multivariate normally distributed	Float

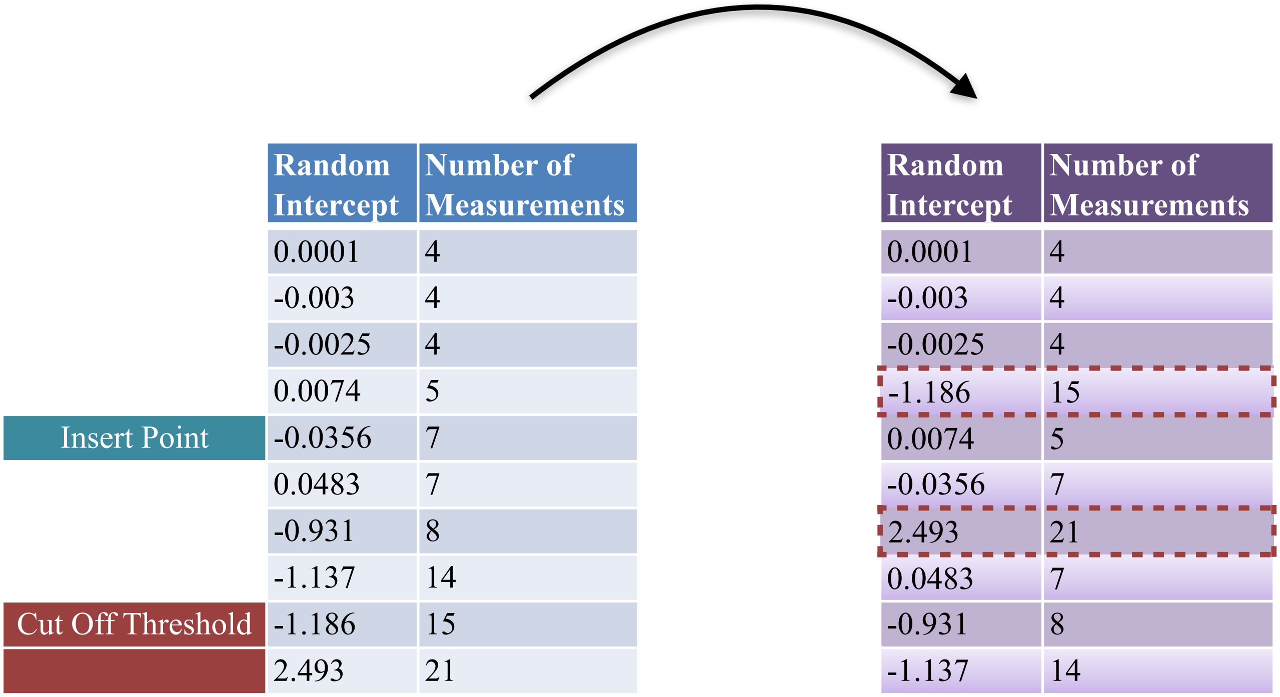
Table 6: Attributes of the Longitudinal Data Set Class

When creating a new data set, the Longitudinal Data Set Class first validates or selects global change points (the stationarity of the outcome process) for patient models. These change points are meant to represent fundamental changes to the outcome process over time, such as the development of a new therapy. The change point is a piecewise break where the models for all subjects shift to different models. If the user does not define their own change points, then the package will randomly select the number of desired change points from a random uniform distribution. The package bounds time between the continuous interval of [0,1). This interval can be mapped to any other interval/set by supplying a mapping function to the class's transform_variable_feature method after the data set has been created.

Next, the package determines the number of measurement occasions for each subject by drawing from the user defined distribution with the user defined location and shape parameters. The user has several choices of distributions for measurement occasions: equal (point mass distribution for balanced data), poisson, discretized (rounded to nearest integer) normal, discretized log normal, and discretized gamma. This vector of measurement occasion counts is then sorted in ascending order. Sorting the count of measurement occasions is important to future steps.

The package then proceeds to generate each subject's random effects. As mentioned previously, the random effects are assumed to follow a multivariate normal distribution (MVN) centered at zero. The package constructs a covariance matrix where the diagonal entries are equal to the user specified random effect variance and the off diagonal terms are equal to the absolute value of the square root of the random effect variance multiplied by the user specified random effect collinearity term. The covariance matrix is a square matrix whose dimension is equal to the number of random effects specified by the user. The package then samples from the MVN resulting in matrix of effects where the rows correspond to subjects and the columns correspond to random effects.

At this point, the code comes to its first realism function. If the user set the "measurements" switch, then a patient's random effect will be correlated with the number of measurement occasions they have. This is done by sorting (in ascending order) the absolute value of a random effect. The sort value chooses one of the following in this priority: 1) intercept, 2) time, 3) trended-time, 4) first listed random effect. The purpose of this switch is to allow the random intercept (or other random effect) to act as an unobserved severity of illness variable. When paired with a measurement distribution with a large number of outliers (fat-tailed), this switch helps to correlate the random intercept with the number of measurements. The result is a quantitative facsimile of the subset of patients with extreme lengths of stay[56]. Some patients are in such critical condition that despite the efforts of clinicians these patients do not survive the healthcare encounter. To accommodate this dichotomy of patients that are very unwell having both a few/average number of measurement occasions or having an extreme number of measurement occasions, we added a mechanism to shuffle the patients with the most extreme random effects back into a different portion of the measurement distribution. This redistribution mechanism is only meaningful if the “measurements” switch is active. It works by taking two user defined percentiles of the random effect. One percentile represents the cut point where more extreme values will be reshuffled. The other represents the average location in the distribution where the extreme random effects will be inserted. Figure 17 provides some visual intuition for this redistribution mechanism.



	Random Intercept	Number of Measurements
	0.0001	4
	-0.003	4
	-0.0025	4
	0.0074	5
Insert Point	-0.0356	7
	0.0483	7
	-0.931	8
	-1.137	14
Cut Off Threshold	-1.186	15
	2.493	21

Random Intercept	Number of Measurements
0.0001	4
-0.003	4
-0.0025	4
-1.186	15
0.0074	5
-0.0356	7
2.493	21
0.0483	7
-0.931	8
-1.137	14

Figure 17: Measurement Link Reshuffling Example

The step is to select coefficient values, if the user has not pre-specified them. The package draws one coefficient value for each feature as well as an intercept for each model period. If there was a model change point, then there would be two coefficients for each feature and two coefficients for the intercept. Model coefficients, if randomly generated, are drawn from uncorrelated zero-centered normal distributions with a user specified variance.

Next, the code comes to another optional switch. The “timespan” switch adaptively tunes the observation period so that subjects with more measurement occasions always have a longer observation period, even in the presence of non-random sampling. The sampling mechanisms will be detailed in a later paragraph. However, in some sampling schemes the value of the features can increase or decrease the observation frequency, thereby increasing or decreasing the sampling period. After this optional switch, the package begins to initialize and create patient class object for each subject.

Each patient object first generates the unobserved measurement error for all of the measurement occasions based on the distribution associated with the canonical link function. The package has four choices: an identity link function (normal error), a log link function (Poisson error), a legit link function (binomial error), and an multiplicative-inverse link function (gamma error). The error is generated with the specified user variance. The package makes the appropriate calculations to achieve the desired error variance. For example, binomial error variance is dependent on the number of draws. Therefore, the probability of an error is adjusted for each patient to maintain a constant level of variance across the dataset.

At this point, the code determines the timing of the measurement occasions as well as the values of the features for the patient object. The feature values and measurement occasion timing are intimately related and behave differently depending on the sampling scheme. There are three built-in sampling schemes: random sampling, equally-spaced sampling, and non-random sampling. If the sampling scheme is random or equally-spaced, then the timing is selected first and then used to produce the feature values. However, if the sampling scheme is non-random, then the feature values are chosen first and are used to select the measurement times. In the random case, the measurement times inform the feature values, while in the non-random case the feature values inform the measurement times. Random sampling, as the name suggests, draws uniformly random sample times. The equally-spaced sampling method divides up the observation period equally based on the number of measurement occasions for that patient. The length of the sampling period in both schemes is determined by the relative number of measurement occasions for this subject compared to the largest number present in the data set.

In the random or equal sampling case the timing of the measurement occasions is chosen first, and then the package samples from different Gaussian processes at the selected time points to produce the feature measurements. The package has a switch that determines what numerical value the Gaussian process is centered on. The center of the Gaussian process affects the values of the feature measurements produced. If the feature-link parameter is active, then the Gaussian processes for each feature are centered at the subject-specific intercept. The effect of this parameter is that each subject has processes that are centered at different values. This difference in centering of the Gaussian process between subjects can induce a correlation between a subject's feature measurements (the value of those features) and the outcome. The induced correlation effect is especially strong in models where the random intercept has a strong effect on the outcome. If the feature-link switch is inactive, then the Gaussian process is centered at one. The Gaussian processes use the Matern covariance function where the length scale and nu parameters are set by the desired mix of collinearity and autocorrelation[134]. This covariance function offers a flexible fit and is frequently used in the literature[134]. We define the collinearity/ autocorrelation of the features in three qualitative buckets: low, moderate, and high. We define low collinearity/autocorrelation as values between [0, 0.33), moderate as values between [0.33, 0.66), and high as values between [0.66, 1). The observed collinearity/autocorrelation is dependent on the sampling density of the data set. This means that the true autocorrelation/collinearity may differ from what is observed.

In the non-random sampling case, feature values are drawn from a correlated multivariate normal

(MVN) distribution with a covariance matrix of ones on the diagonal and the user defined collinearity value on the off diagonals. The mean vector of the MVN distribution is also affected by the feature link parameter. As in the equal/random sampling case, if the feature-link parameter is active, then the center is at the subject-specific random intercept. If the the feature-link is inactive, then the MVN is centered at zero. With the mean and covariance specified, the package then draws values for all the features from the MVN distribution. The drawn feature measurements from the MVN distribution are then used to help generate the measurement occasion timings. The non-random samples begin equally spaced over the sampling period, but are shifted based on the features. The shifts are done according to a step function based on the largest deviation from zero. If there is a feature greater than one or less than negative one for a measurement occasion, then the next measurement will occur 10% sooner. If the largest feature deviation is greater than two, then the next measurement occasion will be 25% sooner. If the largest feature deviance from zero is more than three, then the next measurement occasion will be 50% sooner. The general intuition of this type of sampling scheme is that the more irregular the value observed, the sooner the next observation will occur. We chose to apply a step function to mimic human decision thresholds instead of a continuous function[135]. Furthermore, clinical variables tend to have a physiological bound of how soon one can retest and expect a different result[136].

Once the time points and feature values have been set for a patient object, then the package proceeds to sort the measurement occasions into the different modeling periods. The measurement occasions are sorted into different sets based on where they fall relative to the data-set-level change points. Each modeling period (set of measurement occasions) creates its own patient model object. The patient model is a repository of information that can easily execute the matrix algebra necessary to create outcome data. This outcome data is then aggregated in the Patient and Longitudinal Data Set classes. Figure 18 provides a flowchart of the high-level logic discussed in previous paragraphs.

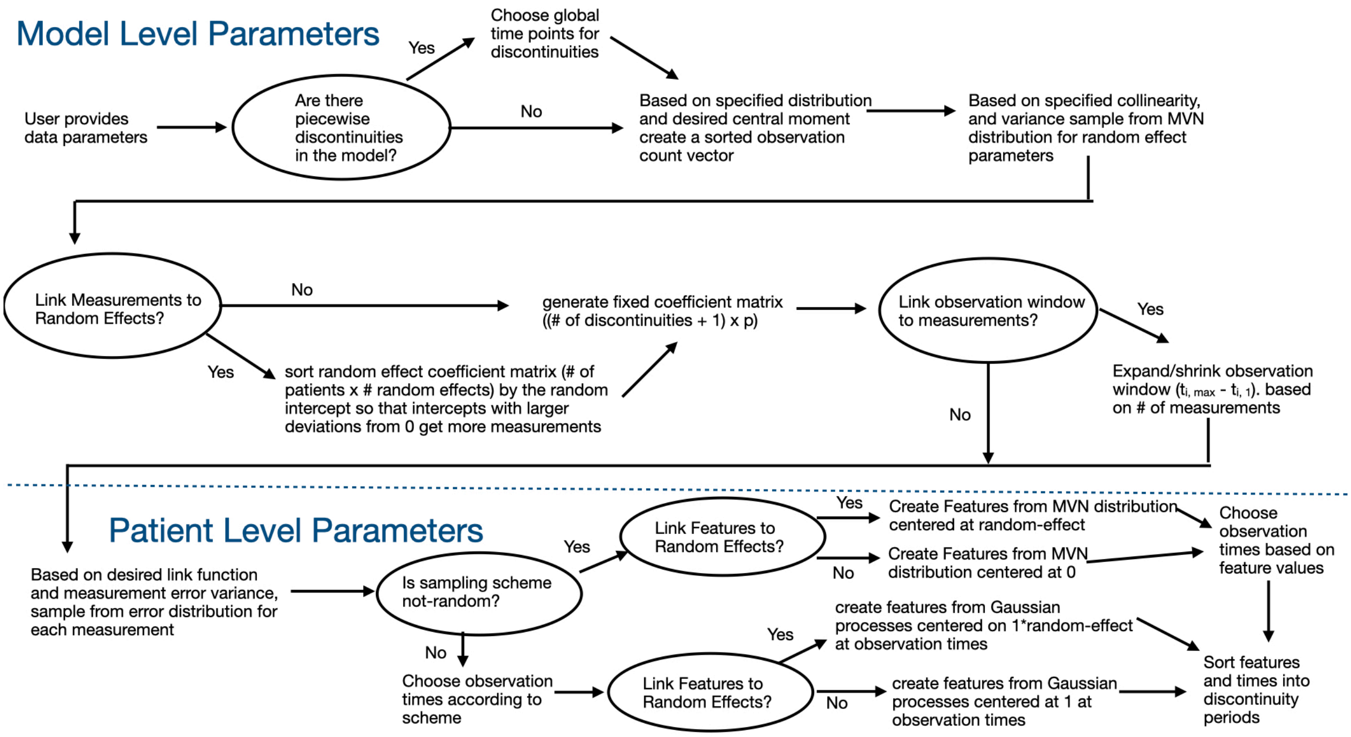

Model Level Parameters
User provides data parameters
Are there piecewise discontinuities in the model?
Yes
Choose global time points for discontinuities
No
Based on specified distribution and desired central moment create a sorted observation count vector
Based on specified collinearity, and variance sample from MVN distribution for random effect parameters
Link Measurements to Random Effects?
No
generate fixed coefficient matrix ((# of discontinuities + 1) x p)
Yes
sort random effect coefficient matrix (# of patients x # random effects) by the random intercept so that intercepts with larger deviations from 0 get more measurements
Link observation window to measurements?
Yes
Expand/shrink observation window ($t_{i, max}$ - $t_{i, 1}$). based on # of measurements
No
Patient Level Parameters
Based on desired link function and measurement error variance, sample from error distribution for each measurement
Is sampling scheme not-random?
Yes
Link Features to Random Effects?
Yes
Create Features from MVN distribution centered at random-effect
No
Create Features from MVN distribution centered at 0
Choose observation times based on feature values
No
Choose observation times according to scheme
Link Features to Random Effects?
Yes
create features from Gaussian processes centered on 1*random-effect at observation times
No
create features from Gaussian processes centered at 1 at observation times
Sort features and times into discontinuity periods

Figure 18: Flowchart of Long-Gen Package Logic

There is an additional parameter that is not shown in Figure 18 that is relevant to certain binary classification problems. In some prediction problems, the modeler is attempting to prognosticate the transition of a patient from one state to another. A state of interest, such as death, cannot be transitioned out of once made. To accomplish the behavior of an absorbing state (to use Markov Chain terminology), the package has a probability threshold parameter. When active, the threshold parameters acts a line of demarcation between cases and controls. All measurement occasions with a probability less than the threshold are controls, while all those above the threshold are cases. When the threshold is inactive, cases occur based on their respective probability.

Package Evaluation Methods

To evaluate the quality of the data generated through this package we undertook two experiments. The first experiment sought to quantitatively establish that a GLME model could recover the model coefficients used in the generation of data. The second experiment sought to qualitatively compare real-world ICU data to the synthetic data to ensure that critical attributes could be synthetically generated. These experiments demonstrate the reliability of the synthetic data used in the experiments in Chapter III.

In Experiment 1 we generated data with four features: x_1, x_2, an intercept, and time. Each subject had a random intercept and time slope. We generated 10 data sets for each parameter combination detailed in Table 7. We only varied the link function, resulting in a total of 20 datasets.

Attribute	Experimental Values
Autocorrelation	High-[0.66, 1)
Coefficient_Values	Randomly drawn from Normal(0,1)
Collinearity	Moderate-[0.33, 0.66)
Link_Function	Identity OR logit
Measurement_Distribution	Log-Normal(0.75, 10, 3)
Num_Extraneous_Variables	0
Number_of_Features	2 (Excluding model intercept and time)
Number_of_Model_Changes	0
Number_of_Subjects	1,000
Probability_Threshold	None
Random_Effects	intercept & time
Random_Effect_Collinearity	0.13
Random_Effect_Cut_Point	None
Random_Effect_Insert_Point	None
Realism_Functions	None
Sampling_Scheme	Random
Temporal_Trend	Linear
Time_Breaks	None

Attribute	Experimental Values
Variance_of_Betas	1
Variance_of_Error	0.05
Variance_of_Random_Effects	1

Table 7: Long-Gen Evaluation Parameters

After generating a variety of data sets, we validated the data generating process by fitting a GMLE model on all of the observable features (x_1, x_2, the intercept, and time) using the STATA program's xtmixed procedure with an unstructured covariance matrix and restricted maximum likelihood estimators for the identity-link data and the xtmelogit procedure for logit-link data. We defined random intercept and time slope parameters to be estimated as well. It is worth noting that the STATA models did not have access to the measurement error term included in each measurement occasion. We then compared the 95% confidence interval of all of the predicted model coefficients (x_1, x_2, the intercept, and time) to the true model coefficients. We also compared and contrasted the estimated coverage probability of the predicted model coefficients to the specified coverage of the procedure (95%) using Wilson confidence intervals[137]. If STATA is able to recover the model coefficients from data, then the data generating process is true to the underlying GLME design.

For the qualitative Experiment 2 we analyzed the distribution of measurement occasions, the distribution of observation periods (lengths of stay), the event rate (in the case of 24-hour ICU mortality), and patterns in the outcome in MIMIC III data. We attempted to replicate these findings in synthetic data. We took a sample of 1,000 adult (age ≥ 18) ICU admissions for this analysis from the MIMIC III data set. Only a subset of clinical events were counted as measurements. Chapter IV has more details on the exact cohort definition including which events were chosen. These decisions were based on a benchmarking study[73]. Comparison of distributions between real and synthetic data were generally done visually through histograms. We estimated the event rate of ICU mortality using Wilson confidence intervals[137]. Patterns in the outcome were also assessed graphically using plots of the outcome over time. Again, we attempted to recreate the observed patterns in MIMIC data using the synthetic data generator.

Evaluation Results

The GLME models fit in STATA were able to capture 40/40 of the model coefficients for the linear (identity link function) models in the estimated confidence interval. The resulting 95% Wilson estimate for the coefficient coverage was [91.24%, 100%]. The resulting interval contained the coverage probability of the procedure (95%), suggesting that the model efficiently estimated the coefficients. Table 8 shows the predicted coefficients along with the true model coefficient for each feature in each identity-link-function produced data set.

Identity Data Set	Feature	Predicted Coefficient	Actual Coefficient	Accurate Prediction
1	intercept	[-0.785, -0.657]	-0.739	Yes
1	time	[-1.239, -1.115]	-1.167	Yes
1	x1	[-2.055, -2.048]	-2.053	Yes
1	x2	[0.015, 0.026]	0.019	Yes
2	intercept	[-0.364, -0.238]	-0.310	Yes
2	time	[0.465, 0.587]	0.530	Yes
2	x1	[1.748, 1.755]	1.752	Yes
2	x2	[1.626, 1.637]	1.637	Yes
3	intercept	[0.801, 0.923]	0.858	Yes
3	time	[0.034, 0.159]	0.048	Yes
3	x1	[-0.504, -0.497]	-0.502	Yes
3	x2	[-0.648, -0.637]	-0.642	Yes
4	intercept	[-1.266, -1.141]	-1.190	Yes

Identity Data Set	Feature	Predicted Coefficient	Actual Coefficient	Accurate Prediction
4	time	[1.701, 1.828]	1.752	Yes
4	x1	[-0.097, -0.090]	-0.096	Yes
4	x2	[-1.949, -1.939]	-1.941	Yes
5	intercept	[1.622, 1.754]	1.718	Yes
5	time	[-0.003, 0.125]	0.054	Yes
5	x1	[1.799, 1.806]	1.801	Yes
5	x2	[0.121, 0.132]	0.123	Yes
6	intercept	[-0.680, -0.553]	-0.608	Yes
6	time	[0.735, 0.860]	0.852	Yes
6	x1	[1.909, 1.916]	1.912	Yes
6	x2	[-0.834, -0.824]	-0.826	Yes
7	intercept	[-1.272, -1.144]	-1.210	Yes
7	time	[-0.284, -0.162]	-0.223	Yes
7	x1	[-0.021, -0.014]	-0.016	Yes
7	x2	[-1.586, -1.574]	-1.581	Yes
8	intercept	[-0.775, -0.655]	-0.754	Yes
8	time	[-0.102, 0.019]	-0.061	Yes
8	x1	[-0.754, -0.747]	-0.752	Yes
8	x2	[1.865, 1.875]	1.870	Yes
9	intercept	[0.037, 0.162]	0.070	Yes
9	time	[-1.132, -1.014]	-1.09	Yes
9	x1	[-1.620, -1.614]	-1.619	Yes
9	x2	[0.740, 0.750]	0.751	Yes
10	intercept	[0.453, 0.577]	0.491	Yes
10	time	[-1.287, -1.163]	-1.243	Yes
10	x1	[-0.821, -0.814]	-0.818	Yes
10	x2	[2.059, 2.069]	2.061	Yes

Table 8: Detailed Results of Identity Data Sets

The GLME binomial-family models were able to accurately estimate 37/40 of the coefficient values. The resulting 95% Wilson interval for these estimates was [80.14%, 97.42%]. This estimate, like the previous estimate for the identity data, contains the coverage probability of the coefficient estimation procedure. Again, this implied that the models were able to produce accurate estimates of the coefficients. Table 9 provides the estimation details for the logit-link data sets.

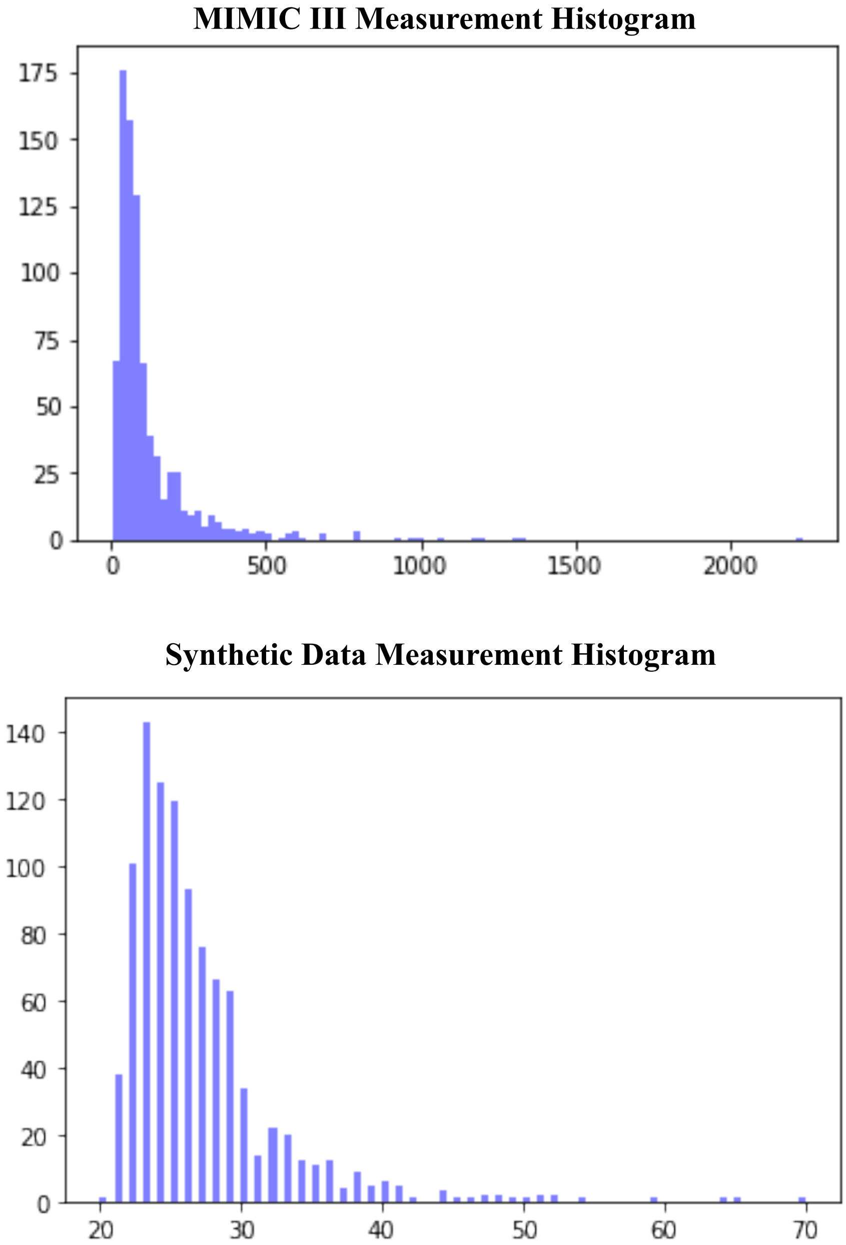
Binomial Data Set	Feature	Predicted Coefficient	Actual Coefficient	Accurate Prediction
1	intercept	[1.873, 3.616]	1.978	Yes
1	time	[1.790, 3.610]	2.146	Yes
1	x1	[-0.937, 0.0493]	-0.974	No
1	x2	[-0.961, -0.610]	-0.757	Yes
2	intercept	[0.035, 0.685]	-0.007	No
2	time	[-0.090, 0.290]	0.153	Yes
2	x1	[0.529, 0.896]	0.536	Yes
2	x2	[0.702, 0.904]	0.701	Yes
3	intercept	[0.092, 0.710]	0.421	Yes
3	time	[0.118, 0.497]	0.203	Yes
3	x1	[1.014, 1.341]	1.197	Yes
3	x2	[0.151, 0.333]	0.212	Yes
4	intercept	[0.090, 0.665]	0.370	Yes
4	time	[0.323, 0.680]	0.556	Yes
4	x1	[-0.574, -0.276]	-0.421	Yes
4	x2	[0.468, 0.662]	0.590	Yes
5	intercept	[-1.742, -0.998]	-1.190	Yes
5	time	[0.075, 0.572]	0.186	Yes
5	x1	[-2.448, -2.025]	-2.114	Yes
5	x2	[-0.124, 0.010]	0.060	No
6	intercept	[-0.845, -0.293]	-0.721	Yes
6	time	[-1.397, -1.055]	-1.125	Yes

Binomial Data Set	Feature	Predicted Coefficient	Actual Coefficient	Accurate Prediction
6	x1	[-0.872, -0.583]	-0.782	Yes
6	x2	[-0.224 -0.057]	-0.188	Yes
7	intercept	[-1.810, -1.237]	-1.519	Yes
7	time	[-2.324, -1.961]	-2.146	Yes
7	x1	[1.986, -1.679]	-1.829	Yes
7	x2	[-0.138, 0.053]	-0.042	Yes
8	intercept	[-0.396, 0.207]	-0.146	Yes
8	time	[-0.403, -0.056]	-0.160	Yes
8	x1	[-0.165, 0.145]	-0.003	Yes
8	x2	[0.291, 0.480]	0.349	Yes
9	intercept	[-1.129, -0.537]	-1.052	Yes
9	time	[0.358, 0.708]	0.663	Yes
9	x1	[-0.881, -0.573]	-0.832	Yes
9	x2	[-0.547, -0.354]	-0.402	Yes
10	intercept	[-1.402, -0.685]	-0.911	Yes
10	time	[-1.183, -0.766]	-0.890	Yes
10	x1	[-1.783, -1.400]	-1.516	Yes
10	x2	[-1.626, -1.411]	-1.503	Yes

Table 9: Detailed Results of Logit Data Sets

The distribution of measurement occasions in the MIMIC III sample has a median of 69 observation episodes with an interquartile range of 93 (25th at 40.0 measurement occasions and 75th at 123 measurement occasions). Looking at Figure 19, the MIMIC III data does appear to follow a log-normal distribution when it comes to the number of measurement occasions per subject. The synthetic data histogram below the MIMIC III histogram was created using a log-normal distribution with a shape parameter of 0.75, a location parameter of 60, and a scale parameter of five.

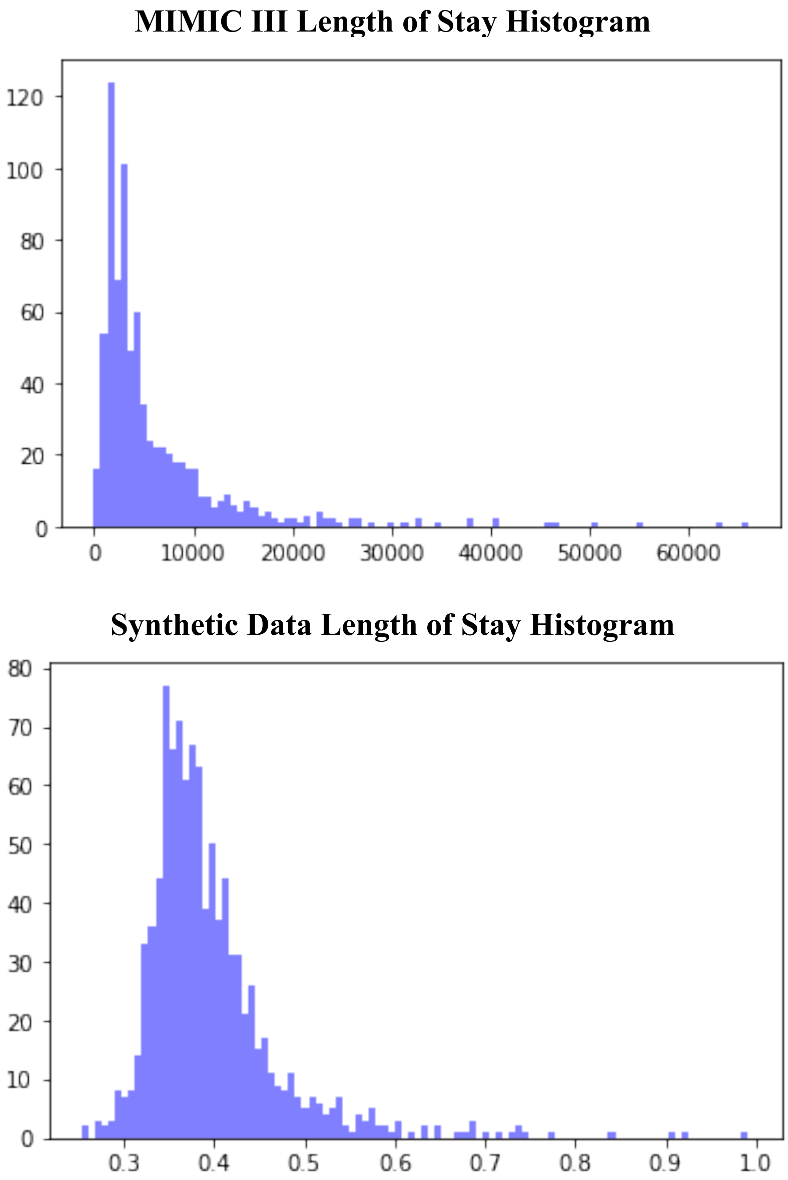
The distribution of the length of ICU stays across subjects follows a similar log-normal looking


MIMIC III Measurement Histogram

Synthetic Data Measurement Histogram

Figure 19: Measurement Occasion Distribution Comparison

distribution in the MIMIC III data set. In the MIMIC III dataset the length of stay is the observation period for each subject. We were able to recreate a similar distribution of subject observation periods in the synthetic data. The “measurements” and “timespan” mechanisms were the key to generating this similarly shaped distribution. We will describe the full parameters set needed to generate synthetic data of this form later in this chapter. Figure 20 shows histograms for both the MIMIC III sample and the synthetic data.


