Title
Causal Inference for Personalized Educational Systems

Author
Tadayon, Manie

Publication Date
2021

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Causal Inference for Personalized Educational Systems

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Electrical and Computer Engineering

by

Manie Tadayon

2021

ABSTRACT OF THE DISSERTATION

Causal Inference for Personalized Educational Systems

by

Manie Tadayon
Doctor of Philosophy in Electrical and Computer Engineering
University of California, Los Angeles, 2021
Professor Gregory J. Pottie, Chair

Educational systems have traditionally been evaluated using cross-sectional studies, namely, examining a pretest, posttest, and single intervention. Although this is a popular approach in education, it does not model valuable information such as confounding variables, feedback to students, and other real-world deviations of studies from ideal conditions. Moreover, learning inherently is a sequential process and should involve a sequence of interventions. Nowadays, due to the availability of a large volume of educational data, researchers can develop more intelligent inference algorithms.

We propose to exploit the rich features in time series data and use them to develop more intelligent and individualized educational systems. Our approach is five-fold: First, we model the sequential nature of education using hidden Markov models and show that analysis of a sequence of student actions is predictive of posttest results. Second, we propose more intelligent experimental designs by collecting richer data from students by including questions on potential confounders in the diagnostic test and instructor interventions during office hours. Third, we propose various experimental and quasi-experimental designs for educational systems and quantify them using the graphical model and directed acyclic (DAG) graph language. We discuss the application and limitations of each method in education. Fourth, we propose to model the education system as time-varying treatments, confounders, and time-varying treatments-confounders feedback. We

show that if we control for a sufficient set of confounders and use appropriate inference techniques such as the inverse probability of treatment weighting (IPTW) or g-formula, we can close the back-door paths and derive the unbiased causal estimate of joint interventions on the outcome. Fifth, we compare the g-formula and IPTW performance and discuss the pros and cons of using each method.

The dissertation of Manie Tadayon is approved.

Abeer A. Alwan

Jonathan chau-yan Kao

Vwani P. Roychowdhury

Gregory J. Pottie, Committee Chair

University of California, Los Angeles

2021

To my parents, my brother

and all my good friends that support me in this journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

A | B | C | D | H | I | L | M | P | R | S

A

AIC Akaike Information Criterion.
AR Autoregressive.
ARIMA autoregressive integrated moving average.
ARMA autoregressive moving average.

B

BIC Bayesian Information Criterion.
BKT Bayesian knowledge tracing.

C

CHMM continuous hidden Markov model.
CPDs conditional probability tables.
CV Cross-Validation.

D

DAG directed acyclic graph.
DBN dynamic Bayesian network.
DHMM discrete hidden Markov model.

H

HMM hidden Markov model.

I

ITS intelligent tutoring system.

L

LSTM long short term memory.

M

MA Moving Average.

P

PO Potential Outcome.

R

RNN recurrent neural network.

S

SCM Structural causal model.
SP Save Patch.

LIST OF SYMBOLS

A | R

A

α Smoothing Parameter.

R

ϵ Residual/error term.

ACKNOWLEDGEMENTS

During my graduate journey at UCLA, I met many amazing and inspiring people from whom I learned so much. I want to thank and acknowledge them for all their help and supports.

First of all, I would like to express my sincere gratitude to my advisor, Professor Pottie, for his invaluable advice and continuous support during the Ph.D. program. I would also like to thank my committee members, Professor Jonathan Kao, Professor Vwani Roychowdhury, and Professor Abeer Alwan, for providing technical support during my study.

I would like to thank the faculties whom I had the honor to be their TA's, especially Professor Lorenzelli, whose dedication and work ethic are always in my mind.

I would like to thank everyone at Jet Propulsion Laboratory (JPL) and Qualcomm, especially my managers Yumi Iwashita, Curtis Padgett, Norman Lay, Arby Argueta, and Greg Steele that teach me so much about conducting original research in the industry.

I would like to thank my labmates, Sunny Bhat and Jeffery Jiang, for their support and all the good memories.

I would like to thank the following organizations for funding my graduate program, UCLA Graduate Division, to provide the departmental fellowship and the TAship support, Office of Naval Research (ONR), and National Science Foundation (NSF).

I would like to thank my wonderful friends for all the fantastic memories; I would especially like to thank Behnam Khaki for sharing his knowledge and wisdom on conducting better research. I thank him for all the great discussions and memories in the dining halls and UCLA campus. I thank Firouz Shahriarian for continuous support and philosophical advice on living a better life and coping with stress during difficult times. I thank Hamed Nilchi for being a very supportive friend and organizing all the wonderful events at UCLA. I thank Ramin Essalat for being a supportive and caring friend and sharing his experience and knowledge on different technical and non-technical matters. I would like to thank Mohammad Ali Panahi for all his great advice in the Engineering IV hallways and all the motivational phone calls during the Covid-19 pandemic. Thank you, Mo-

hammad Ali, for keeping me motivated. I would like to thank Alireza Asadi and Soroush Souri for being very supportive friends during my entire UCLA journey. I would like to thank Sajjad Satvati, Hamidreza Motaman, and Morteza Hashemi for being such amazing friends since my childhood and bring so many good memories to my life. I would like to thank Hossein Razavi, who was one of the most supportive and respectable students I have seen at UCLA. I would like to thank Borna Zandkarimi for his technical and philosophical discussions at UCLA. I thank him for all the good memories during different events at UCLA. I would like to thank Shadi Mohagheghi for her exceptional support and help during my journey at UCLA. Without her support during the difficult and stressful times, I would not be where I am right now.

Finally, I would like to express my gratitude to my parents, my brother Macan, my aunts Akram and Mina, Tahereh, and my cousins Masstooreh, Yousef, Ali, Mehrdad, Arash, etc. Without their tremendous understanding and encouragement, it would be impossible for me to complete my study. I would especially thank my mother for all her emotional support during my life. It is so sad she passed away from cancer in 2014 and could not see what she always dreamt of seeing me achieving. Thank you, mom, I learned a great deal from you, and you will always remain my role model.

VITA

2012-2015	B.S. Electrical and Computer Engineering, University of California Los Angeles
2015-2017	M.S. Electrical and Computer Engineering, University of California Los Angeles

PROFESSIONAL EXPERIENCE

2015	Jet Propulsion Laboratory, Signal Processing Intern
2016	Jet Propulsion Laboratory, Signal Processing Intern
2017	Qualcomm, Signal Processing Intern
2019	HRL, Machine Learning Intern
2019-2020	Jet Propulsion Laboratory, Machine Learning Intern

PUBLICATIONS

M. Tadayon, G. Pottie, "Predicting student performance in an educational game using a hidden markov model", IEEE Transactions on Education, 2020.

M. Tadayon, G. Pottie, "tsBNgen: A Python Library to Generate Time Series Data from an Arbitrary Dynamic Bayesian Network Structure", arXiv preprint, 2020.

M. Tadayon, G. Pottie, "Comparative Analysis of the Hidden Markov Model and LSTM: A Simulative Approach", arXiv preprint, 2020.

M. Tadayon, Y. Iwashita, "A clustering approach to time series forecasting using neural networks: A comparative study on distance-based vs. feature-based clustering methods", arXiv preprint,

2021.

M. Tadayon, G. Pottie, "Causal Inference in Educational Systems: A Graphical Modeling Approach", arXiv preprint, 2021.

Chapter 1

INTRODUCTION

Educational systems have traditionally been evaluated using cross-sectional studies, namely, examining a pretest and posttest. Considerable research such as [1] and [2] has been done in designing and analyzing the pretest and posttest. Although this is a popular approach, it does not model valuable information such as confounding variables, feedback to students, and other real-world deviations of studies from ideal conditions. Moreover, it does not exploit rich information available in time series data. Also, learning inherently is a sequential process and should involve a sequence of interventions. Nowadays, a large volume of educational data is available, allowing researchers to develop more intelligent and more complex modeling and inference algorithms. Numerous ideas such as intelligent tutoring systems (ITS), Bayesian knowledge tracing (BKT), dynamic Bayesian networks, and hidden Markov models (HMMs) have been proposed to overcome the limitations of current educational systems.

1.1 Hidden Markov Model

The hidden Markov model (HMM) is a popular method to model the time series data because of its rich mathematical structure and the availability of many practical algorithms for computing model components [3]. Numerous papers such as [3], [4], [5], about the HMM applications in speech and the stock market have been published. There has also been some research done in recent years about applications of the HMM in education [6], [7]. In [6], the authors used an HMM to predict student performance in educational video games. They exploit and extract the rich information and features in time series data to develop more intelligent models in education. Dynamic Bayesian networks are the generalization of HMMs, which have numerous applications in the real world.

DBNs has been successfully used in education [8], [9], and [10].

1.2 Causality

Although ITS, BKT, and HMM have been successfully used in education, they have many limitations, such as: 1- they do not model multiple interventions thorough time. 2- They do not model time-varying confounders. 3- They do not model time-varying treatments and confounders feedback. Our objective is to have a tractable approach for the actual educational system by accounting for all the potential confounders, and design suitable interventions to reduce the negative influence of confounders through time. To design intelligent and individualized educational systems, one must incorporate all the parameters and factors that can influence the system in the model and try to collect rich data for these factors through suitable experimental and quasi-experimental designs. The next step is to develop suitable algorithms for inference and learning the parameters of the model. Finally, the last step is to test the model and modify it accordingly. Causal inference is a framework that allows one to query about all this necessary information. It has been used successfully in various domains such as econometric, epidemiology and education [11–13].

The potential outcome and Pearl's Structural causal model (SCM) are two well-known frameworks for causal inference. SCM is a popular framework that uses a directed acyclic graph (DAG) to model real-world phenomena [14]. DAGs have been used in many domains [15], [16], and [17]. In [17], the authors used dynamic Bayesian networks to generate synthetic time series data. In [13], we incorporated the potential outcome and causal DAGs in education. We analyzed experimental, quasi-experimental designs with a single intervention and quasi-experimental designs with multiple interventions. We borrowed tools from various domains such as econometric and epidemiology to derive unbiased causal estimates. The use of causal DAGs provides the following advantages: 1- They explain the data generating process. 2- They demonstrate the d-separation and variables independence using graphs. 3- They provide easy and efficient algorithms for inference and learning. In this dissertation, we frequently use DAGs and explain various models using graphical modeling language.

1.3 The Structure of this Dissertation

This dissertation is organized as follows: Chapter 2 provides the background and prerequisite knowledge needed for other chapters. Chapter 3 examines sequential modeling and inference using algorithms such as HMMs in educational systems. The HMM was our first algorithm to model time series, and it was quite successful in predicting student trajectory in the game. Chapter 4 discusses tsbngen, a Python package to generate synthetic time series data using arbitrary dynamic Bayesian network structures. The motivation behind this package is to generate data for more complex structures than the HMM to approximate real-world situations. Chapter 5 provides a comprehensive simulative comparison of both discrete and continuous HMM with long short term memory (LSTM). The results indicate that the HMM can outperform the LSTM when the amount of data is limited. Chapter 6 discusses causal inference in education systems and provides a detailed treatment of experimental and quasi-experimental designs in educational systems. It breaks from approximating with HMMs to determine what kinds of measurements must be made for a causal model to explain data and guide the sequence of interventions. It also discusses various algorithms to model quasi-experimental designs with multiple interventions. Chapter 7 provides the conclusion and final remarks.

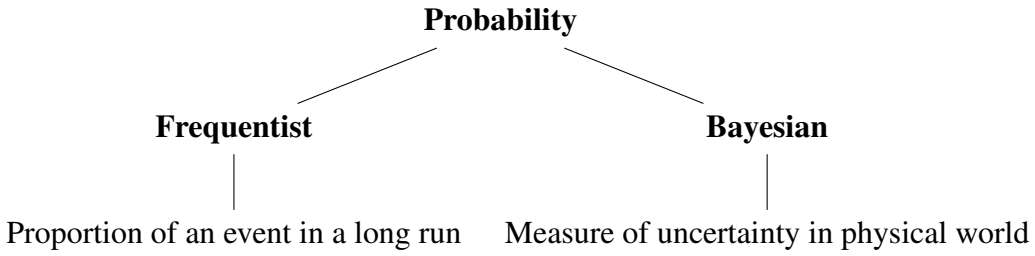
Chapter 2

BACKGROUND

In this section, I discuss the background and prerequisite knowledge used in other chapters.

2.1 Statistical Concepts

Probability: There are two popular schools of thought to define the probability:



The Bayesian approach assigns a probability measure to each event (prior probability) and updates this probability with observing data (posterior probability). The frequentist approach is mainly concerned with the likelihood function as opposed to the posterior distribution. To better distinguish between the frequentist and Bayesian approach consider the following theorem:

$$P(x|D) = \frac{P(D|x)P(x)}{P(D)} \tag{2.1}$$

The equation 2.1 is referred to as the Bayes theorem, which states the relationship between likelihood and posterior probability. The $P(x)$ is known as the prior probability, the $P(D|x)$ is called the likelihood function, and $P(x|D)$ is the posterior probability.

p-value: The p-value is a measure of the probability that an observed difference could have occurred just by random chance. The lower the p-value, the greater the statistical significance of the observed difference. The p-value is the evidence against the null hypothesis: the smaller the p-value, the stronger the evidence against the null hypothesis (Rejecting the null hypothesis).

For example, reject the null hypothesis if p-value $\leq \alpha$ and fail to reject the null hypothesis if p-value $\geq \alpha$. The p-value is a common technique in frequentist driven studies.

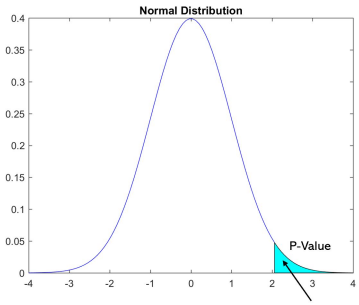


Figure 2.1: p-value, probability of the shaded area

Regression to the Mean: Formally, it says if a random variable is an extreme or outlier in a first observation, it will be more towards the average for future observations.

Table 2.1 depicts various data types in real-world applications:

Table 2.1: Various Data Type

Data Type	Definition
Cross-Sectional	Multiple individuals at the same time.
Time Series	Single individual at multiple points in time.
Panel Data (Longitudinal Data)	Multiple individuals at multiple time points.

2.2 Time Series

White noise: Each element in a time series is a random draw from a population with zero mean and constant variance.

Autoregressive (AR): AR(p) is an AR model with p lags. They are the models in which the value of a time series at time t is related to its previous values.

$$AR(1) : y_t = \mu + \gamma y_{t-1} + \epsilon_t \tag{2.2}$$

$$AR(p) : y_t = \mu + \sum_{i=1}^{i=p} \gamma_i y_{t-i} + \epsilon_t \tag{2.3}$$

where μ and γ are constant values.

Moving Average (MA): MA(q) is a MA model with q lags. They are the models in which the

value of a time series at time t is expressed as a linear combination of previous error terms.

$$MA(1) : y_t = \mu + \theta\epsilon_{t-1} + \epsilon_t \tag{2.4}$$

$$MA(q) : y_t = \mu + \sum_{i=1}^{i=q} \theta_i y_{t-i} + \epsilon_t \tag{2.5}$$

Autoregressive moving average (ARMA): ARMA(p,q) combines the AR(p) and MA(q) processes as follows:

$$ARMA(p,q) : y_t = \mu + \sum_{i=1}^{i=p} \theta_i y_{t-i} + \sum_{j=1}^{j=q} \phi_j \epsilon_{t-j} + \epsilon_t \tag{2.6}$$

Stationary: A stationary process has a mean and variance that do not change over time.

De-trending: A variable can be detrended by regressing the variable on a time trend and obtaining the residual.

$$y_t = \mu + \beta t + \epsilon_t \tag{2.7}$$

$$\hat{\epsilon_t} = y_t - \hat{\mu} - \hat{\beta} t \tag{2.8}$$

Differencing: It is a process of subtracting a current value of a time series from its previous values. Equation 2.9 is an example of first order differencing where the current value of the time series at time t is subtracted from the value at time $t-1$. Differencing is a simple yet very powerful tool to remove the trend and seasonality in the time series.

$$\Delta y_t = y_t - y_{t-1} \tag{2.9}$$

Equation 2.9 can iteratively be used to make the time series stationary.

Auto-correlation Function (ACF): Ratio of the autocovariance of y_t and y_{t-k} to the variance of dependent variable y_t.

$$ACF(k) = \rho_k = \frac{\mathrm{Cov(y_t, y_{t-k})}}{\mathrm{Var(y_t)}} \tag{2.10}$$

Partial Auto-correlation Function (PACF): It is a correlation between y_t and y_{t-k} for different values of k minus the part explained by the intervening lags.

2.2.1 Time Series Forecasting Techniques

Suppose y_t is time series of length T denoted by $y_1, y_2, ..., y_T$. The forecasting or prediction is a problem of estimating the future value of y_t given observed values up to time T.

Naive: The future value of time series is equal to the last observed value of the time series.

$$\hat{y} = y_T \tag{2.11}$$

Moving Average: The future value of the time series is equal to the average values of the time series up to time T.

$$\hat{y} = \frac{1}{T}\sum_{t=1}^{t=T} y_t \tag{2.12}$$

Simple Exponential Smoothing: The future value of a time series is equal to the weighted average of all the observations up to time T. The weights decrease exponentially and are controlled by smoothing parameter α. α is a parameter and is selected based on how important are the past values compared to the more recent values in a given time series. For example Table 2.2 shows the single exponential smoothing coefficients for the five most recent values in a time series. As Table 2.2 shows, a smaller value of α puts more weight on the more distant values and a larger α puts more weight on the more recent values of a time series. The naive and moving average methods are both special cases when α is one and when weights are all the same.

Table 2.2: Smoothing Coefficients for various α

	$\alpha = 0.05$	$\alpha = 0.1$	$\alpha = 0.5$	$\alpha = 0.9$
y_T	0.05	0.1	0.5	0.9
y_{T-1}	0.0475	0.09	0.25	0.09
y_{T-2}	0.0451	0. 0.081	0.125	0.009
y_{T-3}	0.0429	0.0729	0.0625	0.0009
y_{T-4}	0.0407	0.06561	0.03125	0.00009

$$\hat{y} = \alpha y_T + \alpha(1-\alpha)y_{T-1} + \alpha(1-\alpha)^2 y_{T-2} + \cdots \tag{2.13}$$

Regression: Express the dependent variable as a linear combination of predictors. Suppose the predictors are $X_{1,t}, X_{2,t}, ...X_{k,t}$ then, mathematically this can be represented as follows:

$$y_t = \beta_0 + \beta_1 X_{1,t} + \beta_2 X_{2,t} + ... + \beta_k X_{k,t} + \epsilon_t \tag{2.14}$$

The regression model should satisfy the Gauss-Markov assumptions. This implies that the error term ϵ is independent and uncorrelated at different points in time. The covariance between the error term and the independent predictors is zero and the error term is zero mean.

$$E[\epsilon_t] = 0 \tag{2.15}$$

$$Cov(\epsilon_t, X_{l,t}) = 0 \qquad \forall l = 1, 2, ...k \tag{2.16}$$

Autoregressive integrated moving average (ARIMA): ARIMA(p,q,d) is defined as applying the ARMA(p,q) model to d-differenced time series (time series after performing equation 2.9 d times). ARIMA is a powerful univariate time series forecasting technique, which is used when the residual time series exhibits autocorrelation at different points in time.

2.3 Hidden Markov Model

In this section, HMM algorithms are briefly reviewed. Both the discrete hidden Markov model (DHMM) as well as the continuous hidden Markov model (CHMM) are discussed. The HMM is the extension of the Markov process in which the observations are a probabilistic function of the states. In an HMM, states are considered as hidden and should be inferred by the sequence of observations. The HMM is characterized by the following:

N: Number of the hidden states. Although this is unknown since the states are hidden, it usually can be initialized to a reasonable number depending on the problem and the dataset and later can be learned using various statistical analysis tools which will be discussed later.

M: Number of the observation symbols per state.

$\overrightarrow{S}$: State sequence where $\overrightarrow{S} = (s_1, s_2, ...s_T)$, T is the length of the sequence, and each $s_i \in \{1, 2, ..., N\}$.

$\overrightarrow{O}$: sequence of the observation symbols where $\overrightarrow{O} = (o_1, o_2, ...o_T)$ and each $o_i \in \{1, 2, ..., M\}$.