MIMIC III Length of Stay Histogram

Synthetic Data Length of Stay Histogram

Figure 20: Observation Period Distribution Comparison

In the MIMIC III sample there were 98 ICU admissions with a mortality event, for an observed event rate of 9.8%. We estimated the 95% Wilson confidence Interval for the event rate for the whole data set was [8.11%, 11.8%]. Figure 20 provides examples of the different outcomes patterns observed for the remaining ICU length of stay outcome and the 24-hour ICU mortality outcome. The figure shows these two outcomes as functions over time for a single subject. We were able to replicate these outcome patterns in synthetic data. The settings used to generate synthetic data with the properties shown in

Figures 19-22 are shown in Table 10. Figure 21 presents the real and synthetic outcome patterns as lines with a constant slope of negative one, where the y-axis intercept (the total length of stay) is the primary difference between subjects. The second set of graphs below display our recreations of this pattern. While the scales between the simulated data and the MIMIC III data may be different, the shape is what is important.

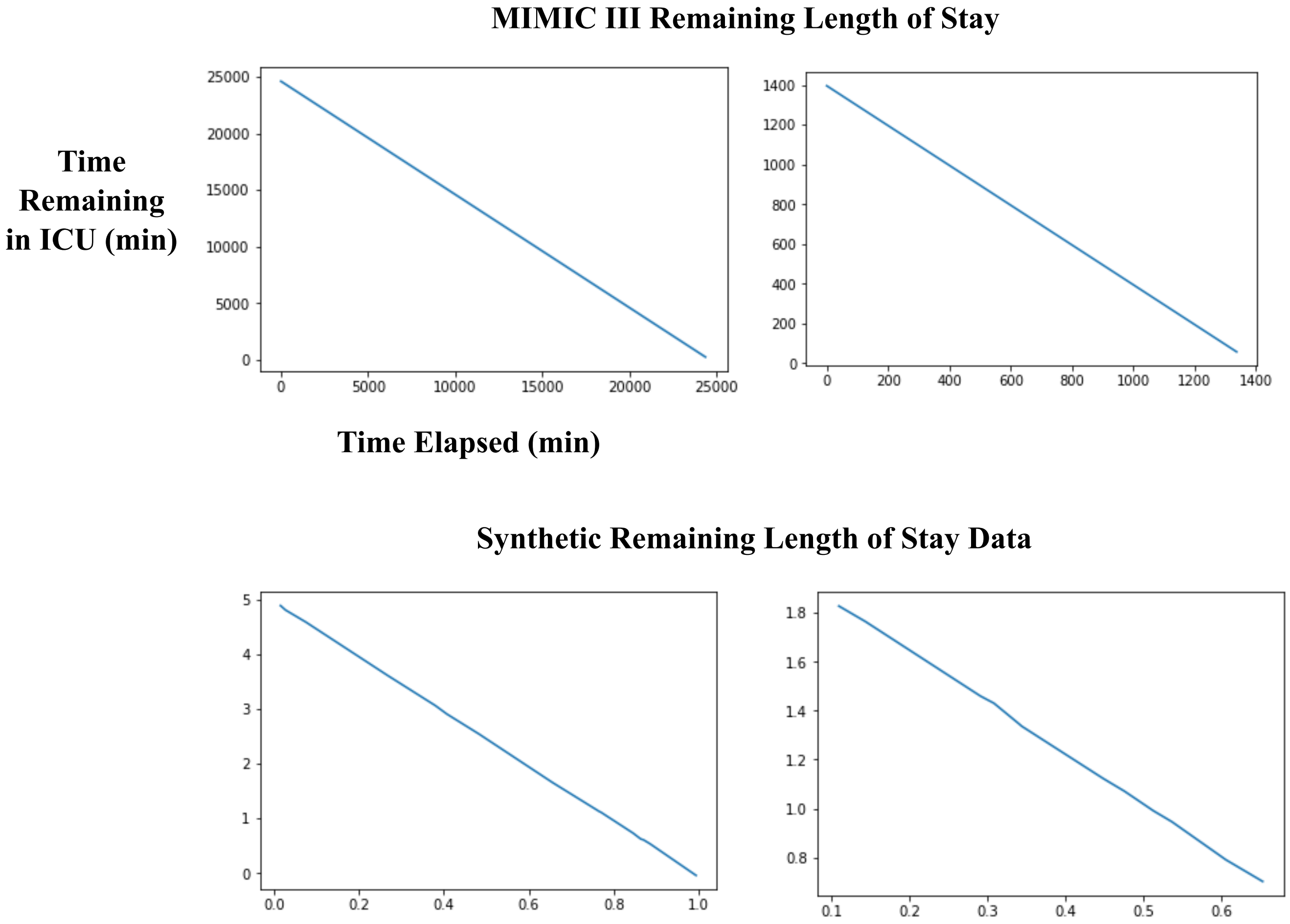


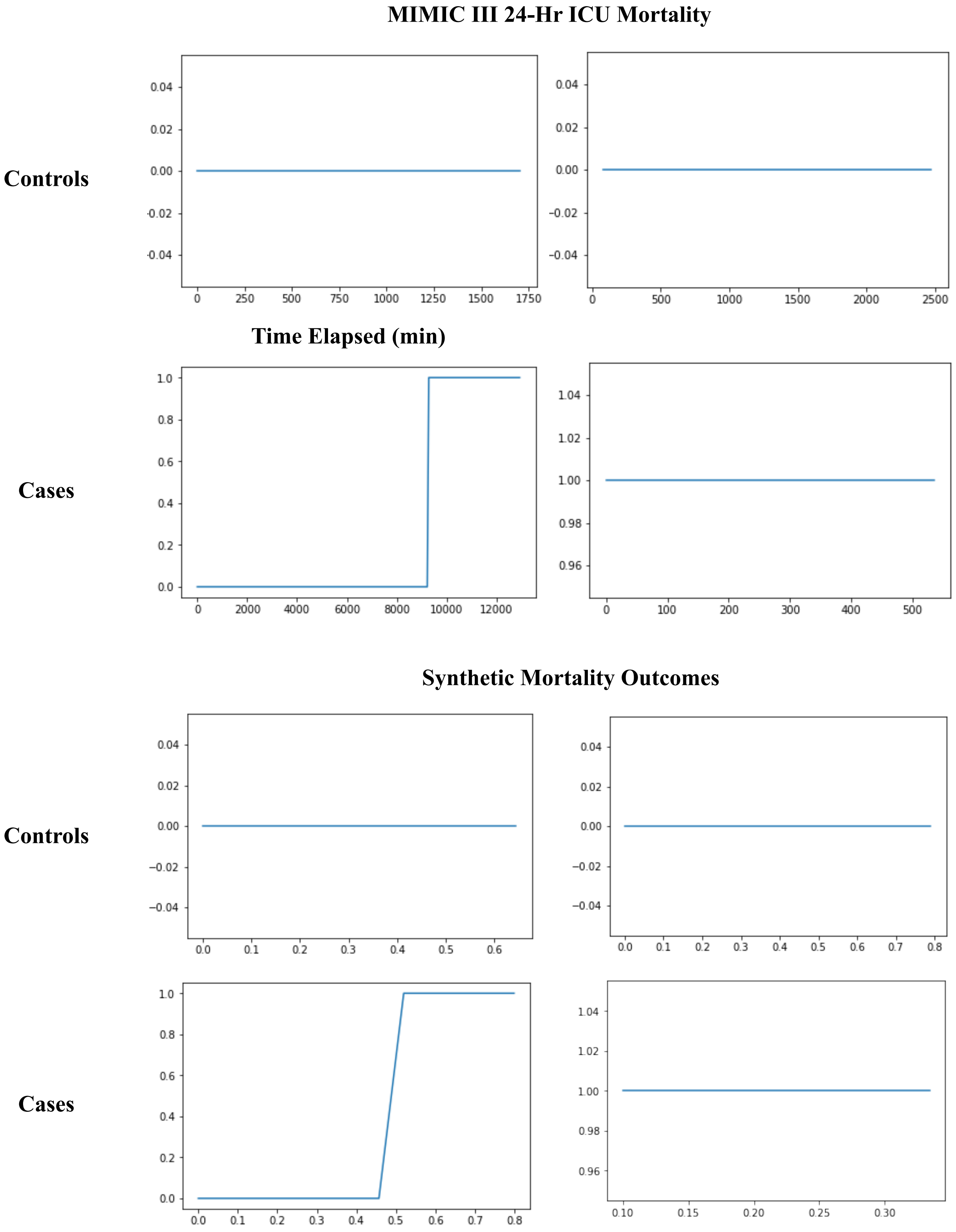
Figure 21: Length of Stay Outcome Patterns

Table 10 contrasts the different features sets to create the synthetic Length of Stay and synthetic ICU morality data. The biggest difference between the two is the exclusion of a random time coefficient in the Length of Stay data. The fixed slope between subjects precludes the use of subject specific time coefficients. The collinearity value is based on the average collinearity between MIMIC III features (0.47). Given the unequal observation times of predictors within MIMIC, we based the autocorrelation on previous literature[138, 139].

Attribute	Length of Stay	24-Hr Mortality
Autocorrelation	High-[0.66, 1)	High-[0.66, 1)
Coefficient_Values	Intercept: 8 Time: -1	Intercept: -2 Time: 1.6
Collinearity	Moderate-[0.33, 0.66)	Moderate-[0.33, 0.66)
Link_Function	Identity	Logit
Measurement_Distribution	Log-Normal(0.75, 60, 5)	Log-Normal(0.75, 60, 5)
Num_Extraneous_Variables	2	2
Number_of_Features	0	0
Number_of_Model_Changes	0	0
Number_of_Subjects	1,000	1,000
Probability_Threshold	None	0.65
Random_Effects	Intercept	Intercept & time
Random_Effect_Collinearity	0.13	0.13
Random_Effect_Cut_Point	0.95	0.95
Random_Effect_Insert_Point	0.6	0.6
Realism_Functions	Measurements Timespan	Measurements Timespan
Sampling_Scheme	Random	Random
Temporal_Trend	Linear	Linear
Time_Breaks	None	None
Variance_of_Betas	1	1
Variance_of_Error	0	0
Variance_of_Random_Effects	1	1

Table 10: Synthetic ICU Data Parameters

In Figure 22 the controls show only one type of pattern over a varying length of stay, while the cases have two patterns with a variable length of stay. Some cases pass on after 24 hours in the ICU and other patients succumb to their illness within 24 hours. We replicated the same patterns (and only those patterns) in synthetic data for cases and controls.



MIMIC III 24-Hr ICU Mortality

Synthetic Mortality Outcomes

Figure 22: ICU Mortality (24 Hr) Outcome Patterns

Discussion & Limitations

In this chapter we demonstrated the statistical foundation of our data generating software package. We also validated that the package is capable of producing data that has similar distributional properties to two real-world problems of interest. In the first experiment, the same family of statistical model was able to produce coefficient estimates that met their intended coverage probability. The confidence intervals produced were neither too wide (the coverage probability exceeded 95%) nor too narrow (coverage probability fell below 95%). The ability to capture various combinations of coefficient values in unbalanced data with a non-trivial error term, provided confidence of the package's statistical underpinnings.

In the more exploratory analysis of Experiment 2, we reasonably replicated several important characteristics of the MIMIC data: the distribution of measurement occasions, the distribution of observation periods (lengths of stay), and outcome patterns. These characteristics are important because they inform how many observations the models must learn from, how much time those observations take place in (which is linked to what patterns might be found over time), and what forms the outcome can take. The distributions of the synthetic data loosely replicated the shape and skew of the MIMIC data, but the size of the tail probabilities were not reproduced. Differences in shape are largely the result of the larger tail probabilities making the primary concentration of the real data look more compact than in the synthetic distribution.

The parameters required to reproduce the outcome space for each modeling problem bear some interpretation. The measurement occasion distribution is a straightforward result of the parameters for the specified log-normal distribution. Replicating the length of stay distribution took more effort, and was the inspiration for the "measurements" and "timespan" mechanisms. The "measurements" mechanism links the number of measurement occasions to the value of the random intercept. This link acts like a scaling parameter for the observation period distribution. Said another way, the measurement mechanism helps create the spread and tail probabilities in the observation period distribution. The timespan mechanism instead acts as a shape parameter (the skew and concentration of the distribution). The two mechanisms work in tandem by correlating the random intercepts to the number measurement occasions, and then correlating the number of measurement occasions to the observation period. Neither of the parameter sets include directly causal features. The strict shapes of the outcomes do not allow for directly causal features. In the remaining length of stay problem, the slope decreases linearly at the same rate over time for all subjects. There is no room for features to have a direct impact on this type of outcome in a GLME model. Similarly with the mortality problem, once a patient succumbs to their malady, there is no room for variation. In this situation either the feature also monotonically increases/decreases with time or the feature cannot be directly causal. Vital signs and other human chemistry measurements generally do not monotonically increase/decrease over the course of an ICU stay. Therefore the GLME must model a process that looks like Figure 23, where the causal variable is not directly observed. In this setting the model is using variables that are partially correlated with the cause to predict the outcome. Figure 23 is drawn utilizing a directed acyclic graph in the style of Judea Pearl's causal inference framework[140].

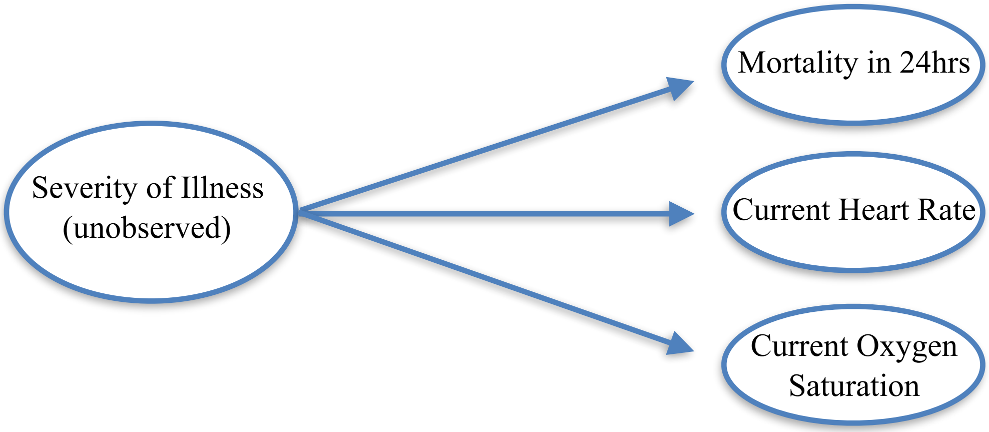


Figure 23: Modeling Limitation as a Causal Diagram

The experiments of Chapter II have several key limitations. In the first experiment the number of validation data sets are limited and do not evaluate the full suite of functionality. It may be that with a larger and more comprehensive sample the coverage probabilities would fail to include the 95% level with which the coefficient estimates were created. That said, the evidence is fairly strong that data produced by the Long-Gen package does follow the desired distribution.

Experiment 2 has its own limitations, namely that the distributions are only an approximate replication. The synthetic data is more concentrated and has smaller tail probabilities than its real world counter part. The smaller tail probabilities were a deliberate part of the design as an accommodation for the computational requirements of Chapter III. As stated in the introductory chapter, deep longitudinal models use a wide-format representation, which can lead to a significantly greater consumption of random access memory (RAM) and computational time in highly skewed data. Therefore, we needed to find a compromise between run-time and realism. We also thought it less confusing to keep the parameters and characteristics of the synthetic data as consistent as possible between chapters. The random effect mechanisms did help create a more realistic distribution of observation periods; however, these mechanisms are difficult to express and validate mathematically. We attempted to ground as many of the data generation parameters as possible in observed values from the MIMIC III sample. Given the underlying statistical model, there are limitations as to how close the synthetic data can come to the true data. We believe that despite these limitations the essential characteristics of the MIMIC data can be replicated and that the inferences produced analyzing synthetic data has the potential to inform real world applications.

The work described in this chapter provides the means to reliably produce (and reproduce) “gold-standard” data where the distributional characteristics are known. Through some basic analyses, we believe that this package can produce simplifications of clinical data, such as MIMIC III, that grants the user the ability to study some data properties while holding other constant. Model developers could use this package to learn more about the conditions (data properties) that lead to consistent as well as variable model performance. This data generation package is not specific to MIMIC III and can be generalized to other modeling problems and datasets through its wide assortment of pre-built functionality as well as the option to incorporate user-specified sampling and feature transformation functions. As previously mentioned, the long-gen package is publicly available on *pypi.org*. In future work, we hope to continue to expand the functionality of the long-gen package to be able to accept a dataset sample and then self-tune to produce replica data.

Chapter III

COMPARING TEMPORAL REPRESENTATIONS IN SYNTHETIC DATA

Study Design

Using the software package described in Chapter II, we sought to study whether there are characteristics in data that cause one temporal representation to be advantageous compared to others in LSTM and Attention models. In our study, we generated four distinct data set groups where we gradually added characteristics to match our dataset of interest, MIMIC III. Within each data set group there are numerous replicas, each with variations in the longitudinal data characteristics described in Chapter I. The first group has directly causal features (the feature values directly influence the outcome) with synchronous data (all features are observed for every measurement occasion). The second data group also has causal features, but the synchronicity of measurements is relaxed, allowing for missing values. In the third data group the features do not directly influence the outcome, but are correlated with the outcome. This third data group lowers the signal to noise ratio, but brings back the requirement for synchronous measurements. The fourth data group has the lowest signal to noise ratio, as it has non-causal features as well as missing data. We then evaluated how well each temporal representation performed using both the LSTM and the Attention model for each data group. We hoped that by studying the model performance metrics as a function of the data characteristics we could learn associations that may be useful to future model builders. We performed our analysis using statistical inference through a regression framework to isolate the effects of temporal representation on model performance by adjusting for the data characteristics. We were also interested in identifying significant interaction terms between the temporal representations and said data characteristics. In summary, in this chapter will describe how we varied parameters in each data set group, how we architected/tuned/trained the deep models, how we evaluated model performance, how we analyzed the results, and what those results mean.

Materials

For developing the software necessary for these experiments, we used a 2015 MacBook Pro with four 2.9 GHz Intel processors and eight GB of RAM. We piloted the development of the deep models on a 2014 Alienware X51 desktop running Windows 10 with four 1.6 GHz processors, 32 GB of RAM, and a 6GB NVIDIA GTX 1060 graphics processor (GPU). The overwhelming majority of the computation was done using the resources of the Advanced Computing Center for Research and Education (ACCRE) at Vanderbilt University, Nashville, Tennessee.

> The ACCRE high-performance computing cluster has over 10,000 processor cores and is growing. Typical nodes each have 128 or 256 GB of memory. Compute nodes all run a 64-bit Linux OS and have a 250 GB – 1 TB hard drive and dual copper gigabit Ethernet ports. Fifty five compute nodes are each equipped with 4X Nvidia Titan X or GeForce RTX 2080 Ti GPU cards, and are also interconnected with a low-latency 25 or 40/56 Gb/s RoCE network. All compute nodes are monitored via Nagios. Resource management, scheduling of jobs, and usage tracking are handled by an integrated scheduling system by SLURM. These utilities include an "advance reservation" system that allows a block of nodes to be reserved for pre-specified periods of time (e.g., a class or lab session) for educational or research purposes.
>
> IBM's General Parallel File System (GPFS) is used for user home and data directories and scratch space. The ACCRE filesystem provides over 2 PB of usable disk space and can sustain more than 100 Gb/s of I/O bandwidth to the cluster. The home directories of all users are backed up daily to tape. The disk arrays are attached to a SAN fabric along with the storage nodes that then exports the file system to the rest of the cluster using a fully redundant design with no single point of failure[142].

In the ACCRE environment, we personally had use of 31GB of storage, 20 standard processor cores (CPUs), and 34 CPUs connected to 17 GPUs in a configuration of two CPUs per GPU. The use of the 34 GPU-connected CPUs were generously provided by a miniature grant from the Vanderbilt University Data Science Institute[143].

We used the Python programming language version 3.6.3 to develop the data generation controller, the deep models, and the data aggregation and visualization software. When evaluating model performance for synchronous data we used a GLME model as a reference. Fitting GLME models over hundreds of data sets necessitated the use of the R programming language (version 3.6.1). We used Stata version 29 Jan 2018 to analyze the aggregated performance data.[125]. We used IPython’s[126] Jupyter[127] notebooks, the Sublime Text editor[128] (version 2), and vim[141] as the editors to do the actual development in Python. We used RStudio[144] version 1.0.143 for developing the R scripts that benchmarked the deep models. Table 11 lists the software dependencies of our Python and R software for these experiments. We used Stata version 29 Jan 2018 for statistical inference on the advantages and disadvantages of the different temporal representations[125].

Package	Language	Version
IPython[125]	Python	7.12.0
Joblib[145]	Python	0.15.1
Jupyter[126]	Python	1.0.0
Long-Gen[124]	Python	0.2.3
Matplotlib[129]	Python	3.1.3
Numpy[130]	Python	1.18.1
Pandas[131]	Python	1.0.1
Scipy[132]	Python	1.4.1
Scitkit-learn[133]	Python	0.22.1
Torch[146]	Python	1.6.0
caret[147]	R	6.0-85
dplyr[148]	R	0.8.3
doMC[149]	R	1.3.6
lme4[150]	R	1.1-21
nlme[151]	R	3.1-140

Package	Language	Version
pROC[152]	R	1.16.2

Table 11: Synthetic Data Evaluation Software Dependencies

Causal Data Group Methods

The features in the Causal Data Group have a direct effect on the outcome. Consequently, this means that the outcomes in these data sets do not look like the outcome data in our MIMIC III cohort. The goal of this Causal Data Group analysis was to identify relationships between longitudinal data characteristics (that we will vary) and the temporal representation of the data as a function of model performance. The Causal Data Group is further subdivided into a synchronous subgroup and an asynchronous subgroup. This divide is meant to check for differences that may materialize due to missing data. To produce asynchronous data, we took our synchronous dataset and then removed values in a non-random way from the original data sets, thus converting the data from synchronous to asynchronous. The results from the causal data groups set a baseline from which we added further complexity to the data generating process that cause the data to look more and more like our MIMIC III cohort.

Key parameters such as the number of subjects, number of measurement occasions, and the number of data sets that required consideration before proceeding to generate data. We anchored the rate of cases to the MIMIC III estimate, deciding to allow the case rate to vary between 4% and 25%. The event rate in MIMIC III was about 10%. Using these approximate event rates, we made us of the equations from Riley et. al.'s work to estimate the number of subjects required to accurately estimate model parameters[58]. We made our power calculations with a desired margin of error/mean absolute prediction error less than 0.01. We chose this level of precision, because we believed differences in the Brier Score less than 0.01 were not practically meaningful for model developers. This determination was subjective and may vary based on application. However, this kind of framing leads to a more consequential discussion of statistical results by casting the null hypothesis as an interval of irrelevance on the scale of the data, as opposed to a point mass null hypothesis that casts the results in the statistical standard deviation space[153].

The 24-hour mortality (classification) model incorporated four features (x_1, x_2, an intercept, and time) with two random effects (intercept and time). This model structure leads to the covariance matrix shown in Equation 7. This covariance matrix has four unique variance parameters (two random effect variance parameters, one error variance parameter, and one autoregressive correlation parameter) resulting in eight total model parameters. The covariance matrix of Equation 7 translates to a model where the covariance between measurement occasions will decay over time to fixed baseline. In Equation 7, t symbolizes the absolute time of a patient specific measurement occasion. Patients were indexed with the variable i, where the last measurement occasion for a specific patient was represented by the variable j.

$$Cov\left(Y_{-\mathrm{i}}\right) = (\sigma_I^2 + \sigma_\epsilon^2)\begin{pmatrix} 1 & \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} & \cdots & \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} \\ \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} & 1 & \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} & \vdots \\ \vdots & \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} & \ddots & \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} \\ \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} & \cdots & \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} & 1 \end{pmatrix} + \sigma_2^2\begin{pmatrix} 1 & \rho_\epsilon^{\frac{1}{|t_{-\mathrm{i}1} - \mathrm{t}_{-\mathrm{i}2}|}} & \rho_\epsilon^{\frac{1}{|t_{-\mathrm{i}1} - \mathrm{t}_{-\mathrm{i}3}|}} & \cdots & \rho_\epsilon^{\frac{1}{|t_{-\mathrm{i}1} - \mathrm{t}_{-\mathrm{i}j}|}} \\ \rho_\epsilon^{\frac{1}{|t_{-\mathrm{i}2} - \mathrm{t}_{-\mathrm{i}1}|}} & 1 & \rho_\epsilon^{\frac{1}{|t_{-\mathrm{i}2} - \mathrm{t}_{-\mathrm{i}3}|}} & \cdots & \rho_\epsilon^{\frac{1}{|t_{-\mathrm{i}2} - \mathrm{t}_{-\mathrm{i}j}|}} \\ \rho_\epsilon^{\frac{1}{|t_{-\mathrm{i}3} - \mathrm{t}_{-\mathrm{i}1}|}} & \rho_\epsilon^{\frac{1}{|t_{-\mathrm{i}3} - \mathrm{t}_{-\mathrm{i}2}|}} & 1 & \cdots & \rho_\epsilon^{\frac{1}{|t_{-\mathrm{i}3} - \mathrm{t}_{-\mathrm{i}j}|}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho_\epsilon^{\frac{1}{|t_{-\mathrm{i}j} - \mathrm{t}_{-\mathrm{i}1}|}} & \rho_\epsilon^{\frac{1}{|t_{-\mathrm{i}j} - \mathrm{t}_{-\mathrm{i}2}|}} & \rho_\epsilon^{\frac{1}{|t_{-\mathrm{i}j} - \mathrm{t}_{-\mathrm{i}3}|}} & \cdots & 1 \end{pmatrix}$$

Equation 7: Classification Model Covariance Matrix

The length-of-stay (regression) model uses the same four features, but only has one random effect (intercept). This difference alters the covariance matrix to the form in Equation 8. The covariance matrix of Equation 8 has two variance parameters: one for the error and one for the random effect. We used a small scale pilot of 20 data sets with 1,000 subjects and 25 measurement occasions to estimate the population variance required for the power calculations for the identity model with the data generation settings from Table 10. We chose a multiplicative margin of error at less than 5% in accordance with the recommendation from Riley et. al.[58].

$$Cov\left(Y_{-\mathrm{i}}\right) = (\sigma_I^2 + \sigma_\epsilon^2)\begin{pmatrix} 1 & \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} & \cdots & \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} \\ \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} & 1 & \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} & \vdots \\ \vdots & \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} & \ddots & \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} \\ \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} & \cdots & \frac{\sigma_I^2}{\sigma_I^2 + \sigma_\epsilon^2} & 1 \end{pmatrix}$$

Equation 8: Regression Model Covariance Matrix

These parameter estimates are for GLME models and are not specific to LSTM or Attention models, which have orders of magnitude more parameters. However, the equations from Riley et. al. are specifically for statistical models[58] and deep models have shown themselves to be resilient to overfitting their great number of parameters[154, 155]. Given that we are using a GLME as a baseline model, it seemed reasonable to calculate the model power from the perspective of a GLME model.

The number of measurement occasions was largely decided by computational constraints fitting the baseline GLME model in R. We wanted to replicate the measurement occasion distribution in MIMIC III, which was centered around 68. However, we found that the lme4 and nlme packages would struggle fitting models that involved thousands of subjects with a median of 68 measurement occasions. Thus, we scaled back the median number of measurement occasions until we found a number that we could reliably fit with the GLME models in a reasonable amount of time.

We chose to create 120 data sets for each modeling problem. Given the two sequence models, four temporal representations, and a three-fold cross validation strategy, this number of datasets would create 2,880 data points per modeling problem. Leaning on Riley et. al. again, we validated that this amount of data would be more than adequate to power a statistical inference model focused on teasing out the effect of temporal representation on model performance[58].

We designed software that used the Long-Gen package described in Chapter II to vary certain longitudinal data characteristics while creating 120 data sets per modeling problem (a total of 240 data sets). The controller would uniformly at random select from a range of possible values for the variable longitudinal characteristics. We chose to focus on parameters that would alter the signal to noise ratio: inter-subject variance (random effect collinearity and random effect variance), the unobserved measurement error variance (noise), sampling method (signal), and the amount of feature collinearity & autocorrelation (signal). We studied the same characteristics in both modeling problems. Table 12 enumerates the data settings we used to randomly create the 240 data sets. We found that the regression data required more than 2,000 subjects per data set to be adequately powered, while we needed more than 2,500 subjects for each classification data set. Within our data generation controller, we also had a quality control module that kept the event rate between 3% and 30% and also checked that all of the previously mentioned outcome patterns (and only those patterns) occurred in each data set.

Attribute	Length of Stay	24-Hr Mortality
Autocorrelation	[Low-[0.01, 0.33), Moderate-[0.33, 0.66), High-[0.66, 1.0)]	[Low-[0.01, 0.33), Moderate-[0.33, 0.66), High-[0.66, 1.0)]
Coefficient_Values	Intercept: 1 Time: 5 x_1: -0.5 x_2: -0.5	Intercept: -5 Time: 5 x_1: -0.5 x_2: -0.5
Collinearity	[Low-[0.01, 0.33), Moderate-[0.33, 0.66), High-[0.66, 1.0)]	[Low-[0.01, 0.33), Moderate-[0.33, 0.66), High-[0.66, 1.0)]
Link_Function	Identity	Logit
Measurement_Distribution	Log-Normal(0.75, 20, 5)	Log-Normal(0.75, 20, 5)
Num_Extraneous_Variables	0	0
Number_of_Features	2	2
Number_of_Model_Changes	0	0
Number_of_Subjects	2,007	2,556

Attribute	Length of Stay	24-Hr Mortality
Probability_Threshold	None	0.65
Random_Effects	Intercept	Intercept & time
Random_Effect_Collinearity	NA	[0.05, 0.99]
Random_Effect_Cut_Point	0.95	0.95
Random_Effect_Insert_Point	0.6	0.6
Realism_Functions	None	None
Sampling_Scheme	[Random, Not-Random, Equal]	[Random, Not-Random, Equal]
Temporal_Trend	Linear	Linear
Time_Breaks	None	None
Variance_of_Error	[0, 0.125]	[0, 0.05]
Variance_of_Random_Effects	[0.5, 2]	[0.5, 2]

Table 12: Synthetic Causal Data Group Parameters

To go from synchronous data to asynchronous data we created software to non-randomly eliminate feature values from existing longitudinal data sets. This program enabled us to change a specific data feature without altering the settings of other characteristics. The asynchronous data creator has two built in methods for punching out values: random and not-random. The random mechanism has a uniform random probability of whether a predictor was observed or not in each measurement occasion. If this random process decides a value was not observed, then the value is replaced with Numpy's nan value (a place holder for missing data). In the non-random mechanism, the actual value of the data informs how likely it is to be observed. The non-random mechanism takes a predictor's population mean and maximum absolute value as parameters. The non-random process has a base probability of observing a value, which can be decreased based on how little a predictor value deviates from the population mean. We use the max value to normalize the deviance, and we use a scale parameter to set an upper bound on how much the base probability can be reduced. Our non-random process is mathematically expressed in Equation 9. Predictors in the MIMIC III sample had an average missing rate between [48%, 60%]. Therefore we set the base missing probability to 27% and the scale parameter to 27%.

$$p_{observed} = p_{base} - scale\_parameter \times (1 - \frac{|\mu_x - x_{-,i,j}|}{x_{max}})$$

Equation 9: Non-Random Data Creation Formula

We evaluated two imputation strategies on the asynchronous data to see if there were any performance advantages of temporal representation that depended on the imputation method. The first strategy we evaluated was mean imputation, where we would substitute the population mean of a predictor for missing values. The second imputation method we evaluated was the mean-imputation + indicator method introduced in Chapter I.

Non-Causal Data Group Methods

In the Non-Causal Data Group we sought to explore datasets that increasingly reproduced the characteristics of our MIMIC III cohort. We took a step-wise approach to adding these characteristics (the measurement link, timespan link, and feature-link switches). We started by changing the predictors to no longer causally affect the outcome with measurement inactive, timespan, but the feature mechanisms set to active. Without the feature link-mechanism the feature would be totally uncorrelated with the outcome. Next, we generated data with the pairwise combinations of the feature-link and measurement link, as well as the feature-link and timespan link. Lastly we produced data with all the mechanisms active. We generated 20 datasets for each combination of characteristics for a total of 80 data sets per modeling problem. We varied fewer data characteristic parameters compared to the Causal Data Group: the autocorrelation and feature collinearity were held constant in the same bucket (high autocorrelation, moderate feature collinearity). In the classification data sets we varied the random effect collinearity, the random effect variance, the event rate, and the sampling scheme. In the regression data sets we varied the random-effect size, the measurement error variance, and the sampling scheme. Table 13 depicts the data generation parameters we used for the Non-Causal Data Group. As in the previous data group, we also used the asynchronous data creator with the same non-random method to transform our synchronous data sets into asynchronous ones. We again evaluated both mean imputation and the mean imputation + indicator missing data representation in the asynchronous data.

Attribute	Length of Stay	24-Hr Mortality
Autocorrelation	Moderate-[0.33, 0.66)	Moderate-[0.33, 0.66)
Coefficient_Values	Intercept: 1 Time: 5 x_1: 0 x_2: 0	Intercept: -5 Time: 5 x_1: 0 x_2: 0
Collinearity	Low-[0.01, 0.33)	Low-[0.01, 0.33)
Link_Function	Identity	Logit
Measurement_Distribution	Log-Normal(0.75, 20, 5)	Log-Normal(0.75, 20, 5)
Num_Extraneous_Variables	0	0
Number_of_Features	2	2
Number_of_Model_Changes	0	0
Number_of_Subjects	2,007	2,556
Probability_Threshold	None	[0.5, 0.8]
Random_Effects	Intercept	Intercept & time
Random_Effect_Collinearity	NA	[0.05, 0.99]
Random_Effect_Cut_Point	0.95	0.95
Random_Effect_Insert_Point	0.6	0.6
Realism_Functions	[None, Timespan, Measurement, Timespan & Measurement]	[None, Timespan, Measurement, Timespan & Measurement]
Sampling_Scheme	[Random, Not-Random, Equal]	[Random, Not-Random, Equal]
Temporal_Trend	Linear	Linear
Time_Breaks	None	None
Variance_of_Error	[0, 0.125]	0
Variance_of_Random_Effects	[0.5, 2]	[0.5, 2]

Table 13: Synthetic Non-Causal Data Group Parameters

LSTM and Attention Model Architecture, Tuning, and Evaluation Strategy

The data generating processes are the means by which we hope to learn more about the LSTM and Attention models. We will now discuss the architecture of our LSTM and Attention models, the tuning and training methods, as well as the evaluation strategy we used across all of the data groups. For a given temporal representation, we used the wide data format to feed a given measurement occasion directly into the LSTM or sequential encoder (Attention). That measurement occasion had access to information of past inputs either through the hidden memory state (LSTM) or the self-attention mechanism. That sequential block/neuron then produces an output that propagates into a dropout layer, which randomly introduces error into the recurrent output. The dropout layer helps the model from overfitting its parameters to the training data. This altered output then is fed into a feedforward network. The feedforward network interprets the input and at the last layer produces a prediction. Both the LSTM and the Attention models produce a prediction for each measurement occasion (input). Figure 24 shows the LSTM Architecture The number of ReLU layers was a tunable hyper-parameter.

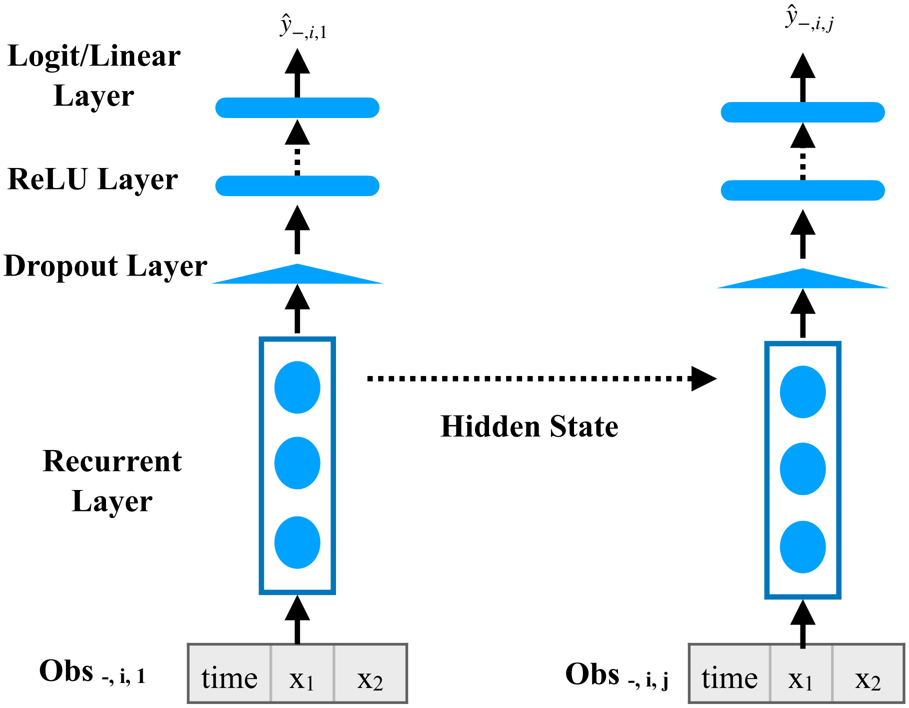


Figure 24: LSTM Model Architecture

As seen in Figure 25, we explicitly designed the attention mechanism to mask future data from informing current forecasts. We only use the encoder portion of the transformer proposed in Vaswani et. al.'s work as our Attention architecture[45]. In the LSTM model (Figure 24), the hidden memory state is only passed forwards in time and not backwards. In the case of the classification problem, the last layer of the feed forward network would be a sigmoid function. The last layer is a linear layer for the regression problem. The positional encoder and Multi-Head Attention Mechanism in Figure 25 can be further broken down into subcomponents.

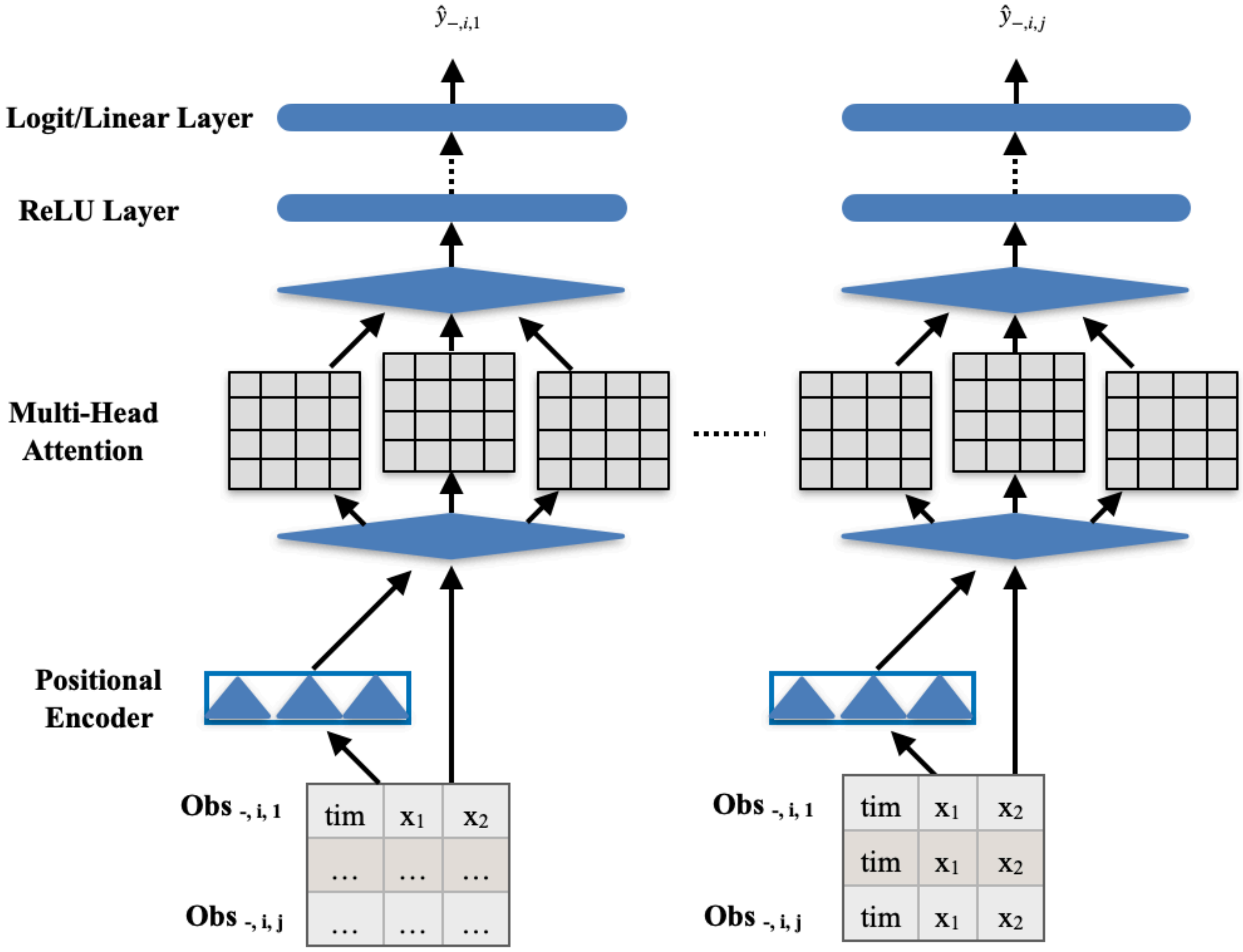


Figure 25: Attention Model Architecture

The positional encoder takes dimension of the full feature matrix for a subject where the rows are measurement occasions and the columns are predictors and adds two timing signal columns. The timing signal columns are created by calculating an incrementing factor based on the dimension of the hidden layer within the Attention model. Equation 10 shows how we calculate the incrementing factor. This incrementing factor is then multiplied by the sequential position (row number) of each measurement occasion. We then take the sin of all these scaled position values and also take the cosine of all these scaled positions. The timing signals are the results of these two trigonometric functions.

$$time\_increment = \frac{ln(10{,}000)}{(hidden\_size//2) - 1}$$

Equation 10: Time Increment for Positional Encoder

As mentioned in the introductory Chapter, Multi-head Attention is going to tune weights that focus the attention of the model on different measurement occasions depending on the value of the current observation.With multi-head Attention we simultaneously repeat the Attention weighting process multiple times and then take an average of the weights from the different Attention heads. The Attention mechanism we implemented applies a normalization layer (a layer that scales and shifts inputs to have a mean of zero and variance of one). Those outputs then go to the multi-head Attention (which splits the data to the different heads and then averages the results). Next, the data flow through a drop out layer, before going through another normalization layer. These outputs then go through a feed forward network made up of three layers (a linear, a ReLU, and then a linear layer). Finally, that output goes through another dropout layer. Figure 26 presents our implementation of the Attention-based Transformer encoder.

We held the loss function constant in our architecture. We used the MSE as the loss function (see Equation 3) for the LSTM and Attention regression models. This loss function was also used by the authors of the study we used as a benchmark[73]. Likewise, both deep classification models used the binary cross-entropy loss function (log loss). This function is equivalent to the log likelihood function of

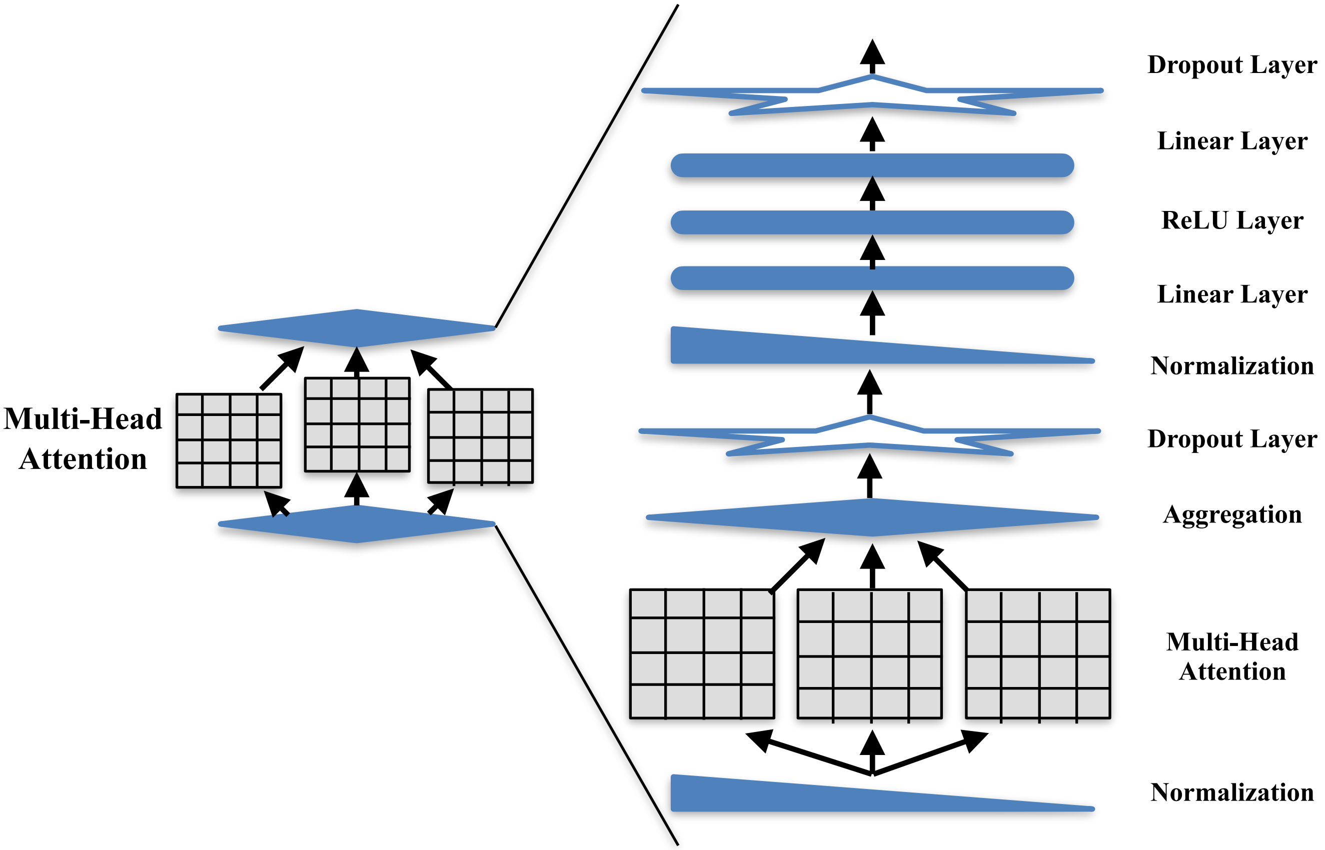


Figure 26: Attention Encoder Implementation

a generalized linear model for an unordered categorical outcome (with only two categories). One can see the components of the log loss function in Equation 11.

$$\frac{-1}{n}\sum_{i=1}^{n} y_i log(p(y_i)) + (1 - y_i)log(1 - p(y_i))$$

Equation 11: Binary Cross-Entropy Loss

We used nested 3-fold cross-validation as our evaluation strategy coupled with a uniformly random search of hyper-parameters. We used stratified cross-validation, which preserves the original event rate in each fold by dividing the cases and controls separately, preserving the proportions in each fold. The nested cross-validation strategy entails that we evaluate each combination of hyper-parameters with an inner cross-validation of data made of the outer training fold. We selected the hyper-parameters that maximized the cross-validated MSE (regression) or log loss (classification). For each outer fold we evaluated 25 unique sets of hyper-parameters. Each set of tuning hyper-parameters trained on the inner-

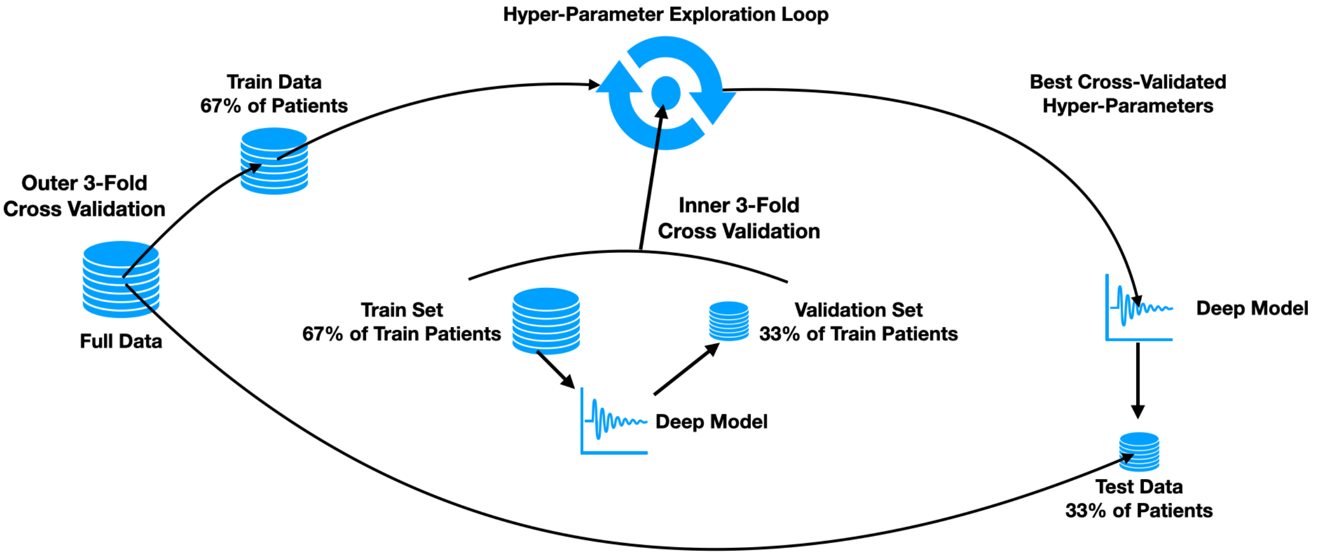


Figure 27: Evaluation and Hyper-Parameter Tuning Strategy

fold five times (also known as five epochs). After selecting the best cross-validated hyper-parameters we trained the model on the outer-fold for 30 epochs (30 times through the entire data set). We chose the parameter values three folds, 25 draws, five epochs, and 30 epochs after a series of benchmarking experiments. More epochs or hyper-parameter draws could have led to a better performing model, but the cost in compute time is non-linear. Even with these modest parameters, our deep models trained on 1,215 separate epochs (3 outer-cross-validation folds x (25 hyper-parameter draws x 3 inner-cross-validation folds x 5 tuning epochs + 30 final training epochs)). Figure 27 provides a visual of our tuning, training, and evaluation scheme.

We chose to tune as many hyper-parameters as possible in an attempt to find the best fitting model empirically, as well as to prevent developer-induced overfitting via a trial and error process. As previously mentioned we produce sets of hyper-parameters in a uniformly random way. Table 14 shows the hyper-parameters tuned and the possible range of values. Many hyper-parameters are restricted to multiples of eight, because of the performance advantages of working with NVIDIA cuda-tensors (which are restricted to vectors and matrices with a dimension divisible by eight)[156]. Some hyper-parameters, such as depth, are a functions of other hyper-parameters.

Hyper-parameter	Related Model	Value Range
Batch size	Attention, Feedforward, LSTM	[8, 16, 24, …, 136]
Depth	Attention	[1, 2, 3, …, 17] x number_of_heads
Drop out rate	Attention, Feedforward, LSTM	(0, 0.1)
Hidden dimension	Attention, Feedforward, LSTM	[8, 16, 24, …, 104]
Learning rate	Attention, Feedforward, LSTM	(0.0001, 0.001)
Number of layers	Feedforward	[1, 2, 4, 8]
Number of heads	Attention	[1, 2, 4, 8]
Optimizer	Attention, Feedforward, LSTM	[ADAM, Wighted ADAM, Rprop, Centered RMS]
Weight decay	Attention, Feedforward, LSTM	(0, 0.25)

Table 14: Hyper-parameter Tuning Values

Model Performance Analysis and Inference Methods

After tuning and training the model, we used the MSE and explained variance score to evaluate the test-folds of the regression data sets. In the classification data sets we used the AUROCC and the brier score metrics for evaluation. We evaluated each data set with both deep sequence model architectures and each temporal representation of interest (absolute time, relative time, sequence time, and window-time). We treated window-size as a tunable hyper-parameter that was specific only to window-time. The window size could vary from (0.036, 0.5], note that time was bounded to the (0, 1) interval. The (0, 1) interval can be mapped to any interval of real numbers, therefore this representation should generalize to more conventional definitions of time.

For the synchronous data sets, we fit the perfectly specified GLME model using the absolute time representation to act as a theoretical baseline. We replicated the stratified cross-validation strategy we used for the deep models. These methods split the folds non-randomly and sequentially, meaning that all of the different models evaluated the exact same folds of data for all the temporal representations.

We designed our model fitting programs to fit and produce output for all the files within a directory. Each program would produce a separate output of the evaluation data for each fold of each unique data set for each temporal representation for each model. Each model fit would produce a different file with the selected hyper-parameters for that fold. This created thousands of output files, which we used a separate program to aggregate. We chose this design to minimize data loss in the event of job interruption. We aggregated all the unique files into a single data frame which we then saved off for final analysis.

Using the aggregated data frame, we used the Matplotlib package to visualize some aggregate differences before doing deeper analysis. In this deeper analysis we used robust regression within STATA that makes use of Huber-White standard errors[157]. In these analyses we would specify all data characteristics that varied. We added interaction terms between the temporal representation and other variables in an attempt to minimize the Bayesian Information Criterion (BIC). In our model the MSE was the outcome of interest for the length of stay problem, and the brier score was the outcome of interest for the classification. The random effect collinearity, random effect variance, unobserved measurement error variance, sampling method, feature collinearity, the feature autocorrelation, and the temporal representation were all predictors in our analysis. The goal of using a regression analysis was to ascertain the effect of the temporal representation on performance, while accounting for the variable parameters in the data generating process. This analysis also allowed us to search for interactions terms between the temporal representation used and the variable data generating parameters. An interaction would suggest that there is a relationship between that data characteristic, e.g., feature collinearity and the temporal representation used. Such a relationship between a data characteristic could lead to useful rules of thumb or model building decision support. For example, if the average feature collinearity exceeds 0.5, then use a relative time representation.

In the asynchronous data group, the imputation method was included as a predictor. We assessed the quality of our inference model through residual versus fitted (RVF) plots and residual versus predictor (RVP) plots. We adjusted relationships between a predictor and outcome to a non-linear form based on the RVP plot. Again, we attempted to minimize the BIC when adding non-linear terms to the base model. We report the final model, its fit characteristics, and the Second Generation p-values of the coefficients. We used a simple interval for categorical predictors, but took a different approach for continuous predictors. We multiplied the effect-size estimates of a continuous predictors by their inter-quartile range (IQR) to address differences of scale in the parameters. We chose not to standardize the data, so that the coefficients would be interpretable.

Causal Synchronous Data Group Results

For the Causal Synchronous Data Group, we generated 120 data sets for each modeling problem. Signal to noise parameters, such as the random effect variance, feature collinearity, random effect collinearity, and measurement error were varied for study. A summary of these data characteristics is presented in Table 15. In the table, we also contrast the collinearity and autocorrelation values we used to create the data to what was observable. Depending on the sampling frequency, the observed collinearity and autocorrelation can be significantly divergent from the underlying process.

Parameter	Modeling Problem	Summary Statistics
Autocorrelation Type (Mean %)	Classification	55.0% High 25.0% Moderate 20.0% Low
	Regression	62.5% High 16.7% Moderate 20.8% Low
Observed Autocorrelation (Median [IQR])	Classification	0.293 [0.001, 0.911]
	Regression	0.261 [0.002, 0.913]
Collinearity Type (Mean %)	Classification	20.8% High 16.7% Moderate 62.5 % Low
	Regression	24.1% High 21.7% Moderate 54.2% Low
Observed Collinearity (Median [IQR])	Classification	0.034 [0.022, 0.162]
	Regression	0.037 [0.025, 0.075]
Measurement Error Variance (Median [IQR])	Classification	0.028 [0.015, 0.041]
	Regression	0.074 [0.038, 0.098]
Random Effect Collinearity (Median [IQR])	Classification	0.574 [0.282, 0.673]
	Regression	NA
Random Effect Variance (Median [IQR])	Classification	1.24 [0.833, 1.70]
	Regression	1.17 [0.840, 1.70]
Sampling Type (Mean %)	Classification	30.8% Equal 32.5% Not-Random 36.7% Random
	Regression	35.0% Equal 33.3% Not-Random 31.7% Random

Table 15: Causal Synchronous Data Group Summary Statistics

We performed three fold cross-validation for two different deep models using four different temporal representations over 240 unique data sets (120 for each modeling problem). We also used three fold cross validation to fit a GLME baseline model on all 240 data sets. Between all the different data sets, models, folds, and temporal representations, we had 6,480 data points (3,240 per modeling problem) to analyze in the Causal Synchronous Data Group (CSDG). Figure 28 depicts some differences between the temporal representation and the models, but the significance of these differences is unclear without the regression analysis to separate effects. In Figure 28, tiles that have a darker blue shade are better, while tiles that have a darker red shade are worse. Each figure displays the metric average (avg) and its asymptotic standard error at the 95% level (α = 0.05). In the classification setting, the true GLME model has the best discrimination (AUROCC), but not the best calibration (Brier score). The true GLME serves as a benchmark in this setting, representing the best achievable fit based on the population mean. In both figures we see evidence that the absolute time and the window-time representations may be generally superior to relative-time and sequence-time.

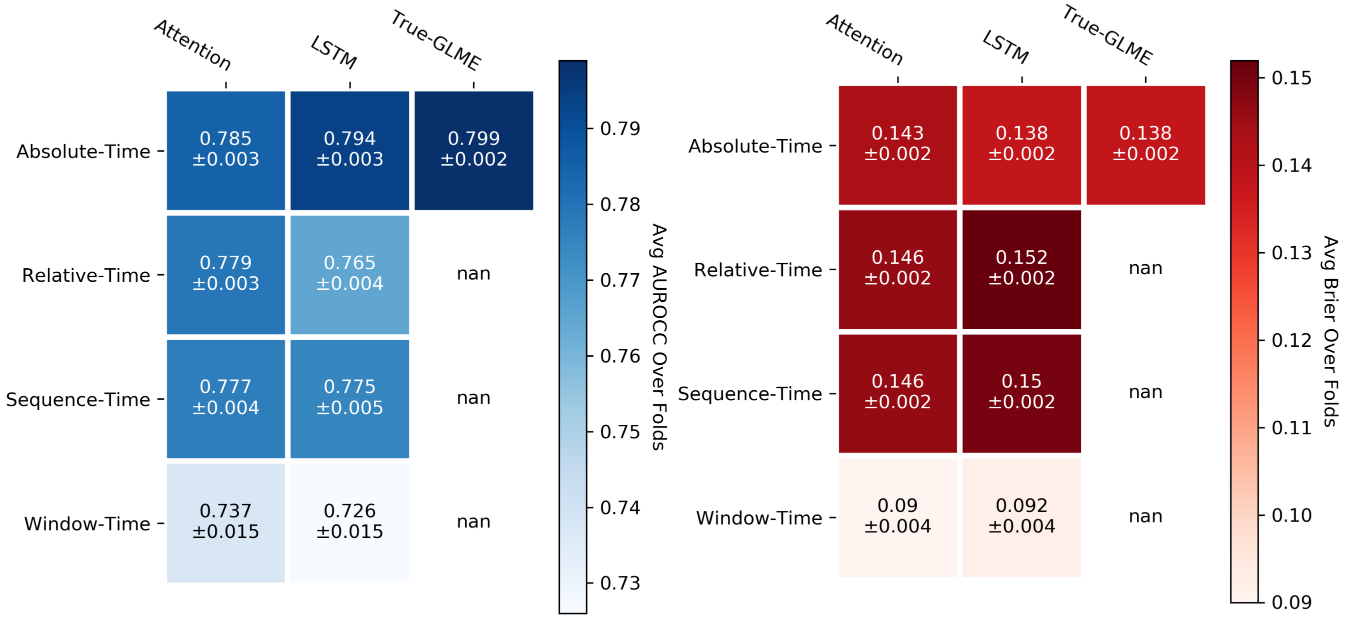


Figure 28: CSDG Classification Model Performance vs. Temporal Representation

We used the Brier score as the outcome of interest in our regression analysis as the outcome of interest for the classification problem in the CSDG. We used the Brier score as opposed to the AUROCC because the Brier score better satisfied the linear hypothesis of the predictors, leading to a higher quality analysis. The interpretation of the coefficient signs is reversed, where a negative coefficient (less Brier error) is associated with a better fit model, while a positive coefficient is associated with a worse fit model. We found several non-linear relationships between the data characteristics and the Brier score, as well as several interactions with temporal representation that significantly contributed to the analysis BIC. Table 16 shows the robust confidence intervals for the analysis coefficients as well as the Second-Generation p-values. As previously mentioned, we used an interval null hypothesis (H_0) to discriminate against effects that were statistically significant, but practically meaningless. We used the IQRs from Table 15 to inform our scaling adjustment when comparing continuous predictors to the interval null hypothesis. Our interval null hypothesis spanned the range of -0.1 to 0.1, and we set 5% as our acceptable amount of Type-1 error ($H_0 = [-0.01, 0.01]$, $\alpha = 0.05$). The overall model R^2 was 0.896 and the BIC was -17,372 with 30 degrees of freedom. We observed several practically significant features that affected the Brier score. The autocorrelation of the predictors significantly increased the Brier score in a non-linear fashion. Increased amounts of inter-subject variability (random effect variance) also increased the Brier score. However, the deep models were able to harness the mechanism of the non-random sampling scheme to significantly decrease the Brier score loss. Furthermore, smoothing through the window-time representation significantly improved model calibration, though those improvements could be offset by higher levels of inter-subject variability.