A: State transition probability. It defines the probability of going from state i to the state j and is denoted by

$$a_{ij} = p(s_{t+1} = j | s_t = i) \tag{2.17}$$

B: Observation distribution per each state, which is denoted as follows:

$$b_i(k) = p(o_t = k | s_t = i) \tag{2.18}$$

$\overrightarrow{\pi}$: Initial state distribution that is defined as follows:

$$\overrightarrow{\pi_i} = p(s_1 = i) \tag{2.19}$$

λ: HMM parameters together are usually denoted by the following:

$$\lambda = (A, B, \overrightarrow{\pi}) \tag{2.20}$$

The above equations together can be used to fully define any HMM with discrete observations. Forward and backward algorithms [3] are used to calculate $P(\overrightarrow{O}|\lambda)$, the probability of observing a sequence given λ. If the time series is not labeled and the mapping between the observations and the states is not available, then HMM parameters should be estimated using the Baum-Welch or EM algorithm [18]. If the observations are continuous (CHMM) as opposed to discrete, the emission probability distribution should be adjusted to account for this change. Continuous observations are modeled by fitting the probability density functions (pdf) to the data. A Gaussian distribution or mixture of Gaussian distributions are typically used for modeling the data. If the observations for each state can be modeled using a single Gaussian distribution, then equation 2.18 will be changed to the following:

$$b_i(x) = p(x | s_t = i) = N(x; \mu_i, \Sigma_i) \tag{2.21}$$

In equation 2.21, μ_i and Σ_i are the mean and the covariance matrix of the Gaussian distribution for state i respectively. If a single distribution is not a reasonable fit to the data, then a mixture of Gaussian distributions can be used to model the observations. In this case, equation 2.22 can be

used to model the observations for each state.

$$b_i(x) = p(x|s_t = i) = \sum_{m=1}^{M} c_{im} N(x; \mu_{im}, \Sigma_{im}) \tag{2.22}$$

c_{im} is the mixture coefficient and determines the weight each component has in modeling the data. μ_{im} and Σ_{im} are the mean and covariance matrices of each mixture component corresponding to the state i. Decoding the optimal state sequence given the observation can be done using the Viterbi algorithm [19]. It finds the sequence of the states that best explains the observed data:

$$S^* = \textit{argmax}_S P(S|O, \lambda) \tag{2.23}$$

2.4 Model Selection

Model selection is the process of choosing between different machine learning approaches, or choosing between different hyperparameters or sets of features for the same machine learning approach. Some popular metrics used to choose an optimal model:

- Minimizing the test error.
- Maximizing the log likelihood.
- Minimizing the misclassification error.

Some well-known methods for model selections are:

- K-fold Cross-Validation (CV)
- Bayesian Information Criterion (BIC).
- Akaike Information Criterion (AIC).

AIC and BIC are the model selection algorithms that are used to combat overfitting by introducing the penalty terms. AIC is defined by the following formula:

$$AIC = -2 \ln L + 2k \tag{2.24}$$

where $\ln L$ is the log likelihood function and k is the number of parameters in the model; therefore $2k$ is the penalty term.

BIC is another well known model selection algorithm that measures the trade off between the model fit and the complexity. The formula for the BIC is given below:

$$BIC = -2 \ln L + k \ln N \tag{2.25}$$

$\ln L$ and k are the same parameters as in AIC, and N is the number of observations. By comparing equations 2.24 and 2.25 it appears that BIC has a larger penalizing term. Therefore it penalizes the complex model more than AIC. Minimizing the test and the misclassification errors are usually the objective in the supervised learning since there is ground truth data. CV is a resampling procedure used to evaluate machine learning models on a limited data sample. K-fold is a type of CV that has a single parameter K, and K refers to the number of groups that a given data sample is to be split into. K-fold CV works as follows:

1. Shuffle the data randomly.
2. Split the data into K groups.
3. For each group:
    - Take a group as test data and the remaining K-1 groups as training data.
    - Fit a model on training data and evaluate its performance on the test data.
    - Repeat this procedure K times.
    - Take the average or combine the K results to have a score for the model.

Figure 2.2 shows the procedure the for K-fold cross validation, where in each step one fold is used for the testing and K-1 folds are used for the training.

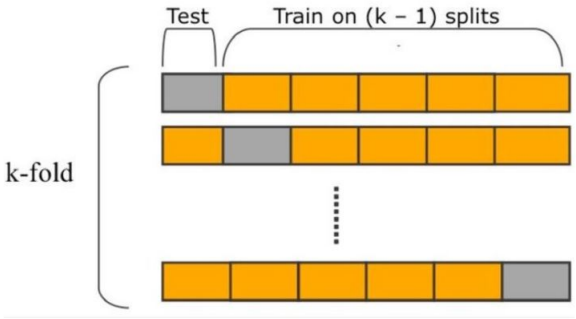


Figure 2.2: K-fold Cross Validation

2.5 Long Short Term Memory

Recurrent neural network (RNN) are mainly designed for the sequence prediction problem. Unlike feed-forward networks that consider inputs and outputs to be independent, RNNs consist of memory cells that can remember the long term dependencies between elements of a sequence. In theory, a RNN can remember arbitrarily long time steps, but in practice, they suffer from the vanishing gradient problem [20], [21]. The LSTM is designed to address the vanishing gradient problem. The first step in the operation of LSTM is the use of a cell state, which is a memory element to store information. The LSTM has the ability to remove and update the cell state via various gates. A forget gate decides what information should be kept or removed. This is done using a sigmoid function that produces a number between 0 and 1, where one means keeping and zero means removing the data. The input gate is responsible for adding or updating the information to the cell state. It consists of a component that involves a sigmoid function to determine which information should be added to the cell state. It also has a component using a tanh to squeeze the data to the -1 and 1 range. The output gate decides what information from the cell state should be passed to the output at time t. This is done similarly to the input gate in which the cell state data is squeezed to -1 and 1 range and then is multiplied by the output of a sigmoid function to determine the useful information passed to the output.

2.6 K-means Clustering

K-means is a well-known distance-based clustering method that uses the Euclidean distance to measure the similarity between data points. The inherent problem with the K-means is that the value of K should be specified in advance. This can be done by the domain knowledge or techniques such as the elbow method, Silhouette index, and gap statistics.

The elbow method works by iterating through different values of K and plots the sum of squared distances of samples to their closest cluster center. This quantity is called inertia or the sum of squared errors (SSE). Then the optimal number of clusters is selected by the value of K at the "elbow", namely the point after which the SSE/inertia starts decreasing linearly. Silhouette index is a measure of how similar a data point is to the points in its cluster compared to points in other clusters. It works by calculating the Silhouette coefficient for each instance of clustering as follows:

$$c = \frac{a - b}{max(a, b)} \tag{2.26}$$

In equation 2.26, *a* represents the mean intra-cluster distance, and *b* represents the mean nearest-cluster distance for each sample. Equation 2.26 shows that the silhouette coefficient is a number between -1 and 1 in which the number closer to 1 represents that a sample is assigned to the correct cluster and a number closer to -1 shows that the sample is assigned to the wrong cluster.

2.7 Bayesian Networks and Probabilistic Graphical Models

Bayesian networks are a class of directed graphical models which consist of nodes to denote the variables, and edges to denote the connection and relationships between the nodes. Bayesian networks do not contain loops or cycles and hence they are called directed acyclic graphs. Consider the following Bayesian network: The variable A is called the root, which is a variable that does not have any parents (any arrow into it). The variables B and C are its children since there is an edge from A to them. The variable D is the leaf node, which is a variable that has no children. The joint

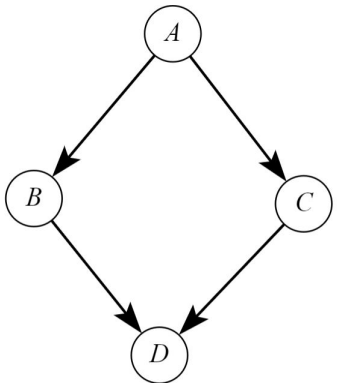


Figure 2.3: Bayesian Network

distribution on Bayesian networks respect the Markov property and is expressed as follows:

$$P(X_1, X_2, ..., X_n) = \prod_{i=1}^{i=n} P(X_i | \textit{Parents}(X_i)) \tag{2.27}$$

For example, the joint distribution for figure 2.3 is as follows:

$$P(A, B, C, D) = P(A)P(B|A)P(C|A)P(D|B, C) \tag{2.28}$$

Gaussian Bayesian Networks: It is a Bayesian network where all of its variables are continuous and all the conditional probability tables (CPDs) are linear Gaussians. Each node in a Gaussian Bayesian network is represented as follows:

$$X_j = \sum_{i \in pa(j)} \beta_{ij} X_i + \epsilon_j \tag{2.29}$$

where $\epsilon_j \sim N(0, \sigma^2)$.

Figure 2.4 shows an example of Gaussian Bayesian network.

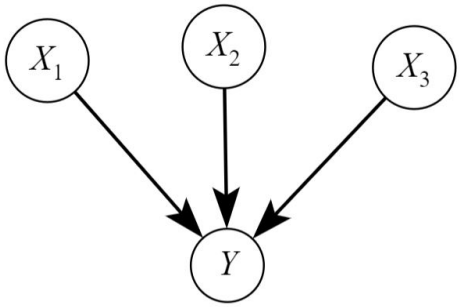


Figure 2.4: Gaussian Bayesian Network

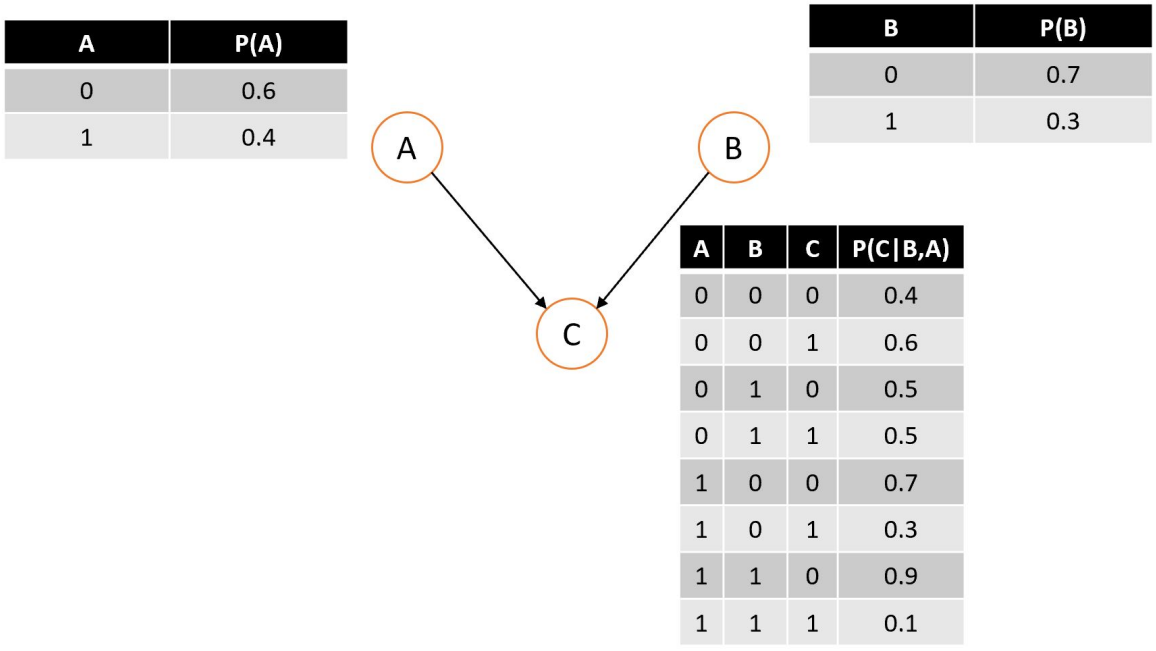
$$P(X_1) \sim N(\mu_1, \sigma_1^2) \tag{2.30}$$

$$P(X_2) \sim N(\mu_2, \sigma_2^2) \tag{2.31}$$

$$P(X_3) \sim N(\mu_3, \sigma_3^2) \tag{2.32}$$

$$P(Y|X_1, X_2, X_3) \sim N(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3, \sigma_y^2) \tag{2.33}$$

Discrete Bayesian Networks: It is a Bayesian network where all of its variables are discrete and all the CPDs are multinomial distributions. The CPDs in discrete Bayesian networks are usually represented as tables. Figure 2.5 shows an example of discrete Bayesian network, in which the variables A ,B, and C are Bernoulli random variables.



A	P(A)
0	0.6
1	0.4

B	P(B)
0	0.7
1	0.3

A	B	C	P(C|B,A)
0	0	0	0.4
0	0	1	0.6
0	1	0	0.5
0	1	1	0.5
1	0	0	0.7
1	0	1	0.3
1	1	0	0.9
1	1	1	0.1

Figure 2.5: Discrete Bayesian Network

2.8 Causal Inference

Figure 2.8 shows the popular causal inference frameworks, namely Pearl's structural and graphical models and Rubin's potential outcome (PO) model. The PO model was first discovered by

Neyman for randomized experiments and then later on extended to non-randomized settings with confounders by Rubin and Robins. Pearl shows that the there is one-to-one mapping between PO and the structural causal model.

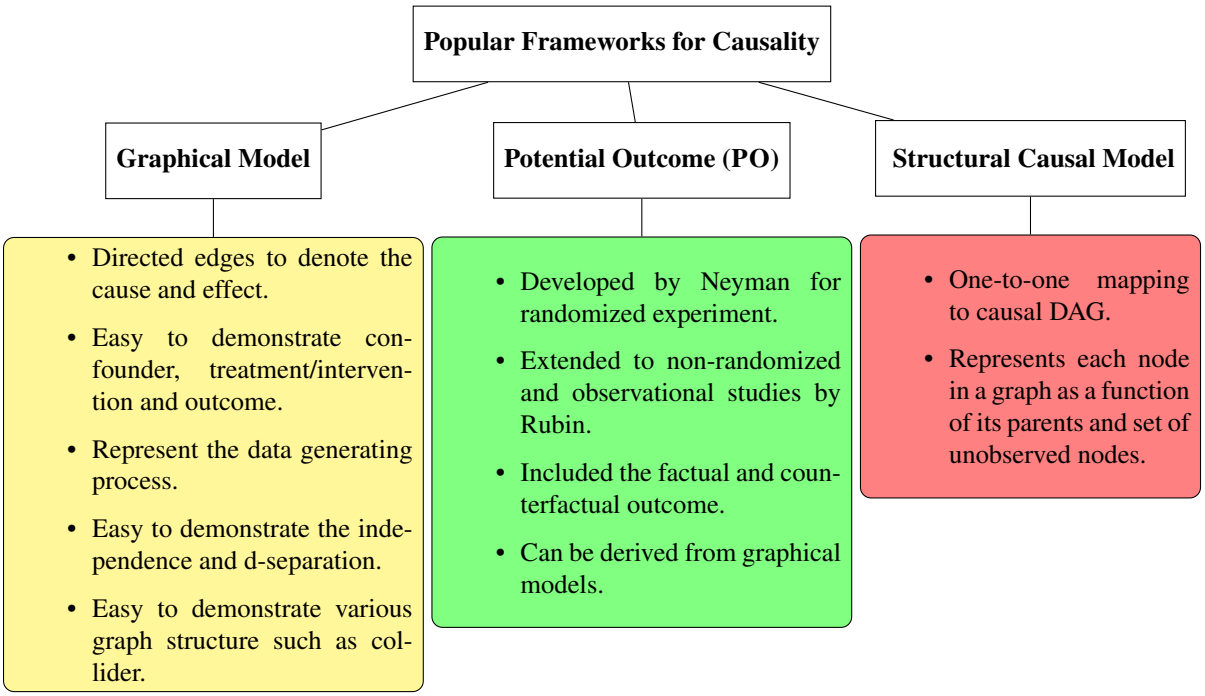


Figure 2.6: Well-known Frameworks for Causality

2.8.1 Pearl Ladder of Causation

The following are the rungs in Pearl's hierarchical ladder of causation:

- **Association**: This is the first layer in the ladder of causation corresponding to the first row of figure 2.7. This is where most of the current statistical literature lies. An example of association or correlation is the conditional probability of $P(y|x)$, which means the probability of obtaining y given we observe or see x.

- **Intervention**: This is the second layer in ladder of causation corresponding to the second row of figure 2.7. This is the layer responsible for the do calculus. An example of this is $P(y|do(x))$, which means the probability of obtaining y given we are forcing or intervening

Level (Symbol)	Typical Activity	Typical Questions
1. Association $P(y\|x)$	Seeing	What is? How would seeing X change my belief in$Y$?
2. Intervention $P(y\|do(x), z)$	Doing Intervening	What if? What if I do X?
3. Counterfactuals $P(y_x\|x', y')$	Imagining, Retrospection	Why? Was it X that caused Y? What if I had acted differently?

Figure 2.7: Pearl Ladder of Causation

$X = x$.

Note: $P(y|do(x))$ is commonly written as $P(y_x)$ (Potential outcome notation, explained later in this section).

- **Counterfactual**: This is the third layer in the ladder of causation corresponding to the third row of figure 2.7. The counterfactual output is always unobserved. For example, consider $x = 1$ denotes receiving the treatment and $x = 0$ denotes otherwise. Then $P(y_{x=1}|x = 0)$ is a counterfactual outcome since we are asking about the probability of outcome among the people who did not receive the treatment had we forced them to receive the treatment.

2.8.2 Potential Outcome (PO)

Y^x or Y_x is the outcome that would be observed if the treatment was set to X=x; in other words, it is the outcome under different possible treatment options. It consists of observed or factual and unobserved or counterfactual segments.

Assume the case of binary treatments, where $X = 1$ denotes receiving the treatment and $X = 0$ denotes otherwise. $C = c$ denotes observing participants among those that are in group c. For example, in education, c can represent students who are in a specific class, have a certain GPA, etc. Then based on the definition of the PO and the counterfactual in the last section, we can define

the following equations.

$$\text{Average Treatment Effect} = ATE = E[y_{x=1}] - E[y_{x=0}] \tag{2.34}$$

$$\text{Average Treatment Effect Among Treated} = ATT = E[y_{x=1}|X = 1] - E[y_{x=0}|X = 1] \tag{2.35}$$

$$\text{Average Treatment Effect Among Untreated} = ATU = E[y_{x=1}|X = 0] - E[y_{x=0}|X = 0] \tag{2.36}$$

$$\text{Conditional Treatment Effect} = CATE = E[y_{x=1}|C = c] - E[y_{x=0}|C = c] \tag{2.37}$$

2.8.3 Graphical Model (Causal DAG)

In this framework, the nodes and variables are represented as vertices of the graph, and directed edges denote causal relationships between them. Besides having directed edges, there is no cycle or loop in a causal DAG. The benefit of using a DAG is to define a pictorial view of the problem, to represent the data generating process, and to demonstrate independence and d-separation clearly. For example, in figure 2.8, X is the treatment, Y is the outcome, and Z is the confounder (cause of both X and Y and is placed on the backdoor path between X to Y). In causal inference, we are usually interested in finding the average causal effect of treatment on outcome. If Z were not present, then the problem would be reduced to correlational studies such that causation and correlation would be the same.

In figure 2.8, the red path $X \leftarrow Z \rightarrow Y$ is called the backdoor path, which is a non-causal path pointing to a treatment X and outcome Y. Our goal in causal inference is to block backdoor paths since they are non-causal paths and introduce bias in our analysis. To account for this backdoor path and determine unbiased causal estimate of X on Y we use the Pearl backdoor criterion as follows:

Backdoor Adjustment: If Z is a backdoor variable relative to X and Y then the causal effect of

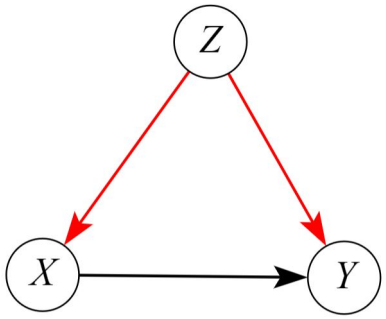


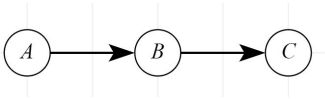
Figure 2.8: Graphical Representation of Treatment, Outcome and Confounder

X on Y is calculated by:

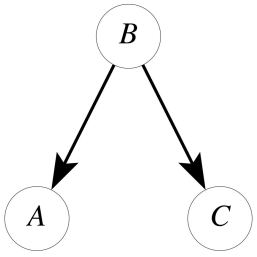
$$P(Y|do(X)) = \sum_{Z} P(Y|X,Z)P(Z) \tag{2.38}$$

2.8.4 Structures in DAG

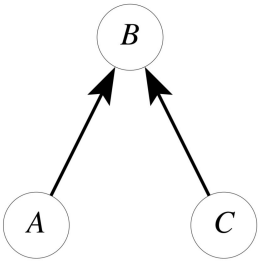
Figure 2.9 shows various independence structures on DAGs. Figures 2.9a and 2.9b are structures in which A and C are d-separated conditioned on B. However 2.9c denotes a collider structure, where A and C are independent but they will be dependent conditioned on B. Conditioning on B results in what is known as a collider stratification bias.



(a) $A \perp\!\!\!\perp C|B$



(b) $A \perp\!\!\!\perp C|B$



(c) $A \perp\!\!\!\perp C$

Figure 2.9: Various structures on a DAG

2.8.5 Pre-Intervention and Post-Intervention DAG

The pre-intervention DAG is a data generating process, or a DAG designed by an expert or given by nature. The post-intervention DAG is a pre-intervention DAG after performing the intervention. This is done by removing the parents of the nodes we are doing an intervention on. For example figure 2.10a shows a pre-intervention DAG and 2.10b shows the post-intervention DAG after performing the intervention on X.

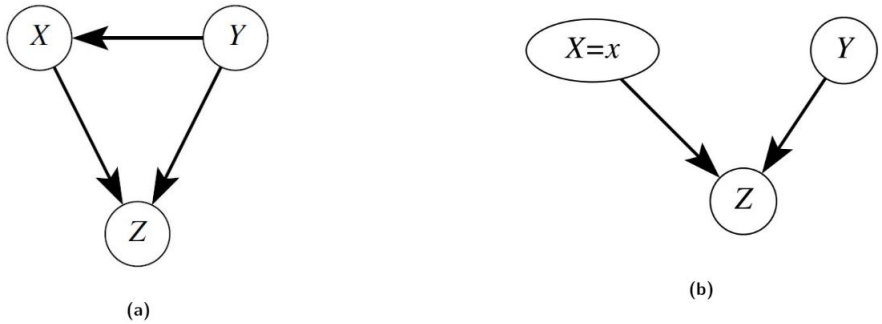


Figure 2.10: **a)** The Pre-Intervention DAG **b)** The Post-Intervention DAG

2.8.6 Structural Causal Model (SCM)

The SCM is a system of structural equations relating each node in a DAG to their parents and the set of unobserved causes. It has a one-to-one mapping to graphical models. The following are the components of the SCM model:

- Variables/Nodes.
- Unobserved cause of nodes.
- Functions mapping each node to their parents.

For example, in figure 2.10, the following equations demonstrate the SCM for the pre (the left equation) and post (the right equation) intervention DAGs.

$$Y = f_Y(U_Y) \qquad Y = f_Y(U_Y) \tag{2.39}$$

$$X = f_X(Y, U_X) \qquad X = x \tag{2.40}$$

$$Z = f_Z(X, Y, U_Z) \qquad Z = f_Z(x, Y, U_Z) \tag{2.41}$$

$$U_X \perp\!\!\!\perp U_Y \perp\!\!\!\perp U_Z \tag{2.42}$$

2.8.7 Identifiability Condition

- Well-Specified Variables.
  - All variables should be properly defined.
  - The treatment, outcome and measured confounders should be correctly specified.
- Positivity.
  - No deterministic treatment within levels of the confounder. For example, in figure 2.8, for positivity to hold we need:

    $$P(X = x|Z = z) > 0 \qquad \text{If} \quad P(Z = z) > 0 \tag{2.43}$$

    If the equation 2.43 is violated then $E(Y|X, Z)$ is not defined. This is because:

    $$E(Y|X, Z) = \sum_y y\, p(y|X, Z) = \sum_y y \frac{p(y, x, z)}{p(x, z)} = \sum_y y \frac{p(y, x, z)}{p(x|z)p(z)} \tag{2.44}$$

    since the numerator of equation 2.44 will be zero.
- Consistency.
  - For those who receive $X = x$ then $Y_x = y$.
  - For those who receive $X = x^\star$ then $Y_{x^\star} = y$.

For example using consistency, $E[Y_x]$ can be written as follows:

$$E(Y_x) = \sum_x E(Y_x|X = x)p(x) = E(Y_x|X = x)p(x) + E(Y_x|X = x^\star)p(x^\star) \tag{2.45}$$

$$= E(Y|X = x)p(x) + E(Y_x|X = x^\star)p(x^\star) \tag{2.46}$$

- Conditional Exchangeability.
  - Conditional exchangeability states that the potential outcome is independent of the treatment given the sufficient set of confounders. Mathematically this is represented as follows:

    $$Y_X \perp\!\!\!\perp X|Z \tag{2.47}$$

    Conditional exchangeability states if one accounts for sufficient confounders then treatment and control groups are the same.
  - Z in equation 2.47 is sufficient for controlling any non-causal path.
  - It states that the treated and untreated groups are similar conditioned on the set Z.
- Defining Sources of Bias.
  - No measurement error.
  - No selection bias (loss-to-follow-up, etc).
- Well-specified Model.

  Define a valid statistical model for inference and testing. Make sure the statistical model can match the problem requirements in terms of amount of data, number of parameters, nature of data (whether data is cross-sectional or longitudinal), etc.

2.8.8 Internal and External Validities

External validity is the ability to generalize the findings to other populations and settings. Lack of external validity suggests that the outcome cannot be related to other people or contexts outside the

current study. For example, consider studies where the participants are all male or students who all have perfect GPA, then the findings cannot be generalized to the entire population.

Table 2.3 summarizes some well-known threats to external validities. Internal validity is defined

Table 2.3: Threats to External Validity

Threats	Definition
Participation Characteristics	Participants should be a good representative (sample) of entire population.
Experiment Setting	How similar is the experiment setting to real-world.
Timing	Study over the past might not be the same as now.

as the degree to which the causal relationship between dependent and independent variables can be established. Internal validity is trying to answer the following questions: Is the independent variable the only cause of the change in the dependent variable? Have we considered all the confounders in the study? Table 2.4 indicates some well-known threats to internal validity according to [22].

Table 2.4: Threats to Internal Validity

Threats	Definition
Ambiguous Temporal Precedence	Lack of clarity between which variable occurs first.
Regression	Problem of regression to the mean.
Confounding Variables	Something other than treatment is causing the change in outcome.
History	External events that affect how participants responds.
Maturation	Natural process in which participants change during the course of study.
Instrumentation	An instrument or a measure in a study change over the course of study .
Testing/Practice Effects	Repetition of test to enhance the performance.
Attrition/Mortality	Participants dropping out during the course of study.

Chapter 3

SEQUENTIAL MODELING AND INFERENCE IN EDUCATION

In this chapter, we discuss sequential modeling and inference and their applications to education. Bayesian knowledge tracing (BKT), proposed in 1995 by Corbett and Anderson [23], is one of the most popular approaches to model student knowledge as two-state latent variables of either learned or did not learn the subject. The BKT can be considered as a two-state hidden Markov model where the probability of forgetting a concept is set to zero. In [24], the authors proposed to add student-specific parameters to BKT as opposed to only including the skill-specific parameters. In [25], and [26] the authors used a RNN to model student learning. They showed that neural networks result in substantial improvements in prediction performance over the more traditional BKT. Although BKT is a popular approach in education, it suffers from the following limitations: 1- It only considers two hidden states, which might not be optimal for a given dataset and application. 2- It assumes once the subject is learned, it will not be forgotten. This is not a valid assumption in education and can lead to an invalid conclusion. Intelligent tutoring system (ITS) is another popular approach to model student learning in education. It consists of computer software that dynamically evaluates student knowledge and provides feedback. Numerous papers have been published such as [27], [28], [29], [30], and [31] about designs and applications of ITS in education. In [27], the authors discussed the ITS architecture and its application to education. In [28], the authors introduced AutoTutor, an ITS that helps college students learn computer literacy. The hidden Markov model (HMM) is a powerful technique to model sequential and time series data. HMM can handle discrete and continuous observations. It can be used in both supervised and unsupervised applications, and in particular we show how it can be used to model the trajectory of student attainments. In this

section, we explain modeling educational videos games using HMM and present a novel approach to predict student performance using a video game as opposed to the exam [6].

3.1 Dataset

The dataset used in this section belongs to the Save Patch (SP) game designed by the National Center for Research on Evaluation, Standards, and Student Testing (CRESST). This game is one out of four fraction games designed to teach the concept of a unit in rational numbers. It is intended to teach the following two concepts: 1- Rational numbers are defined relative to a whole unit; 2- Rational numbers can be added only if they have a common denominator [32].

Save Patch is a single-player game with 56 levels that integrates math learning with game mechanics. It was designed to target the concepts in sixth-grade math. The game was played in four 40-minute classroom periods [33], [34].

The goal in the game is to reach a cat statue across an archeological dig site. The challenge lies in only being able to travel on a one or two-dimensional grid by determining the correct length of rope to travel between posts. On some levels, the student is required to stop at prescribed points. Students are given a set number of rope pieces. They should identify the correct size of rope needed to get to the statues. Sometimes they need to break the ropes to a smaller size or even add multiple small ropes together. Figure 3.1 shows some snapshots of the Save Patch game.

Along with the four games, a pretest and a posttest were designed to test the students' understanding of the concepts before and after each game. A set of questions targeted by each game in the posttest and pretest is carefully identified. This is very beneficial since it permits each game to be analyzed and verified independently of all the other games.

Students who played the game were evenly distributed to 50% male and 50% female. Students were administered a pretest to evaluate their prior math knowledge before playing the game and a posttest upon completion of the game.

There were 22 questions in the pretest and 23 questions in the posttest that examine various mathematical concepts such as understanding the meaning of numerator, denominator, and fraction

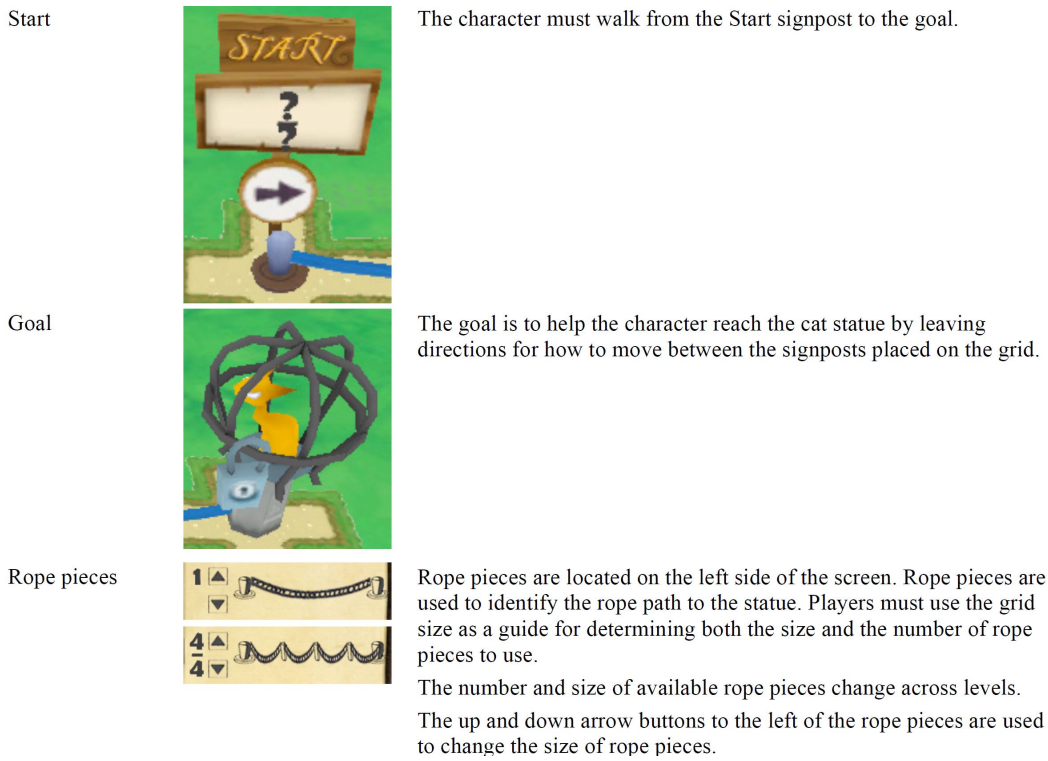


Figure 3.1: Snapshot of the Save Patch game

additions. They are designed to measure the impact of the games on students' fraction knowledge [33].

3.2 Problem Formulation

In this section, the problem formulation and the prediction algorithm using the HMM are discussed. Prediction begins by dividing the students according to their score in the questions targeted by the SP game in the posttest into two classes: Class 1 consists of 433 students and are those who score low, and Class 2 consists of 421 students and are those who score high in those questions. This is because there are multiple different trajectories as students go through different levels of the game; hence students with more similar trajectories should be trained with each other. Each class is trained separately using the HMM, and the optimal parameters are determined by the model selection algorithms, which were explained in chapter 2. Testing or decoding is done by running the Viterbi algorithm on the observation sequences.

The research questions investigated in this section are 1- Can educational video games be used to predict student performance in a test? 2- Can the hidden Markov model be used as a reliable tool to predict student mastery levels using game data? 3- Can trajectories from the game provide useful information that cannot be obtained by examining the postest? The prediction problem using the HMM is solved as follows: Hidden states are defined to be the students' mastery levels, and the goal is to predict their final mastery level as they go through different levels of the game. This can be formulated as predicting the final mastery level $\hat{S}$ given all the past mastery levels $(s_1, s_2, ...s_n)$. The following techniques can be used to perform the prediction.

1. **Naive**: This is the most basic method in which the predicted value is simply equal to the last observed value of the time series.

$$\hat{S} = s_n \tag{3.1}$$

2. **Linear averaging**: This means that the final predicted value is the average of all the other mastery levels.

$$\hat{S} = \frac{\sum_{i=1}^{i=n} s_i}{n} \tag{3.2}$$

One extension to this method is to perform averaging over a window of time length p, which means to consider only the most recent p values.

$$\hat{S} = \frac{\sum_{i=n-p+1}^{i=n} s_i}{p} \tag{3.3}$$

Another extension of this method is the exponential smoothing. The idea is to perform the linear averaging by choosing larger weights for the most recent values and smaller weights for the distant values. This is described by the following formula:

$$\hat{S} = \alpha s_n + \alpha(1-\alpha)s_{n-1} + \alpha(1-\alpha)^2 s_{n-2} + ...$$

$$0 \leq \alpha \leq 1 \tag{3.4}$$

Equation(13) can also be written recursively as follows:

$$\hat{S}_n = \alpha s_n + (1-\alpha)\hat{S}_{n-1}$$
$$0 \leq \alpha \leq 1 \tag{3.5}$$

3. **Mode**: This means that the final mastery level is the mastery level that appears most in the sequence. Mathematically this can be represented as follows:

$$\hat{S} = \underset{j}{\operatorname{argmax}} \sum_{i=1}^{i=n} 1(S_i = j) \tag{3.6}$$

To learn more about the exponential smoothing method, refer to the background materials in chapter 2.

3.3 Results and Discussions

In this section, the results for the prediction task described in the last section are presented and discussed. The prediction is done by the DHMM by discretizing the observations to a certain number of bins using the domain knowledge or the k-mean algorithm [35]. Since the HMM training is done using expectation maximization (EM), and EM might converge to the local optimum instead of the global optimum, multiple initial conditions are used to test the algorithms and the best model is selected using the AIC or BIC. The prediction is performed under the following cases:

1. The total number of attempts per level is used as the observations.

2. The total number of moves per level is used as the observations.

Case 1: Total number of attempts per level is used as the observation.

The observations are discretized (DHMM) to four levels according to the following rules: 1 or 2 attempts per level is label 1; 3 or 4 attempts is label 2; 5,6 or 7 attempts is label 3, and anything above is label 4. According to the Tables 3.1 and 3.2 that provide the results for the model selections using the BIC algorithm, training is done by assigning the number of states (Q) to be 3 with the initial condition to be 35 for the class 1 and Q=2 with initial condition=26 for the

class 2. The goal is to train two separate HMMs with the above parameters for class 1 and class 2 and make the prediction of the final mastery level and compare it to the class label.

For instance consider the following examples in Tables 3.3, 3.4 and 3.5.

Class 1: Table 3.3 shows an example of a game trajectory for a student who finishes four levels of the game and obtained 1.5 out of 8 in the posttest. For class 1 students, 1 in the "State Sequence" column indicates the lowest mastery level, and 3 indicates the highest mastery level. Since there are 3 states, scores in the [2,3] are mapped to label 2, and scores in the [1,2) are mapped to label 1. The following cases illustrate how the final mastery level is calculated using the methods discussed in the last section.

1. **Naive**: $\hat{S} = s_4 = 1$. This method ignores all the past states and makes the predicted value to be the most recent state.

2. **Average**: $\hat{S} = \frac{1+1+1+1}{4} = 1$. This method assigns equal weight to each state and could be the best prediction method for the game since levels of the game are independent of each other and should be treated separately.

3. **Mode**: $\hat{S} = 1$. This method predicts the final value to be the state that is repeated the most.

Table 3.4 shows another example of comparison between the posttest and the state trajectories for the class 1 students. The final prediction for this student is done as follows:

1. **Naive**: $\hat{S} = s_{42} = 1$. This method predicts the class label correctly but it ignores all the past states and does not take any past performance into account.

2. **Average**: $\frac{\sum_{i=1}^{i=42} s_i}{42} = 1.81$. Since 1.81 is less than 2 therefore the predicted label would be 1.

3. **Mode**: $\hat{S} = 1$. The predicted label using this method is also 1, since 1 is repeated more than any other states.

Class 2: Table 3.5 shows an example of a student with ID 1627 in class 2 who scored 6.17 out of 8 in the posttest and completed 49 levels of the SP game. Scores in the [1.5,2] are mapped

Table 3.1: BIC for Class 1 in Case I

Row Number	Initial Condition	BIC	Number of States
1	1	21885	2
2	26	21720	3
3	35	21636	3
4	55	21709	3
5	64	21779	4
6	100	21880	2

Table 3.2: BIC for Class 2 in Case I

Row Number	Initial Condition	BIC	Number of States
1	1	17716	2
2	26	17538	2
3	35	17541	4
4	55	17557	3
5	64	17631	4
6	100	17726	2

to label 2 and scores in [1,1.5) are mapped to label 1. The final prediction for this student is as follows:

1. **Naive**: $\hat{S} = s_{49} = 1$. Since $1 < 1.5$, the predicted label is 1. This is an example of forecasting error since only the last state is used for the prediction.

2. **Average**: $\frac{\sum_{i=1}^{i=49} s_i}{49} = 1.88$. Since 1.88 is greater than 1.5 the predicted label would be 2.

3. **Mode**: $\hat{S} = 2$. Since $2 > 1.5$ the predicted label is 2.

Table 3.6 summarizes the accuracy for the various methods discussed in the last section by comparing the class label to the predicted value from the state trajectory. According to Table 3.6 the best prediction accuracy for class 1 is for the naive method and the best prediction accuracy for class 2 is for the average and the mode methods. Among all the prediction methods described

Table 3.3: HMM Trajectory for class 1 student in Case I

ID	Posttest	State Sequence
1994	1.5	1111

Table 3.4: HMM Trajectory for class 1 student in Case I

ID	Posttest	State Sequence
1764	2.5	1111111111111111233333 3333333333211111111

Table 3.5: HMM Trajectory for class 2 student in Case I

ID	Posttest	State Sequence
1564	6.5	22221222222222212222222222 22222222122222121221

in the last section, the average method is the most reliable one; this is because the naive method only accounts for the most recent mastery level and ignores all the past values. This cannot be a reliable method for the prediction using a game since different levels of the game have different game mechanics and difficulties. Therefore, every level should make a contribution to the final prediction. The average method is also more informative and provides more detail than the mode method. To better understand this consider the following cases for two different students whom each finish five levels of the game:

- Student A trajectory is 11222.
- Student B trajectory is 22222

For both students A and B the predicted label is 2 using both the mode and the average methods. However, for the student A the score using the average method is 1.6 while for student B the score using the average method is 2. This shows that student B has a better performance that student A in the game. This information cannot be gained using the mode method since in both cases the state 2 is repeated the most.