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Observed Autocorrelation	[-0.018, 0.005]	0.652	No
Squared Observed Autocorrelation	[0.032, 0.050]	0.0	Yes
Observed Collinearity	[0.0, 0.006]	1.0	No

Predictor	**Robust 95% Confidence Interval**	**2nd-Gen p-value**	**Significant?**
Not-Random Sampling (Equal as Reference)	[-0.016, -0.011]	0.0	Yes
Random Sampling (Equal as Reference)	[0.0, 0.003]	1.0	No
Random Effect Variance	[0.014, 0.016]	0.0	Yes
Random Effect Collinearity	[0.010, 0.013]	1.0	No
Error Variance Restricted Cubic Spline Component 1	[0.058, 0.356]	1.0	No
Error Variance Restricted Cubic Spline Component 2	[-0.868, -0.028]	1.0	No
Error Variance Restricted Cubic Spline Component 3	[-0.24, 2.254]	1.0	No
LSTM model (Attention as Reference)	[-0.006, -0.004]	1.0	No
Relative-Time (Absolute as Reference)	[0.001, 0.006]	1.0	No
Sequence-Time (Absolute as Reference)	[0.003, 0.008]	1.0	No
Window-Time (Absolute as Reference)	[-0.084, -0.072]	0.0	Yes
Relative-Time x LSTM	[0.010, 0.012]	0.173	No
Sequence-Time x LSTM	[0.008, 0.011]	0.691	No
Window-Time x LSTM	[0.004, 0.010]	1.0	No
Relative-Time x Random	[0.002, 0.006]	1.0	No
Sequence-Time x Random	[0.0, 0.004]	1.0	No
Window-Time x Random	[0.001, 0.007]	1.0	No
Relative-Time x Not-Random	[-0.001, 0.003]	1.0	No
Sequence-Time x Not-Random	[-0.001, 0.002]	1.0	No
Window-Time x Not-Random	[-0.010, 0.0]	1.0	No
Relative-Time x Random Effect Variance	[-0.003, 0.0]	1.0	No
Sequence-Time x Random Effect Variance	[-0.005, -0.002]	1.0	No
Window-Time x Random Effect Variance	[0.014, 0.020]	0.0	Yes
Relative-Time x Autocorrelation	[-0.004, 0.001]	1.0	No

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Sequence-Time x Autocorrelation	[-0.001, 0.004]	1.0	No
Window-Time x Autocorrelation	[0.007. 0.016]	0.443	No

Table 16: CSDG Classification Performance Results

The diagnostic plots for the classification analysis signified solid inferential characteristics of the model. The RVF plot in Figure 29 shows some heteroskedasticity in the residuals between the interval before 0.1 and the interval after. However, the residuals appear randomly centered on zero (see the Lowess curve) across the whole x-axis, meaning that the plot does not show bias.

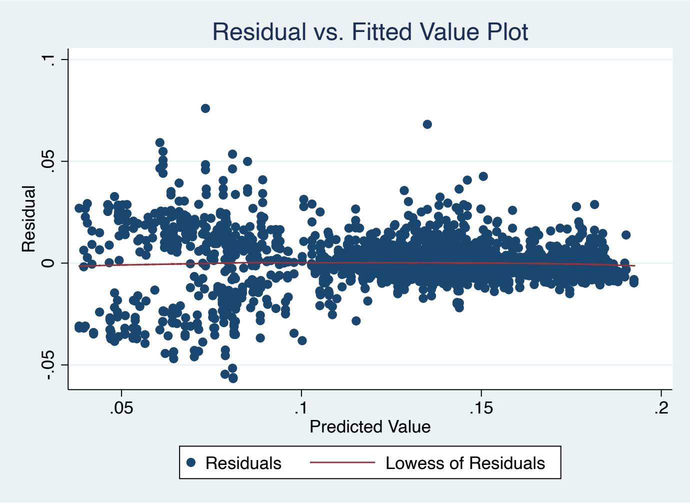


Figure 29: CSDG Classification RVF Plot

In the regression setting, the true GLME model both maximizes the explained variance and minimizes the MSE as shown by Figure 30. We see some minor differentiation between the Attention and LSTM models, with the Attention models performing slightly better. As in the classification case, absolute-time and window-time appear to generally maximize model performance better than other representations.

The performance analysis for the regression data sets used MSE as the outcome of interest. Again, we found several non-linear relationships between the data characteristics and the MSE that

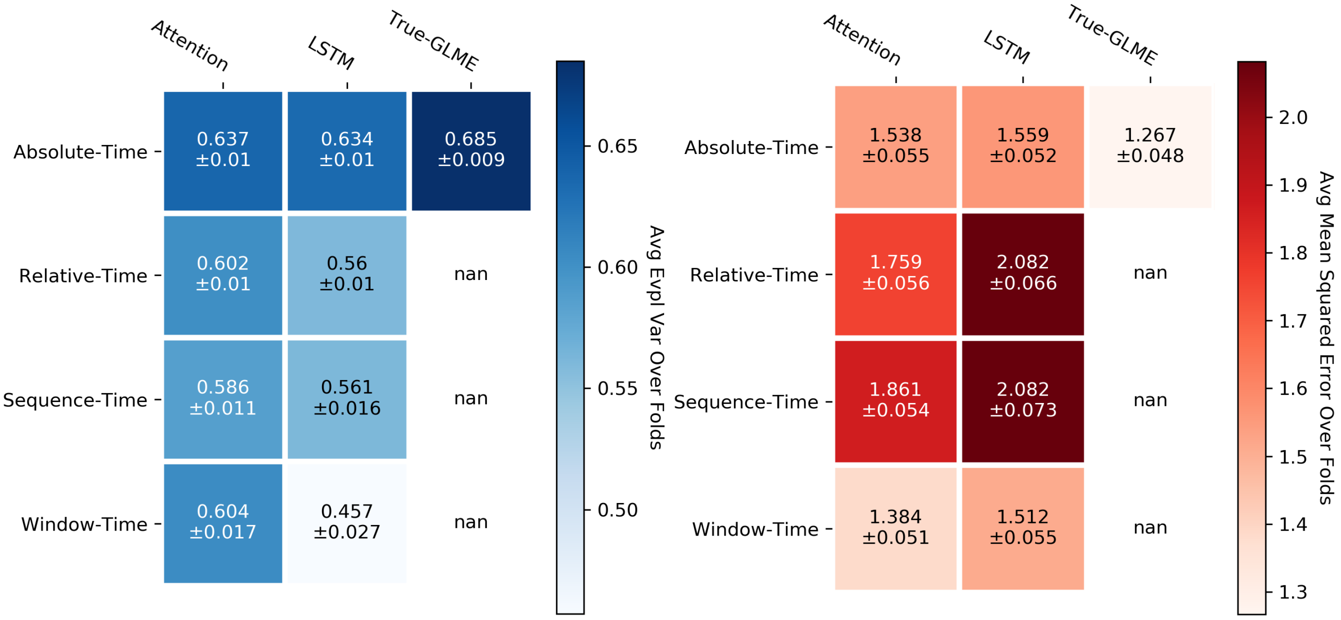


Figure 30: CSDG Regression Model Performance vs. Temporal Representation

lowered the BIC. Table 17 shows the robust confidence intervals for the regression model coefficients with the Second-Generation p-values ($H_0 = [-0.125, 0.125]$, $\alpha = 0.05$). The regression inference model achieved an R^2 of 0.712 and the BIC was 2,009.6 with 31 degrees of freedom. Increases in the amount of inter-subject variance (random effect variance) led to increases in model error. The measurement error had a large effect on MSE that grew exponentially. Relative to absolute-time, the sequence-time and relative-time representations significantly increased model error. We found a couple significant interactions between the temporal representation and model architecture as well as between temporal representation and sampling type. The LSTM model saw significant increases in MSE when used with relative-time and sequence-time relative to the Attention model. The window-time representation performed significantly worse when used in randomly sampled data rather than data that is regularly or semi-regularly spaced over time.

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Observed Autocorrelation	[0.071, 0.187]	0.595	No
Observed Collinearity	[-0.013, 0.515]	1.0	No

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Squared Observed Collinearity	[-0.619, -0.081]	1.0	No
Not-Random Sampling (Equal as Reference)	[-0.043, -0.092]	1.0	No
Random Sampling (Equal as Reference)	[-0.112, -0.036]	1.0	No
Random Effect Variance	[0.341, 0.800]	0.0	Yes
Squared Random Effect Variance	[0.102, 0.288]	0.233	No
Random Effect Collinearity	[-0.253, 0.160]	0.0	Yes
Squared Random Effect Collinearity	[-0.179, 0.193]	1.0	No
Error Variance	[-4.694, -1.788]	0.102	No
Squared Error Variance	[12.508, 33.868]	1.0	No
LSTM model (Attention as Reference)	[-0.009, 0.051]	1.0	No
Relative-Time (Absolute as Reference)	[0.201, 0.459]	0.0	Yes
Sequence-Time (Absolute as Reference)	[0.537, 0.846]	0.0	Yes
Window-Time (Absolute as Reference)	[-0.221, -0.075]	0.342	No
Relative-Time x LSTM	[0.239, 0.365]	0.0	Yes
Sequence-Time x LSTM	[0.118, 0.269]	0.046	Yes
Window-Time x LSTM	[0.066, 0.142]	0.776	No
Relative-Time x Random	[0.129, 0.276]	0.0	Yes
Sequence-Time x Random	[0.043, 0.236]	0.425	No
Window-Time x Random	[0.140, 0.236]	0.0	Yes
Relative-Time x Not-Random	[-0.034, 0.174]	0.764	No
Sequence-Time x Not-Random	[-0.255, -0.013]	0.463	No
Window-Time x Not-Random	[-0.126, -0.005]	0.992	No
Relative-Time x Random Effect Variance	[-0.168, -0.017]	0.715	No
Sequence-Time x Random Effect Variance	[-0.307, -0.010]	0.387	No

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Window-Time x Random Effect Variance	[-0.053, 0.044]	1.0	No
Relative-Time x Autocorrelation	[-0.314, -0.094]	0.197	No
Sequence-Time x Autocorrelation	[-0.429, -0.108]	0.091	No
Window-Time x Autocorrelation	[-0.161, -0.019]	0.833	No

Table 17: CSDG Regression Performance Results

The diagnostic plots for the regression analysis do not show evidence of poor fit. The RVF plot in Figure 31 shows a good shape that is centered at zero (see Lowess curve). There is not much evidence of heteroskedasticity and the plot does not give cause for concern about the inference models.

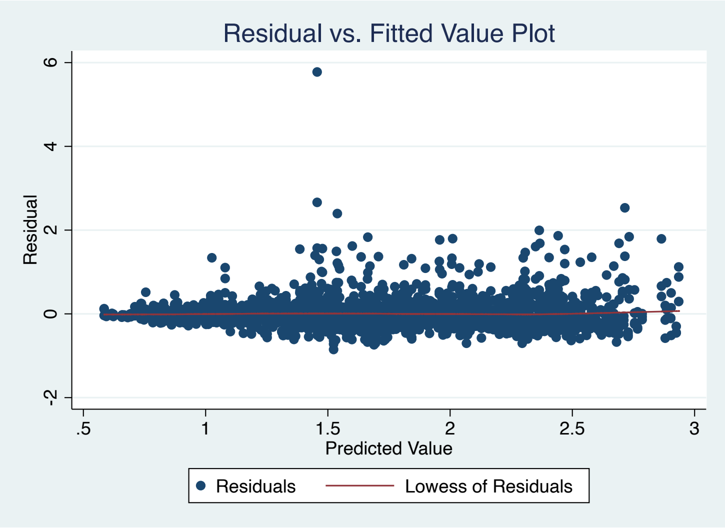


Figure 31: CSDG Regression RVF Plot

Causal Asynchronous Data Group Results

The Causal Asynchronous Data Group (CADG) was derived from the synchronous group by inducing missing values in X_1 and X_2. The median rate of missing data in X_1 was 48.6% with an IQR of

[47.9%, 49.1%]. The median rate of missing data in X_2 was 47.7% with and IQR of [45.6%, 48.8%]. Based on these rates, one would approximately expect a measurement occasion to observe either X_1 or X_2 50% of the time, both X_1 and X_2 25% of the time, and neither X_1 nor X_2 25% of the time. Despite the missing values, the discrimination was equivalent to the fully informed models of the CSDG, though the calibration was significantly worse when compared to the fully synchronous data sets.

The inference model on classification performance required squared terms on autocorrelation and the random effect variance variables to satisfy the linear hypothesis. There were also a few interactions that improved the model BIC (temporal representation x model type, temporal representation x sampling method, temporal representation x autocorrelation, and temporal representation x random effect variance). Table 18 lays out the robust confidence intervals for the model coefficients as well as the Second-Generation p-values ($H_0 = [-0.01, 0.01]$, $\alpha = 0.05$). The inference model had an R^2 of 0.869 and a BIC of -27,750. We discovered several significant effects in the causal asynchronous setting for classification models. The autocorrelation decreased the Brier score, though in a nonlinear fashion. It is important to note that the autocorrelation is bound between [-1, 1]. Non-random sampling decreased the Brier score, suggesting that the deep models were able to take advantage of the structure of the sampling. Not surprisingly, increases in the error variance or random effect variance increased the Brier score loss. The window-time representation significantly decreased the Brier score loss. The performance gain for window-time specifically was significantly reduced by increases in random effect variance and by increases in feature autocorrelation.

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Observed Autocorrelation	[-0.036, -0.016]	0.0	Yes
Squared Observed Autocorrelation	[0.043, 0.058]	0.0	Yes
Observed Collinearity	[-0.002, 0.004]	1.0	No

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Not-Random Sampling (Equal as Reference)	[-0.021, -0.016]	0.0	Yes
Random Sampling (Equal as Reference)	[0.003, 0.005]	1.0	No
Random Effect Variance	[0.024, 0.035]	0.0	Yes
Squared Random Effect Variance	[-0.008, -0.003]	0.0	No
Random Effect Collinearity	[0.009, 0.012]	0.399	No
Error Variance	[0.034, 0.087]	1.0	No
LSTM model (Attention as Reference)	[-0.006, -0.004]	1.0	No
Relative-Time (Absolute as Reference)	[0.005, 0.008]	1.0	No
Sequence-Time (Absolute as Reference)	[0.003, 0.007]	1.0	No
Window-Time (Absolute as Reference)	[-0.085, -0.076]	0.0	Yes
Mean Imputation (Mean + Indicator as Reference)	[-0.001, 0.000]	1.0	No
Relative-Time x LSTM	[0.010, 0.012]	0.173	No
Sequence-Time x LSTM	[0.009, 0.011]	0.691	No
Window-Time x LSTM	[0.004, 0.009]	1.0	No
Relative-Time x Random Effect Variance	[-0.005, -0.002]	1.0	No
Sequence-Time x Random Effect Variance	[-0.005, -0.002]	1.0	No
Window-Time x Random Effect Variance	[0.014, 0.019]	0.0	Yes
Relative-Time x Autocorrelation	[0.001, 0.003]	1.0	No
Sequence-Time x Autocorrelation	[0.002, 0.005]	1.0	No
Window-Time x Autocorrelation	[0.013. 0.019]	0	Yes

Table 18: CADG Classification Performance Results

The diagnostics for the classification inference model show some heteroskedasticity in RVF plot shown in Figure 32. There appear to be some patterns in the residuals before 0.1 on the X-axis, signifying some potential for unaccounted correlation. The lowess of the residuals is well centered at zero. Beyond 0.1 on the X-axis, the plot appears to have a well fitted stochastic distribution about zero.

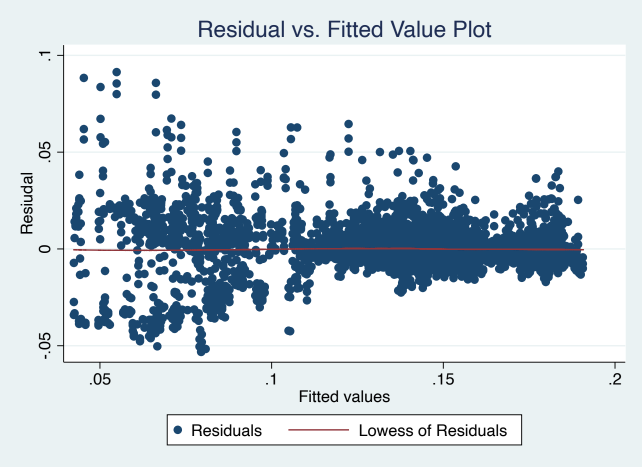


Figure 32: CADG Classification RVF Plot

Unlike the classification models, the regression models generally saw performance degrade in comparison to the CSDG as measured by MSE as well as explained variance. There was also more differentiation between temporal representations. There was also some small differentiation between the Attention and LSTM architectures, with a slight advantage appearing for the Attention models across both MSE and explained variance.

In the inference model on regression performance for the CADG, we found that increases in the size of the random intercepts as well as the measurement error increased the MSR. The temporal representations of relative and sequence time were significantly worse than absolute time, while the window-time representation was significantly better. As in the classification model case, the performance gains from using window-time can be negatively affected by increases in feature autocorrelation. Relative time and sequence time both benefited from increases in autocorrelation. The interactions with of temporal representation and model architecture remained significant. Table 19 visualizes the coefficient estimates and the Second Generation p-values ($H_0 = [-0.125, 0.125]$, $\alpha = 0.05$) for the regression inference model on the CADG. This inference model had an R^2 of 0.760 and a BIC of

2,821. Autocorrelation, collinearity, random effect variance and the measurement error variance all had quadratic terms that improved model BIC. This model included interactions between the temporal representation and the model type, the temporal representation and the sampling type, the temporal representation and the random effect variance, and the temporal representation and the autocorrelation.

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Observed Autocorrelation	[-0.193, 0.318]	1.0	No
Squared Observed Autocorrelation	[-0.536, -0.105]	0.106	No
Observed Collinearity	[0.144, 0.561]	1.0	No
Squared Observed Collinearity	[-0.640, -0.194]	1.0	No
Not-Random Sampling (Equal as Reference)	[-0.194, -0.039]	0.555	No
Random Sampling (Equal as Reference)	[-0.119, -0.060]	1.0	No
Random Effect Variance	[1.067, 1.207]	0.0	Yes
Squared Random Effect Variance	[0.062, 0.221]	0.559	No
Error Variance	[-3.062, -0.653]	0.594	No
Squared Error Variance	[9.513, 27.793]	1.0	No
LSTM model (Attention as Reference)	[0.016, 0.064]	1.0	No
Relative-Time (Absolute as Reference)	[0.367, 0.636]	0.0	Yes
Sequence-Time (Absolute as Reference)	[0.395, 0.649]	0.0	Yes
Window-Time (Absolute as Reference)	[-0.629, -0.500]	0.0	Yes
Mean Imputation (Mean + Indicator as Reference)	[-0.023, 0.013]	0.0	No
Relative-Time x LSTM	[0.278, 0.385]	0.0	Yes
Sequence-Time x LSTM	[0.174, 0.277]	0.0	Yes
Window-Time x LSTM	[0.034, 0.091]	1.0	No
Relative-Time x Random	[0.068, 0.193]	0.456	No
Sequence-Time x Random	[0.096, 0.220]	0.234	No
Window-Time x Random	[0.108, 0.176]	0.25	No
Relative-Time x Not-Random	[-0.199, 0.021]	0.664	No
Sequence-Time x Not-Random	[-0.214, -0.015]	0.553	No
Window-Time x Not-Random	[-0.126, -0.005]	0.992	No
Relative-Time x Random Effect Variance	[-0.188, -0.056]	0.677	No

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Sequence-Time x Random Effect Variance	[-0.197, -0.072]	0.587	No
Window-Time x Random Effect Variance	[-0.010, 0.057]	1.0	No
Relative-Time x Autocorrelation	[-0.446, -0.220]	0.0	Yes
Sequence-Time x Autocorrelation	[-0.362, -0.160]	0.0	Yes
Window-Time x Autocorrelation	[0.263, 0.369]	0.0	Yes

Table 19: CADG Regression Performance Results

The regression inference diagnostic plots show fewer signs of bias and correlated residuals than the classification analysis. The RVF plot shown in Figure 33 is well centered at zero. Figure 38 has a stochastic pattern around zero that does not imply problems in fit.

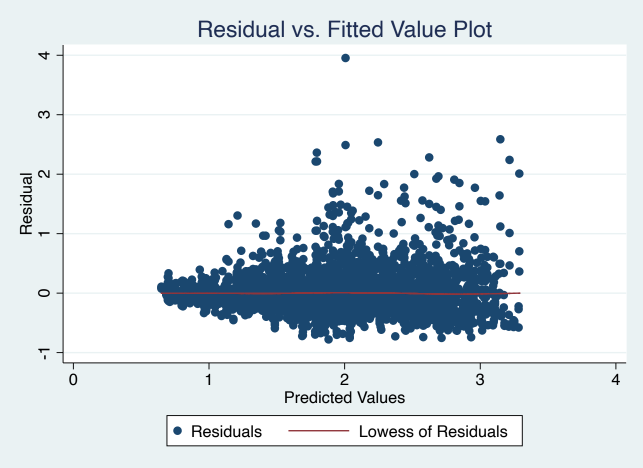


Figure 33: CADG Regression RVF Plot

Non-Causal Synchronous Data Group Results

For the Non-Causal Synchronous Data Group (NSDG), we generated 80 data sets for each modeling problem (four batches of 20 data sets). Each batch used a different combination of realism mechanisms (timespan link, feature link, and/or measurement link). Within each batch we uniformly randomly varied some signal to noise parameters such as the random effect variance and the measurement error. A summary of the stochastic data characteristics are laid out in Table 20.

Parameter	Modeling Problem	Summary Statistics
Event Rate (Median [IQR])	Classification	9.7% [5.4%, 16.9%]
	Regression	NA
Measurement Error Variance (Median [IQR])	Classification	0 [0, 0]
	Regression	0.066 [0.022, 0.092]
Probability Threshold	Classification	0.600 [0.610, 0.744]
	Regression	NA
Random Effect Collinearity (Median [IQR])	Classification	0.445 [0.254, 0.776]
	Regression	NA
Random Effect Variance (Median [IQR])	Classification	1.273 [0.985, 1.684]
	Regression	1.213 [0.932, 1.579]
Sampling Type (Mean %)	Classification	100% Equal
	Regression	100% Equal

Table 20: Non-Causal Synchronous Data Group Summary Statistics

As with the previous data group, we performed three fold cross-validation for two different deep models using four different temporal representations over the 80 data sets per modeling problem. Again, we used three fold cross-validation to fit a GLME baseline model on the NSDG. There were a total of 4,320 data points (2,160 per modeling problem).

Based on Figure 34, the GLME performed poorly compared to the attention models. This finding suggests that the attention models were able to find meaningful associations in the longitudinal data for each patient that the GLME could not. Not surprisingly, the non-causal data, was generally more difficult for the models to fit in comparison to the causal data. Figure 34 does not suggest meaningful differences between temporal representations.

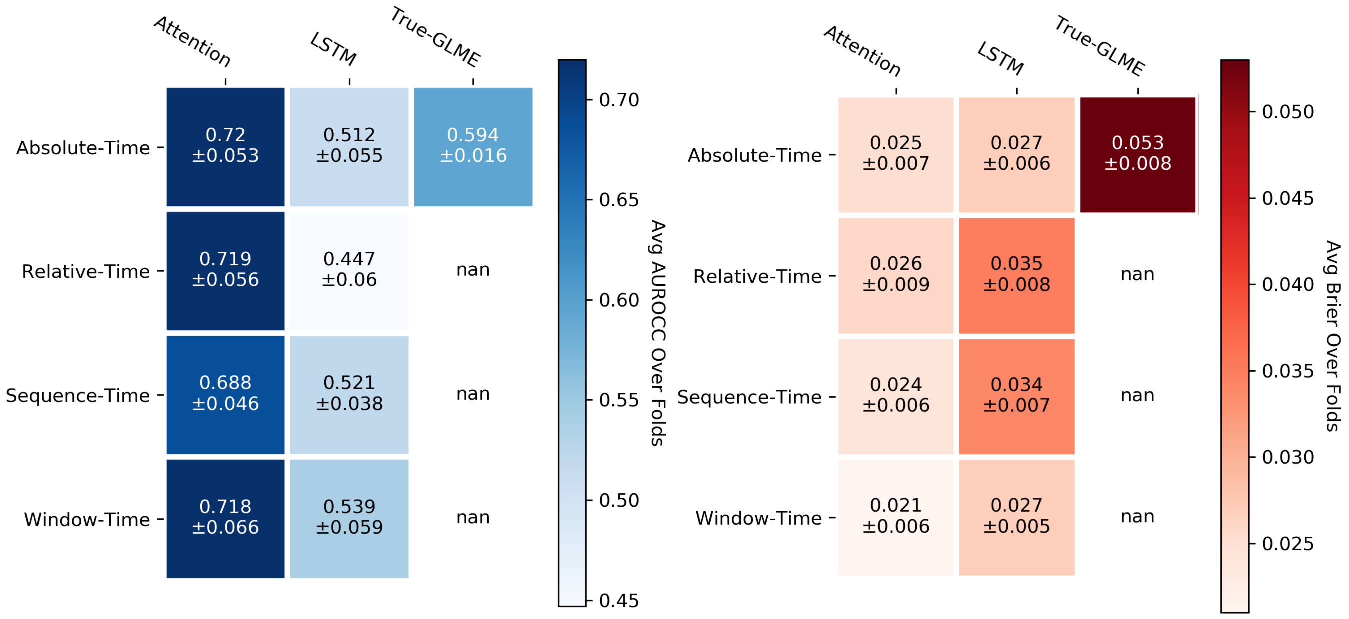


Figure 34: NSDG Classification Model Performance vs. Temporal Representation

In the analysis of the performance data the Brier score was our outcome of interest for the classification data analysis, because it produced the best inferential characteristics. The analysis did not show that any one temporal representation or modeling architecture had advantages over another. This finding suggests that the temporal representation is not significant in the non-causal synchronous setting. As in the causal data groups, we observed that increases in the random effect variance or collinearity significantly increased the Brier score. As previously mentioned, increases in the spread of subject averages from the population mean intuitively decrease the performance of a mean model. We found two non-linear terms that helped to minimize the BIC of the NSDG classification inference model: a square term on the random effect variance and a square term on the probability threshold. We did not find any meaningful interactions of other variables with temporal representation. Table 21 depicts the robust confidence intervals for the model coefficients as well as the Second-Generation p-values (H_0 = [-0.01, 0.01], α = 0.05). The model R^2 was 0.664 and had a BIC of -10,392 with 12 degrees of freedom. Decreasing the event rate (increasing the probability threshold), significantly and nonlinearly improved the Brier score. The timespan link between the number of measurement occasions and the length of the observation period also improved model performance. Increases in inter-subject variability had a strong effect on increasing the Brier score loss. There did not appear to be significant differences between model architectures or temporal representations.

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Probability Threshold	[-0.13, 0.144]	1.0	No
Square of Probability Threshold	[-0.283, -0.069]	0.0	Yes
Timespan Link On	[-0.063, -0.059]	0.0	Yes
Measurement Link On	[-0.004, 0.000]	1.0	No
Measurement Link x Timespan Link	[0.001, 0.007]	1.0	No
Random Effect Variance	[0.057, 0.076]	0.0	Yes
Random Effect Variance Squared	[-0.017, -0.008]	0.222	No
Random Effect Collinearity	[0.021, 0.026]	0.0	Yes
LSTM model (Attention as Reference)	[0.004, 0.007]	1.0	No
Relative-Time (Absolute as Reference)	[-0.001, 0.003]	1.0	No
Sequence-Time (Absolute as Reference)	[-0.001, 0.003]	1.0	No
Window-Time (Absolute as Reference)	[-0.004, -0.001]	1.0	No

Table 21: NSDG Classification Performance Results

The fit of the classification NSDG inference model was not as good as in the Causal Synchronous Data Group. There is some clear curvature in the RVF plot shown in Figure 35, suggesting model misspecification; however, there does not appear to be evidence of heteroskedasticity. These characteristics imply that there may be some biases and significant unexplained variance in the model specification. While coefficient estimates are biased, the inferences may still relevant given the linear specification (identity link function) of the inference model.

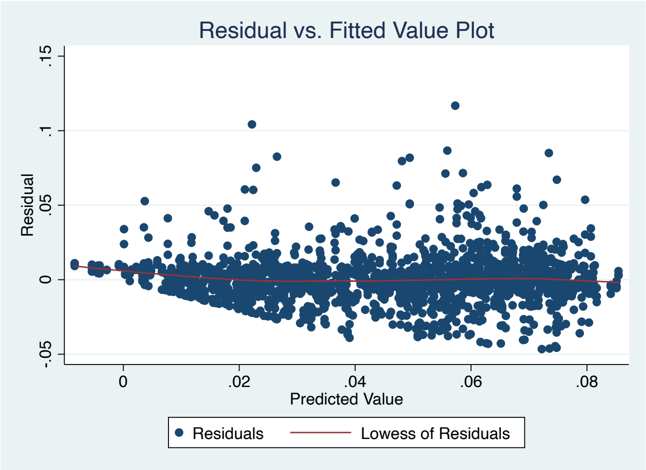


Figure 35: NSDG Classification RVF Plot

Based on Figure 36, all of the temporal representations struggled to fit the regression problem. Some models experienced more variance in the error than the outcome, leading to a negative explained variance score. The true GLME model maximized the explained variance, but also had the greatest MSE. From the MSE perspective, the deep architectures significantly exceeded the benchmark performance established by the GLME. There was not much differentiation between temporal representations in the figures; however, the Attention models tended to perform better than the LSTM.