Case II: Total number of moves per level is used as the observations.

The observations are discretized to four levels according to the following rule: An expert plays all the levels of the game, and the total number of moves to finish each level is recorded, then there is a $\alpha = 1.5$ which can be called as a compensation factor which compensates for the game

Table 3.6: Prediction Accuracies for various Methods in Case I

Method	Class I Accuracy	Class II Accuracy
$Naive$	97.48%	86.09%
$Average$	86.55%	100%
$Mode$	86.55%	100%

mechanics and the game difficulties. The compensation factor is chosen by comparing the students' total number of moves per level and the expert's total number of moves per level. This comparison results in the $\alpha = 1.5$ as the most reasonable candidate. The compensation factor is then multiplied by total moves per level for the expert, and the observations are discretized according to this rule per level since different levels might have different difficulties. According to the Tables 3.7 and 3.8 the training is done by letting the number of states to be 3 for both class 1 and 2 and the initial conditions to be 35 for the class 1 and 26 for the class 2. A similar analysis to the case I is done here to predict the final mastery level.

For instance consider the following examples in Tables 3.9 and 3.10.

Class 1: Table 3.9 shows an example of a class 1 student with ID 1768 who scored 3.88 out 8 in the posttest. The predicted mastery level using all three methods is 1. This is because 1 is repeated more than other states, the most recent state is 1 and the average value of the state trajectory is 1.85 which is less than 2. Although all three methods predict the final mastery level correctly, the average method is more informative since it provides more detail that the given student has done well on some levels since his score is close to the boundary between the class 1 and the class 2.

Class 2: Table 3.10 shows another example where the naive method can make an incorrect prediction. The mode or average methods predict the final mastery level to be 2 while the naive method predicts the final value to be 1 since the most recent state is 1.

Table 3.11 presents the prediction accuracies for various algorithms for the class 1 and the class 2 students. According to Table 3.11 the highest prediction accuracy for class 1 is for the naive method and for class 2 is for the average and mode methods. Similar to case I, it can be argued that the average method is better than naive and is more informative than the mode method.

Table 3.7: BIC for Class 1 in Case II

Row Number	Initial Condition	BIC	Number of State
1	1	25895	2
2	26	25850	7
3	35	25596	3
4	55	25613	3
5	64	25682	4
6	100	25872	2

Table 3.8: BIC for Class 2 in case II

Row Number	Initial Condition	BIC	Number of State
1	1	24462	2
2	26	24206	3
3	35	24239	4
4	55	24231	3
5	64	24315	4
6	100	24456	5

Table 3.9: HMM Trajectory for class 1 student in Case II

ID	Posttest	State Sequence
1768	3.88	2111111111111111123333333333333333211111111111

Table 3.10: HMM Trajectory for class 2 student in Case II

ID	Posttest	State Sequence
1573	7.5	22233333333333312333333312333331312333122233123331

Table 3.11: Prediction Accuracies for Various Methods in Case II

Method	Class I Accuracy	Class II Accuracy
$Naive$	89.92%	92.17%
$Average$	75.63%	100%
$Mode$	76.47%	100%

Table 3.12: Statistical Results for Average Method

	Case I	Case II
$Accuracy$	93.16%	87.61%
$Recall$	Class 1: 86.55% Class 2: 100.0%	Class 1: 75.63% Class 2: 100.0%
$Precision$	Class 1: 100.0% Class 2: 87.79%	Class 1: 100.0% Class 2: 79.86%
F_1Score	Class 1: 92.79% Class 2: 93.50%	Class 1: 86.12% Class 2: 88.80%
$AUCScore$	0.8644	0.8818

Other metrics that are widely used in evaluating a model are recall, precision, accuracy, F_1 and AUC scores. Table 3.12 summarizes the results for the average method for both class 1 and class 2 students under both case I and case II. High values of accuracy, recall, precision, F_1 and AUC scores under both case I and II suggest that the proposed method can perform strong prediction of student mastery levels. It is important to note that since accuracies are high for all prediction methods and are obtained from the sufficiently large test sample size, then it is safe to claim that results are statistically significant.

One important topic that needs more attention is the confounding variables. They are defined as the variables that affect both the independent and dependent variables, and if not controlled properly, they might change the results of experiments. For example, the transfer of knowledge between the game and the posttest is a confounding variable. Transfer of knowledge is the application of the previously learned skills in a new domain. This can be the main reason why accuracy for class 1 is lower than class 2 students since class 1 students could have a harder time connecting the game concepts to the posttest. Game mechanics and the difficulty of the different levels are other confounding variables in the SP game. However, there might be other confounding variables that are hard to control and might cause the change in the exam score such as students' interest, family situation, health and many more. Since in this study a retrospective analysis of the data was conducted, it was not possible to query such factors.

3.4 conclusion

In this section, the HMM algorithm is used to predict the students' final mastery level, given their performance in various levels of the game. It was shown that despite various confounding variables affecting the students, the HMM can be used as a promising solution in educational environments to model students' actions and make the prediction throughout the game.

The results indicate that examining time series data from the game can lead to dynamic evaluation of student mastery levels throughout the time which cannot be obtained by examining only the posttest. This can be useful to make a timely intervention and provide efficient feedback. In particular, such dynamic analysis can enable students to be guided through a sequence of concepts that build on each other, thus enabling learning at their own pace. It can also be used to target human intervention for students who are struggling.

While for this study there was no ground truth to quantify student attainment throughout the game, the strong prediction of final mastery level shows that there is considerable promise in applying the HMM to this purpose. A focus of the future research will be on how to design the interactive game experience to enable such inferences to be of high quality.

Chapter 4

TSBNGEN: SYNTHETIC TIME SERIES DATA GENERATION USING ARBITRARY DYNAMIC BAYESIAN NETWORK STRUCTURES

Real-world data collection and labeling are time-consuming and error-prone tasks that can take up to years to finish. Furthermore, some real-world data, due to its confidential nature, cannot be distributed or analyzed. Moreover, many modern algorithms, such as neural networks, require a large amount of data for efficient training. For example, in [36], the authors show that under limited data, a more classical algorithm such as a hidden Markov model (HMM) can outperform the long short term memory (LSTM) algorithm.

Scientists and researchers have proposed various methods to generate synthetic data such as [37], [38] and [39] using generative adversarial network (GAN). In [39], the authors proposed a Recurrent GAN (RGAN) and Recurrent Conditional GAN (RCGAN) to produce realistic real-valued multi-dimensional time series data. Although the GAN has numerous image processing and computer vision applications and produces satisfactory results, it suffers from the following limitations: 1- It is hard to train 2- it is sometimes unstable and might not never converge. 3- It is significantly harder to train for text than images. 4- It generally requires lots of data for training and might not be a good choice when there is limited or no available data.

In this section, the tsBNgen package is introduced. tsBNgen is a Python package to generate synthetic time series data according to arbitrary dynamic Bayesian network structures [17]. The package, along with its documentation, can be downloaded from
https://github.com/manitadayon/tsBNgen (tsBNgen).

The motivation behind implementing this package is three folds: 1- Designing a model-based technique to generate synthetic time series data since the prior work mainly focused on data-driven approaches that required a large amount of data to start with. 2- It automates the data generating and sampling process once the structure of the network is known. This is especially useful if there is a mixture of discrete and continuous nodes and the covariates are high dimensional. 3- It allows more complex models to be studied. This will be used in future chapters to study more sophisticated sequential models than the HMM. For example, this package is suitable for testing deep learning models since it can generate an arbitrary number of observations with an arbitrary number of time points per observation.

4.1 Features and Capabilities

tsBNgen is an open-source Python library released under MIT license. It generates time series data corresponding to any Bayesian network structures. The following is the list of supported features and capabilities of the tsBNgen.

- Easy and simple interface.
- Support for discrete, continuous, and hybrid networks (a mixture of discrete and continuous nodes).
- Supports an arbitrary number of nodes with any interconnections.
- Use multinomial distributions for discrete and Gaussian distributions for continuous nodes.
- Supports arbitrary loopback values for temporal dependencies.
- Easy to modify and extend the code to support for the new structure.

Loopback is defined as the length of temporal dependency. For example, loopback of one means node at time t is connected to some nodes at time t+1.

4.2 Instruction

To use tsBNgen, either clone this repository tsBNgen or install the software using the following commands:

```
pip install tsBNgen
```

After the software is successfully installed. Import necessary libraries and functions as follows.

```
from tsBNgen import *
from tsBNgen.tsBNgen import *
```

The above two lines import all the functions for tsBNgen. Next, we review an example of how to use tsBNgen to generate an arbitrary DBN structure. Before going over an example, we need to define some notations.

T = Length of each time series.

D = Dimension of time series.

N = Number of samples

$u_{i0}, u_{i1}, ...u_{iT}$ = Time series for node i

N_{ui} = Number of possible symbols for node i (if it is discrete).

4.2.1 Example 1

This Example refers to the figure 4.1, which is the diagram of an HMM in which $u_{00}, u_{01}, ..., u_{0T}$ refer to the state sequence and $u_{10}, u_{11}, ..., u_{0T}$ refer to the observations. The following table summarizes the setting and configuration corresponding to this model.

The conditional probability distributions (CPDs) for this network are listed in Tables 4.2,4.3, and 4.4. The code to model and generate data for figure 1 is as follows:

```
import time
START=time.time()
T=20
```

Table 4.1: Parameter Setting for Example 1

Architecture	Value
T	20
N	1000
Loopback	1
# of discrete nodes per time step	1
# of possible levels for node u_0 per time point	4
# of continuous nodes per time step	1

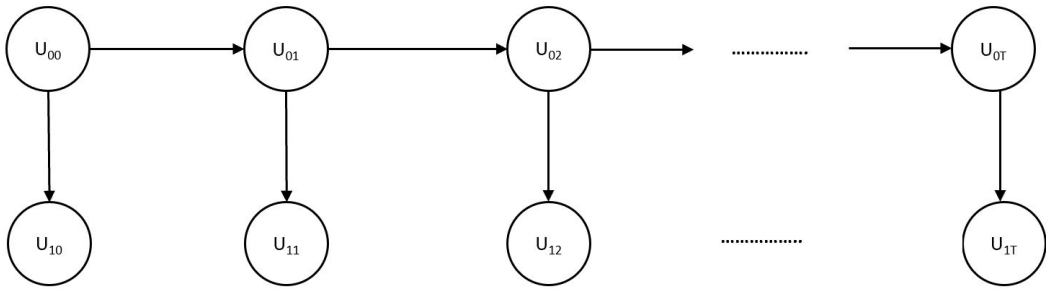


Figure 4.1: Synthetic Time Series Data Under Example 1

```
N=1000
N_level=[4]
Mat=pd.DataFrame(np.array(([0,1],[0,0]))) # HMM
Node_Type=['D','C']
CPD={'0':[0.25,0.25,0.25,0.25],'01':{'mu0':20,'sigma0':5,
   'mu1':40,'sigma1':5,'mu2':60,'sigma2':5,'mu3':80,
   'sigma3':5}}
Parent={'0':[],'1':[0]}
CPD2={'00':[[0.6,0.3,0.05,0.05],[0.25,0.4,0.25,0.1],
[0.1,0.3,0.4,0.2],[0.05,0.05,0.4,0.5]],'01':{'mu0':20,
'sigma0':5,'mu1':40,'sigma1':5,'mu2':60,'sigma2':5,
'mu3':80,'sigma3':5
}}
loopbacks={'00':[1]}
Parent2={'0':[0],'1':[0]}
```

Table 4.2: CPD for Initial Time

u_{00}	$P(u_{00})$
1	0.25
2	0.25
3	0.25
4	0.25

Table 4.3: CPD for Time Step t_1 to t_T

u_{0t-1}	u_{0t}	$P(u_{0t}\|u_{0t-1})$
1	1	0.6
2	1	0.3
3	1	0.05
4	1	0.05
1	2	0.25
2	2	0.4
3	2	0.25
4	2	0.1
1	3	0.1
2	3	0.3
3	3	0.4
4	3	0.2
1	4	0.05
2	4	0.05
3	4	0.4
4	4	0.5

Table 4.4: CPD for the Continuous Node

u_{1t}	μ	σ
1	20	5
2	40	5
3	60	5
4	80	5

```
Time_series1=tsBNgen(T,N,N_level,Mat,Node_Type,CPD,Parent,
CPD2,Parent2,loopbacks)
Time_series1.BN_data_gen()
FINISH=time.time()
print('Total Time is',FINISH-START)
```

The above code generates a 1000 time series with a length of 20 correspondings to states and observations. Observations are normally distributed with a particular mean and standard deviation. The states are discrete (hence the 'D') and take four possible levels determined by the N_level variable. The loopbacks is a dictionary in which each key has the following form: node+its parent. Since in architecture 1, only states, namely node 0 (according to the graph's topological ordering), are connected across time and the parent of node 0 at time t is node 0 at time $t-1$; therefore, the key value for the loopbacks is '00' and since the temporal connection only spans one unit of time, its value is 1. Node_Type determines the categories of nodes in the graph. For example, the first node is discrete ('D'), and the second one is continuous ('C'). Mat represents the adjacency matrix of the network. The total time is 2.06 (s), and running the model through the HMM algorithm gives us more than 93.00 % accuracy for even five samples. Now let's take a look at a more complex example. From now on, to save some space, I avoid showing the CPD tables and only show the

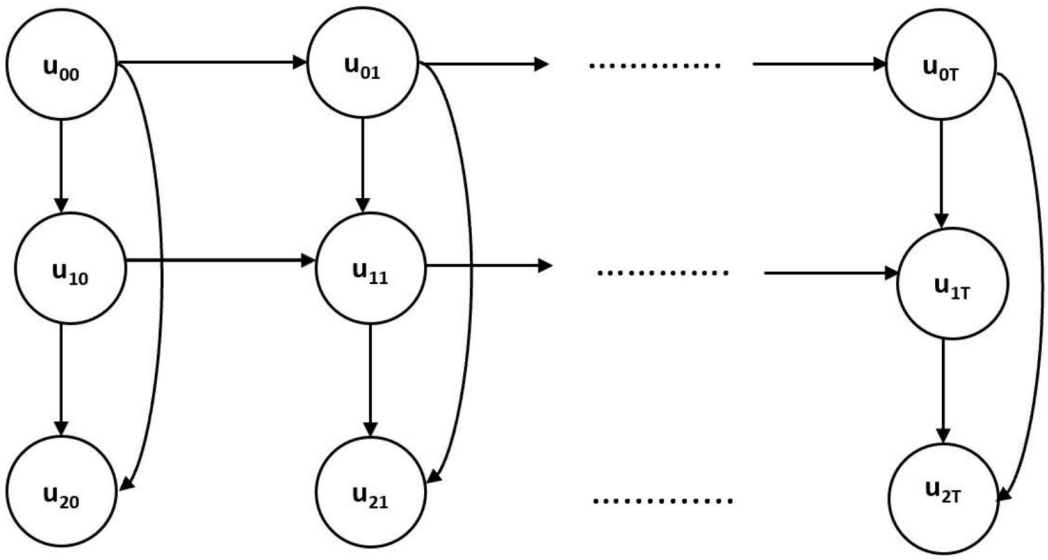


Figure 4.2: Synthetic Time Series Data Under Example 2

architecture and the python code used to generate data.

4.2.2 Example 2

Example 2 refers to the architecture in figure 4.2, where the nodes in the first two layers are discrete and the last layer nodes(u_2) are continuous. Based on the graph's topological ordering, you can name them nodes 0, 1, and 2 per time point. Let's say you would like to generate data when node 0 (the top node) takes two possible values (binary), node 1(the middle node) takes four possible values, and the last node is continuous and will be distributed according to Gaussian distribution for every possible value of its parents. The following python codes simulate this scenario for 2000 samples with a length of 20 for each sample.

```
T=20
N=2000
N_level=[2,4]
Mat=pd.DataFrame(np.array(([0,1,1],[0,0,1],[0,0,0])))
Node_Type=['D','D','C']
CPD={'0':[0.6,0.4],'01':[[0.5,0.3,0.15,0.05],
```

```
[0.1,0.15,0.3,0.45]],'012':{'mu0':10,'sigma0':2,'mu1':30,
'sigma1':5,'mu2':50,'sigma2':5,'mu3':70,'sigma3':5,
'mu4':15,'sigma4':5,'mu5':50,'sigma5':5,'mu6':70,
'sigma6':5,'mu7':90,'sigma7':3
}}
Parent={'0':[],'1':[0],'2':[0,1]}
CPD2={'00':[[0.7,0.3],[0.2,0.8]],'011':[[0.7,0.2,0.1,0],
[0.6,0.3,0.05,0.05],[0.35,0.5,0.15,0],
[0.2,0.3,0.4,0.1],[0.3,0.3,0.2,0.2],
[0.1,0.2,0.3,0.4], [0.05,0.15,0.3,0.5],[0,0.05,0.25,0.7]],
'012':{'mu0':10,'sigma0':2,'mu1':30,'sigma1':5,
'mu2':50,'sigma2':5,'mu3':70,'sigma3':5,'mu4':15,
'sigma4':5,'mu5':50,'sigma5':5,'mu6':70,
'sigma6':5,'mu7':90,'sigma7':3
}}
Parent2={'0':[0],'1':[0,1],'2':[0,1]}
loopbacks={'00':[1],'11':[1]}
Time_series2=tsBNgen(T,N,N_level,Mat,Node_Type,CPD,Parent,CPD2,
Parent2,loopbacks)
Time_series2.BN_data_gen()
```

As the above code shows, node 0 (the top node) has no parent in the first time step (This is what the variable Parent represents). This is sometimes known as the root or an exogenous variable in a causal or Bayesian network. Node 1 is connected to node 0, and node 2 is connected to both nodes 0 and 1. To represent the structure for other time-steps after time 0, the variable Parent2 is used, which says node 0 is connected to itself across time (since '00' is [1] in loopbacks). Node 1 is connected to node 0 for the same time and to node 1 in the previous time. Since tsBNgen is a model-based data generation, you need to provide the distribution (for exogenous node) or

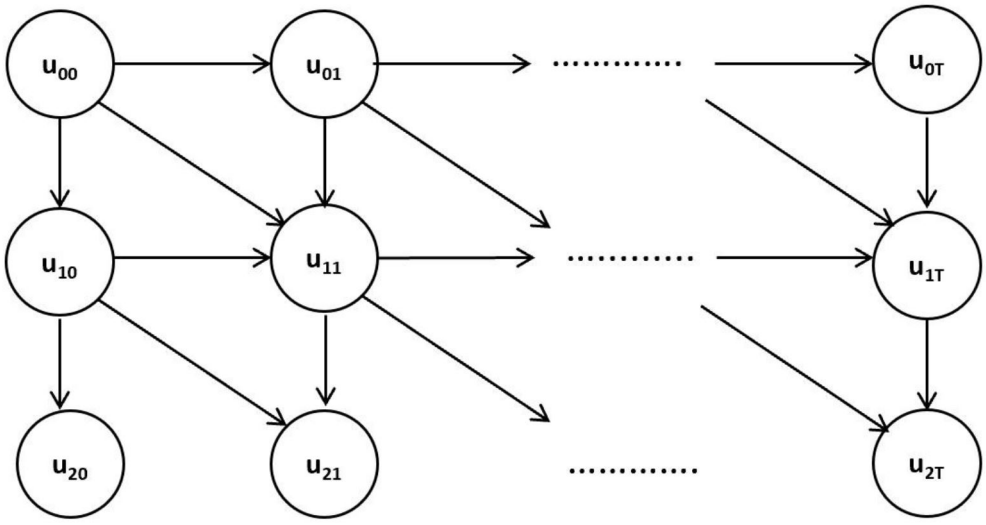


Figure 4.3: Synthetic Time Series Data Under Example 3

conditional distribution of each node. If you would like to generate synthetic data corresponding to architecture with arbitrary distributions, you can choose CPD and CPD2 to be anything you like as long as the sum of entries for each discrete distribution is 1.

4.2.3 Example 3

Example 3 refers to the architecture in figure 4.3, where the nodes in the first two layers are discrete and the last layer nodes(u_2) are continuous. Assume you would like to generate data when node 0 (the top node) is binary, node 1(the middle node) takes four possible values, and node 2 is normally distributed for every possible value of its parents. The following python codes simulate this scenario for 1000 samples with a length of 10 for each sample.

```
T=10
N=1000
N_level=[2,4]
Mat=pd.DataFrame(np.array(([0,1,0],[0,0,1],[0,0,0])))
Node_Type=['D','D','C']
CPD={'0':[0.5,0.5],'01':[[0.6,0.3,0.05,0.05],[0.1,0.2,0.3,0.4]]
```

```
,'12':{'mu0':10,'sigma0':5,'mu1':30,'sigma1':5,
    'mu2':60,'sigma2':5,'mu3':80,'sigma3':5}}
Parent={'0':[],'1':[0],'2':[1]}
CPD2={'00':[[0.7,0.3],[0.3,0.7]],'0011':[[0.7,0.2,0.1,0],
[0.5,0.4,0.1,0],[0.45,0.45,0.1,0],[0.3,0.4,0.2,0.1],
[0.4,0.4,0.1,0.1],[0.2,0.3,0.3,0.2],[0.2,0.3,0.3,0.2],
[0.1,0.2,0.3,0.4],[0.3,0.4,0.2,0.1],[0.2,0.2,0.4,0.2],
[0.2,0.1,0.4,0.3],[0.05,0.15,0.3,0.5],[0.1,0.3,0.3,0.3],
[0,0.1,0.3,0.6],[0,0.1,0.2,0.7],[0,0,0.3,0.7]],
'112':{'mu0':10,'sigma0':2,'mu1':30,'sigma1':2,'mu2':50,
'sigma2':2,'mu3':60,'sigma3':5,'mu4':20,'sigma4':2,
'mu5':25,'sigma5':5,'mu6':50,'sigma6':5,
'mu7':60,'sigma7':5,
 'mu8':40,'sigma8':5,'mu9':50,'sigma9':5,'mu10':70,
 'sigma10':5,'mu11':85,'sigma11':2,'mu12':60,'sigma12':5,
'mu13':60,'sigma13':5,'mu14':80,'sigma14':3,
'mu15':90,'sigma15':3}}
Parent2={'0':[0],'1':[0,0,1],'2':[1,1]}
loopbacks={'00':[1], '01':[1],'11':[1],'12':[1]}
Time_series2=tsBNgen(T,N,N_level,Mat,Node_Type,CPD,Parent,CPD2,
Parent2,loopbacks)
Time_series2.BN_data_gen()
```

In the same way, you can generate time series data for any graphical models you want.

4.3 Conclusion

In this section, we introduced the tsBNgen, a python library to generate synthetic data from an arbitrary Bayesian network. We discussed the features and capabilities of our software. We discussed

how to use the software using three examples. For more examples, up-to-date documentation, and instructions, please visit the GitHub page at https://github.com/manitadayon/tsBNgen.

Chapter 5

COMPARATIVE ANALYSIS OF THE HIDDEN MARKOV MODEL AND LSTM

The fundamental challenges in time series and sequential data analysis are: 1- Observations at different points in time are correlated. 2-order of observations in time matters. This makes some of the algorithms that change or permute the order of data unusable. Scientists and researchers have done extensive research in time series analysis, such as [40], [41],[42],[43]. They have borrowed tools from various domains, such as graphical modeling, statistics, and machine learning to model and forecast the time series data. In [44], the authors used ARIMA to make predictions of the stock market. In [40], authors used deep learning, specifically LSTM, to predict multivariate time series data. They showed that even simple LSTM architectures can make an accurate prediction of future values. In [42], the author combined ANN and ARIMA to design a hybrid methodology to model time series data. In [43], the author used machine learning, specifically a support vector machine (SVM), to predict the stock market index.

Deep learning has been used extensively in various applications. For instance, LSTM, a variant of the recurrent neural network, has been successful in modeling the sequence data in recent years. For example it is widely used in the natural language processing (NLP) to model sequences [45], [46]. Despite their effectiveness and their numerous applications, neural networks, in general, suffer from the following problems: 1- They require a massive volume of (labeled) data for training. 2-The number of trainable parameters, even for a simple model, is huge. 3- Since they have a huge number of parameters, the models are not easily interpretable. For these reasons, simpler models with fewer parameters are usually preferred if the performance degradation is negligible.

Advances in statistics and optimization allow researchers to develop various algorithms from

graph theory and Bayesian learning such as [47], [48], [49] to design sophisticated graphical models for better inference and learning. However, the more complex the model, the more computationally difficult inference and learning become. HMM and its variants such as [50], [51] have been successfully used in applications with latent variables. This is due to the tree structure of its graph, which makes the inference and learning very efficient.par In this section, we compare HMM and LSTM in terms of performance, the number of trainable parameters, and how much data is needed for training using synthetic data corresponding to various graph structures. The data is generated by the tsBNgen package explained in chapter 4. The contribution of this chapter is twofold: 1- Comparing LSTM to both supervised and unsupervised HMM in terms of performance and complexity. 2- Proposing some methods to efficiently convert continuous HMM to discrete HMM by discretizing the observations.

5.1 Dataset

In this section, We create the synthetic data corresponding to different dynamic Bayesian network (DBN) structures. We proposed three DBN structures with various levels of complexity. The following are the parameters and notations for the synthetic data:

T = Length of each time series.

D = Dimension of time series.

N = Number of samples (time series)

$S_1, S_2, ..., S_T$ = sequence of discrete states.

$u_{i1}, u_{i2}, ...u_{iT}$ = sequence of input i.

$O_{i1}, O_{i2}, ...O_{iT}$ = sequence of observations i.

N_{ui} = Number of symbols for input i (if it is discrete).

N_s = Number of symbols for state i (if it is discrete).

Case I: we generate a structure shown in figure 5.2 with two confounding variables (inputs) and two continuous observations per time point. For this case, $N = 2000$, $T = 50$, $D = 2$, $N_s = 3$, N_{u1} and N_{u2} are both 2.

Case II: we generate a structure shown in figure 5.3 with one input variable and one continuous observation per time point. This is more complex than the case I since inputs are connected across time in addition to states. Moreover, the input is connected to observation and state per time point. For this case, $N = 2000$, $T = 20$, $D = 1$, $N_s = 4$, $N_{u1} = 2$.

Case III: we generate a structure shown in figure 5.4 with one input variable and one continuous observation per time point. This is the most complex case since input at time t is connected to state at time t and input and state at time $t + 1$. Moreover, the state at time t is connected to state at time $t + 1$ and both observations at time t and t+1. For this case, $N = 1000$, $T = 10$, $D = 1$, $N_s = 4$, $N_{u1} = 2$.

Case IV: we generate a structure shown in figure 5.5 with one input variable and one continuous observation per time point. This is a more complex version of case II since the state at time t is connected to the state at time $t + 1$ and the state at time $t + 2$. For this case, $N = 2000$, $T = 20$, $D = 1$, $N_s = 4$, $N_{u1} = 2$.

For all these cases, a multinomial distribution is used to model discrete nodes, and a Gaussian distribution is used to model continuous nodes.

This method of generating time series has the advantage that parameters of the time series such as the dimension, length, number of samples, distribution of each node, and graph structures can be arbitrarily chosen.

5.2 Problem Formulation

In this section, the problem formulation and the prediction algorithms are discussed. The procedure to train and test the LSTM is as follows:

1. Pick a training ratio as the amount of data used for training.

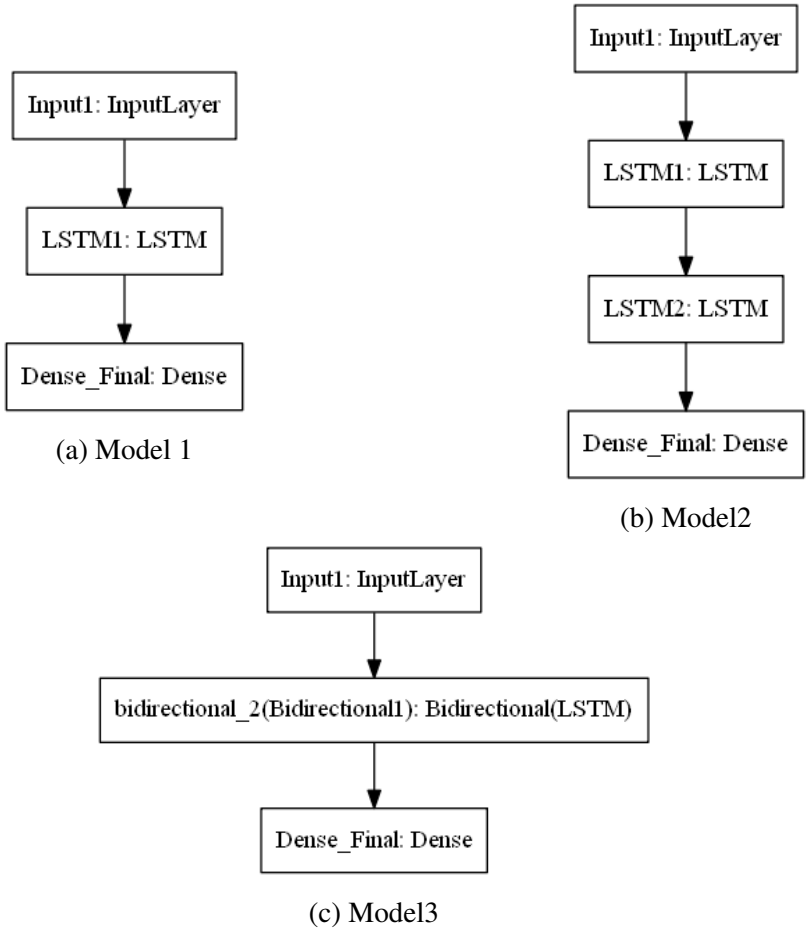
2. Perform transformation on data to scale it to [0,1] range.

3. Perform model selection to find the optimal parameters.

4. Test the model on the test (unseen) data.

5. Keep reducing the training ratio and perform steps 1 to 4 again.

Figure 5.1 shows the architectures used for LSTM training: The procedure to train and test the



(a) Model 1

(b) Model2

(c) Model3

Figure 5.1: LSTM Architectures for Time Series Data

HMM is as follows:

1. Pick a training ratio as the amount of data is used for training.

2. Perform model selection to find the optimal parameters.

3. Train model with parameters from step 2.

4. Perform the testing by applying the Viterbi algorithm on test data.

5. Perform state mapping

6. Keep Reducing the training ratio and perform steps 1 to 5 again.

The above procedure works very well for the HMM if there is one feature but if there are more than one feature then a more reasonable method to approach the prediction problem is as follows. First, find the correlation between the features. Second, train one HMM for each set of features that have positive correlation with each other. Third, combine the predictions together.

The differences between discrete hidden Markov model (DHMM) and continuous hidden Markov model (CHMM) lie in the input and how data is modeled. CHMM can be converted to the DHMM problem by discretizing the observations. The critical question is how to perform discretization in order not to degrade the predictive performance. In this section, we propose the following:

1. **Network structure is the same through time**: Since the network structure is unchanged throughout time and number of nodes, their distributions, and confounding variables remain the same. Then time series for different samples can be concatenated to create a long vector, and discretization by domain knowledge or K-means can be performed on this vector.

2. **Network structure is varying through time**: In this case, we perform the discretization per time slice. This is because node distribution, interconnections, and confounding variables are changing from time to time, and it would not be meaningful to concatenate the time series. For example, consider that the observation corresponds to how long it takes to finish the levels of a game. Different levels have different difficulties, game mechanics, and narratives. Therefore, if some levels are more straightforward than others, then they are expected to take less time, which should be taken into consideration during discretization.

The first case above is used more often than the second one in modeling time series and sequential data since it reduces the inference and learning complexities. This is because it allows us to use dynamic programming to find the estimate of the parameters iteratively while if a network structure is changing with time, a new set of equations is needed to update and estimate the parameters. This section aims to measure the trade-off between predictive accuracy and the complexity of HMM

and LSTM. We use CHMM for unsupervised HMM, DHMM for both the supervised HMM and unsupervised HMM. K-means clustering is also used for the discretization. We find the number of trainable parameters for LSTM using simulation. The following are the equations to find the number of parameters in CHMM with diagonal covariance matrix (although the procedure for other cases is very similar) and DHMM.

- CHMM with a diagonal covariance matrix:

$$C = M * (M-1) + M * (N-1) + (N * M) * D + (M * N * D) + M - 1 \quad (5.1)$$

  where M is the number of states, N is the number of mixture components, and D is the dimension of time series.

- DHMM:

$$C = M * (M-1) + M * (N-1) + (M-1) \quad (5.2)$$

  where M is number of states, N is number of symbols.

It is worth noting that to measure the accuracy of unsupervised HMM (parameters learned from EM), we set the number of states to be Ns and find the optimal number of mixture components.

5.3 Results and Discussions

In this section, we present and discuss the results for the comparison of HMM and LSTM.

Case I

This case corresponds to figure 5.2 and consists of two continuous observations. This can be a simple model of student learning, in which $S_1, S_2, ..., S_T$ corresponds to student mastery levels and u_{1j}, $j = 1, 2, 3, ...T$ corresponds to study habit (it has two levels corresponding to bad and good) and u_{2j}, $j = 1, 2, 3, ...T$ corresponds to other factors (like the personal issues, interest,...) and observations can correspond to time to finish an assignment and the assignment score. Tables 5.1 and 5.2 summarize the prediction accuracy and the number of parameters for the LSTM and HMM as a function of the training ratio. It is worth noting that N=2000 is the total number of time series.

Therefore training ratio of 0.8 results in 1600 time series for training and 400 time series for testing. The length of each time series is 50. To train the LSTM model, we usually concatenate the samples, which means we will have $50 * 1600 = 80000$ samples. In Table 5.1 for training ratio=$0.8, 0.3, 0.1$, the optimal parameters are as follows: the number of neurons in the hidden layer is 8, the optimizer is Adam, and the batch size is 4. The optimal parameters for training ratio =0.005 and 0.001 are the same, except the number of neurons in the hidden layer is 32. As Table 5.1 demonstrates the prediction accuracy for unsupervised HMM even for two time series ($2 * 50 = 100$) samples is very accurate. Both DHMM and CHMM are outperforming LSTM when the number of samples is low. HMM has a significantly lower number of parameters than a simple one layer LSTM. The correlation between the two observations is -0.87, which shows they are negatively correlated. Therefore two different HMMs will be trained, one for each set of observations, and the final prediction will be a combination of both of them. This method has an advantage that predictions can be weighted with greater weights to more important observations. The unsupervised DHMM provides accuracy comparable to the unsupervised CHMM, which shows that if the discretization is done correctly, there is not much information loss.

We observe that although under case I, there are two confounding variables per time step, Markov's assumption for states hold and observations are conditionally independent. To make enough samples for HMM and LSTM prediction, instead of increasing the number of cases (N), we can increase the length of time series (T). This is a crucial observation since, in many domains such as education, it is hard to have a large number of time series and sequence data, but it is rather easier to increase the length of time series. Here we generate synthetic time series, and so we have access to labels (states); therefore, we should manually set the number of states and find the optimal number of mixture components, which is less optimal to perform model selection over states and mixture components. Finally, if discretizing is done correctly, i.e., finding the optimal number of clusters, this can be used as an alternative solution to fit a Gaussian or mixture of Gaussian distributions to data.

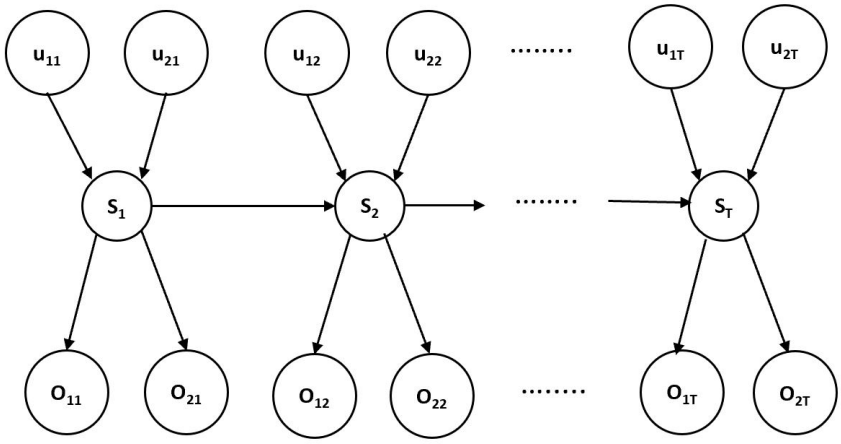


Figure 5.2: Synthetic Time Series Under Case I

Table 5.1: LSTM Prediction Accuracy and Number of Parameters for Case I

Architecture	Number of Parameters	Training Ratio	Number of Training Sample	Accuracy(%)
Model 1	379	0.8	80000	98.55
		0.3	30000	98.34
		0.1	10000	98.1
	4579	0.005	500	93.63
		0.001	100	64.2
Model 2	971	0.8	80000	98.4
		0.3	30000	98.12
		0.1	10000	98.06
	12899	0.005	500	93.83
		0.001	100	69.5
Model 3	755	0.8	80000	98.25
		0.3	30000	98.06
		0.1	10000	98.06
	9155	0.005	500	94.23
		0.001	100	76.68

Case II

This case corresponds to the figure 3 and is more complex than case 1 in terms of interconnection between the nodes. In this network, the Markov assumption does not hold since

$$S_{n+1} \not\perp\!\!\!\perp S_{n-1} | S_n \tag{5.3}$$

The prediction procedure is similar to the case I and is summarized in Tables 5.3 and 5.4. As Tables 5.3 and 5.4 show, the prediction accuracy for this case, especially for an unsupervised HMM, is lower than the case I, and this is because the Markov assumption no longer holds for this network. However, unsupervised HMM still provides a reasonable estimate and can outperform

Table 5.2: HMM Prediction Accuracy and Number of Parameters for Case I

Model Type	Number of Parameters	Training Ratio	Number of Training Sample	Accuracy(%)
Unsupervised CHMM	46	0.8	80000	93.89
	28	0.3	30000	92.24
		0.1	10000	88.50
		0.005	500	84.01
		0.001	100	81.23
Unsupervised DHMM	28	0.8	80000	90.89
		0.3	30000	87.81
		0.1	10000	86.51
		0.005	500	78.32
		0.001	100	67.01
Supervised DHMM	28	0.8	80000	96.78
		0.3	30000	96.39
		0.1	10000	96.12
		0.005	500	96.12
		0.001	100	96.12

LSTM when there are only 200 and 40 samples. The supervised HMM is done by first discretizing the observations according to K-means to five clusters. The elbow and Silhouette methods are used to verify the optimal number of clusters. Comparing the number of parameters between different models suggests that the HMM is much more efficient and less complex than a simple one layer LSTM. Moreover, the sub-optimal unsupervised HMM provides a reasonable prediction even when the structure is not the tree, and the Markov assumption is not satisfied. As Table 5.4 shows again there is not much loss in terms of prediction accuracy after discretizing the observation.

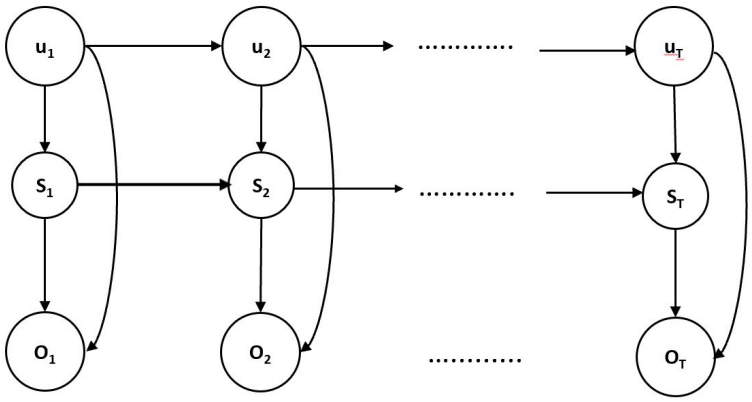


Figure 5.3: Synthetic Time Series Under Case II

Case III

This case corresponds to the figure 5.4 and is much more complex than cases I and II. In this

Table 5.3: LSTM Prediction Accuracy and Number of Parameters for Case II

Architecture	Number of Parameters	Training Ratio	Number of Training Sample	Accuracy(%)
Model 1	356	0.8	32000	88.75
		0.3	12000	87.33
		0.1	4000	87.20
	4484	0.005	200	58.28
		0.001	40	38.23
Model 2	900	0.8	32000	88.52
		0.3	12000	88.36
		0.1	4000	87.32
	12804	0.005	200	58.47
		0.001	40	33.46
Model 3	708	0.8	32000	87.62
		0.3	12000	87.23
		0.1	4000	87.19
	8964	0.005	200	62.47
		0.001	40	38.12

Table 5.4: HMM Prediction Accuracy and Number of Parameters for Case II

Model Type	Number of Parameters	Training Ratio	Number of Training Sample	Accuracy(%)
Unsupervised CHMM	47	0.8	32000	75.21
	23	0.3	12000	74.5
		0.1	4000	72.14
		0.005	200	70.04
		0.001	40	74.12
Unsupervised DHMM	27	0.8	32000	70.07
		0.3	12000	71.36
		0.1	4000	70.14
		0.005	200	62.83
		0.001	40	60.15
Supervised DHMM	31	0.8	32000	88.91
		0.3	12000	88.15
		0.1	4000	88.04
		0.005	200	85.09
		0.001	40	80.05

network, the Markov assumption is not satisfied. Tables 5.5 and 5.6 summarizes the prediction accuracy and the number of parameters for this case. In this case, N is 1000, and T is 10. Some interesting observations from this case are:1- Lower prediction accuracies for the HMM and LSTM for this case suggest that a more complex model is needed to model time series. 2- Adding more layers to the LSTM does not necessarily increase the prediction accuracy but can significantly increase the model complexities. 3- Unsupervised HMM, even when the number of states is not optimal, can compete with the supervised HMM or LSTM.