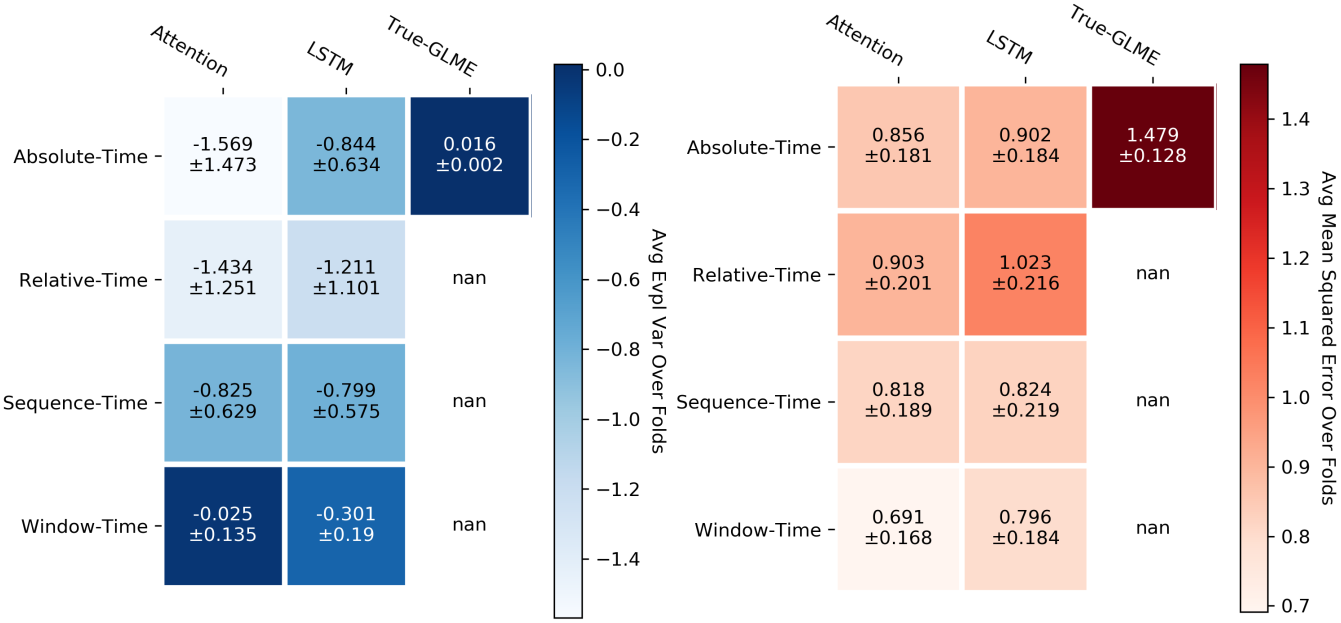


Figure 36: NSDG Regression Model Performance vs. Temporal Representation

The poor fit characteristics of the deep models on the regression portion of the NSDG was reflected in the inference model. There were no non-linear terms that resulted in smaller BIC values. There was a significant interaction between the measurement link and the random effect variance, however there were no meaningful interactions with temporal representation. Table 22 shows the robust confidence intervals for the model coefficients as well as the Second-Generation p-values (H_0 = [-0.125, 0.125], α = 0.05). The model R^2 was 0.463 and had a BIC of 2,859 with 11 degrees of freedom. The measurement link interaction and the random effect variance (inter-subject variability) were the only significant variables in this model. As the inter-subject variability increased, so did the MSE of the deep models. The measurement link significantly decreased MSE, and this effect grew as the random effects grew larger. This result signified that the deep models were able to pick up on the correlation between the number of measurement occasions and the random effect as the random effects grew more varied.

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Error Variance	[-0.311, 0.937]	1.0	No
Timespan Link On	[0.047, 0.141]	1.0	No
Measurement Link On	[0.363, 0.637]	0.830	No
Random Effect Variance	[0.941, 1.015]	0.0	Yes
Random Effect Variance Exponential	[0.046, 0.246]	0.377	No
Measurement Link On x Random Effect Variance	[-0.716, -0.467]	0.0	Yes
LSTM model (Attention as Reference)	[-0.009, 0.081]	1.0	No
Relative-Time (Absolute as Reference)	[-0.023, 0.104]	1.0	No
Sequence-Time (Absolute as Reference)	[-0.048, 0.081]	1.0	No
Window-Time (Absolute as Reference)	[-0.108, 0.011]	1.0	No

Table 22: NSDG Regression Performance Results

There are some clear patterns related to model bias in the diagnostic plots shown in Figure 43.

The RVF plot in Figure 37 has a clear linear pattern of residuals below 0 on the y-axis. This pattern signifies a clear correlation in the residual violating a key assumption of our NSDG classification inference model. Furthermore, there appear to be a greater dispersion of residuals above 0 than below demonstrating heteroskedasticity. The lowess fit of the residuals is decently level at zero, but that is only a small consolation given the other patterns.

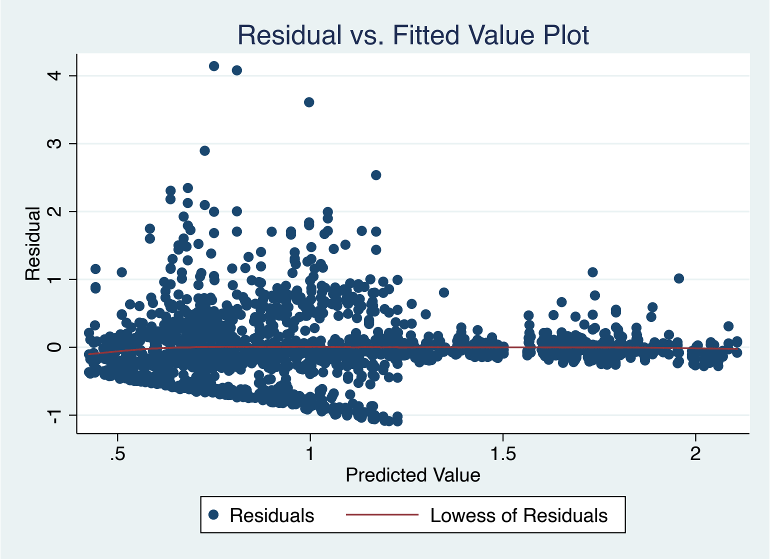


Figure 37: NSDG Regression RVF Plot

Non-Causal Asynchronous Data Group Results

In the Non-Causal Asynchronous Data Group (NADG) the properties of interest (variance of the measurement error, sampling distribution, collinearity of the random effects, variance of the random effect(s), and the event rate) were unaffected by removing values of X_1 and X_2. The median rate of missing data in X_1 was 48.6% with an IQR of [48.3%, 48.8%]. The median rate of missing data in X_2 was 47.9% with and IQR of [47.6%, 49.0%].

The introduction of missing data in the NADG produced different results than the fully synchronous NSDG. This is similar phenomena to what we saw in the causal data groups, that experienced greater differentiation between temporal representations with the introduction of missing data. Unlike previous analyses the outcome of interest for the classification performance analysis was the AUROCC. The interpretation of positive coefficients is an association with better model discrimination, and negative coefficients with worse model discrimination. In our performance analysis, we saw that window-time had significant performance advantages, while sequence time had performance advantages specific only to the LSTM architecture. These findings establish that for models with correlated, but not causal features and missing data, window-time can have significant performance advantages. We also found that the Attention model was generally the dominant architecture. However, that advantage did not hold when using sequence time. Said another way, the LSTM architecture appears to be just as good as the Attention model, if the data is ordered, but without time or date. Table 23 has the coefficient estimates and the Second Generation p-values ($H_0 = [-0.01, 0.01]$, $\alpha = 0.05$) for the classification inference model. This analysis had an R^2 of 0.4616 and a BIC of -3,977 with 15 degrees of freedom. The analysis also showed significant positive association with the measurement link, and significant negative associations with the random effect variance, and the random effect collinearity on AUROCC. The probability threshold lost significance in this analysis compared to the NSDG.

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Probability Threshold	[-0.074, 0.077]	1.0	No
Timespan Link On	[-0.009, 0.020]	1.0	No
Measurement Link On	[-0.177, -0.154]	0.0	Yes
Timespan Link On x Measurement Link On	[-0.111, -0.073]	0.0	Yes
Random Effect Variance	[0.009, 0.032]	0.031	Yes
Random Effect Collinearity	[0.002, 0.035]	0.223	No
LSTM model (Attention as Reference)	[-0.144, -0.110]	0.0	Yes
Relative-Time (Absolute as Reference)	[-0.020, 0.011]	1.0	No
Sequence-Time (Absolute as Reference)	[-0.054, -0.020]	0.0	Yes

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Window-Time (Absolute as Reference)	[0.015, 0.052]	0.0	Yes
Mean Imputation (Mean + Indicator as Reference)	[0.000, 0.018]	1.0	No
LSTM x Relative-Time	[-0.036, 0.010]	1.0	No
LSTM x Sequence-Time	[0.031, 0.080]	0.0	Yes
LSTM x Window-Time	[-0.012, 0.041]	1.0	No

Table 23: NADG Classification Performance Results

The model diagnostic plot in Figure 38 for the classification performance analysis shows the discreteness of the AUROCC with the residual distribution experiencing several discontinuities along the x-axis. The plot also shows evidence of heteroskedasticity and bias. The inference characteristics of this analysis are suspect, and these results should be treated with some skepticism.

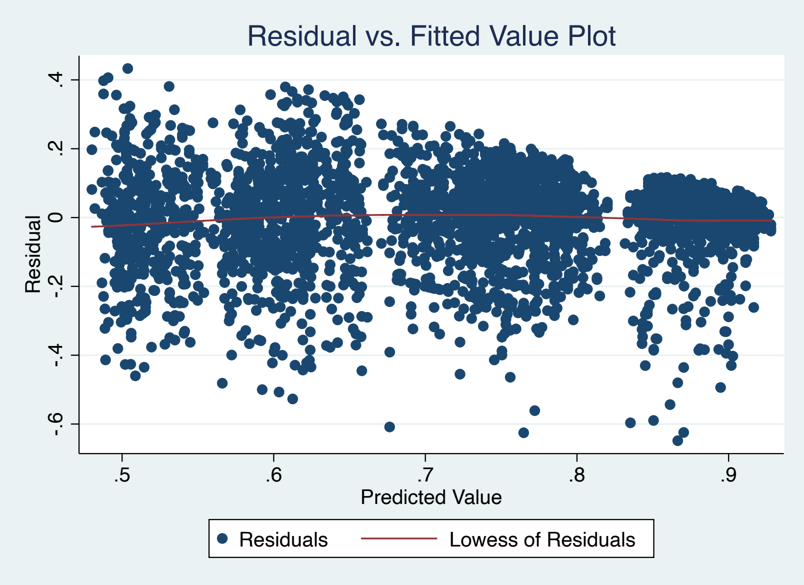


Figure 38: NADG Classification RVF Plot

The regression model struggled to find a good fit in the asynchronous data group, which is not surprising given the difficulty the models had in the synchronous data group. In the performance analysis, we added a term for the different imputation methods. The other control variables were largely the same as the NSDG inference analysis. We again used the MSE as the outcome of interest. The random effect variance and measurement error remained significant contributors to MSE compared to the NSDG results. The coefficient estimates and the Second Generation p-values ($H_0 = [-0.125, 0.125]$, $\alpha = 0.05$) are available in Table 24. This inference model had an R^2 of 0.354 and a BIC of 6,911 with 11 degrees of freedom.

Predictor	Robust 95% Confidence Interval	2nd-Gen p-value	Significant?
Error Variance	[0.795, 1.925]	0.0	Yes
Timespan Link On	[0.028, 0.113]	1.0	No
Measurement Link On	[0.514, 0.775]	0.0	Yes
Random Effect Variance	[0.999, 1.098]	0.0	Yes
Measurement Link On x Random Effect Variance	[-0.774, -0.552]	0.0	Yes
LSTM model (Attention as Reference)	[-0.006, 0.070]	1.0	No
Relative-Time (Absolute as Reference)	[-0.050, 0.050]	1.0	No
Sequence-Time (Absolute as Reference)	[-0.059, 0.045]	1.0	No
Window-Time (Absolute as Reference)	[-0.057, 0.052]	1.0	No
Mean Imputation (Mean + Indicator as Reference)	[-0.011, 0.064]	1.0	No

Table 24: NADG Regression Performance Results

The diagnostic plot for the regression inference model shown in Figure 39 is almost identical to Figure 37 from the NSDG. The diagnostic plot shows curvature, suggesting correlated residuals, as well as heteroskedasticity. These similarities imply that the results are of similarly low quality and should be treated with some reservation.

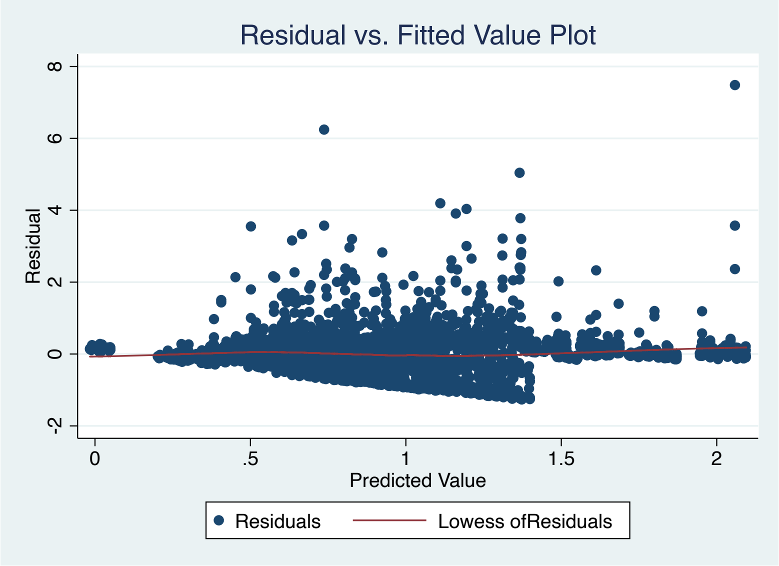


Figure 39: NADG Regression RVF Plot

Classification CSDG & CADG Discussion

In this chapter we detailed the setup and briefly described the results of eight different analyses over four different data groups. Over all of the experiments we explored the effect of temporal representation on model performance in datasets with causal and non-causal predictors as well as with complete (synchronous) and missing data (asynchronous). In the first two experiments, we focused on data with causal predictors. When working with synchronous data we found that the deep models were able to achieve equal performance to the True-GLME. This GLME was not trained on the test set and simply produced a mean fitted value for each prediction. The parity of the deep model to the True-GLME verified that our power calculations for the sample size of each data set were correct and that our deep models produced well fit estimates. The purpose of the causal data group experiments was three fold: 1) determine if there was an overall dominant temporal representation, 2) find data characteristics that had a meaningful impact on performance, and 3) to determine if any of those data characteristics signaled that one temporal representation would be superior to another.

The window-time representation generally exceeded the calibration of the true mean-model for classification problems. However, we found that the temporal representation with the best calibration (window-time) did not necessarily have the best discrimination (absolute-time). This result held in both the synchronous and asynchronous data groups.We believe that the Brier score performance advantage of the window-time representation was in part due to it making fewer predictions compared to other temporal representations; the window-time representation made one prediction per window instead of one prediction per measurement occasion. In both the asynchronous and synchronous results we found a strong positive correlation between Brier Score and AUROCC in the window-time results. The other temporal representation had strong negative correlations between the AUROCC and the Brier Score. This trend implies that as the window-time representation produced a greater variety of predicted probabilities, its discrimination increased, but the calibration decreased. However, the models fit with the other temporal representations generally saw improvements in both discrimination and calibration. The models fit with the window-time representation generally predicted more controls than other representations and predicted controls with greater confidence than other representations. Window-time is a common representation in the literature[66, 68, 73, 74, 91]. We found significant tradeoffs between calibration performance and discrimination performance using window-time compared to other representations. This finding is not discussed in other studies. The tradeoff we observed may be related to how we defined the window-time representation. Our definition did not use the average sequence, relative, or absolute time of the window as a feature, instead it used the count of measurement occasions observed within each window as a predictor. Another potential difference between our definition and others may derive from the window-time outcome. In the classification case, we used a majority vote to determine the outcome value for a specific window, where ties went to the controls. The effect sizes we observed related to window-time were large.

We noted several factors with a significant influence on the Brier score. The noise terms such as random effect variance, random effect collinearity, and the error variance all significantly increased the Brier score. All of these noise terms represent unexplainable variance from the model's perspective, so it is logical that these factors should contribute to the Brier score. The strongest contributor was the variance of the random effects (the size of the random intercept and slope coefficients). Given the generating model setup, these factors had the greatest influence on the outcome value, as their effect size in the generative model was larger than that of the measurement error. The autocorrelation of the predictors was also significantly associated with larger Brier scores in both the synchronous and asynchronous data groups. We believe that increases in feature autocorrelation increased the time dependence of the outcome process. This effect made model calibration more difficult because the time time effect was more difficult to distinguish from the feature effects. The window-time representation showed diminishing returns on improved calibration in the face of increased inter-subject variability. This effect suggests that the window-time representation may be no better than the other representations when there is a high amount of noise or unexplained variability in the data. In the asynchronous setting, features with a high amount of autocorrelation can also curb the calibration advantages of the window-time representation. Excluding the window-time interaction, the effect of these noise parameters are intuitive and are supported by previous findings[58, 66]. The non-random sampling scheme did significantly decrease the Brier score. This is likely a result of the data creation mechanism, where the features in the non-random sampling scheme were created before the sampling times were selected. The features in the non-random sampling have low autocorrelation, which is where the performance gain likely comes from.

It is also important to discuss the trivial/non-significant effects observed in the classification model performance. We did not observe significant differences between absolute, relative, and sequence time performance. This finding signifies that these representations of time are practically interchangeable. We expected that these sequence architectures would realize performance gains on equally spaced inputs. However, a performance difference between equally spaced measurement occasions and randomly spaced measurement occasions never materialized. We did not observe a significant difference in the performance between the two different architectures, though we should note that tuning the Attention models took longer and involved more hyper-parameters than the LSTM. The effect of increases in measurement error was relatively small compared to other effects. We would expect data with highly leveraged measurement error to experience significantly lower model performance. Feature collinearity also lacked a significant showing as an effect. While the collinearity of densely sampled features during testing was in the desired ranges, the observed feature collinearity in sparsely sampled data was low. The low observed collinearity likely depressed any notable effects on model performance.

The addition of missing data in the asynchronous group led to greater differentiation between models and the association of an additional interaction term. In the CADG, window-time was the representation that significantly maximized the Brier score. The addition of missing data did not significantly degrade the discrimination of any of the architecture/temporal representation combinations, but it did significantly degrade the calibration of the absolute-time models. This finding means that if one is able to observe directly causal features for a process with a manageable level of randomly distributed noise, then, in a linear setting, it does not matter if the data has non-random missing values. However, in a nonlinear process, such as one with piecewise discontinuities, missing values may prove much more consequential.

Both the CSDG and the CADG identified an interaction between window-time and the random effect variance. This interaction suggests that populations that have large inter-subject deviance from the population mean, have no clear choice of temporal representation for maximizing the Brier score. On the other hand, if one is looking to maximize calibration on a population whose outcomes are relatively clustered to the population mean, then window-time can yield a clear advantage in either the LSTM or Attention architecture. Said another way, the greater the inter-subject variability, the less that the temporal representation matters, because a mean model will struggle to fit a process that varies greatly from subject to subject. In this situation, we found that adding past outcomes as a feature can help the deep models learn the personalized subject coefficients.

In the presence of missing data, autocorrelation also has a significant impact on the advantages of window-time. The greater the autocorrelation in the feature space, the less it matters whether one chooses window-time or another representation. If the feature space is not strongly self-correlated over time (highly variable from one measurement to the next), then the smoothing action of window-time can offer significant Brier score performance benefits.

The causal data group experiments in the classification setting revealed some generalizable intuition for future model builders working with directly causal features that have a linear relationship. The common use of window-time by model developers appears justified. Window-time can minimize the Brier Loss in some circumstances. When window-time is not advantageous, there does not appear to be a significant difference between temporal representations. This finding can also offer some comfort to model developers, as implementation consideration can inform temporal representation without fear of performance loss.

Regression CSDG & CADG Discussion

For the regression models, the true-GLME baseline had greater separation in terms of performance compared to the deep models. However, the performance difference between the GLME baseline and the deep models did not meet our threshold for significance of 0.125 MSE in all cases. This result implied that our power calculations for sample size were accurate enough to detect the desired differences in effects. We set the significance threshold prior to experimentation, and in retrospect our chosen threshold may be too large, as it was 6%-8% difference relative to the largest average MSEs shown in Figure 29 and Figure 35. This high bar may have lead us to discount effects that others might consider significant.

When it came to temporal representation, we found approximately the same number of significant effects in the synchronous data group as the asynchronous one. We found strong evidence that the sequence-time and relative-time representations are inferior to the absolute-time reference in the complete and missing data settings. The preference for absolute time is likely a result of the data generating process. Absolute-time was the representation used as a model predictor. It is logical that absolute-time yielded performance gains. What is surprising, is that window-time demonstrated a performance advantage in the missing data group and had a non-zero, but trivial performance advantage in the complete data group. Window-time avoided large MSEs by smoothing out large fluctuations in the outcome. The explained variance scores suggested that the predictions from window-time fit models are much more variable relative to the variance of the outcome than other temporal representations. The efficacy of a window-like abstraction especially in the presence of missing data is consistent with previous research[54].

We found only one significant data characteristic on MSE in our inference analysis on the synchronous data. The random effect variance (our main noise parameter) was significant. The personalized unobserved offset to the regression intercept had a large effect on the MSE. As individual subjects deviated more from the group mean, naturally, the MSE increased, because the models generally trained toward a population mean fit. As mentioned earlier, the data generating GLME assumed the distribution of personalized effects to be centered at 0. The baseline GLME we fit for comparison was able to find the true mean and minimized the MSE effectively. Yet, the MSE of the GLME was strongly positively correlated with the variance of the random effect. The random effect variance was also significantly associated with more MSE in the asynchronous data group.

However, we again found that autocorrelation can significantly degrade the performance benefits of using window-time. Window-time had an interesting dynamic with autocorrelation compared to the other representations in the asynchronous data group. As the autocorrelation increased the other representations reduced their MSE, while window-time increased. This finding implies that smoothing may lose many of its benefits when features already behave in a smoothed manner with respect to time (highly autocorrelated). The relative-time and sequence-time representations saw significant performance losses in the LSTM architecture.

The regression analysis found similar non-effects in the data generating parameters as the classification analysis. Feature collinearity and measurement error variance had trivial effects on MSE. We detected some significant interactions between the temporal representations and the random sampling scheme. These effects suggested that random sampling was less preferred than equally-spaced sampling for relative and window-time representations. The reference sampling scheme was the equally-spaced scheme, which we expected, based on previous literature, to perform the best[91]. We believe that the preference of relative and sequence time for equally spaced samples has to do with the models being able to easily learn how much absolute time has passed. The importance of absolute time to the outcome makes it a key relationship to learn. Autocorrelation was positively associated with the Brier score, but negatively associated with the MSE. The change in effect direction between the classification and regression data sets cannot readily be explained. The feature coefficients were -0.5 in the classification and the regression data sets, but the nonlinear probit transformation may have amplified the effect size in the classification data sets to reach significance. Adding an indicator variable per feature to signify whether that features was observed during that measurement occasion also had a trivial effect on model performance. Previous studies have reported that it took a large number of samples to associate the indicator variables with the their numeric/categorical counterparts[68]. We may not have had a large enough sample of subjects and/or measurement occasions for the deep networks to find this association. Using window-time or absolute-time with either the LSTM or Attention modeling architectures were found have indistinguishable results. This finding is supported by previous literature[45]. However, the LSTM did have diminished performance when used with the relative and sequence time representations. The LSTM hidden memory, may have struggled to learn how these representations associated with absolute time. When considering the two architectures, it is worth noting that the hyper-parameter tuning and training time of the Attention models generally exceeded that of the LSTM models by 20%-25%.

In the causal regression based data group, we again found significant performance advantages of window-time. This performance advantage manifested itself in the presence of missing data. We hypothesize that the absence of values during each measurement occasion made the features more variable and less self-correlated over time. Both of those factors were shown to significantly contribute to the advantage of window-time. Despite a different modeling problem with its own range of parameters, this experiment produced findings that are consistent with the classification setting.

Classification NSDG & NADG Discussion

The non-causal data group was meant to create data that gradually approached the complexity of the MIMIC III modeling problems. Different data generations mechanisms were added in a stepwise fashion so that we could isolate the performance effects of those mechanisms, therefore providing a better estimate of the performance effects attributable to the temporal representation. We varied fewer data generating parameters than in the causal data groups, as we needed to vary only the significant factors found in the CSDG & CADG analyses. Specifically, we allowed the random effects variance, the random effects collinearity, and the probability threshold to vary for the classification data. The random effects variance and random effects collinearity are noise terms. An increase in either increases the irreducible error of a predictive model. The probability threshold controls the event rate, also known as the ratio of cases to controls.

The primary differentiation of these data groups with the causal data group was that the outcome was not a function of the features, but the outcomes values were correlated with the latent subject specific intercept term. Many predictive models in healthcare use features that are not causal themselves, but are correlated to causal factors. This artificial structure that we specified for the non-causal data group parallels the situation of many clinical predictive models and may allow our findings to generalize beyond the modeling problems chosen. We induced variability in the data (event rate and signal to noise) to allow our findings to generalize beyond the MIMIC III dataset.

In the synchronous data group we found that the temporal representation is not of particular import. This finding is consistent to that of the CSDG. When given complete information (all data for all features at all time steps), the model performance is not affected by the chosen temporal representation. Such situations are exceedingly rare in healthcare, but may exist in other domains, specifically those with automated data collection.

What is more interesting, is that the temporal representation does matter when data is incomplete and imputation is used. As mentioned in the methods, we used a non-random mechanism to create missing values to better simulate real data collection processes. The imputation methods we selected (mean imputation and mean+indicator imputation) are those that appear frequently in the literature. This scenario is common in healthcare, as data is rarely complete. Simple imputation methods are commonly used in the development of clinical machine learning models, making our findings of use to other model builders in the domain. In both the CADG and the NADG, window-time demonstrated performance advantages. One key caveat to this finding is that we allowed the window-size to be a tunable parameter, meaning that the window-size is a function of the training data. Window-time would likely not have performance advantages if the window-size was determined a priori. We believe that window-time demonstrated performance advantages, because it makes fewer predictions than the other temporal representation and the feature values for an individual window are an average of observed and imputed values. For each subject we generate the full matrix of measurement occasions with imputed values for each feature before we divide those measurement occasions into windows and average the measurement occasions within each window. This approach results in a shrinkage-like effect, as windows with more observed measurements can differ more from the population mean than windows with fewer observed measurements. We believe that the window size is dependent on the rate of change in the feature space. In the CADG experiment we found that high autocorrelation mitigated the advantages of window-time, suggesting that features that change slowly require larger windows than those that change quickly. This hypothesis is based on some of the theories of signal processing, where the variability of the signal determines the optimal sampling rate. A variable process requires more frequent sampling than data that is more consistent.

Another finding from these experiments was the suggestion that the Attention model may have performance advantages over the LSTM in asynchronous data. This finding is not consistent with the results of the NSDG or CADG. As with window-time, the architecture may not matter much when data is complete, but appears to make significant performance contributions when data is incomplete and features are not causal. The advantage of the multi-head Attention model over the LSTM may have to do with the number of parameters. Multi-head Attention trains several matrices that are specific to the current feature value, while the LSTM shares its hidden memory cells across all feature values.

Regression NSDG & NADG Discussion

The inference models on the non-causal regression data struggled to find a decent fit on the data. Many of the findings of the non-causal regression inference models are similar to those of the causal data groups. The non-causal regression datasets had a small variable amount of measurement error. We did not observe any tradeoff between MSE and explained variance. The average explained variance scores for some of the models went below zero, suggesting that the model prediction had greater variance than the outcome.

Only the random effect variance remained a significant effect on MSE in both the synchronous and asynchronous data groups. Random effect variance was a significant effect in all eight data groups. We also discovered a significant interaction between the measurement link and the random effect variance in both data groups. The measurement link reduced MSE, because it associated subjects with larger random intercepts with more measurement occasions. The deep models were able to use this structure to reduce MSE, because the inter subject variance decreased as one moved from one measurement occasion to the next. Said another way, the population mean is not particularly accurate on the first measurement occasion; however, with the measurement link active, the outcome variance between subjects on the 30th+ measurement occasion is much reduced. The subjects that have 30 or more measurement occasions also have much more similar random intercepts and therefore more similar outcomes.

The regression heat maps appeared to show large differences in performance between the Attention and LSTM modes, but that difference never formally materialized in the inference models. The error cutoff of 0.125 for the MSE (roughly a %10 difference) may have been too large to capture the differences shown on the heat maps. As in the classification case, no temporal representation stood out, and adding missing indicators did little to improve model performance. These findings are intuitive given the non-causal relationship of the predictors to the outcome, implying that the representation of a predictors in a high noise setting is not particularly important.

Strengths and Limitations

The strength of this work largely rests on its use of precisely created data that allowed for a careful interrogation of different temporal representations. We searched a large combinatoric space with our data creation machinery and came away with several key findings that are consistent with less formal experiments in the literature. This work was methodical in its choice of experimental parameters. We made great effort to meaningfully capture as much of the realistic elements of clinical data (non-random missing values, non-random measurement occasions, and mechanisms to match the measurement occasion distribution), while simplifying the data generation process enough to have a true gold-standard. We carefully determined sample sizes and the number of measurement occasions through pilot studies and formal calculation. We attempted to anchor experimental parameters on real world data through a detailed exploratory analysis of MIMIC III. We set thresholds for meaningful effect sizes prior to analysis and determined the significance of findings on the data scale, and not on the standard deviation scale. The deep architectures we built incorporated as many design decisions as possible as hyper-parameters. We rigorously tuned and evaluated said hyper-parameters with a uniformly random search using a nested cross-validation strategy. We compared the performance of our deep models to that of a true baseline model to validate that the deep models trained as expected. We used stepwise methods in the non-causal data groups to isolate the effects of the different data generations mechanisms we used. We performed inference on the resulting model performance measures with robust statical methods, and we evaluated the quality of those inferences.The robust (Huber-White corrected standard errors) inference method helped to prevent the underestimation of model variation and the over estimation of effect sizes. The regression framework allowed us to isolate the effects of different data parameters from one another, helping to generalize the findings to other datasets. We believe that the internal consistency of these experiments will allow our work to generalize to other datasets with different generating processes. The primary finding of window-time being advantageous in asynchronous data has been observed in other studies and there is a theoretical basis for its advantages in the signal processing domain.

Despite these methodological strengths, our work was not without weaknesses. While the GLME model used to generate the data allowed for fine tuned control and a gold-standard baseline, it came at the cost of some major assumptions as well as unrealistic artifacts. The most difficult assumptions to defend are the zero-centered normal distribution of random effects. We do not have an empirical basis for this assumption, and generally saw log-normal distributions in the characteristics of the MIMIC data. Another major assumption of our generating model was that of a time-specific coefficient. Modeling time explicitly did lead to the desired correlation structure in the residuals, but it also baked in absolute-time with a relative anchor as the default temporal representation. Absolute-time likely had a structural advantage compared to other representations. There may be some defense of such an assumption, as our birth date acts as a relative anchor point for many health risks and comorbidities. However, it may have been of greater value to evaluate a different representation instead of absolute-time to ensure equal footing. The non-random sampling scheme, measurement link, and timespan links are simplifications and/or workarounds to improve the realism of the simulated data in comparison to MIMC III. It may have been worth the while to use MIMIC III data to develop a more realistic sampler. The long-gen package can make use of custom user-designed sampling functions, opening the door for more practically grounded future work. The GLME was also an awkward fit for the problems modeled (remaining ICU LOS and 24hr ICU mortality). A time to event model such as the cox proportional hazard model may have better fit these problems, while being able to handle unbalanced longitudinal data. Another limitation was that the predictor collinearity dimension was not well explored. During development the collinearity was within the desired parameter ranges. However, the limited number of measurement occasions made the theoretical collinearity difficult to translate into observed collinearity. Finally, there were other longitudinal properties such as stationarity that were not explored at all. In future work we would hope to address these limitations, as they are important properties of clinical data. We also hope to better explore window-time specifically and to better understand the conditions in asynchronous data that make it advantageous. Another area of future study is that of interpreting the window size. Currently we hypothesize that the window size is correlated to the average time until a significant change in the feature space, but this question is worthy of further study.