Case IV

This case corresponds to the figure 5.5. In this network, the Markov assumption is not satisfied, and

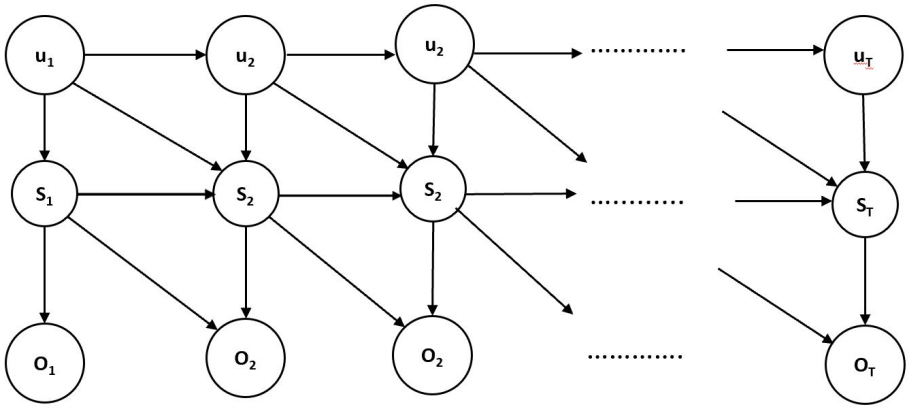


Figure 5.4: Synthetic Time Series Under Case III

Table 5.5: LSTM Prediction Accuracy and Number of Parameters for Case III

Architecture	Number of Parameters	Training Ratio	Number of Training Sample	Accuracy(%)
Model 1	4484	0.8	8000	61.59
		0.3	3000	58.36
		0.1	1000	56.12
		0.005	50	33.84
		0.001	10	30.23
Model 2	12804	0.8	8000	62.33
		0.3	3000	61.25
		0.1	1000	59.51
		0.005	50	34.78
		0.001	10	29.10
Model 3	8964	0.8	8000	62.32
		0.3	3000	61.12
		0.1	1000	59.4
		0.005	50	34.26
		0.001	10	30.12

state dependencies are over two time steps. Tables 5.7 and 5.8 summarizes the prediction accuracy and the number of parameters for this case. In this case, N is 2000, and T is 20. Some observations from this case are:1- DHMM provides comparable accuracy to CHMM if discretization is done correctly. 2- Supervised HMM outperforms LSTM for a lower number of samples even when the Markov assumption does not hold, and state dependencies are more than one step time.

Some observations from all these four cases are as follows: 1- HMM, unlike LSTM, can be used in an unsupervised domain to model sequential and time series data. This is important especially since the amount of labeled time series is limited.2- HMM is a model-based generative model that operates based on the first-order Markov assumption, however LSTM is a data-driven

Table 5.6: HMM Prediction Accuracy and Number of Parameters for Case III

Model Type	Number of Parameters	Training Ratio	Number of Training Sample	Accuracy(%)
Unsupervised CHMM	35	0.8	8000	60.8
	23	0.3	3000	60.15
		0.1	1000	58.4
		0.005	50	51.74
		0.001	10	37.67
Unsupervised DHMM	31	0.8	8000	60.95
		0.3	3000	60.01
		0.1	1000	57.9
		0.005	50	50.16
		0.001	10	37.2
Supervised DHMM	35	0.8	8000	61.32
		0.3	3000	60.10
		0.1	1000	60.10

discriminative model that can learn information from arbitrarily long time-steps. 3- HMM has significantly fewer parameters than LSTM, which makes it easier to interpret.4- LSTM is a data-driven method and needs lots of samples for efficient training.5- LSTM has a huge number of hyper-parameters, such as the number of layers, batch-size, choice of the optimization method, learning rate, and the number of neurons per layer. It is very hard, and there is no ground rule to find optimal parameters in LSTM. 6- HMM is very sensitive to the initial condition due to the EM algorithm; therefore, the model might never converge to the global optima. One way to overcome this issue is to start the EM algorithm from multiple different initial conditions and choose the model that optimizes some criteria. 7-Supervised HMM provides a very high prediction accuracy even when the Markov assumption fails, and state dependencies are more than one time step.

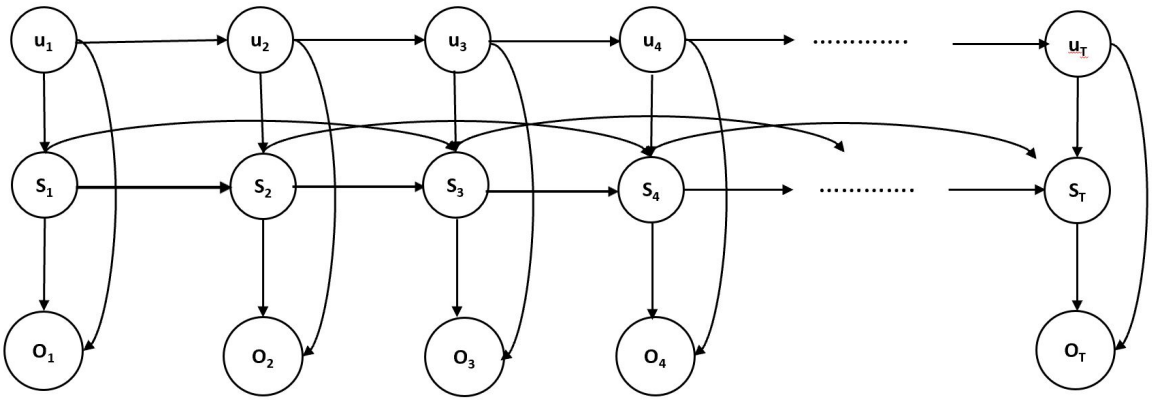


Figure 5.5: Synthetic Time Series Under Case IV

Table 5.7: LSTM Prediction Accuracy and Number of Parameters for Case IV

Architecture	Number of Parameters	Training Ratio	Number of Training Sample	Accuracy(%)
Model 1	4484	0.8	32000	82.97
		0.3	12000	81.68
		0.1	4000	80.54
		0.005	200	36.84
		0.001	40	34.33
Model 2	12804	0.8	32000	83.30
		0.3	12000	81.92
		0.1	4000	80.70
		0.005	200	43.36
		0.001	40	35.07
Model 3	8964	0.8	8000	82.90
		0.3	12000	81.72
		0.1	4000	81.01
		0.005	200	40.07
		0.001	40	33.89

Table 5.8: HMM Prediction Accuracy and Number of Parameters for Case IV

Model Type	Number of Parameters	Training Ratio	Number of Training Sample	Accuracy(%)
Unsupervised CHMM	59	0.8	32000	68.4
	23	0.3	12000	66.2
		0.1	4000	63.47
		0.005	200	61.12
		0.001	40	58.12
Unsupervised DHMM	27	0.8	32000	64.17
	35	0.3	12000	62.58
	27	0.1	4000	62.40
	35	0.005	200	60.12
	35	0.001	40	57.89
Supervised DHMM	35	0.8	32000	83.89
		0.3	12000	83.77
		0.1	4000	83.11
		0.005	200	81.36
	23	0.001	40	50.91

5.4 Conclusion

In this section, we created synthetic time series corresponding to various DBN structures with different degrees of complexity. We calculated and compared the prediction accuracies and number of parameters for the LSTM and supervised and unsupervised HMM. We showed that even an unsupervised HMM can be a reliable method when the amount of labeled data is limited. Furthermore, we showed that unsupervised DHMM can provide comparable prediction performance to unsupervised CHMM when observations are continuous. We also showed that a supervised DHMM outperforms LSTM and produces reliable and accurate predictions when the number of samples is limited. We proposed a method to discretize the observation under two different scenarios, i.e., 1-

The network structure is changing with time 2- network structure remains the same. The second option is used in synthetic data generation as it significantly reduces the number of parameters.

Chapter 6

CAUSAL INFERENCE FOR EXPERIMENTAL AND QUASI-EXPERIMENTAL DESIGNS IN EDUCATION

In this section, first, we use causal DAG and graphical modeling frameworks to quantify the experimental and quasi-experimental studies. We study some well-known experimental and quasi-experimental designs in education and discuss their limitations in terms of threats to internal and external validities mentioned by [22]. Second, we propose more intelligent experimental designs by collecting richer data from students by including questions on potential confounders in the diagnostic test and instructing the TAs and instructors to inquire during office hours about potential confounders. Third, we propose to model the education system as time-varying treatments, confounders, and time-varying treatments-confounders feedback. We show that if we control for a sufficient set of confounders and use appropriate techniques such as the marginal structural model along with IPTW [52] or g-formula, which is the extension of Pearl's backdoor criterion, we can close a sufficient set of backdoor paths and derive the unbiased causal estimate of joint interventions on the outcome.

6.1 Experimental Designs in Education

Experimental designs are designs where the researcher has complete control over the study. Their main characteristic is randomization, where the participants are assigned to control and treatment groups randomly. Perhaps the most well-known type of experimental design is the randomized controlled trial (RCT). It assigns the participants to the control and the treatment groups randomly. Hence, it will remove the confounding variables from the study. The ideal RCT (full compliance) is considered the gold standard in medicine; however, in practice, participants might leave the study

or do not comply with the treatment, which creates a bias in the study. Figure 6.1a demonstrates a candidate DAG for the RCT under the non-compliance, where R is the instrument and denotes the randomization to receive the treatment, X is whether you actually receive the treatment, L and W are the confounders (They may be observed or unobserved), and Y is the outcome. For example, in education, X can be either solving some problems, reading sections of a book, or getting a referral to an expert; L and W are either the observed or unobserved confounders such as family issues, socioeconomic status, prior knowledge and stress, and psychological problems. Y denotes the outcome of interest, for example, the exam score. In figure 6.1a X is a collider, hence conditioning on X will open the non-causal path $R \rightarrow X \leftarrow L \rightarrow Y$ and $R \rightarrow X \leftarrow W \rightarrow Y$.

Figure 6.1b shows the same DAG but an ideal RCT, where the effect of confounding variables is removed. In figure 6.1b, R and X are the same, since if you are randomized to receive the treatment, then you will receive the treatment (Full-compliance)

However, even if the RCT is done with full compliance, it still suffers from the following limitations:

- It is expensive. In education usually funding is limited, which makes the RCT not practical as a tool for widespread use.
- It requires careful planning and correct strategy to keep participants in the study and perform the correct randomization.
- It is unfeasible and unethical in many situations. For example, you may not be able to force students to talk to the TA or visit a counselor.

Sequentially Multiple Assignment Randomized Trial (SMART) [53] is another type of experimental design that consists of randomization at each stage. Figure 6.2 shows a proposed SMART design in education, where initially students are randomly assigned to three groups based on interventions they receive:1- Students who are recommended to solve some problems (SP), 2- Students are recommended to talk to the TA (TT). 3 - Students who are recommended to talk to a counselor (TC). Students then continue with the intervention for some duration, such as five weeks, and are

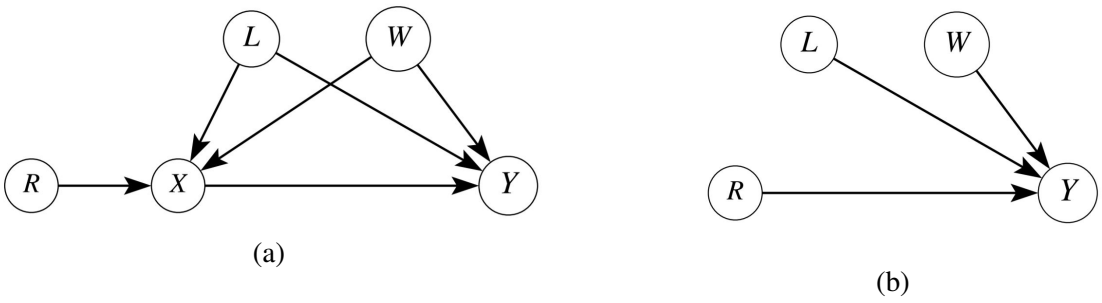


Figure 6.1: a) DAG for a RCT with non-compliance. b) DAG for RCT with full compliance

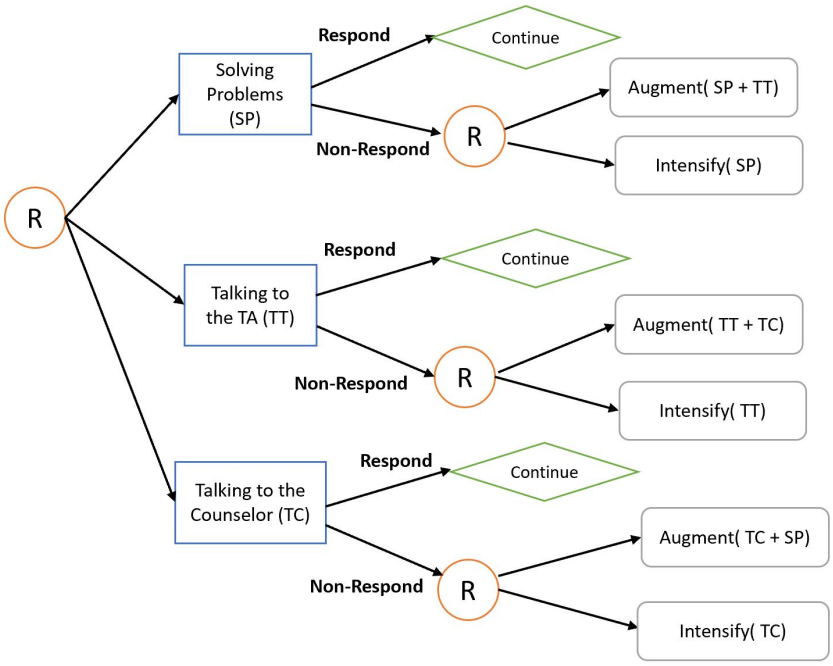


Figure 6.2: SMART data collection design in education

assessed after that. Those who respond to treatment, meaning that treatment is effective for them, continue with this treatment, and those who are non-respondent will be randomized again to students who receive the augmented treatments, namely combining the previous treatment with the new one and students who receive the previous treatment with the higher intensity. This process may continue until a certain mastery level for all students is achieved or until the duration of study is over. Figure 6.2 shows a two-stage SMART design with three treatment options at the first stage. Since SMART is a multi-stage randomization design, then similar to the RCT, it is costly and requires careful planning. Therefore it may not be a practical option for educational design. In the

next sections, we will discuss non-randomized (quasi-experimental) designs in education.

6.2 Non-randomized Designs with a Single Intervention

Non-randomized or quasi-experimental designs are designs used when the randomized experiment is not possible due to ethical, timing, or cost restrictions. They are sometimes also referred to as observational studies [54]. Figure 6.3 represents some well-known quasi-experimental designs in literature:

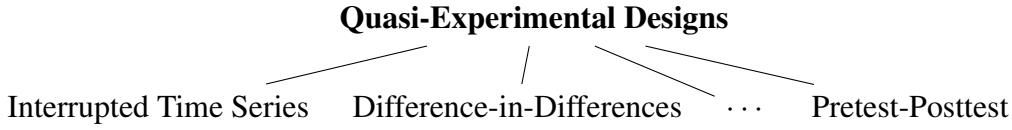


Figure 6.3: Some Well-Known Quasi-Experimental Designs

In [22], the authors introduced a four levels of hierarchy for quasi-experimental designs demonstrated in table 6.1. In Table 6.1, D is the best design, followed by C, B and A. Next, we discuss the use of designs in table 6.1 in educations.

Table 6.1: Hierarchy of Quasi-Experimental Design

Type of Experiment
A. Design with no control groups.
B. Design with control groups but no pretest.
C. Design with control groups and pretest
D. Design for time series data.

Note: The common notation is to use X for intervention, O for measurement (observation), and the NR for non-randomized assignment.The position from left to right determines the temporal order.

6.2.1 Designs with no control groups

1. **The One group posttest design:**

$$\text{X} \qquad O_1$$

Figure 6.4 shows an example of a DAG for this scenario. X is a treatment or an intervention, O_1 is an outcome or the posttest, W_1 is observed, and U_1 and U_2 are unobserved confounders. For

example, suppose a researcher is interested in finding the causal effect of doing some problem sets in a textbook on student learning by taking a final exam or posttest immediately after the study ends. Then X is doing the problem sets, and the O_1 is the final exam. This design is subject to many threats to internal validities such as history since other events besides the treatment might cause the change in exam score. Confounding variables such as prior knowledge of the topic, intelligence, family, and psychological problems (unobserved confounder) might cause a change in outcome. Lack of a pretest prevents query of students' prior knowledge and the absence of a control group, makes it difficult to determine if treatment is the only cause of change in the outcome. To better understand the problem with this design, consider the DAG in figure 6.4; ideally, we would like to find the direct causal effect of X on O_1; however, the X $\leftarrow W_1 \rightarrow O_1$ is a non-causal backdoor path, which should be closed. But, even blocking the backdoor path through W_1 still is not sufficient due to the backdoor path via unobserved confounders.

To better understand the problem with this design consider the following simulation. The DAG

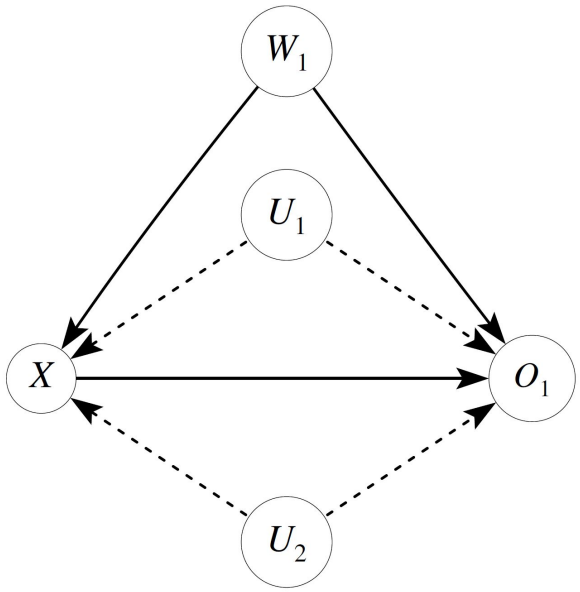


Figure 6.4: DAGs for a single posttest design

for the data generating process is the same as figure 6.4. The treatment and all the confounders (observed and unobserved) are Bernoulli random variables and the outcome O_1 is normally distributed with mean to be the linear combination of X, U_1, U_2 and W_1 and the standard deviation is

0.6.

Simulation Setup

$$
\begin{aligned}
N &= 500 \\
P(W_1 = 1) &= 0.8 \\
P(U_1 = 1) &= 0.4 \\
P(U_2 = 1) &= 0.6 \\
P(X = 1) &= 0.30 + 0.2U_1 + 0.5U_2 - 0.3W_1 \\
\mu &= 5 + 7X + 4W_1 - 2U_1 - 2U_2 \\
\sigma &= 0.6 \qquad O_1 \sim N(\mu, \sigma^2)
\end{aligned}
$$

Table 6.2 presents simulation results for this design. According to this table, the causal effect of

Table 6.2: Simulation Result for a design with a single intervention

Control for No Confounders		
Coefficient	True	Estimate
Intercept	5.0	7.13
X	7.0	5.04
Control for W_1		
Intercept	5.0	3.88
X	7.0	5.58
Control for W_1 **and** U_1		
Intercept	5.0	4.27
X	7.0	6.55
Control for W_1**,** U_1 **and** U_2		
Intercept	5.0	4.99
X	7.0	7.03

X on O_1 will be biased if we do not control for unobserved confounders.

2. **The One-Group Pretest-Posttest Design:**

$$O_1 \qquad \text{X} \qquad O_2$$

Figure 6.5a is a candidate DAG for this case, where the outcome O_2 is the posttest score (It can be a continuous or discrete quantity). U_1 and U_2 are unobserved confounders and can refer

to psychological and family problems. W_1 represents any observed confounder. This is similar to the DAG in figure 6.4. This is because, in causal inference, we study the causal effect of intervention/treatment on the outcome and the pretest-posttest design has similar characteristics as one-group posttest design after the intervention. Including a pretest, however, can provide some information about the student's prior knowledge, which in turn helps us to control for some confounders compared to the first design. However, repeating the same problems in the consecutive exams results in testing/practice effects that threatens internal validity. This design is also susceptible to other threats to internal validities such as maturation and history. This is because participants may drop out during the study (for example, after taking the pretest). This is sometimes referred to as a loss-to-follow-up, which causes the selection bias. This design also suffers from regression to the mean since if there is only one observation before the intervention (the pretest), then the score in the pretest can be due to chance. Therefore, the pretest would be a very weak counterfactual had the treatment never happened. Figure 6.5b is another candidate DAG, which includes the pretest O_1 and L_1 as a covariate causing students to drop from the study. C_1 is a binary variable that represents the censoring where $C_1 = 0$ students are not censored and continue the study.

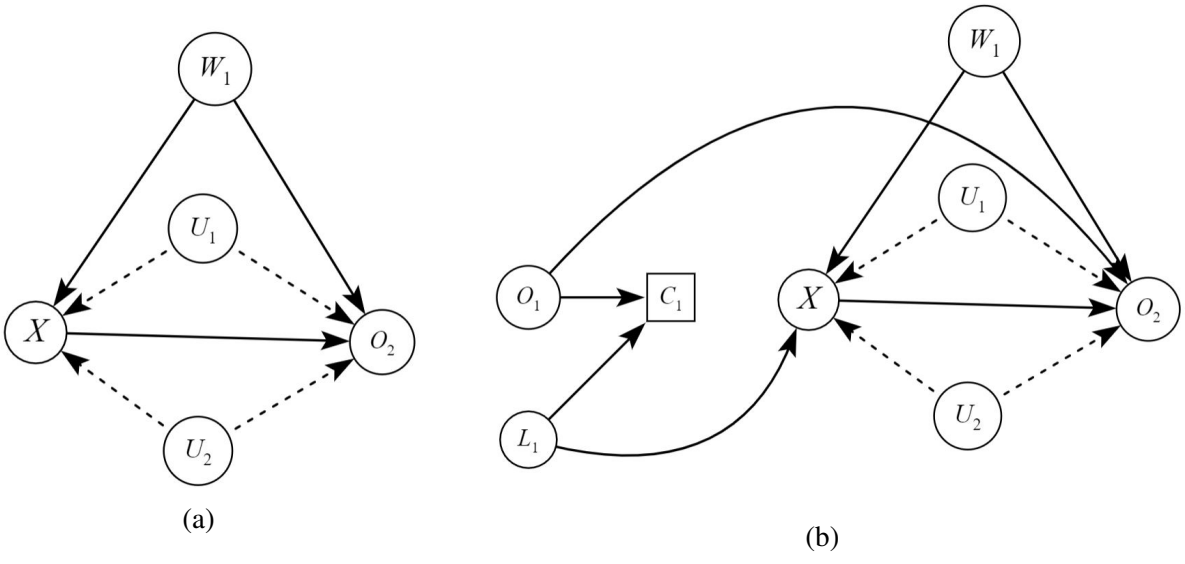


Figure 6.5: DAGs for a pretest-posttest Design

6.2.2 Designs with a control group but no pretest

3. **Posttest-Only Design With Nonequivalent Groups:**

NR	X	O_1
NR		O_1

This is similar to a one-group posttest design with the difference of adding a control group. The control group is supposed to provide a counterfactual outcome for the treated group had they not received the treatment. The DAG for this is demonstrated in figure 6.6 and is similar to the DAG in figure 6.4 with the inclusion of variables Z. Z is a variable indicating the treatment assignment. It determines the conditions on how to assign the participants to treatment and control groups, such as choosing some classrooms for the treatment and some others for the control group. One major problem with the nonequivalent design is the selection bias since the participants are not assigned randomly. Therefore, the participants in the control and treatment groups do not share identical characteristics. For example, assume participants in treatment groups are from more educated family backgrounds, or they are academically stronger candidates. Therefore they perform better in the posttest even under the null treatment effect.

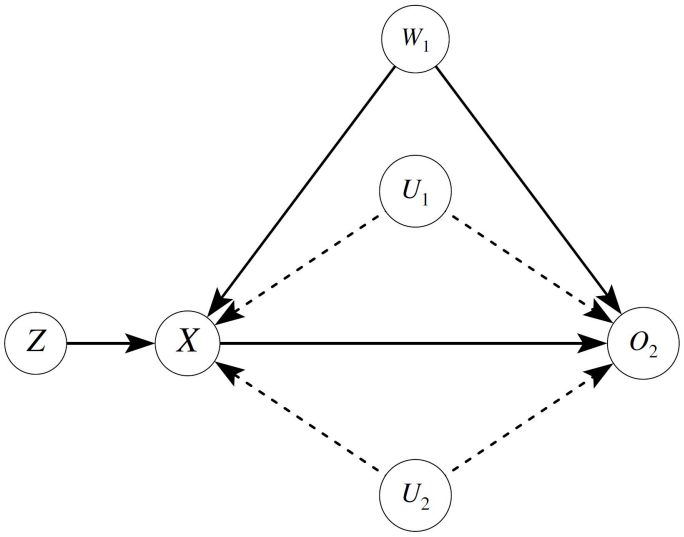


Figure 6.6: The DAG for Posttest-Only Design With Nonequivalent Groups

6.2.3 Designs with both control groups and pretests

4. **The Pretest-Posttest Design with Nonequivalent Groups:**

NR	O_1	X	O_2
NR	O_1		O_2

This is a common design in the educational systems. Students are divided into treatment and control groups based on some factors, such as the classroom. The DAG after the intervention for this design is similar to figure 6.6. The control and pretest provide a counterfactual for the treated group and help to remove some of the threats to the internal validities. However some flaws with this design are as follows: 1- A single intervention may not be sufficient to influence the outcome (posttest). 2- Since the participants are not randomly assigned, the treatment and control groups may have very different characteristics (confounder). 3- A single observation before and after the intervention is not enough to draw any conclusion about the participants' mastery of the subject and is prone to regression to the mean threats. 4- Since there is a single intervention then there is no opportunity to provide feedback to students.

6.2.4 Design for time series data

5. **Interrupted Time series Design**

O_1	O_2	...	O_n	X	O_{n+1}	. . .	O_{2n}

Interrupted time series (ITS) design consists of repeated measurements over time with an intervention, or a shock is introduced at a certain time. It is considered the most powerful design in the hierarchy of quasi-experimental designs. This is because it can rule out many threats to internal validity such as regression to the mean since many measurements are taken before and after the intervention; therefore, the probability that the result is only due to chance is very slim. However, history and instrumentation are still considered as threats to internal validities since due to administrative change, the measure for student learning can change through time. Attrition is a big threat for this design since the loss-to-follow-up can happen after each measurement. Therefore the number of participants from the beginning to the end of the study can be very different.

Figure 6.7 shows a candidate DAG for this design for four measurements (2-time points before and 2-time points after the intervention). The O_1, O_2, . . ., O_4 denote the measurements, L_1, L_2, . . ., L_4 are the time-varying covariates such as family, financial or prerequisite problems that influence the outcome and treatment. It is important to note that figure 6.7 is a simplified DAG for most real-world phenomena. Interrupted time series (ITS) is a type of design used when a randomized

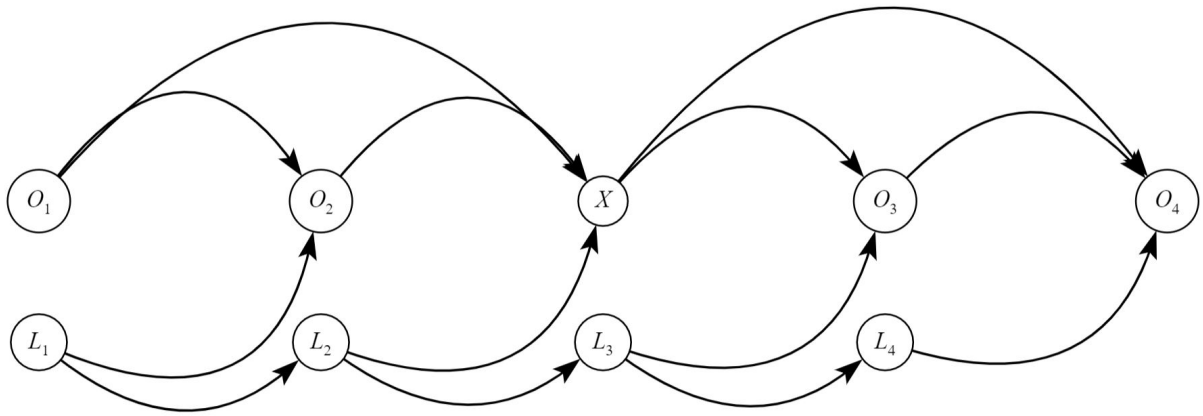


Figure 6.7: A DAG for the Interrupted Time Series Design

experiment is either too expensive or impractical. ITS measures the change in intercept and slope of the time series before and after the intervention. It comprises two distinct periods, namely the pre-intervention and the post-intervention, and may include the control group. There are two major methods to model the ITS design. The segmented regression by far is the most widely used one, in which a linear regression model is fit for before and after intervention with a dummy variable representing the pre and post-intervention phase. This model assumes a linear trend before and after intervention time [55], [56],and [57]. The segmented regression technique assumes the residual term to be normally distributed, but if residuals are autocorrelated, then autoregressive integrated moving average (ARIMA) can be used to model the ITS [55], [58], and [59]. This method works by first building the ARIMA model of the time series using only the pre-intervention series. Then apply the model for the complete time series data and add the intervention as a dummy variable, where 1 is coded as the post-intervention phase, and 0 is for the baseline period. ARIMA only

handles the univariate time series data. ITS can be used in education to evaluate the influence of a single intervention introduced in a semester as a form of a study skill or measure the influence of adding an online textbook.

6.2.5 Regression Discontinuity (RD)

C O_1 X O_2

C O_1 O_2

RD is another type of quasi-experimental design used when there is some kind of criterion (threshold) that must be met before participating in the intervention. This threshold identifies the eligibility for participation in the program.

It is a type of pretest-posttest comparison group, where participants are assigned to treatment and comparison groups based on the threshold or cutoff on a pretest. This can be represented mathematically as follows: A is called the running variable and C is the threshold. If A is above the threshold then you receive the intervention.

$$X = \begin{cases} 1 & A \geq C \\ 0 & A < C \end{cases}$$

Figure 6.8 represents a candidate DAG for RD design. According to this figure, A, L, and U are confounders (U is an unobserved confounder). and $X \leftarrow A \leftarrow L \rightarrow O$, $X \leftarrow A \leftarrow U \rightarrow O$, and $X \leftarrow A \rightarrow O$ are the backdoor paths. Therefore, one way to obtain the unbiased estimate of X on O is to control for A. In RD, we consider the effect of the intervention on the outcome when the running variable (A) is near the threshold (condition on A). Hence, by doing this, we will close all the backdoor paths.

In education, A can be a pretest, X is the intervention or treatment such as talking to a TA or a tutor, O is the posttest, L is the observed confounder such as the prerequisite knowledge, U is the unobserved confounder such as family background, and C is the threshold on the pretest for participation in the program. In RD, we study the influence of the intervention on the posttest score

for those participants whose pretest score is near the threshold.

To analyze the RD for the above problem in education, let A denote the pretest score,O denote the posttest score, and Z be a binary dummy variable indicating the treatment or control group. Then select the participants whose pretest score is just below and above the threshold, and regress the posttest score on the pretest and other variables.

$$O = \beta_0 + \beta_1 A + \beta_2 Z + \beta_3 AZ + \epsilon \tag{6.1}$$

The β_3 coefficient is the causal effect of A on O.

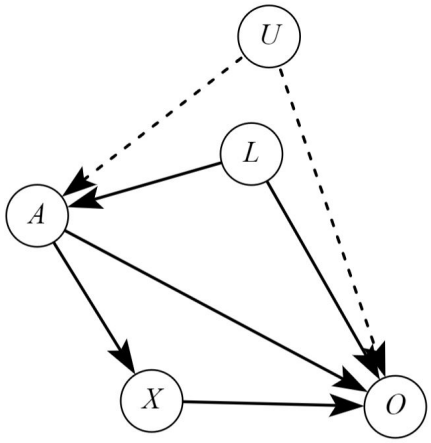


Figure 6.8: A DAG for Regression Discontinuity Design

Other popular quasi-experimental designs are Difference-in-differences (DID) and instrumental variable (IV) that can provide unbiased causal estimates when a randomized experiment is not possible [60],[61], [62],[63], [64], and [65].

Although quasi-experimental designs such as ITS, RD, DID, and IV provide unbiased causal estimates for many real-world phenomena, they may not be suitable options for education since 1- They mainly model a single intervention yet, learning is more effective using a sequence of interventions. 2- They do not model time-varying treatment and confounder feedback; therefore, they do not incorporate any mechanisms to reduce the negative effect of confounders through timely intervention.

Our objective is to have a tractable approach for the actual educational system by accounting

for all the potential confounders and design suitable interventions to reduce the negative influence of confounders through time. Thus, in the next section, we discuss observational studies in education and introduce techniques to overcome limitations of quasi-experimental designs with a single intervention.

6.3 Non-randomized Designs with Multiple Interventions in Education

Before discussing non-randomized designs with time varying interventions, We first define parameters and notation used in this section:

- $X_1, X_2, \ldots X_T$ are time varying treatments, such as:
  - Reading chapter of book.
  - Solving some problems.
  - Talking to the TA/instructor/counselor.
- $L_1, L_2, \ldots L_T$ are observed and $U_1, U_2, \ldots U_T$ are unobserved time varying confounders, such as:
  - Prior knowledge of subject.
  - Psychological/stress problems.
- O is the outcome. such as exam score.
- N is the total number of students.

The **Objective** is to find the causal effect of joint interventions X_1,X_2, . . . , X_T on O. The Joint effect of interventions is defined as a collection of direct effects unmediated by other interventions [66]. For example, the joint effect of interventions X_1 and X_2 on O in figure 6.9 consists of $X_1 \rightarrow O$ and $X_2 \rightarrow O$. In this section, we consider the following three cases, which characterize scenarios happening in real-world educational systems.

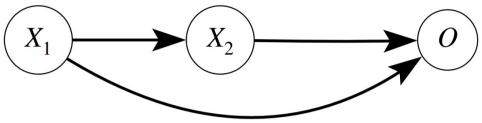


Figure 6.9: Joint Effect of X_1 and X_2 on O

1. Time-Varying Treatment and Confounder with no unmeasured confounders.

2. Time-Varying Treatment and Confounder with unmeasured confounders.

3. Time-Varying Treatment-Confounder feedback.

6.3.1 Time Varying Treatments and Confounders With No Unmeasured Confounders

Figure 6.10 presents an example of this scenario, where X_1, X_2 and X_3 are referred to as time-varying treatments/interventions. They can represent solving problems, talking to the TA, and talking to the counselor. L_1, L_2, and L_3 are observed confounders such as student background knowledge through time. O is the posttest exam. We use the outcome regression model to find the average causal effect of joint intervention X_1,X_2, and X_3 on O. The outcome regression measures the correlation coefficients but not necessarily the causal coefficients. Therefore we need to investigate if the correlation and causation are the same in equation 6.2. Controlling for L_1, L_2 and L_3 will close all the backdoors such as $X_1 \leftarrow L_1 \rightarrow O$, $X_2 \leftarrow L_2 \rightarrow O$, and $X_3 \leftarrow L_3 \rightarrow O$. Furthermore controlling for X_1, X_2 and X_3 will block the mediated and all other backdoor paths such as $X_2 \leftarrow X_1 \rightarrow O$. Hence the outcome regression in equation 6.2 results in unbiased estimates of causal coefficients. In equation 6.2, the multiplication terms are interaction terms, for example X_1X_2 suggests that the causal effect of X_1 on O is different for different values of X_2.

$$E[O|X_1, X_2, X_3, L_1, L_2, L_3] = \alpha_0 + \alpha_1 X_1 + \alpha_2 X_2 + \alpha_3 X_3 + \alpha_4 X_1 X_2 + \alpha_5 X_1 X_3 + \alpha_6 X_2 X_3 + \alpha_7 X_1 X_2 X_3 + \alpha_8 L_1 + \alpha_9 L_2 + \alpha_{10} L_3 \quad (6.2)$$

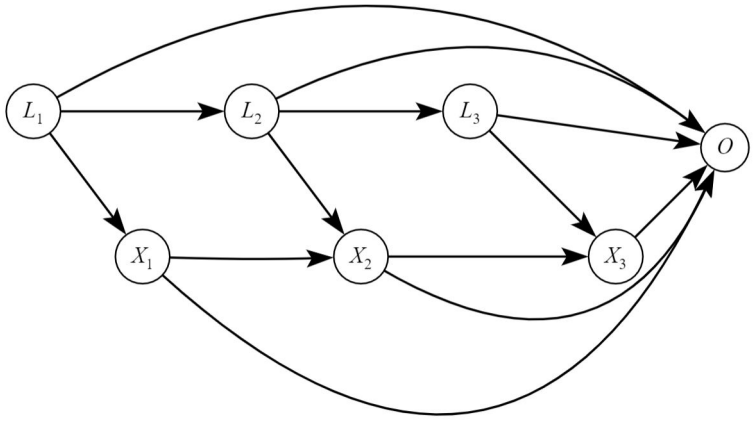


Figure 6.10: Time Varying Treatment and Confounder for three time points

The following is a simulation setup for this case:

$$
\begin{aligned}
N =& 500 \\
P(L_1 = 1) =& 0.5 \\
P(X_1 = 1) =& 0.2 + 0.4L_1 \\
P(L_2 = 1) =& 0.2 + 0.6L_1 \\
P(X_2 = 1) =& 0.3 + 0.5X_1 + 0.2L_2 \\
P(L_3 = 1) =& 0.3 + 0.5L_2 \\
P(X_3 = 1) =& 0.3 + 0.4X_2 + 0.1L_3 \\
\mu =& 0.2 + 2X_1 + 6X_3 + 5X_2 + X_1X_2 + X_1X_3 + X_2X_3 + 4L_1 + 4L_2 + 3L_3 \\
\sigma =& 0.2 \qquad O \sim N(\mu, \sigma^2)
\end{aligned}
$$

N represents number of cases, confounders and treatments are all binary variables and the posttest is normally distributed. Table 6.3 shows the simulation result for this scenario. It indicates all the coefficients except $X_1X_2X_3$ are significant for P-value of 0.05. Therefore the coefficient of $X_1X_2X_3$ may be assumed to be zero and the regression model estimates all coefficients correctly. Table 6.3 only includes the coefficients for treatments and their interaction terms and not any information about the confounders. Table 6.4 shows the 95% confidence interval (CI) for the coefficients using bootstrapping. According to this table the true values for all the coefficients lie

within the 95% confidence interval. The results in this section suggest that if all the confounders

Table 6.3: Simulation Result for Time-Varying Treatments, Confounders with No Unmeasured Confounders

Coefficient	True	Estimate	P-Value
Intercept	0.2	0.17	0.0008
X_1	2	1.90	2×10^{-16}
X_2	5	4.98	2×10^{-16}
X_3	6	6.02	2×10^{-16}
X_1X_2	1	0.91	2×10^{-7}
X_1X_3	1	1.34	9.6×10^{-7}
X_2X_3	1	0.90	1.9×10^{-7}
$X_1X_2X_3$	0	-0.091	0.81

Table 6.4: Simulation Result for Time-Varying Treatments, Confounders with No Unmeasured Confounders

Coefficient	True	95% CI
Intercept	0.2	$(0.0702, 0.2826)$
X_1	2	$(1.710, 2.302)$
X_2	5	$(4.875, 5.336)$
X_3	6	$(5.826, 6.180)$
X_1X_2	1	$(0.4688, 1.2230)$
X_1X_3	1	$(0.755, 1.918)$
X_2X_3	1	$(0.5542, 1.1484)$
$X_1X_2X_3$	0	$(-0.8461, 0.4669)$

are measured properly, then the unbiased causal effect of joint interventions on the outcome can be estimated accurately. Next, we investigate what would happen if there are unmeasured confounders.