In this work we demonstrated that window-time can be a dominant representation for modeling problems (regression and classification) with asynchronous features. Window-time was best used where the predictors were not strongly time dependent (low to moderate autocorrelation). As the autocorrelation of the feature space grew, other temporal representations became suitable alternatives. In situations datasets where all features are observed, the temporal representation became less important. Our work explored a few commonly used representations and methodically evaluated their effects on model performance, accounting for many commonly seen characteristics in clinical data.

Chapter IV

COMPARING TEMPORAL REPRESENTATIONS IN HEALTHCARE DATA

Study Design

In this chapter we sought to test if the intuition afforded to us by our experiments on synthetic data would hold in real clinical data. As previously mentioned, we used the MIMIC III ICU data set as a source of inspiration for many of the different parameters of the synthetic data[122]. We made use of MIMIC III to evaluate the predictive performance of different temporal representations using cohort definitions and modeling tasks from a benchmarking study on this same data[73]. The high level architecture designs of Figure 24 and Figure 25 for the deep models were based off of a benchmarking study[73]. We replicated two of the modeling problems of interest (24 hour ICU mortality and remaining ICU length of stay prediction) as well as the cohort from Harutyunyan et. al.'s study[73]. In the original benchmark, Harutyunyan et. al. only reported results for a relative-time representation used with a mean+indicator imputation method in an LSTM model[73]. Other studies have examined other topics using MIMIC III such as imputation methods, architecture components, and training strategies[68, 61, 73, 74, 95, 158]. In this experiment, we evaluated four temporal representations (absolute, relative, sequence, and window) with both mean imputation and mean+indicator imputation in both LSTM and Attention architectures.

Materials

We used a 2015 MacBook Pro with four 2.9 GHz Intel processors and eight GB of RAM to pre-process and clean the MIMIC III data into a cohort ready for model fitting. We used a 2014 Alienware

X51 desktop running Windows 10 with four 1.6 GHz processors, 32 GB of RAM, and a 6GB NVIDIA GTX 1060 graphics processor (GPU) to develop the deep models that would train and run on MIMIC III data. Our primary compute environment for the experiment was ACCRE. In the ACCRE environment, we used 31GB of storage and 34 CPUs connected to 17 GPUs in a configuration of two CPUs per GPU. The 34 GPU-connected CPUs were graciously provided through a miniature grant from the Vanderbilt University Data Science Institute[143].

We used the Python programming language version 3.6.3 for data preprocessing, predictive model development, and data aggregation/visualization. Jupyter notebooks[127], the Sublime Text editor[128] (version 2), and vim[141] were used as the primary Python editors. Our Python package dependencies are listed in Table 25.

Package	Version
IPython[125]	7.12.0
Joblib[145]	0.15.1
Jupyter[126]	1.0.0
Long-Gen[124]	0.2.3
Matplotlib[129]	3.1.3
Numpy[130]	1.18.1
Pandas[131]	1.0.1
Scipy[132]	1.4.1
Scitkit-learn[133]	0.22.1
Torch[146]	1.6.0

Table 25: MIMIC III Modeling Software Dependencies

Methods

Our study cohort was made of adult ICU patients admitted to Beth Israel Deaconess Hospital in Boston, MA between June 2001 and October 2012[122]. We used the same inclusion and exclusion criteria defined by Harutyunyan et. al.[73]. We excluded ICU admissions that resulted in a transfer between critical care wards and we also excluded ICU admissions where the patient returned to the ICU after being discharged to a medical ward. ICU admissions without admit and discharge date-times were also excluded. The benchmark study applied four criteria sequentially to build their cohort: 1) no ICU admissions with critical care transfers, 2) no admissions with multiple ICU stays for the same inpatient admission, 3) age ≥ 18, and 4) must have ICU admit and discharge date-times. The sequential application of these criteria resulted in a logical error, because ICU admissions with multiple stays were allowed into the cohort if all but one of those ICU stays involved a transfer between critical care wards. We intentionally replicated their logical error allowing these 56 corner-case ICU admissions into the cohort of 33,798 patients with 42,276 ICU admissions. The study used 16 different clinical variables to make predictions for a variety of tasks. We reproduce their table of variables in Table 26, which includes the MIMIC III source file, the population mean value used for imputation, and how the clinical feature was represented. Harutyunyan et. al. chose the clinical variables that were most complete per each measurement occasion.

Variable	MIMIC-III table	Impute Value	Modeled As
Blood pH	chartevents, labevents	7.4	continuous
Capillary Refill Rate	chartevents	0-Normal <3 secs	categorical
Diastolic Blood Pressure	chartevents	59mmHg	continuous
Fraction Inspired Oxygen	chartevents	0.21	continuous
Glascow Coma Scale Eye Opening	chartevents	4-spontaneously	categorical
Glascow Coma Scale Motor Response	chartevents	6-obeys commands	categorical

Variable	MIMIC-III table	Impute Value	Modeled As
Glascow Coma Scale Verbal Response	chartevents	5-oriented	categorical
Glucose	chartevents, labevents	128mg/dL	continuous
Heart Rate	chartevents	86beats/min	continuous
Height	chartevents	170cm	continuous
Mean Blood Pressure	chartevents	77mmHg	continuous
Oxygen Saturation	chartevents, labevents	98%	continuous
Respiratory Rate	chartevents	19breaths/min	continuous
Systolic Blood Pressure	chartevents	118mmHg	continuous
Temperature	chartevents	36.6C	continuous
Weight	chartevents	81kg	continuous

Table 26: Clinical Variables Used as Predictors from MIMIC III

For this chapter we decided to focus on only two prediction tasks. The first task we chose was a classification task, where we would attempt predict whether a patient was likely to deteriorate past the point of the clinical staff being able to resuscitate them (ICU mortality). We chose a prediction window of the next 24 hours from the time the forecast was issued. This choice was primarily to parallel the benchmarking study, but the intuition is that 24 hours may be enough time to intervene on patients with treatable causes of death. Figure 40 provides a model of how this event is defined over a set of

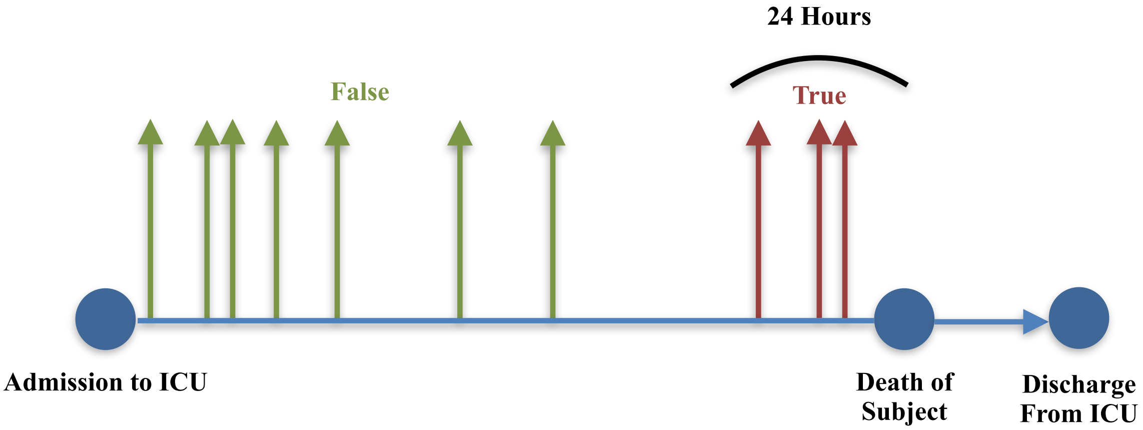


Figure 40: 24 Hour ICU Mortality Outcome Definition

measurement occasions for one subject. Figure 40 is a recreation of a figure from Harutyunyan et. al.[73].

In the second task we attempted to predict the remaining length of stay for a patient in the ICU. We took this as a regression problem that attempts to predict the amount of time until discharge relative to the time when the forecast was made. This problem could also be framed as a time-to-event model, but we wanted to follow the definition laid out in Harutyunyan et. al.[73]. Figure 41 provides some visual intuition on our definition of the remaining ICU length of stay outcome, as we make predictions at various points during the ICU admission as to the remaining time until discharge.

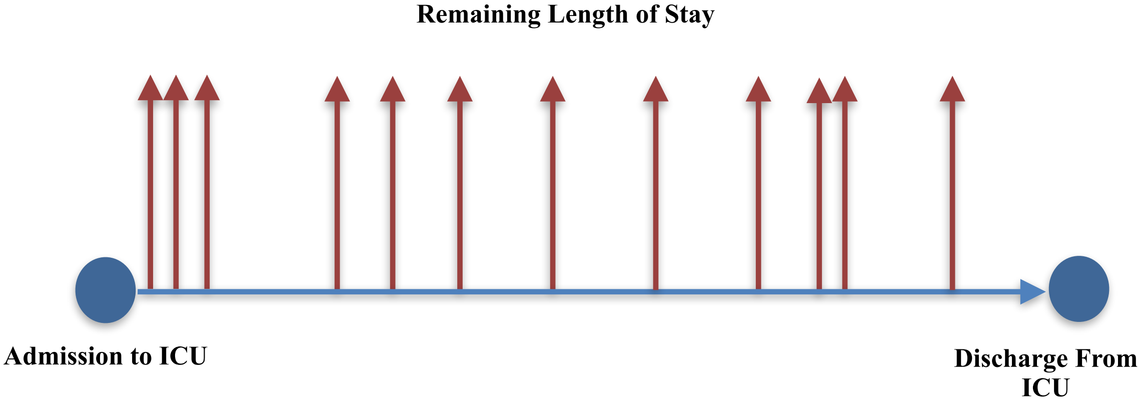


Figure 41: Remaining ICU LOS Outcome Definition

Having selected our modeling problems, the next task we undertook was to preprocess the MIMIC III data to tie clinical measures to ICU admissions present within our cohort. As in the benchmark study, we excluded clinical measurements that could not be associated with either a particular inpatient admission or an ICU admission. As part of pre-processing we standardized the units of the clinical measurements using standard conversions (e.g., imperial to metric). Our preprocessing performed limited inference of the units of measure based on the value if the units were not present, as was done in the benchmark study[73]. We also screened the clinical predictors for implausible values using the value ranges from the benchmark study. We did deviate from Harutyunyan et. al.'s value ranges for height and weight, because zero kilograms and zero centimeters are not valid cutoff thresholds for low-end erroneous values for adult patients. For height we chose thresholds that were sex specific based on 99th percentiles for the United States. The weight thresholds were based on the patient's height by using body mass index (BMI) thresholds. While all other clinical measurements were treated as dynamic values that changed over time, we treated height and weight as static values by averaging all valid measurements for that patient. We performed this static smoothing because the measurement error for height and weight in the ICU setting is significantly greater than the actual variance of those features during an ICU stay[159-161]. We used multiple regex patterns to extract a blood glucose value, which led to a large difference in event capture. In total, these choices led us to associate 34,668,291 clinical measurements, compared to only 31,868,114 measurements used in the benchmark study (a difference of 2,800,177 measurements [9%]). The value ranges we used to screen erroneous values can be seen in Table 27. It is important to note that for the 24hr mortality and length of stay prediction problems, the benchmark study excluded admissions less than 4 hours. We did not make that exclusion, because such exclusions add selection biases that may not be obvious to model users.

Variable	Low Threshold	High Threshold
Blood pH	6.3	8.4
Capillary Refill Rate	0-Normal <3 secs	1-Abnormal >3 secs
Diastolic Blood Pressure	0mmHg	375mmHg
Fraction Inspired Oxygen	0.2	1
Glascow Coma Scale Eye Opening	1	4
Glascow Coma Scale Motor Response	1	6
Glascow Coma Scale Verbal Response	1	5
Glucose	0.1mg/dL	2200mg/dL
Heart Rate	0beats/min	350bpm
Height	female: 140cm male: 155cm	female: 190cm male: 205cm
Mean Blood Pressure	0mmHg	375mmHg

Variable	Low Threshold	High Threshold
Oxygen Saturation	0%	100%
Respiratory Rate	0breaths/min	300breaths/min
Systolic Blood Pressure	0mmHg	375mmHg
Temperature	26C	45C
Weight	15BMI	45BMI

Table 27: Thresholds Used for Cleaning Clinical Predictors

Another key difference between our study and the benchmark was how we defined a measurement occasion. The benchmark study used a window-time representation with a window length of one hour. The benchmark study did not include a count of measurements within window count, like we did in our window-time representation. We tuned our window sizes as a hyper-parameter that varied from 14 minutes to 5 days and 21.5 hours. For the absolute, relative, and sequence-time representation, we defined a measurement occasion as a one minute interval, because that was the lowest resolution of the timestamps for clinical events. We made a prediction each time new data was recorded for these three representations. This is an analogous setup to that of the asynchronous data from Chapter III. The outcomes of 24hr ICU mortality and remaining LOS were defined for all measurement occasions. This prediction setup may not be possible for outcomes that require repeated measurement. The MIMIC data had a long left tail in the distribution of measurement occasions that could lead to unwieldy matrix calculations. Therefore, we right-truncated sequences longer than 1,008 measurement occasions. This means we took the last 1,008 measurement occasions of any admission. We chose 1,008 because that was the 99.9th percentile of the number of measurement occasions. This choice means we discarded measurements for approximately 42 admissions. We considered implementing truncation within the learning algorithm (back propagation through time) instead of in the data, but decided against that approach due to the performance considerations of redefining the propagation window at each time step in the training process[162]. We evaluated both mean imputation and mean+indicator imputation, as we did in Chapter III. Harutyunyan et. al. used mean+indicator imputation[73].

After preprocessing the data, we carefully studied the data properties to forecast what we thought would be the best temporal representation based on our findings from Chapter III. We measured the predictor autocorrelation, the inter/intra-subject variability in the feature and outcome spaces, as well as the sampling distribution. To compare and contrast intra-subject variability with inter-subject-variability we compared the distribution of statistics grouped at the subject level with those calculated on ungrouped data. To gain insight into the sampling distribution we plotted the histograms of the relative-time between measurements for a variety of predictors. For the histograms we adjusted the number of bins to standardize an interval of 10 minutes per bin for all the histograms. We hoped to informally examine if the methodology in this chapter was able to inform a real world problem using greatly simplified data. We then compared our prediction to reality by evaluating the four temporal representations on both architectures, using both imputation methods, for both modeling problems leading to a total of 32 combinations.

The architecture setup for the predictive models was essentially unchanged from Chapter III. The primary difference was that the MIMIC models used more features (17 with mean imputation and 33 with mean+indicator imputation). There were some slight downward adjustments to our potential hyper-parameter values, as we ran into memory overflows on the ACCRE GPU's for the Attention models. We list the possible hyper-parameters in Table 28, and highlight differences from Chapter III (see Table 14) in light blue.

Hyper-parameter	Related Model	Value Range
Batch size	Feedforward, LSTM	[8, 16, 24, …, 156]
	Attention	[8, 16, 24, …, 56]
Depth	Attention	[1, 2, 3, …, 8] x number_of_heads

Hyper-parameter	Related Model	Value Range
Drop out rate	Attention, Feedforward, LSTM	(0, 0.1)
Hidden dimension	Attention, Feedforward, LSTM	[8, 16, 24, …, 56]
Learning rate	Attention, Feedforward, LSTM	(0.0001, 0.001)
Number of layers	Feedforward	[1, 2]
Number of heads	Attention	[1, 2]
Optimizer	Attention, Feedforward, LSTM	[ADAM, Wighted ADAM, Rprop, Centered RMS]
Weight decay	Attention, Feedforward, LSTM	(0, 0.25)

Table 28: MIMIC Model Hyper-Parameter Tuning Values

The tuning and evaluation strategy for the predictive models was also very similar to Chapter III. We used three-fold cross-validation nested within three-fold cross-validation to tune, train, and evaluate the different models (please refer to Figure 27 for a visual). We used the exact meta-parameters of Chapter III as well (25 random hyper-parameter draws, 5 training epochs during tuning, and 30 training epochs after selecting the best of the explored hyper-parameters). We recorded the AUROCC and Brier score for the 24hr mortality models. For the length of stay models we recorded the MSE, the mean absolute deviation (MAD), and the Explained Variance score. We compared the different model performance for each representation and architecture combination with Second-Generation p-values. We defined a significant classification difference as a difference of 0.01 AUROCC or more. We defined a significant regression difference as a 12 hour or more difference as measured by the MAD. We chose to scale the MAD to hours to allow for an easier comparison with the benchmark study results.

Results

We compared the median of all values regardless of subject compared to the median of the subject-specific median measurement. The ungrouped medians and IQRs excluded extreme values, while the grouped median of medians excluded extreme subjects. If intra-subject variability was equivalent to inter-subject variability, we would expect these two quantities to be roughly equal. If the ungrouped IQR was wider than the grouped IQR, that would signify that intra-subject variation was large relative to inter-subject variability. From the measurement occasion perspective, we had an event rate of 2.6%, which is fairly case imbalanced. From the subject perspective, 11.3% of the patients in our cohort died in the ICU. Our grouped event rate was greater than the ungrouped rate, suggesting patients that died tended to die shortly after their admission to the ICU, because the proportion of measurement occasions labeled as cases is significantly smaller than the expected proportion (percent of subjects with mortality event x median ratio of mortality possible time to total LOS = 0.113 x (24/30.25) = 0.090). This estimate is conservative, as we might expect a greater rate of measurement occasions in the hours before death instead of the uniform rate we assumed. The remaining length of stay saw much greater intra-subject variability than inter-subject variability. This result was to be expected as the subject with the longest LOS has a larger outcome space (424,103min - 0min) than the average subject (1,815min - 0min). Most of the continuous features saw greater intra-subject variability than inter subject variability. Overall, the data had a moderate amount of variability in the feature space. Though some features, such as blood glucose, saw much more variance than others, such as oxygen saturation. The categorical features tended to be fairly low variance. We observed that the outcomes tended to be much more time dependent than the features. The predictors had an average autocorrelation of 0.406, while the outcomes had an average autocorrelation of 0.987. Collinearity was difficult to measure due to the asynchronicity of observations. The statistics of the data characteristics for the features and the outcomes used for prediction is shown in Table 29.

Variable	Variable Type	Ungrouped Median (IQR)	Median of Subject Grouped Median (IQR)	Autocorrelation
Mortality	Outcome	2.6%	4.5%	0.975
Length of Stay	Outcome	73.2hr [28.3, 200]	30.25hr [18.6, 54.6]	0.999
Glucose	Predictor	127mg/dL [105, 158]	124 mg/dL [110, 143]	0.401
Systolic BP	Predictor	119mmHg [104, 137]	118.5mmHg [108.5, 131]	0.423
Diastolic BP	Predictor	59mmHg [50, 67]	59mmHg [53, 69]	0.420
Mean BP	Predictor	77mmHg [68, 89]	77mmHg [70.8, 84.5]	0.337
Respiratory Rate	Predictor	19breaths/min [15, 24]	18breaths/min [16, 21]	0.221
Temperature	Predictor	37C [36.5, 37.5]	36.9C [36.5, 37.2]	0.625
Blood pH	Predictor	7.39 [7.34, 7.44]	7.38 [7.34, 7.42]	0.302
O_2 Saturation	Predictor	98% [96, 99]	97% [96, 99]	0.013
Heart Rate	Predictor	85 [74, 98]	83 [74, 93]	0.514
% Inspired O_2	Predictor	40% [40%, 50%]	50% [40%, 50%]	0.310
Capillary Refill	Predictor	0-Normal [0, 0]	0-Normal [0, 0]	0.612
GCS Verbal	Predictor	4-Disoriented [1, 5]	5-Oriented [2, 5]	0.594
GCS Motor	Predictor	6-Obeys commands [5, 6]	6-Obeys commands [6, 6]	0.470
GCS Eye	Predictor	4-Spontaneous [3, 4]	4-Spontaneous [4, 4]	0.446
Height	Predictor	NA	170cm [163, 178]	NA
Weight	Predictor	NA	78.9kg [66.5, 93.0]	NA

Table 29: MIMIC Cohort Characteristics

To analyze the sampling scheme we looked at the time between measurements for different predictors. We plotted the distribution of relative sample times to see if there was a wide distribution of relative-times. A wide distribution would imply a more random sampling distribution, while a more concentrated distribution of times would signify a more equally spaced sampling distribution. Figure 42 shows that most variables are regularly sampled with distributions that have sharp peaks near zero and a rapid fall off. The blood glucose and blood pH tests have the widest distributions. Their distributions are still fairly peaked, but the relative sampling appears a bit more random than other predictors.

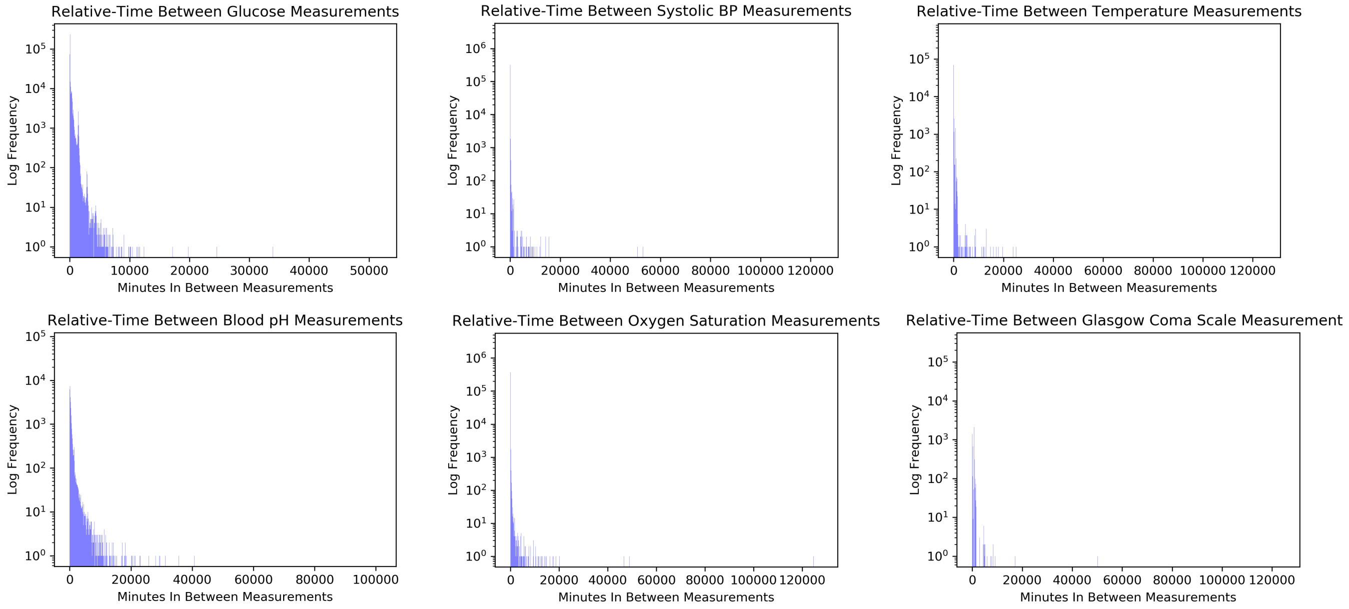


Figure 42: Histograms of Relative Sample Times

Having studied the temporal characteristics of the MIMIC III data, we then sought to make a prediction of what temporal representation (if any) might lead to the best performance for each modeling problem. The asynchronous non-causal data group most closely approximates MIMC III's data characteristics. MIMIC III is an asynchronous data set, and clinical features such as current temperature, heart rate, and blood glucose are likely correlated with future death in the ICU or ICU discharge, but might not necessarily be on the causal pathway of future ICU death/discharge. The results from the NADG Chapter III experiment suggest that window-time paired with an Attention architecture will lead to the best AUROCC for the 24hr mortality model, and that neither missing data representation will out perform the other. In the case of the length of stay prediction problem we predicted that no temporal representation would out perform the others, the architecture choice would not matter, and neither would the missing data representation. Agreement of these predictions with the observed reality would suggest that our methods have discovered meaningful relationships in the synthetic data.

After training and evaluating the models on the MIMIC III data, we found that that window-time had a large positive impact on both the classification model results as well as the regression model results. The Attention architecture also appeared to make a positive contributions to both classification and regression model performance. However, the high variance estimates of the LSTM model performance prevented this difference from being conclusive. The mean+indicator imputation method also appeared to have a strong positive impact on performance. However, many of those differences did not rise to the level of statistical significance.

In the 24hr mortality model results, window-time paired with the Attention architecture clearly produced the best performing models. Figure 43 shows the average performance of the different temporal representations, architectures, and imputation methods from the perspective of the AUROCC, the Brier Score, and the area under the precision recall curve (AUPRC). Below the average performance is the asymptotic confidence interval at the 95% level. No matter the metric, the window-time representation, when used with the Attention architecture, performed best. Absolute time was significantly inferior compared to other representations. Relative and sequence time were equivalent to each other across the different models and imputation-method combinations.

The performance of our window-time mortality model was superior to Harutyunyan et. al's model on the 24-hr ICU mortality[73]. The best 24hr mortality from Harutyunyan et. al. had an AUROCC of [0.908, 0.913] and an AUPRC of [0.334, 0.354][73]. Given that Harutyunyan et. al. used a window-time representation, it was encouraging to see our window-time models reasonably exceed the benchmark. This feat provided evidence that we implemented our models accurately. However, it was not clear how much of the performance difference between our best model and theirs was due to cohort definition, data pre-processing methods, architecture differences, or temporal representation.

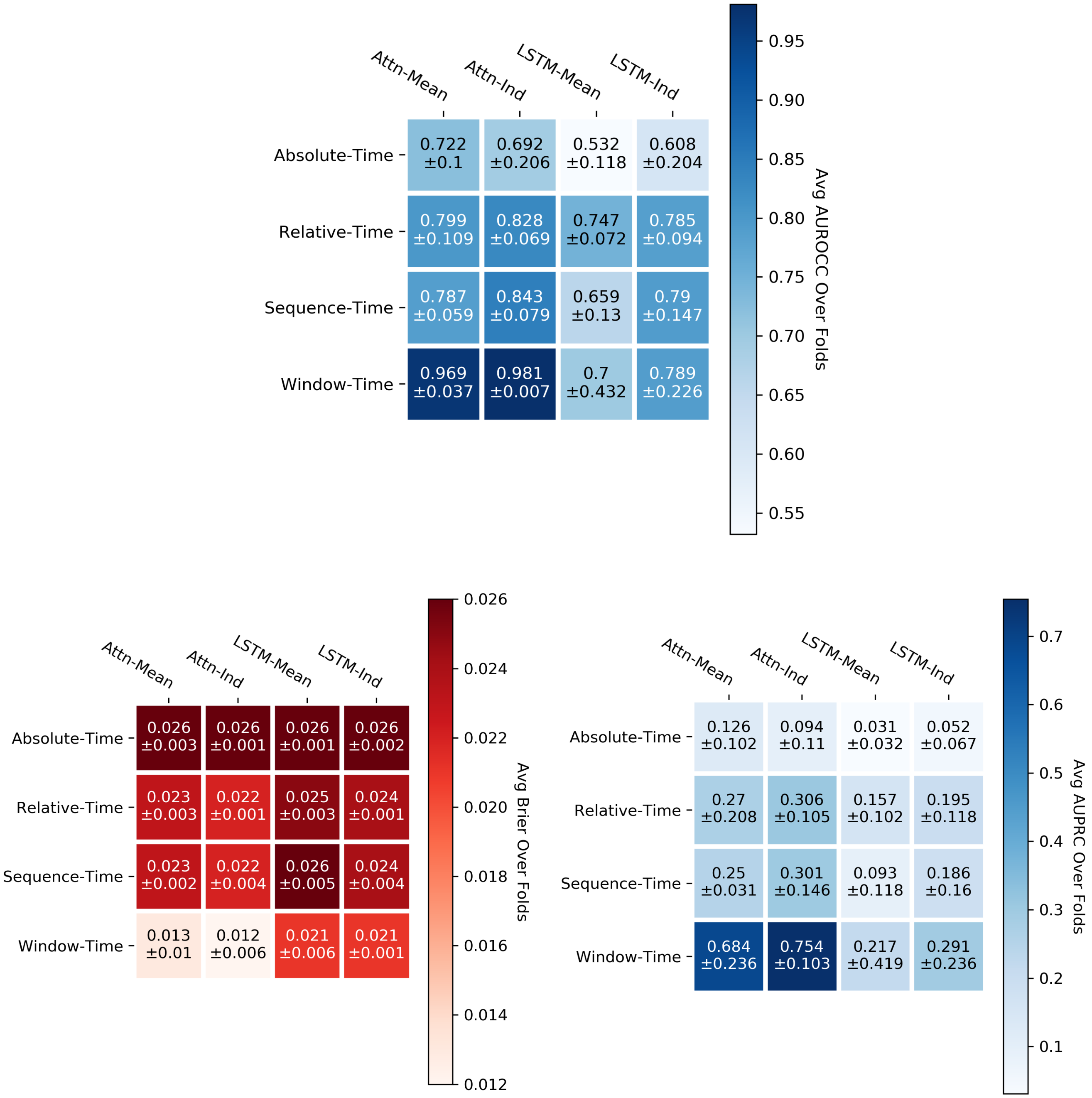


Figure 43: 24hr Mortality Prediction Performance

We believe that the implementation of window size as a hyper-parameter was one of the primary drivers of our state of the art performance result on 24hr ICU mortality prediction[66, 68, 73, 74]. Many of the other studies on the 24hr ICU mortality prediction task used window-time as their default temporal-representation, but none of the studies implemented and tuned the window-size as a hyper-parameter.

We also believe that our formal approach of defining measurement occasions within a window also improved our model's performance.

The length of stay prediction task was relatively more difficult than the ICU mortality task and the average model performance reflects that difficulty. The relative performance was poor and most models struggled to produce predictions within 80 hours of the true value. The Attention model using window-time, again proved superior, as assessed by the explained variance score and the MAD. That combination significantly dominated all of the other combinations. The MAD and explained variance scores of all model and temporal representation combinations is visualized in Figure 45. There were trivial performance differences between mean imputation and mean+indicator imputation in the Attention model + window-time categories.

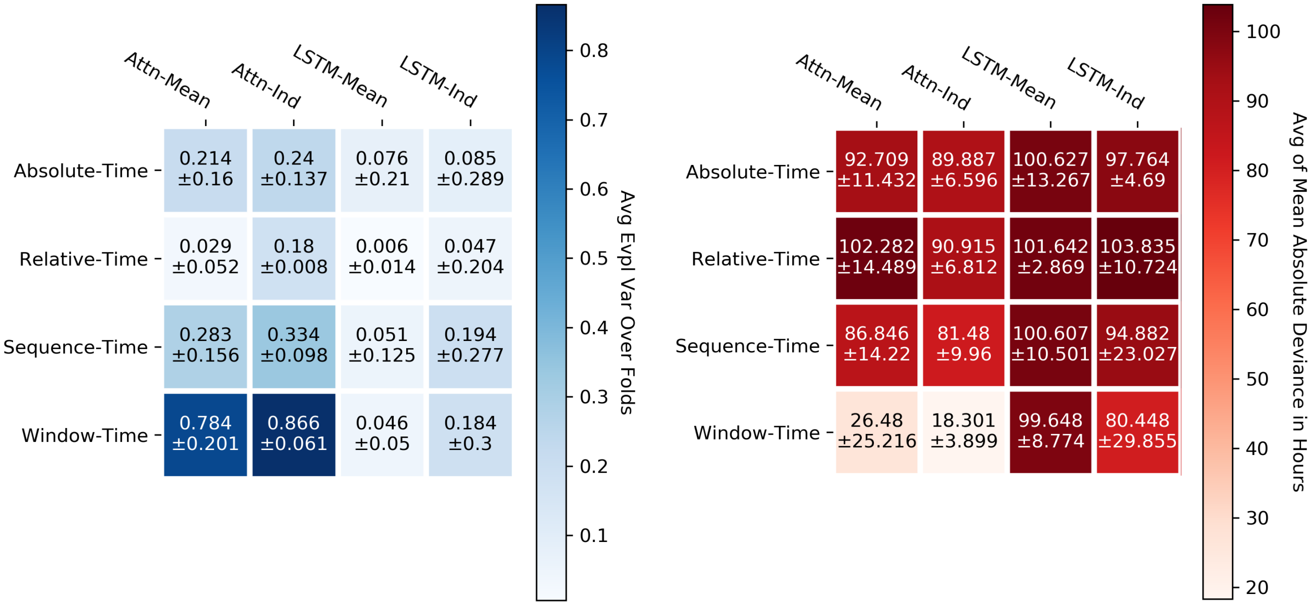


Figure 45: Remaining Length of Stay Prediction Performance

Our best model as able to produce predictions within 18 hours of the true value on average. This model used an average window definition that produced predictions approximately every 4 days and 20.5 hours on average. The model that performed best, made orders of magnitude fewer predictions than other temporal representations (a median of 1 instead of 69). We believe that the structure of this problem is not well suited to the task because it seems that the models that perform best learn to make the fewest number of predictions. Despite the challenges of this modeling task, a few of our LOS models were better than the best benchmark model, which had a MAD of [110.5hr, 111.4hr][73]. This finding provides evidence that our models trained and performed as accurately as one could expect for this task. Again it is unclear how much of the performance gains were attributable to cohort and data cleaning differences and how much was attributable to the temporal representation.

Discussion

To begin this series of experiments, we performed a detailed analysis of the data to understand its characteristics. We focused on the characteristics that had the greatest effect on model performance based on the findings from Chapter III. Those characteristics included inter-subject variability, intra-subject variability, feature autocorrelation, and the general sampling mechanism. Based on those observed characteristics of the MIMIC III data, we made predictions on the best temporal representation to use for each modeling problem (24hr ICU mortality and remaining LOS prediction). We based our prediction on the results from the NADG experiments of Chapter III, because those experiments approximated many of the key characteristics of MIMIC III (asynchronicity, correlated non-causal features, non-random sampling, and outcome distribution). The Chapter III results suggested that window-time would offer the greatest performance advantage in the 24hr ICU mortality problem when paired with the Attention model. The experimental results of this chapter confirmed our prediction exactly. Furthermore, the results from the NADG in Chapter III did not find that the advantages of the window-time and attention model combination were dependent on characteristics of the features or the cohort. Taken together, the confirmation of the best performing window-time-attention-model combination with the results of Chapter III suggest that the performance advantages should also be seen in other modeling problems with asynchronous features, correlated non-causal features, non-random sampling, and an outcome label present at each measurement occasion. Having cases act as an absorbing Markov state may also be a necessary condition for the advantages of window-time paired with the Attention model to hold. Despite all these constraints, there are many classification problems in the clinical domain that may benefit from these findings: mortality prediction in general, Human Immunodeficiency Virus status, cancer stage, and 30-day readmission status. All of these problems have outcomes that act as an absorbing state with features that are sparsely and episodically collected through a non-random inferential process by clinicians over time. Many of the routinely collected clinical measures that would be used as predictors are surrogate measures of vital function that are correlated with disease, but do not generally point to a pathognomonic cause on their own.

The other key findings of this chapter relate to how we realized the performance benefits of the window-time and Attention model combination. Our window-time representation was as effective as it was, because we tuned the window-size as a hyper-parameter. The tuned window size is at the level of the cohort average. Individual subjects within the cohort may be better optimized with smaller/larger windows than the tuned window size. Building in the flexibility to tune the window-size is not commonly done, because it adds complexity to the hyper-parameter tuning process[66, 68, 73, 74]. Many model builders that use window-time pick a window-size to fit socio-technical considerations in model use[66, 74]. Those that do perform trial and error on the window-size, use a predetermined search grid in isolation of other hyper-parameters[68, 73]. The search grid approach may ignore potential interactions the window size might have with other hyper parameters (e.g., the learning rate or the batch size), and unnecessarily constrains the range of possible values.

In the remaining ICU LOS prediction task, although the window size was not tuned to some inflection point, being tuned toward the boundary of possible values (an average of 4.85 days of a 5 day maximum) was also informative and interpretable. The window size was more evidence of features that were ill suited to predicting the outcome as it the outcome was framed. This kind of dynamic may be difficult to detect without tuning the window-size. There are not many options to optimize the representation of predictors that are more or less orthogonal to the outcome.

One might argue that the results from the LOS prediction task contradict our generalizability conclusions, as we found performance advantages with window-time and the attention model where we predicted neither would be particularly beneficial. However, if one were to dig deeper into the results, one would see that the best Attention models using window-time were making one prediction per subject on average. That one prediction required data over a time window greater than the average LOS in the ICU (4.85 day window size versus a 3 day average ICU LOS). Despite making 68 fewer predictions on average than the other representations, the window-time models still had an average MAD of at least 18.3 hours, not a particularly useful model. The lack of predictions made by the window-time representation suggests a fundamental issue with the framing of the prediction task in that there is a mismatch between the data used as predictors and the outcome of interest. As the problem was framed, it suggests that the ICU discharge criteria is correlated to the vital measures of oxygen saturation, heart rate, blood pressure, etcetera. While stability in those vital measures might be necessary to step a patient down to a medical unit (a big assumption), they are certainly not sufficient. We lack details on the organizational policies and structure at Beth Israel Deaconess Medical Center to know the capabilities and roles played by the other hospital units. Furthermore, admission/transfer/discharge are all clinical process outcomes[163]. That means that administrative and organizational factors can have a big effect on the LOS that will not be captured in vital signs[163]. Organizational factors, such as staffing[164] and the use of residents during the course of care[165], can affect the length of stay within a hospital. Hospitals are also subject to different cost and capacity pressures that add variability to the length of stay between institutions and within institutions over time[166]. Differing workflows as well as variable enforcement of care pathways can lead to length of stay differences between service lines and between specialties[167]. Those policies also vary between institutions[168]. The variance in length of stay both within and between institutions makes it a challenging outcome to predict, especially using only clinical variables.

Our methodology may also be of benefit to model builders. To obtain the performance results on 24hr ICU mortality and remaining ICU LOS required weeks of compute time. Model builders working with large data sets under constraints such as computational resources or time, may benefit from creating simplified data generating models. These simplified data generating models can help modelers come to data driven answers to consequential modeling decisions such as the modeling method or temporal representation of the data without the full costs of trial and error on the target dataset. This approach is challenging to pull off and we encountered some difficulties in our simulated data. Matching the type of relationship the features have with the outcome along with the synchronicity of the features, the sampling scheme of the features, along with the outcome distribution is a complex task. However, the creation of synthetic data through generative adversarial networks is an active area of research.

Strengths and Limitations

The major strengths of this work lie in its thorough sensitivity analysis of relevant parameters, comprehensive hyper-parameter tuning, and robust evaluation strategy. This experiment rigorously evaluated different temporal representations in a well studied data set. Not only did we vary the temporal representation, but also the architecture and the imputation methods. To the author’s knowledge no other group has undertaken a study of different temporal representations using state of the art deep architectures. We built and compared our research cohort, modeling problems, and results to a published benchmark. We achieved state of the art performance on 24hr ICU mortality prediction compared to previously published worked on this dataset[66, 68, 73, 74, 169]. However, it is not a straightforward comparison between studies, as prior studies used varying cohort definitions and feature sets. Our methods for hyper-parameter tuning and evaluation were amongst the most rigorous of the works mentioned. Where the other works used high variance test-train splitting to evaluate model performance, we used the lower variance nested cross validation strategy. Our methods would have been intractable without the generous support of the Vanderbilt University Data Science Institute.We also demonstrated the advantages of the window-time representation when working with data with high intra-subject variability, which has been suggested by other researchers[54, 73].

While three-fold cross-validation is generally more robust than a single test-train split, it was not enough to produce high precision performance estimates across all model combinations. Reducing the combinatorics to repeat the three-fold cross-validation multiple times or to add additional folds would be a desirable option in future work. The uniformly random selection of 25 sets of different hyper-parameters may not have sufficiently explored the hyper-parameter space and more targeted hyper-parameter optimization strategies (e.g., Bayesian optimization) may have yielded results with lower variation. The use of a single data set is also a limitation. This work would benefit from a replication study using other publicly available clinical datasets. A replication study would build a greater evidence base for the conclusions made from the artificial data experiments (the advantages of window-time in asynchronous/high intra-subject variability data and the methodology of using artificial data to inform modeling decisions in real data). We did not replicate the benchmark study methods to the letter. The small differences in pre-processing led to difficulties in attributing the performance differences. We could not say for certain what performance gains we observed were due to temporal representation and which were due to data pre-processing differences. Despite these limitations, we believe our study has produced generalizable guidance for longitudinal model builders in terms of model architecture choice, temporal representation, and how to get the most out of the window-time representation.

Chapter V

EXPLORING THE INTERPRETATION OF WINDOW SIZE

Study Design

This chapter sought to extend the results of Chapter IV by attempting to explain the efficacy of window time, particularly a tunable window size, on asynchronous data. Our Attention models in Chapter IV tuned to a window-size of 17.5 hours (1,050 minutes) on average. We hypothesized that the best performing window size maximized the unique information content of the features between windows. We believed that autocorrelation could serve as a surrogate measure, where less autocorrelation would be desirable to more. A lower autocorrelation would suggest more distinct feature values between windows, while a larger autocorrelation would suggest that the feature value of the previous window is increasingly similar to the value of the current window. We believed autocorrelation to be a viable surrogate measure because the windows were equally wide, and the average autocorrelation could be easily measured for each feature. We further hypothesized that the features with the greatest correlation to the outcome would have the greatest influence on the window size. To test these hypotheses we generated two synthetic data sets using the data generator described in Chapter II as well as the MIMIC III data. In the positive case dataset we hoped to observe the window size with the smallest autocorrelation for the dominant feature(s) also maximize model performance. In the negative control we hoped to see no relationship between autocorrelation and AUROCC.

Materials

For these experiments we used a 2015 MacBook Pro with four 2.9 GHz Intel processors and eight GB of RAM. ACCE was again our primary compute environment for the experiment. In the ACCRE environment, we again used 31GB of storage and 34 CPUs connected to 17 GPUs in a configuration of two CPUs per GPU.

We continued our used of the Python programming language version 3.6.3 for data preprocessing, predictive model development, and data aggregation/visualization. Jupyter notebooks[127], the Sublime Text editor[128] (version 2), and vim[141] were used as the primary Python editors. The Python package dependencies for this Chapter are listed in Table 30.

Package	Version
IPython[125]	7.12.0
Joblib[145]	0.15.1
Jupyter[126]	1.0.0
Long-Gen[124]	0.2.3
Matplotlib[129]	3.1.3
Numpy[130]	1.18.1
Pandas[131]	1.0.1
Scipy[132]	1.4.1
Scitkit-learn[133]	0.22.1
Torch[146]	1.6.0

Table 30: Window-Time Exploration Software Dependencies

Methods

We generated a synthetic positive and a negative control dataset for experimentation. The positive control had two features that were both directly causal to the outcome. One of the variables, x1, had ten times the effect size as x2. The negative control also had two variables, but neither were directly causal with the outcome. We did not use any of the realism mechanisms described in Chapter II (timespan link, measurement link, or feature link). Both data sets had subject specific intercepts and time slopes. The specific parameters of the long-gen package used to create the data are described in Table 31.

Attribute	Positive Case	Negative Control
Autocorrelation	Moderate-[0.33, 0.66)	Moderate-[0.33, 0.66)
Coefficient_Values	Intercept: 0 Time: 7 x_1: 10 x_2: -1	Intercept: -5 Time: 0.25 x_1: 0 x_2: 0
Collinearity	Low-[0.01, 0.33)	Low-[0.01, 0.33)
Link_Function	Logit	Logit
Measurement_Distribution	Log-Normal(0.75, 20, 5)	Log-Normal(0.75, 20, 5)
Num_Extraneous_Variables	0	0
Number_of_Features	2	2
Number_of_Model_Changes	0	0
Number_of_Subjects	3,000	3,000
Probability_Threshold	None	None
Random_Effects	Intercept & time	Intercept & time
Random_Effect_Collinearity	0.13	0.13
Random_Effect_Cut_Point	NA	NA
Random_Effect_Insert_Point	NA	NA
Realism_Functions	NA	NA
Sampling_Scheme	Random	Random
Temporal_Trend	Linear	Linear
Time_Breaks	None	None
Variance_of_Error	0.05	0.05
Variance_of_Random_Effects	4	4

Table 31: Synthetic Window-Time Data Parameters

We used a different non-random mechanism to create missing values in these data. We erased all values of x1>-1.2 in the positive case dataset and less than -0.09 in the negative control dataset. This scheme led to eight and ten percent of the values to be missing respectively. We erased all values of x2 less than -1.4 in the positive case dataset and less than -0.03 in the negative control dataset. The rates of missing values for x2 were nine and ten percent respectively.

After creating the data, we plotted the autocorrelation of the features at a variety of window sizes: three windows (0.333 wide), four windows (0.25 wide), five windows (0.2 wide), ten windows (0.1 wide), twenty windows (0.05 wide), and fifty windows (0.02 wide). We then fit an attention model for each fixed window size. Each attention model used mean-imputation to handle missing values. The Attention models had the same architecture specified in Chapter II. The hyper-parameter tuning and evaluation methods were also the same as Chapter II with one exception; we used ten-fold nested cross validation instead of three-fold.

For the MIMIC data, we used the same processed data from Chapter IV. We first analyzed the median time and next measurement to eliminate the measurement frequency as the cause for the preferred window-size of 1,050 minutes. We also measured the correlation of each feature to the outcome to gain an understanding of which variables may be the most influential. The experimental setup follows a similar path as the synthetic data experiments. We measured the average of the per-subject autocorrelation for the six most correlated features at different window sizes: 250min, 500min, 750min, 1000min, 1250min, 1500min, 1750min, and 2000min. Specifically, we measured the autocorrelation of the features for each feature, for each subject and then reported the population average autocorrelation for each feature. From those results, we then selected three window sizes to evaluate model performance on in the hopes of finding a negative relationship between autocorrelation and AUROCC. We chose to fit an Attention model using mean+indicator imputation with the same architecture, hyper-parameter tuning, and evaluation strategy as specified in Chapter II.

Results

The synthetic experiments provided support to our hypotheses. In the positive case data set, the

AUROCC has a clear negative relationship with the autocorrelation of x1, much more so than x2. We visualize the autocorrelation of the features x1 and x2 across a range of window sizes along with the AUROCC performance of an Attention model fixed using that window size in Figure 46.

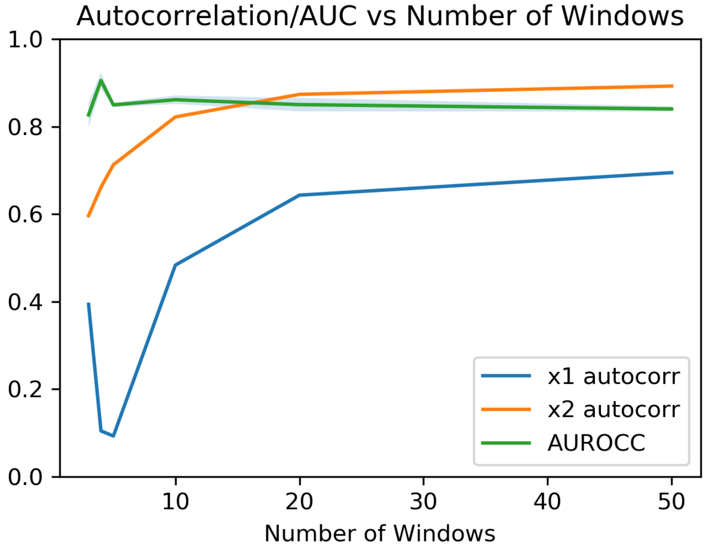


Figure 46: Window Size Positive Case Results

The negative control results did not show the same trend as that of the positive case results. There may be a weak association, but it is unclear if the difference is of significance. Figure 47 shows the model performance and feature autocorrelations of the negative control.

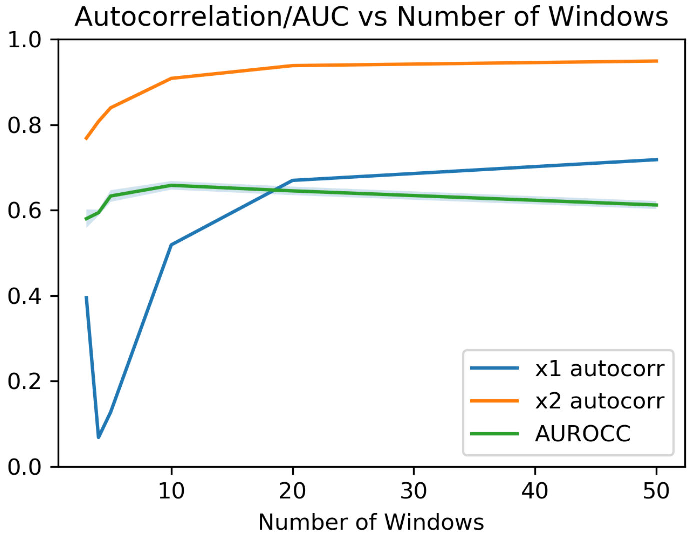


Figure 47: Window Size Negative Control Results

In the MIMIC data we did not find any clear relation between the data collection times and the average preferred window size from the experiments of Chapter IV. This finding suggests that the preferred window size has more to do with the feature values in each window then the presence of data. Table 32 shows the median collection time between observing a new value for each feature as well as the feature's correlation with the outcome. We found that systolic blood pressure, mean blood pressure, respiratory rate, temperature, oxygen saturation and the GCS eye score had the strongest associations with the outcome. We therefore hypothesized that these variables might have the strongest influence on the optimal window size.

Variable	Median Time Until Next Measurement	Correlation to 24hr Mortality
Glucose	120min	0.025
Systolic BP	60min	-0.085
Diastolic BP	60min	-0.038
Mean BP	60min	-0.042
Respiratory Rate	40min	-0.055
Temperature	180min	0.059
Blood pH	127min	0.008
O_2 Saturation	60min	0.056
Heart Rate	60min	-0.025
% Inspired O_2	40min	0.007
GCS Verbal	240min	-0.031
GCS Motor	240min	0.013
GCS Eye	240min	-0.055
Cap Refill	240min	0.017

Table 32: Median Time Until Next Measurement

The autocorrelation of the six features mentioned followed a parabolic path similar, though not as steep, as that of x1 in Figure 47. Based on the average autocorrelation of the features, we selected window sizes of 425min, 1250min, and 1750min to evaluate model performance. Figure 48 shows a slight association between the model performance (pink) and the minimum autocorrelation of the majority of the features. This trend is fairly weak, but it is a good preliminary result to the interpretability of the window size.

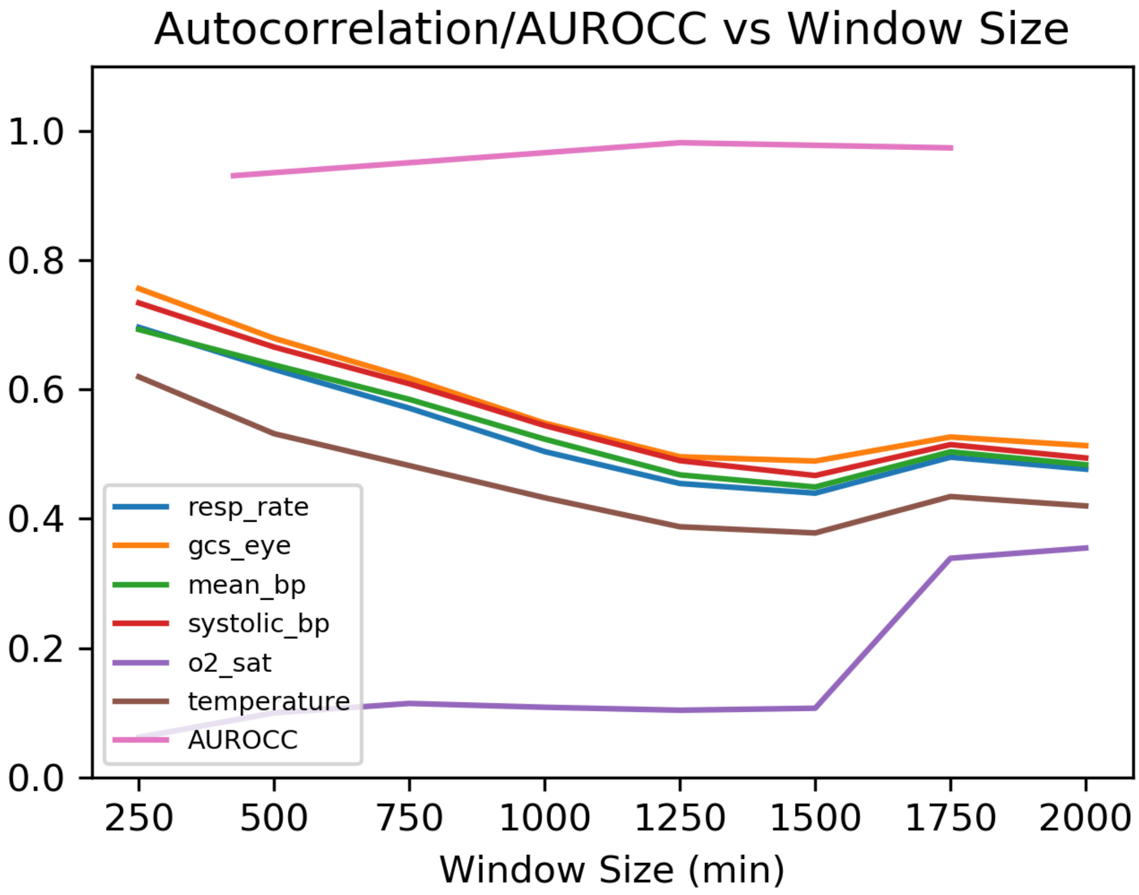


Figure 48: Window Size MIMIC Results

Discussion

Both the synthetic data and MIMIC III provided evidence to our hypothesis that the window size may indeed be related to maximizing the information content of the windows. We based our original hypothesis on the theoretical underpinnings of window size in signal processing[86, 87]. We can see in Figures 46-48 that many of the features' autocorrelation appeared to have a parabolic relationship with the window-size. The synthetic data provided clear evidence of our hypothesis in that not only were the AUROCC and the autocorrelation inversely related, but x1 was the feature that mattered most when minimizing autocorrelation. In the MIMIC III data we saw that the theoretical optimal autocorrelation occurred between a window-size of 1,250 and 1,500 minutes, which is near the range that the Attention models tuned the window size to (1050 minutes on average). Given that we only performed 25 random draws, and each draw contained a full set of new hyper- parameters, it was heartening to see the average preferred window size get so close to the optimal range. In the more explicit experiment we observed a small performance different between the different window sizes, a difference of 0.05 between the 1,250 minute window and the 425 minute window. We observed a difference of 0.01 between the 12,50 minute window and the 1,750 minute window. We believe that these findings suggest that the window size may be interpreted as the optimal rate of new information. Windows that are too small yield trivial feature differences from window to window, and windows that are too big yield the subject average for the admission.

The interpretability of the window size may lead to a couple useful results. Firstly, there may be situations where tuning the window-size may not be practical from an engineering/development perspective. In this case, the interpretability of the window-size can allow model builders to optimize the window-size without tuning by finding the window-size that minimizes the autocorrelation of the features most correlated with the outcome. Secondly, the inheritability of window size can provide insight into the data collection/recording process. Knowing when data is significantly changed enough to optimize prognostication may lead administrators to change how and when data is collected to facilitate better secondary use.

Chapter VI

CONCLUSIONS

In this dissertation we built a means to produce longitudinal data with known distributional characteristics and properties. We evaluated that this data was a gold standard, and that our software package had the flexibility to reproduce the characteristics most important for replicating the MIMIC III data. Through a methodical process we evaluated our software package, produced testable hypotheses, and validated those hypotheses. Model developers have the ability to use this package to learn more about data characteristics that lead to superior model performance, as this package is publicly available. Through the experiments laid out in Chapters II-IV, we learned that it is important to recreate the following properties of the target data set when generating synthetic data: the nature of the relationship between the features and the outcome, the synchronicity of the features and outcome, the sampling scheme of the features and outcome, as well as the outcome distribution.

We varied a variety of data generating parameters to produce generalizable inferences on how those characteristics influence model performance and how they interact with temporal representation. We found that window-time has limited performance benefits in synchronous data sets except in instances where the feature autocorrelation is low and the intra-subject variance is high. Window-time becomes much more advantageous when data is incomplete and imputation is used. The irregular measurements and use of imputation can lead to more intra-subject variance as measured subject specific values are mixed in with imputed means from the population. We found that window-time can be a dominant representation for regression and classification tasks with asynchronous features. When the levels autocorrelation in the feature space are high (0.75+), then other temporal representations became suitable alternatives. These findings were consistent between the causal and non-causal features over a variety of data parameters and signal to noise ratios. Another conclusion gleaned from the artificial data was that the Attention model was the most performant when used with window-time. The additional complexity of the multi-headed attention mechanism was able to capture more temporal patterns than the LSTM. These findings provided testable hypotheses that were validated in real clinical data. We believe that our artificial data findings can generalize to other similarly structured prediction problems, thereby giving model builders evidence to use when trial and error is not possible/practical, as well as providing data on what data characteristics are consequential for model building and which are not.

In Chapter IV we built an internally consistent experiment with real world data. We used the prior literature to benchmark and validate our experimental setup and methods. We confirmed our hypotheses generated in Chapter III, and gained key insights into what made window-time a dominant representation compared to the other temporal representations. We are confident that our findings are valid, because of the validation and reconciliation steps we took when comparing to our benchmark study. Additionally, no other research group has published a better 24hr ICU mortality model, to our best knowledge. The key to this performance was the window size. In Chapter V, we found that the window size should be a tunable parameter because the model will adjust the window size to be large enough to smooth out uninformative noise, but to also be small enough as to maximize the information content between the windows of the key predictors. We believe this result to be intuitive and consistent with how the signal processing domain uses window-time representations and determining the sampling rate of a process. Our autocorrelation analyses provides the means for model builders to approximate the optimal window size without hyper-parameter tuning potentially saving great effort and computational time. We believe our conclusions will generalize because we focused on higher level characteristics of data and not specific data sets or modeling problems themselves. We used the modeling problems of 24hr ICU mortality and remaining ICU LOS prediction to narrow some of the combinatoric possibilities for the data characteristics explored in Chapter III and to provide a tangible example in Chapter IV. However, those problems could have been substituted for other prediction tasks of a similar frame, such as readmission prediction. The focus of this work was not on a particular problem or dataset, but on learning the effect of different data characteristics and temporal representation on the performance of deep longitudinal prognostic models.

In future work we hope to further develop the interpretation of the window-size as well as better define the conditions where window-time is advantageous. We foresee experiments where we attempt to use correlation methods to find the optimal window size on data sets with varying characteristics and forms such as publicly available longitudinal clinical trials or other EHR datasets such as Vanderbilt's Synthetic Derivative. Such experiments would recreate a published model cohort and accompanying experiment to use as a validation benchmark. Then, we would use correlation methods to create the autocorrelation plot of all the features similar to Figures 46-48 of Chapter V. Next, we would constrain the window size tuning set to the window sizes we explored in the autocorrelation analysis. We could then compare the model performance of the different window sizes during tuning to the autocorrelation plot. We could then attempt to establish a mathematical relationship between the model performance at different window sizes, the autocorrelation of different features at different window sizes, and the overall correlation of each feature to the outcome. Such experiments would validate our interpretation of window size as well as provide model builders an easily applied formula or rule of thumb to select a window-size when using a window-time representation.

REFERENCES

1. Stead WW. Clinical Implications and Challenges of Artificial Intelligence and Deep Learning. J Am Med Assoc. 2018:E1-E2. doi:10.1037/h0030806

2. Amarasingham R, Patzer RE, Huesch M, Nguyen NQ, Xie B. Implementing Electronic Health Care Predictive Analytics: Considerations And Challenges. Health Aff. 2014;33(7):1148-1154. doi:10.1377/hlthaff.2014.0352

3. Bose R. Advanced analytics: opportunities and challenges. Ind Manag Data Syst. 2009;109(2):155-172. doi:10.1108/02635570910930073

4. Xiao C, Choi E, Sun J. Opportunities and challenges in developing risk prediction models with electronic health records data: a systematic review. J Am Med Inf Assoc. 2018;0(0):1-10. doi:10.1093/jamia/ocy068

5. Miotto R, Wang F, Wang S, Jiang X, Dudley JT. Deep learning for healthcare: review, opportunities and challenges. Brief Bioinform.

6. Lenert MC, Matheny ME, Walsh CG. Prognostic models will be victims of their own success, unless…. J Am Med Informatics Assoc. 2019;26(12):1645–1650. doi:10.1093/jamia/ocz145.

7. Van Calster B, Wynants L, Timmerman D, Steyerberg EW, Collins GS. Predictive analytics in health care: how can we know it works? J Am Med Inform Assoc. 2019;26(12):1651-1654. doi:10.1093/jamia/ocz130.

8. Novak LL, Holden RJ, Anders SH, Hong JY, Karsh BT. Using a sociotechnical framework to understand adaptations in health IT implementation. *Inter J of Med Informatics*. 2013 Dec 1;82(12):e331-44.

9. Rouse WB and Cortese DA. Engineering the system of healthcare delivery. Vol. 153. IOS Press, 2010.

10. Haynes B, Haines A. Barriers and bridges to evidence based clinical practice. *BMJ*. 1998;317(7153):273-276. doi:10.1136/bmj.317.7153.273.

11. Straus SE, Mcalister FA. Evidence-based medicine: a commentary on common criticisms. *Can Med Assoc J*. 2000;163(7):3-7.

12. Sackett DL, Rosenberg WMC, Gray JAM, Haynes RB, Richardson WS. Evidence based medicine: what it is and what it isn't. *BMJ*. 1996;312(7023):71-72. doi:10.1136/bmj.312.7023.71.

13. Agniel, D., Kohane, I. S., & Weber, G. M. (2018). Biases in electronic health record data due to processes

within the healthcare system: retrospective observational study. *BMJ*, *361*.

14. Janisch J, Pevný T, Lisý V. Classification with Costly Features using Deep Reinforcement Learning. 2017. http://arxiv.org/abs/1711.07364.

15. Lenert MC, Miller RA, Vorobeychik Y, Walsh CG. A Method For Analyzing Inpatient Care Variability Through Physicians' Orders. J Biomed Inform. 2019:103111. doi:https://doi.org/10.1016/j.jbi.2019.103111.

16. Fisk RP, Patrício L. A Brief History of ICD-10-PCS. J Serv Manag. 2011;22(4):2011-2013. doi:10.1108/josm.2011.08522daa.001.

17. Lindberg DAB. Commentary on G . Octo Barnett's Report to the Computer Research Study Section. *J Am Med Informatics Assoc*. 2006;13(2):136-137. doi:10.1197/jamia.M2022.Dr.

18. Wright A, Bates DW. Distribution of Problems, Medications and Lab Results in Electronic Health Records: The Pareto Principle at Work. *Appl Clin Inform*. 2010;1(1):32-37. doi:10.4338/ACI-2009-12-RA-0023.

19. Chen JH, Altman RB. Automated Physician Order Recommendations and Outcome Predictions by Data-Mining Electronic Medical Records. *AMIA Summits Transl Sci Proc*. 2014:206-210.

20. Adler-Milstein J, DesRoches CM, Kralovec P, et al. Electronic Health Record Adoption In US Hospitals: Progress Continues, But Challenges Persist. *Health Aff*. 2015;34(12):2174-2180. doi:10.1377/hlthaff.2015.0992.

21. Tsou A, Lehmann C, Michel J, Solomon R, Possanza L, Gandhi T. Safe Practices for Copy and Paste in the EHR. Appl Clin Inform. 2017;26(01):12-34. doi:10.4338/aci-2016-09-r-0150

22. Tsou A, Lehmann C, Michel J, Solomon R, Possanza L, Gandhi T. Safe Practices for Copy and Paste in the EHR. Appl Clin Inform. 2017;26(01):12-34. doi:10.4338/aci-2016-09-r-0150.

23. Osterman, T. J. (2017). Extracting Detailed Tobacco Exposure from the Electronic Health Record (Masters Thesis, Vanderbilt University).

24. Neilson EG, Johnson KB, Rosenbloom ST, et al. The impact of peer management on test-ordering behavior. Ann Intern Med. 2004;141(3):196-204.

25. Ballesca MA, Laguardia JC, Lee PC, et al. An electronic order set for acute myocardial infarction is associated with improved patient outcomes through better adherence to clinical practice guidelines. J Hosp Med. 2014;9(3):155-161. doi:10.1002/jhm.2149.

26. Hulse NC, Lee J, Borgeson T. Visualization of Order Set Creation and Usage Patterns in Early Implementation Phases of an Electronic Health Record. AMIA . Annu Symp proceedings AMIA Symp. 2016;2016:657-666.

.

27. Eldenburg L, Kallapur S. Changes in hospital service mix and cost allocations in response to changes in Medicare reimbursement schemes. J Account Econ. 1997;23:31-51.

28. Preston AM. The birth of clinical accounting: A study of the emergence and transformations of discourses on costs and practices of accounting in U.S. hospitals. Accounting, Organ Soc. 1992;17(1):63-100. doi:10.1016/0361-3682(92)90036-R.

29. Chen JH, Alagappan M, Goldstein MK, Asch SM, Altman RB. Decaying relevance of clinical data towards future decisions in data-driven inpatient clinical order sets. Int J Med Inform. 2017;102:71-79. doi:10.1016/j.ijmedinf.2017.03.006.

30. Institute of Medicine. Crossing the quality chasm: a new health system for the 21th century. Inst Med. 2001;(March):1-8. doi:10.17226/10027.

31. Singh S, Fletcher KE. A qualitative evaluation of geographical localization of hospitalists: How unintended consequences may impact quality. J Gen Intern Med. 2014;29(7):1009-1016. doi:10.1007/s11606-014-2780-6.

32. Sittig DF, Wright A, Osheroff JA, et al. Grand challenges in clinical decision support. J Biomed Inform. 2008;41(2):387-392. doi:10.1016/j.jbi.2007.09.003.

33. CMS. Department of Health and Human Services Regulations. Fed Regist. 2013;78(184).

34. Pham T, Tran T, Phung D, Venkatesh S. DeepCare: A Deep Dynamic Memory Model for Predictive Medicine BT - Advances in Knowledge Discovery and Data Mining. In: Bailey J, Khan L, Washio T, Dobbie G, Huang JZ, Wang R, eds. Cham: Springer International Publishing; 2016:30-41.

35. Moskovitch R, Choi H, Hripcsak G, Tatonetti N. Prognosis of Clinical Outcomes with Temporal Patterns and Experiences with One Class Feature Selection. IEEE/ACM Trans Comput Biol Bioinforma. 2017;14(3):555-563. doi:10.1109/TCBB.2016.2591539.

36. Hripcsak G, Albers DJ, Perotte A. Exploiting time in electronic health record correlations. J Am Med Informatics Assoc. 2011;109-115. doi:10.1136/amiajnl-2011-000463.

37. Combi C, Shahar Y. Temporal reasoning and temporal data maintenance in medicine: Issues and

challenges. Comput Biol Med. 1997;27(5):353-368. doi:10.1016/S0010-4825(96)00010-8.

38. Du F, Shneiderman B, Plaisant C, Malik S, Perer A. Coping with volume and variety in temporal event sequences: Strategies for sharpening analytic focus. IEEE Trans Vis Comput Graph. 2017;23(6):1636-1649. doi:10.1109/TVCG.2016.2539960.

39. Roddick JF, Spiliopoulou M. A Survey of Temporal Knowledge Discovery Paradigms and Methods. IEEE Trans Knowl Data Eng. 2002;14(4):750-767.

40. Shahar Y. A framework for knowledge-based temporal abstraction. Artif Intell. 1997;90(96):79-133.

41. Elman J. Finding structure in time. Cogn Sci. 1990;211(1):1-28. doi:10.1207/s15516709cog1402_1.

42. Stacey M, McGregor C. Temporal abstraction in intelligent clinical data analysis: A survey. Artif Intell Med. 2007;39(1):1-24. doi:10.1016/j.artmed.2006.08.002.

43. Allen JF. Towards a General Theory of Action and Time. Artif Intell. 1984;23:123-154.

44. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735-1780.

45. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998-6008).

46. Fitzmaurice GM, Laird NM, Ware JH. Applied Longitudinal Analysis. Vol 998. John Wiley & Sons; 2012.

47. Steyerberg, E. W. (2019). Clinical prediction models. Springer International Publishing.

48. Abu-Mostafa, Y. S., Magdon-Ismail, M., & Lin, H. T. (2012). Learning from data (Vol. 4). New York, NY, USA:: AMLBook.

49. Gallager, R. G. (2013). Stochastic processes: theory for applications. Cambridge University Press.

50. Armstrong, J. S. (Ed.). (2001). Principles of forecasting: a handbook for researchers and practitioners (Vol. 30). Springer Science & Business Media.

51. Géron, A. (2019). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems. O'Reilly Media.

52. Harrell Jr, F. E. (2015). Regression modeling strategies: with applications to linear models, logistic and ordinal regression, and survival analysis. Springer.

53. Woolrich, M. W., Ripley, B. D., Brady, M., & Smith, S. M. (2001). Temporal autocorrelation in univariate linear modeling of FMRI data. *Neuroimage.* 14(6), 1370-1386.

54. VanHouten, J. P. (2016). Using Abstraction to Overcome Problems of Sparsity, Irregularity, and Asynchrony in Structured Medical Data (Doctoral dissertation, Vanderbilt University).

55. Lenert MC, Mize DE, Walsh CG. X Marks the Spot: Mapping Similarity Between Clinical Trial Cohorts and US Counties. AMIA Annu Symp proceedings; 2017: 1110-1119.

56. Peterson ED, Coombs LP, Ferguson TB, et al. Hospital variability in length of stay after coronary artery bypass surgery: Results from the Society of Thoracic Surgeon's National Cardiac Database. Ann Thorac Surg. 2002;74(2):464-473. doi:10.1016/S0003-4975(02)03694-9.

57. Fisher AJ, Medaglia JD, Jeronimus BF. Lack of group-to-individual generalizability is a threat to human subjects research. *Proc Natl Acad Sci USA*. 2018;115(27):E6106-E6115. doi:10.1073/pnas.1711978115.

58. Riley RD, Ensor J, Snell KIE, et al. Calculating the sample size required for developing a clinical prediction model. *BMJ*. 2020;441(March):1-12. doi:10.1136/bmj.m441.

59. Groenwold, R. H. (2020). Informative missingness in electronic health record systems: the curse of knowing. Diagnostic and Prognostic Research. 4(1), 1-6.

60. Taheri P a, Butz D a, Greenfield LJ. Length of stay has minimal impact on the cost of hospital admission. J Am Coll Surg. 2000;191(2):123-130. doi:10.1016/S1072-7515(00)00352-5.

61. Wu S, Liu S, Sohn S, et al. Modeling asynchronous event sequences with RNNs. J Biomed Inform. 2018;83(June):167-177. doi:10.1016/j.jbi.2018.05.016.

62. Sterne, J. A., White, I. R., Carlin, J. B., Spratt, M., Royston, P., Kenward, M. G., ... & Carpenter, J. R. (2009). Multiple imputation for missing data in epidemiological and clini cal research: potential and pitfalls. *BMJ. 338*.

63. Rosner, B. (2015). Fundamentals of Biostatistics. Nelson Education.

64. Landermand LR, Land KC, Pieper CF. An Empirical Evaluation of the Predictive Mean Matching Method for Imputing Missing Values. Sociol Methods Res. 1997;26(1):3-33. doi:10.1177/0049124197026001001.

65. Efron B. Missing Data, Imputation, and the Bootstrap. J Am Stat Assoc. 1994;89(426):463-475. doi:10.1080/01621459.1994.10476768.

66. Lipton ZC, Kale DC, Wetzel R. Modeling Missing Data in Clinical Time Series with RNNs. Proc Mach Learn Healthc 2016 JMLR W&C Track. 2016;56.

67. Fletcher Mercaldo, S., & Blume, J. D. (2020). Missing data and prediction: the pattern submodel. Biostatistics, 21(2), 236-252.

68. Che Z, Purushotham S, Cho K, Sontag D, Liu Y. Recurrent Neural Networks for Multivariate Time Series with Missing Values. Sci Rep. 2018;8(1):1-12. doi:10.1038/s41598-018-24271-9.

69. Collins LM. Analysis of Longitudinal Data: The Integration of Theoretical Model, Temporal Design, and Statistical Model. Annu Rev Psychol. 2006;57(1):505-528. doi:10.1146/ annurev.psych.57.102904.190146.

70. Kumar P, Nestsiarovich A, Nelson SJ, Kerner B, Perkins DJ, Lambert CG. Imputation and characterization of uncoded self-harm in major mental illness using machine learning. J Am Med Inf Assoc. 2019;0(0):1-11. doi:10.1093/jamia/ocz173.

71. Luo Y, Szolovits P, Dighe AS, Baron JM. Using Machine Learning to Predict Laboratory Test Results. Am J Clin Pathol. 2016;145(6):778-788. doi:10.1093/ajcp/aqw064.

72. Shahar Y, Musen MA. Knowledge-based temporal abstraction in clinical domains. Artif Intell Med. 1996;8(3):267-298. doi:10.1016/0933-3657(95)00036-4.

73. Harutyunyan H, Khachatrian H, Kale DC, Ver Steeg G, Galstyan A. Multitask learning and benchmarking with clinical time series data. Sci data. 2019;6(1):96. doi:10.1038/s41597-019-0103-9.

74. Purushotham S, Meng C, Che Z, Liu Y. Benchmarking deep learning models on large healthcare datasets. J Biomed Inform. 2018;83(April):112-134. doi:10.1016/j.jbi.2018.04.007.

75. Halpern Y, Choi Y, Horng S, Sontag D. Using Anchors to Estimate Clinical State without Labeled Data. AMIA Annu Symp Proc. 2014.

76. Dorn, J. (1992, July). Temporal reasoning in sequence graphs. In AAAI (pp. 735-740).

77. Bobroske, K., Larish, C., Cattrell, A., Bjarnadóttir, M. V., & Huan, L. (2020). The bird's-eye view: A data-driven approach to understanding patient journeys from claims data. Journal of the American Medical Informatics Association.

78. Huang Z, Dong W, Ji L, He C, Duan H. Incorporating comorbidities into latent treatment pattern mining for clinical pathways. J Biomed Inform. 2016;59:227-239. doi:10.1016/j.jbi.2015.12.012.

79. Huang Z, Dong W, Duan H, Li H. Similarity Measure Between Patient Traces for Clinical Pathway Analysis : Problem , Method , and Applications. IEEE J Biomed Heal Informatics. 2014;18(1):4-14. doi:10.1109/JBHI.2013.2274281.

80. Stein, A., Musen, M. A., & Shahar, Y. (1996). Knowledge acquisition for temporal abstraction. In Proceedings of the AMIA annual fall symposium (p. 204). American Medical Informatics

Association.

81. Shahar Y. Dynamic temporal interpretation contexts for temporal abstraction. Proc Third Int Work Temporal Represent Reason. 1998;22:64-71. doi:10.1109/TIME.1996.555683.

82. Moskovitch R, Shahar Y. Fast time intervals mining using the transitivity of temporal relations. Knowl Inf Syst. 2015;42(1):21-48. doi:10.1007/s10115-013-0707-x.

83. Lasko TA. Processes with Application to Medical Event Data. 2014:469-476.

.

84. Song H, Rajan D, Thiagarajan JJ, Spanias A. Attend and diagnose: Clinical time series analysis using attention models. 32nd AAAI Conf Artif Intell AAAI 2018. 2018:4091-4098.

85. Orphanou K, Stassopoulou A, Keravnou E. Temporal abstraction and temporal Bayesian networks in clinical domains: A survey. Artif Intell Med. 2014;60(3):133-149. doi:10.1016/j.artmed.2013.12.007.

86. Schafer, R. W., & Rabiner, L. R. (1973). A digital signal processing approach to interpolation. Proceedings of the IEEE, 61(6), 692-702.

87. Adams, J. W. (1991). A new optimal window (signal processing). IEEE Transactions on signal processing, 39(8), 1753-1769.

88. West, B. T., Welch, K. B., & Galecki, A. T. (2014). Linear mixed models: a practical guide using statistical software. CRC Press.

89. Kotsiantis SB. Supervised machine learning: A review of classification techniques. Informatica. 2007;31:249-268. doi:10.1115/1.1559160.

90. van der Ploeg T, Austin PC, Steyerberg EW. Modern modeling techniques are data hungry: a simulation study for predicting dichotomous endpoints. BMC Med Res Methodol. 2014;14:137. doi:10.1186/1471-2288-14-137.

91. Tomašev N, Glorot X, Rae JW, et al. A clinically applicable approach to continuous prediction of future acute kidney injury. Nature. 2019;572(7767):116-119. doi:10.1038/s41586-019-1390-1.

92. Pham T, Tran T, Phung D, Venkatesh S. Predicting healthcare trajectories from medical records: A deep learning approach. J Biomed Inform. 2017;69:218-229. doi:10.1016/j.jbi.2017.04.001.

93. Campanella, G., Hanna, M. G., Geneslaw, L., Miraflor, A., Silva, V. W. K., Busam, K. J., ... & Fuchs, T. J. (2019). Clinical-grade computational pathology using weakly supervised deep learning on whole

slide images. Nature medicine, 25(8), 1301-1309.

94. Shickel B, Tighe PJ, Bihorac A, Rashidi P. Deep EHR: A Survey of Recent Advances in Deep Learning Techniques for Electronic Health Record (EHR) Analysis. IEEE J Biomed Heal Informatics. 2018;22(5):1589-1604. doi:10.1109/JBHI.2017.2767063.

95. Beaulieu-Jones BK, Orzechowski P, Moore JH. Mapping patient trajectories using longitudinal extraction and deep learning in the MIMIC-III critical care database. Pacific Symp Biocomput. 2018;0(212669):123-132. doi:10.1142/9789813235533_0012.

96. Bengio Y, Simard P, Frasconi P. Learning Long-Term Dependencies with Gradient Descent is Difficult. IEEE Trans Neural Networks. 1994;5(2):157-166. doi:10.1109/72.279181.

97. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735-1780.

98. Dey R, Salemt FM. Gate-variants of Gated Recurrent Unit (GRU) neural networks. In: Circuits and Systems (MWSCAS), 2017 IEEE 60th International Midwest Symposium On. IEEE; 2017:1597-1600.

99. Ma Y, Xiang Z, Du Q, Fan W. Effects of user-provided photos on hotel review helpfulness: An analytical approach with deep leaning. Int J Hosp Manag. 2018;71(April):120-131. doi:10.1016/j.ijhm.2017.12.008.

100. Al-Rfou, R., Choe, D., Constant, N., Guo, M., & Jones, L. (2019, July). Character-level language modeling with deeper self-attention. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 33, pp. 3159-3166).

101. Alammar J. The Illustrated Transformer. 27 June, 2018. Accessed 19 July, 2020.

.

102. Lenert MC and CG Walsh. "Balancing Performance and Interpretability: Selecting Features with Bootstrapped Ridge Regression." AMIA Annual Symposium Proceedings. Vol. 2018. American Medical Informatics Association, 2018.

103. Andonie, R. (2019). Hyperparameter optimization in learning systems. Journal of Membrane Computing, 1-13.

104. Ghawi, R., & Pfeffer, J. (2019). Efficient Hyperparameter Tuning with Grid Search for Text Categorization using kNN Approach with BM25 Similarity, Open Computer Science, 9(1), 160-180. doi: .

105. Bergstra J, Bengio Y. Random Search for HyperParameter Optimization. J Mach Learn Res. 2012;13:281-305. doi:10.1162/153244303322533223.

106. R. G. Mantovani, A. L. D. Rossi, J. Vanschoren, B. Bischl and A. C. P. L. F. de Carvalho, "Effectiveness of Random Search in SVM hyper-parameter tuning," 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, 2015, pp. 1-8, doi: 10.1109/IJCNN.2015.7280664.

107. Bergstra, J. S., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. In Advances in neural information processing systems (pp. 2546-2554).

108. T. T. Joy, S. Rana, S. Gupta and S. Venkatesh, "Hyperparameter tuning for big data using Bayesian optimisation," 2016 23rd International Conference on Pattern Recognition (ICPR), Cancun, 2016, pp. 2574-2579, doi: 10.1109/ICPR.2016.7900023.

109. Steyerberg EW, Harrell FE, Borsboom GJJM, Eijkemans MJC, Vergouwe Y, Habbema JDF. Internal validation of predictive models: Efficiency of some procedures for logistic regression analysis. J Clin Epidemiol. 2001;54(8):774-781. doi:10.1016/S0895-4356(01)00341-9.

110. Khair, U., Fahmi, H., Al Hakim, S., & Rahim, R. (2017, December). Forecasting error calculation with mean absolute deviation and mean absolute percentage error. In Journal of Physics: Conference Series (Vol. 930, No. 1, p. 012002). IOP Publishing.

111. Hagiwara, K., Toda, N., & Usui, S. (1993, October). On the problem of applying AIC to determine the structure of a layered feedforward neural network. In Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan) (Vol. 3, pp. 2263-2266). IEEE.

112. Wikipedia contributors. (2020, July 10). Sensitivity and specificity. In Wikipedia, The Free Encyclopedia. Accessed 21 July 2020.

.

113. 1. Liu VX, Bates DW, Wiens J, Shah NH. The number needed to benefit: estimating the value of predictive analytics in healthcare. J Am Med Inform Assoc. 2019;26(12):1655-1659. doi:10.1093/jamia/ocz088.

114. Saito T, Rehmsmeier M. The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLoS One. 2015;10(3):1-21. doi:10.1371/journal.pone.0118432.

115. Van Calster B, McLernon DJ, Van Smeden M, et al. Calibration: The Achilles heel of predictive analytics.

BMC Med. 2019;17(1):1-7. doi:10.1186/s12916-019-1466-7.

116. Davis SE, Lasko TA, Chen G, Siew ED, Matheny ME. Calibration drift in regression and machine learning models for acute kidney injury. J Am Med Informatics Assoc. 2017;24(6):1052-1061. doi:10.1093/jamia/ocx030.

117. Walsh CG, Sharman K, Hripcsak G. Beyond discrimination: A comparison of calibration methods and clinical usefulness of predictive models of readmission risk. J Biomed Inform. 2017;76(October):9-18. doi:10.1016/j.jbi.2017.10.008.

118. Huang Y, Li W, Macheret F, Gabriel RA, Ohno-Machado L. A tutorial on calibration measurements and calibration models for clinical prediction models. J Am Med Inform Assoc. 2020;27(4):621-633. doi:10.1093/jamia/ocz228.

119. Stevens RJ, Poppe KK. Validation of clinical prediction models: what does the "calibration slope" really measure? J Clin Epidemiol. 2020;118:93-99. doi:10.1016/j.jclinepi.2019.09.016.

120. Hansen, C. (30 May 2019). Nested Cross-Validation Python Code. MLFromScratch. Accessed 21 July 2020. .

121. Miao Y, Francisco S, Boscardin WJ. SAS Global Forum 2013. Statistics and Data Analysis Estimating Harrell's Optimism on Predictive Indices Using Bootstrap Samples. 2013; 504:1-12.

122. Johnson AEW, Pollard TJ, Shen L, et al. MIMIC-III, a freely accessible critical care database (version 1.4). Sci data. 2016;3:160035.

123. Libraries.io (2017) Pypi. https://libraries.io/pypi. (accessed on 09/08/2018).

124. Matthew C Lenert, Jeffrey Blume, Thomas Lasko, Michael Matheny, Asli Weitkamp, Colin G Walsh. "Deep Aion Project: Longitudinal Data Generator". version 0.2.3.

.

125. StataCorp. 2019. Stata Statistical Software: Release 16. College Station, TX: StataCorp LLC.

126. Fernando Pérez and Brian E. Granger. IPython: A System for Interactive Scientific Computing, Computing in Science & Engineering, 9, 21-29 (2007), doi:10.1109/MCSE.2007.53.

127. Kluyver T, Ragan-kelley B, Pérez F, et al. Jupyter Notebooks—a publishing format for reproducible computational workflows. Position Power Acad Publ Play Agents Agendas. 2016:87-90. doi:10.3233/978-1-61499-649-1-87.

128. Submlime HQ Pty LTd. Sublime Text version 2. Woollahara, Australia. .

129. John D. Hunter. Matplotlib: A 2D Graphics Environment, Computing in Science & Engineering, 9, 90-95 (2007), doi:10.1109/MCSE.2007.55.

130. Stéfan van der Walt, S. Chris Colbert and Gaël Varoquaux. The NumPy Array: A Structure for Efficient Numerical Computation, Computing in Science & Engineering, 13, 22-30 (2011), doi:10.1109/MCSE.2011.37.

131. Wes McKinney. Data Structures for Statistical Computing in Python, Proceedings of the 9th Python in Science Conference, 51-56 (2010).

132. Travis E. Oliphant. Python for Scientific Computing, Computing in Science & Engineering, 9, 10-20 (2007), doi:10.1109/MCSE.2007.58.

133. Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay. Scikit-learn: Machine Learning in Python, Journal of Machine Learning Research, 12, 2825-2830 (2011).

134. Guttorp, P., & Gneiting, T. (2006). Studies in the history of probability and statistics XLIX On the Matern correlation family. Biometrika, 93(4), 989-995.

135. Avorn J. The Psychology of Clinical Decision Making — Implications for Medication Use. N Engl J Med. 2018;378(8):689-691. doi:10.1056/NEJMp1002530.

136. Roy SK, Hom J, Mackey L, Shah N, Chen JH. Predicting Low Information Laboratory Diagnostic Tests. AMIA Jt Summits Transl Sci proceedings AMIA Jt Summits Transl Sci. 2018;2017:217-226.

.

137. Agresti, A., & Coull, B. A. (1998). Approximate is better than “exact” for interval estimation of binomial proportions. The American Statistician, 52(2), 119-126.

138. Lasko TA. Nonstationary Gaussian Process Regression for Evaluating Clinical Laboratory Test Sampling Strategies. Proc Twenty-Ninth AAAI Conf Artif Intell N. 2015:1777-1783.

139. VanHouten, J. P. (2016). A Modified Random Forest Kernel for Highly Nonstationary Gaussian Process Regression with Application to Clinical Data (Doctoral dissertation, Vanderbilt University).

140. Pearl J, Mackenzie D. The Book of Why : The New Science of Cause and Effect.; 2018.

141. Vim (text editor). (2020, July 21). In Wikipedia, The Free Encyclopedia. Accessed 29 July 2020. https://en.wikipedia.org/w/index.php?title=Vim_(text_editor)&oldid=813716105.

142. Summary Describing the ACCRE Facility. Vanderbilt University. Accessed 29 July 2020. https://www.vanderbilt.edu/accre/pi/grant-text/#acknowledging-accre-in-publications.

143. Data Science Mini Grants. Vanderbilt University. Accessed 8 Jan. 2020. https://www.vanderbilt.edu/datascience/data-science-mini-grants/.

144. RStudio Team (2020). RStudio: Integrated Development for R. RStudio, PBC, Boston, MA. http://www.rstudio.com/.

145. Varoquaux, G. Joblib: running Python functions as pipeline jobs. Version 0.15.1. https://joblib.readthedocs.io/en/latest/.

146. Paszke, A, Gross, S, Massa, F, Lerer, A, Bradbury, J, Chanan, G, … Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d extquotesingle Alch&#39;e-Buc, E. Fox, & R. Garnett (Eds.), Advances in Neural Information Processing Systems 32 (pp. 8024–8035).

147. Kuhn, M. (2008). Building Predictive Models in R Using the caret Package. Journal of Statistical Software, 28(5), 1 - 26. doi:http://dx.doi.org/10.18637/jss.v028.i05.

148. Hadley Wickham, Romain François, Lionel Henry and Kirill Müller (2018). dplyr: A Grammar of Data Manipulation. R package version 0.8.3. https://CRAN.R-project.org/package=dplyr.

149. Calaway R, Weston S (2011). doMC. R package version 1.3.6. https://CRAN.R-project.org/package=doMC.

150. Bates D, Mächler M, Bolker B, Walker S (2015). “Fitting Linear Mixed-Effects Models Using lme4.” Journal of Statistical Software, 67(1), 1–48. doi: 10.18637/jss.v067.i01.

151. Pinheiro J, Bates D, DebRoy S, Sarkar D, R Core Team (2020). nlme: Linear and Nonlinear Mixed Effects Models. R package version 3.1-148, https://CRAN.R-project.org/package=nlme.

152. Robin X, Turck N, Hainard A, Tiberti N, Lisacek F, et al. pROC. R package version 1.16.2. https://cran.r-project.org/web/packages/pROC/index.html.

153. Blume JD, Mcgowan LDA, Dupont WD, Greevy RA. Second-generation p-values: Improved rigor, reproducibility, & transparency in statistical analyses. PLoS One. 2018:1-17. doi:10.1371/journal.pone.0188299

154. Poggio, T., Kawaguchi, K., Liao, Q., Miranda, B., Rosasco, L., Boix, X., ... & Mhaskar, H. (2018). Theory of deep learning iii: the non-overfitting puzzle. *CBMM Memo. 073*.

155. Ba, J., & Caruana, R. (2014). Do deep nets really need to be deep? In Advances in neural information processing systems (pp. 2654-2662).

156. Deep Learning Performance Documentation. The NVIDIA Corporation. Accessed 18 August 2019. https://docs.nvidia.com/deeplearning/performance/index.html.

157. White, H. 1982. "Maximum Likelihood Estimation of Misspecified Models,". Econometrica, 50: 1–25.

158. Baytas IM, Xiao C, Zhang X, Wang F, Jain AK, Zhou J. Patient Subtyping via Time-Aware LSTM Networks. Proc 23rd ACM SIGKDD Int Conf Knowl Discov Data Min - KDD '17. 2017:65-74. doi:10.1145/3097983.3097997.

159. Leary, T. S., Milner, Q. J. W., & Niblett, D. J. (2000). The accuracy of the estimation of body weight and height in the intensive care unit. European journal of anaesthesiology, 17(11), 698-703.

160. Maskin, L. P., Attie, S., Setten, M., Rodriguez, P. O., Bonelli, I., Stryjewski, M. E., & Valentini, R. (2010). Accuracy of weight and height estimation in an intensive care unit. Anaesthesia and intensive care, 38(5), 930-934.

161. Young, M. (2006). Estimated versus measured height and weight in the intensive care unit: How do ICU clinicians measure up?. Critical care medicine, 34(8), 2251-2252.

162. Tang, H., & Glass, J. (2018, December). On training recurrent networks with truncated backpropagation through time in speech recognition. In 2018 IEEE Spoken Language Technology Workshop (SLT) (pp. 48-55). IEEE.

163. Peterson ED, Coombs LP, Ferguson TB, et al. Hospital variability in length of stay after coronary artery bypass surgery: Results from the Society of Thoracic Surgeon's National Cardiac Database. Ann Thorac Surg. 2002;74(2):464-473. doi:10.1016/S0003-4975(02)03694-9.

164. Dimick JB, Pronovost PJ, Heitmiller RF, Lipsett PA. Intensive care unit physician staffing is associated with decreased length of stay, hospital cost, and complications after esophageal resection. Crit CARE Med. 2001;29(4):753-758.

165. Riguzzi C, Hern HG, Vahidnia F, Herring A, Alter H. The july effect: is emergency department length of stay greater at the beginning of the hospital academic year? West J Emerg Med. 2014;15(1):88-93. doi:10.5811/westjem.2013.10.18123.

166. Taheri P a, Butz D a, Greenfield LJ. Length of stay has minimal impact on the cost of hospital admission. J Am Coll Surg. 2000;191(2):123-130. doi:10.1016/S1072-7515(00)00352-5.

167. Razavi SA, Johnson J-O, Kassin MT, Applegate KE. The impact of introducing a no oral contrast abdominopelvic CT examination (NOCAPE) pathway on radiology turn around times, emergency department length of stay, and patient safety. Emerg Radiol. 2014;21(6):605-613. doi:10.1007/s10140-014-1240-2.

168. Singh S, Fletcher KE. A qualitative evaluation of geographical localization of hospitalists: How unintended consequences may impact quality. J Gen Intern Med. 2014;29(7):1009-1016. doi:10.1007/s11606-014-2780-6.

169. Johnson, A. E., Pollard, T. J. & Mark, R. G. Reproducibility in critical care: a mortality prediction case study. In Machine Learning for Healthcare Conference, 361–376 (2017).