6.3.2 Time Varying Treatments and Confounders With Unmeasured Confounders

Figure 6.11 presents an example of this scenario, where X_1, X_2 and X_3 are referred to as time varying treatments/interventions. Similar to the figure 6.10 interventions can represent solving problems, talking to the TA, and talking to the counselor. L_1, L_2 and L_3 are observed confounders such as student background knowledge through time. U_1, U_2 and U_3 are unobserved confounders such as family and psychological problems. O is the posttest exam. To find the average causal

effect of joint intervention X_1,X_2, and X_3 on O, we use the same regression model as in the previous case with all observed confounders. We need to investigate if the correlation and causation are the same in equation 6.3. Controlling for L_1, L_2 and L_3 will close backdoors such as $X_1 \leftarrow L_1 \rightarrow O$, $X_2 \leftarrow L_2 \rightarrow O$, and $X_3 \leftarrow L_3 \rightarrow O$. Furthermore controlling for X_1, X_2 and X_3 will block the mediated and all other backdoor paths such as $X_1 \leftarrow X_2 \rightarrow O$. However any backdoors through unobserved confounders such as $X_1 \leftarrow U_1 \rightarrow O$ and $X_2 \leftarrow U_2 \rightarrow O$ remain open. Hence the outcome regression in equation 6.3 results in biased estimate of causal coefficients. Note equations 6.2 and 6.3 are the same since unobserved variables cannot be included in the regression model.

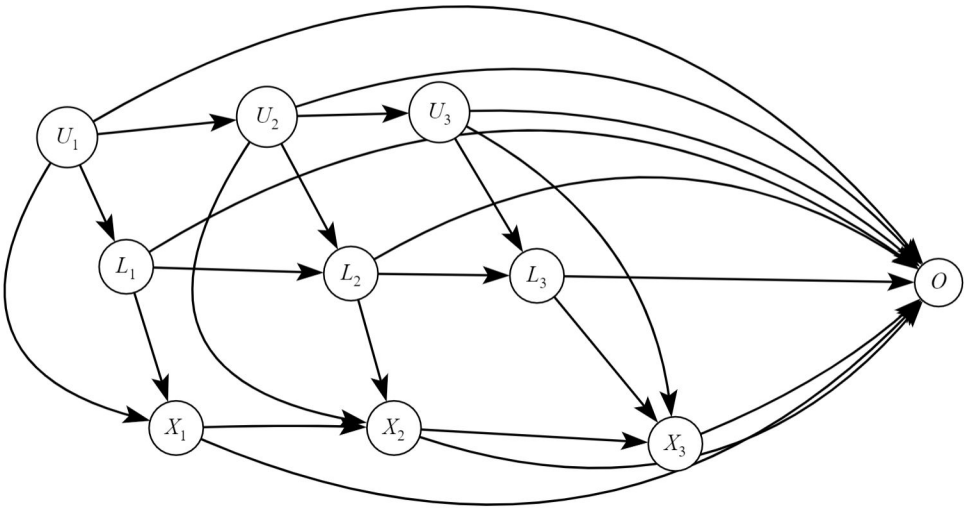


Figure 6.11: Time Varying Treatment and Confounder for three time points with Unobserved Confounders

$$E[O|X_1, X_2, X_3, L_1, L_2, L_3] = \alpha_0 + \alpha_1 X_1 + \alpha_2 X_2 + \alpha_3 X_3 + \alpha_4 X_1 X_2 + \\ \alpha_5 X_1 X_3 + \alpha_6 X_2 X_3 + \alpha_7 X_1 X_2 X_3 + \alpha_8 L_1 + \alpha_9 L_2 + \alpha_{10} L_3 \tag{6.3}$$

The following is a simulation setup for this case:

$$N = 500$$

$$P(U_1 = 1) = 0.6 \qquad P(U_2 = 1) = 0.6 + 0.3U_1$$

$$P(U_3 = 1) = 0.4 + 0.3U_2 \qquad P(L_1 = 1) = 0.5 + 0.4U_1$$

$$P(X_1 = 1) = 0.2 + 0.4L_1 + 0.3U_1 \qquad P(L_2 = 1) = 0.2 + 0.6L_1 + 0.1U_2$$

$$P(X_2 = 1) = 0.3 + 0.2X_1 + 0.2L_2 + 0.2U_2 \qquad P(L_3 = 1) = 0.3 + 0.5L_2 + 0.1U_3$$

$$P(X_3 = 1) = 0.1 + 0.4X_2 + 0.2L_3 + 0.2U_3$$

$$\mu = 4L_1 + 4L_2 + 3L_3 + 5X_2 + 2X_1 + 6X_3 + X_1X_2$$

$$X_1X_3 + X_2X_3 + 2U_1 + 3U_2 + U_3 + 0.2$$

$$\sigma = 0.2 \qquad O \sim N(\mu, \sigma^2)$$

Table 6.5 shows the simulation result for this case. According to this table the coefficients of X_1X_2, X_1X_3, X_2X_3, and $X_1X_2X_3$ are not statistically significant for P-value of 0.05. This is not correct since we know the coefficients of X_1X_2, X_1X_3, X_2X_3 should not be zero. Table 6.6 shows the 95% confidence interval (CI) for the coefficients using bootstrapping. According to this table, the 95% confidence interval for the intercept, X_2 and X_3 do not contain the true values. Moreover, the confidence interval, in this case, is wider than the confidence interval when all the confounders are observed.

Table 6.5: Simulation Result for Time-Varying Treatments, Confounders with Unmeasured Confounders

Coefficient	True	Estimate	P-Value
Intercept	0.2	1.92	10^{-11}
X_1	2	2.97	2×10^{-11}
X_2	5	6.15	2×10^{-16}
X_3	6	7.01	2×10^{-16}
X_1X_2	1	1.24	0.056
X_1X_3	1	0.55	0.42
X_2X_3	1	0.16	0.80
$X_1X_2X_3$	0	0.37	0.67

Table 6.6: Simulation Result for Time-Varying Treatments, Confounders with Unmeasured Confounders

Coefficient	True	95% CI
Intercept	0.2	$(1.844, 2.967)$
X_1	2	$(1.657, 3.638)$
X_2	5	$(5.19, 7.062)$
X_3	6	$(5.486, 8.419)$
X_1X_2	1	$(-0.828, 3.084)$
X_1X_3	1	$(-0.212, 3.917)$
X_2X_3	1	$(-1.4414, 2.6802)$
$X_1X_2X_3$	0	$(-3.3632, 1.8823)$

The accuracy of the estimated coefficient depends on the influence of unobserved confounders. As an example, consider the following two case studies where the simulation setup is the same as before except the mean of the outcome μ varies as a function of unobserved confounders.

Case I :**Unobserved confounders have small influence on the outcome**:

$$\mu_1 = 4L_1 + 4L_2 + 3L_3 + 5X_2 + 2X_1 + 6X_3 + X_1X_2 + X_1X_3 + X_2X_3$$
$$+ 0.1U_1 + 0.2U_2 + 0.3U_3 + 0.2$$
$$\sigma_1 = 0.2$$
$$O_1 \sim N(\mu_1, \sigma_1^2)$$

Case II :**Unobserved confounders have huge influence on the outcome**:

$$\mu_2 = 4L_1 + 4L_2 + 3L_3 + 5X_2 + 2X_1 + 6X_3 + X_1X_2 + X_1X_3 + X_2X_3$$
$$+ 8U_1 + 9U_2 + 10U_3 + 0.2$$
$$\sigma_2 = 0.2$$
$$O_2 \sim N(\mu_2, \sigma_2^2)$$

Table 6.7 summarizes the result for these two case studies and clearly demonstrates the negative effect of unobserved confounders on the estimated coefficients. Under case II, the estimated coefficients are far from the true coefficients, and this is due to the large influence of open backdoor paths. The results in this section suggest that open backdoor paths due to unobserved confounders

Table 6.7: Simulation Result for Time-Varying Treatments, Confounders with Unmeasured Confounders (Comparison of Case I and II)

Coefficient	True	Estimate (Case I)	Estimate (Case II)
Intercept	0.2	0.41	8.97
X_1	2	2.08	7.17
X_2	5	5.12	7.91
X_3	6	5.84	10.05
X_1X_2	1	1.22	0.64
X_1X_3	1	1.34	1.98
X_2X_3	1	1.84	0.04
$X_1X_2X_3$	0	0.44	-1.58

can cause bias in the estimation of the causal coefficients. The amount of bias depends on how large is the influence of the unobserved confounders. This is another important conclusion in designing more intelligent and individualized educational systems that we need to measure and control for as many confounders as we can. It is true that no matter how many confounding variables we measure, there are still some unobserved confounders; however, if we can close for the confounding variables that might plausibly have a large influence on the outcome, then we reduce the bias in the causal coefficients significantly (refer to the case I and case II). In the next section, we discuss time-varying treatments, confounders feedback.

6.3.3 Time Varying Treatments and Confounders Feedback

Figure 6.12 presents an example of this scenario, where X_1, X_2 and X_3 are referred to as time-varying treatments/interventions. Similar to the previous two sections. interventions can represent solving problems, talking to the TA, and talking to the counselor. L_1, L_2, and L_3 are observed confounders such as student background knowledge through time. O is the posttest exam. We assume we control for a sufficient set of confounders, and there are no significant unobserved confounders. The difference between this DAG and the DAG in figure 6.10 is that there are arrows from treatments back to confounders. The motivation behind this design is as follows: assume some students are suffering from a psychological problem, we refer them to the counselor and are hoping through the counselor intervention we reduce the negative effect of the confounder.

Therefore a more intelligent design requires feedback from treatments to interventions. To find the average causal effect of joint intervention X_1,X_2, and X_3 on O, we start again by the same regression model as in previous cases with all observed confounders. We need to investigate if the correlation and causation are the same in equation 6.4 as in other cases. Controlling for L_1, L_2 and L_3 will close backdoors such as $X_1 \leftarrow L_1 \rightarrow O$, $X_2 \leftarrow L_2 \rightarrow O$, and $X_3 \leftarrow L_3 \rightarrow O$. Furthermore controlling for X_1, X_2 and X_3 will block the mediated and all other backdoor paths such as $X_2 \leftarrow X_1 \rightarrow O$. However controlling for L_2 will close the $X_1 \rightarrow L_2 \rightarrow O$ path, which is a causal path (neither mediated nor a backdoor paths), the same way controlling for L_3, will block the causal path $X_2 \rightarrow L_3 \rightarrow O$. Therefore we conclude that the coefficients of X_2 and X_3 are biased and outcome regression produces an unbiased coefficient for X_1.

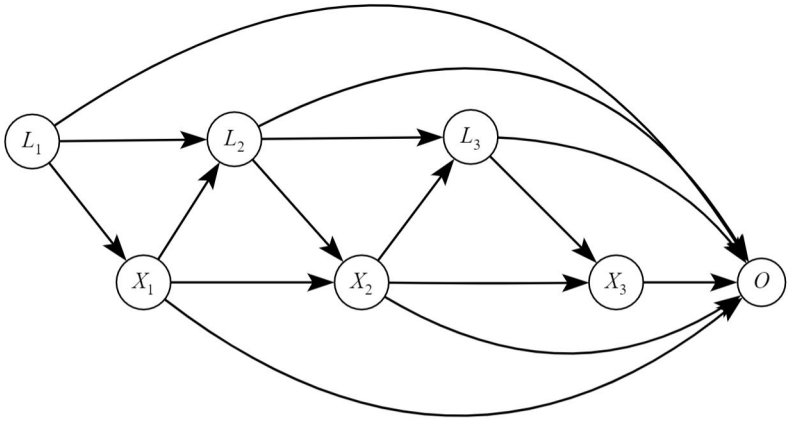


Figure 6.12: Time Varying Treatments and Confounders Feedback

$$E[O|X_1, X_2, X_3, L_1, L_2, L_3] = \alpha_0 + \alpha_1 X_1 + \alpha_2 X_2 + \alpha_3 X_3 + \alpha_4 X_1 X_2 + \\ \alpha_5 X_1 X_3 + \alpha_6 X_2 X_3 + \alpha_7 X_1 X_2 X_3 + \alpha_8 L_1 + \alpha_9 L_2 + \alpha_{10} L_3 \tag{6.4}$$

To find the unbiased causal estimates of joint interventions on the outcome in the case of time-varying treatments, confounders feedback, we use g-formula and the inverse probability of treatment weighting (IPTW). Before discussing how to do inference using g-formula and IPTW, we will briefly discuss each one in the next section.

G-formula and Inverse Probability of Treatment Weighting

G-formula is an extended version of the backdoor criterion or standardization discussed before in the time-varying treatments and confounders setting. Mathematically it is represented as follows:

$$E[O_{\bar{x}}] = \sum_{\bar{l}} E[O|\bar{X} = \bar{x}, \bar{L} = \bar{l}] \prod_{t=0}^{t=T} P(L_t = l_t|\bar{X}_{t-1} = \bar{x}_{t-1}, \bar{L}_{t-1} = \bar{l}_{t-1}) \tag{6.5}$$

where the $\bar{X}$ and $\bar{L}$ represent the treatments and confounders history, T is the number of time steps or the number of interventions. This model requires estimating the conditional expectation and conditional probability from data. Hence it is sensitive to model misspecification and requires correct models for conditional expectation and conditional probabilities.

IPTW is a method to weigh each observation by the inverse of the probability of treatment received. IPTW creates what is called the pseudo-population, which is a confounder-free population. Therefore IPTW is trying to emulate a randomized experiment. The IPTW weights for the subject i is defined as follows [66]:

$$W_i = \frac{1}{\prod_{t=0}^{t=T} P_{X_t|\bar{X}_{t-1},\bar{L}_t}(X_{t,i}, \bar{X}_{t-1,i}, \bar{L}_t, i)} \tag{6.6}$$

$$SW_i = \frac{\prod_{t=0}^{t=T} P_{X_t|\bar{X}_{t-1}}(X_{t,i}|X_{t-1,i})}{\prod_{t=0}^{t=T} P_{X_{t,i}|\bar{X}_{t-1,i},\bar{L}_t}(X_{t,i}, \bar{X}_{t-1,i}, \bar{L_t}, i)} \tag{6.7}$$

The SW_i is referred to as the stabilized weight. It results in a more stable estimator if the denominator of W_i is close to zero. However, a better way to express the IPTW weights is through a causal DAG. The denominator of IPTW weights (It is the same for W_i and SW_i) is the probability distribution of the treatments in the pre-intervention DAG since the treatment at the current time point is conditioned on treatment history and the confounder history. The numerator of the SW weight is the probability distribution on the post-intervention DAG since it is independent of confounders; hence the connections from confounders to treatments are removed. Therefore the stabilized weight can be represented as follows:

$$IPTW_{SW} = \frac{\text{Distribution of Treatment on Post-intervention DAG}}{\text{Distribution of Treatment on pre-intervention DAG}} \tag{6.8}$$

IPTW is used to estimate the parameters of marginal structural model (MSM) [52], [67], and [68].

MSM, as the name suggests, is a marginal model, namely not conditioned on any confounding variables. It is a model for calculating the mean potential outcome for the entire population. To see how g-formula and IPTW can solve the time-varying treatments and confounders feedback problems, consider the following simulation setup corresponding to the DAG in figure 6.12 with the difference that we assume the first treatment X_1 is assigned randomly; hence there is no confounder connected to the X_1:

$$N = 500$$

$$P(X_1 = 1) = 0.6$$

$$P(L_2 = 1) = 0.1 + 0.6X_1$$

$$P(X_2 = 1) = 0.3 + 0.7L_1 - 0.25X_1$$

$$P(L_3 = 1) = 0.2 + 0.3L_2 + 0.3X_2$$

$$P(X_3 = 1) = 0.1 + 0.3L_3 + 0.1X_2$$

$$\mu = 5 + 4X_1 + 5X_2 + 6X_3 + 10X_1X_2 + 11X_1X_3 + 12X_2X_3 - 8L_1 - 9L_2$$

$$\sigma = 0.2 \qquad O \sim N(\mu, \sigma^2)$$

As it was discussed, the coefficients of X_2 and X_3 are not causal; therefore, to find the true causal coefficients we proceed as follows:

$$E[Y_{X_1X_2X_3}] = \sum_{L_2}\sum_{L_3} E[Y_{X_1X_2X_3}|L_2, L_3] = \sum_{L_2}\sum_{L_3} 5 + 4X_1 + 5X_2 + 6X_3 + \tag{6.9}$$

$$10X_1X_2 + 11X_1X_3 + 12X_2X_3 - 8L_1 - 9L_2$$

$$E[Y_{X_1X_2X_3}] = 2.13 - 2.42X_1 + 2.3X_2 + 6X_3 + 10X_1X_2 + 11X_1X_3 + 12X_2X_3 \tag{6.10}$$

As equation 6.10 shows the true value of the intercept and the coefficients for X_1 and X_2 are different from the original model while other coefficients stay the same. The stabilized weight according to equation 6.8 is:

$$IPTW_{SW} = \frac{P(X_2)P(X_3)}{P(X_2|X_1, L_2)P(X_3|X_2, L_3)} \tag{6.11}$$

Table 6.8 compares the 95% CI for the outcome regression and IPTW methods. According to this table, the outcome regression estimate is biased for the coefficients of X_1, X_2, and the intercept term, while IPTW estimates are unbiased for all the coefficients. This clearly indicates that traditional methods such as outcome regression produce biased estimates in the case of treatment and confounder feedback. Table 6.9 compares the outcome regression, MSM(IPTW) and non-parametric g-formula. This table represents that both IPTW and g-formula can handle time-varying treatments and confounders feedback.

We conclude this section by comparing the IPTW and g-formula: They both can handle time-varying treatments, confounders feedback. They both rely on positivity and conditional exchangeability assumptions (no unmeasured confounders). g-formula is more prone and sensitive to model misspecifications since it requires a correct model for conditional expectation and conditional probability. Non-parametric g-formula becomes very tedious to calculate if the dimension of treatments and confounders increases. Therefore, we believe IPTW is more suitable for educational studies.

Table 6.8: Simulation Results for Time Varying Treatments and Confounders Feedback: Comparison of Outcome Regression and MSM in terms of 95% CI

Coefficient	**True**	**Regression** 95% **CI**	**MSM (IPTW)** 95% **CI**
Intercept	2.13	(4.92, 5.02)	(1.910, 3.843)
X_1	-2.42	(3.97, 4.11)	(-3.351, -0.089)
X_2	2.3	(4.965, 5.120)	(-1.0192, 2.3798)
X_3	6.0	(5.816, 6.092)	(4.56, 6.90)
X_1X_2	10.0	(9.823, 10.047)	(7.861, 10.604)
X_1X_3	11.0	(10.87, 11.16)	(7.00, 12.06)
X_2X_3	12.0	(11.78, 12.14)	(9.56, 13.38)
$X_1X_2X_3$	0	(-0.1191, 0.2794)	(-1.80, 3.04)

Table 6.9: Simulation Results for Time Varying Treatments and Confounders Feedback: Comparison of Outcome Regression, MSM and G-formula

Coefficient	**True**	**Regression**	**MSM(IPTW)**	**G-formula**
Intercept	2.13	5.02	3.09	1.80
X_1	-2.42	4.00	-1.85	-2.14
X_2	2.3	4.98	1.94	1.81
X_3	6.0	5.98	5.80	5.96
X_1X_2	10.0	10.04	8.65	9.84
X_1X_3	11.0	11.04	11.22	11.01
X_2X_3	12.0	12.05	12.43	12.31
$X_1X_2X_3$	0	0.09	0.47	0.12

6.4 Conclusion

In this section, we proposed various experimental and quasi-experimental designs for educational systems and quantify them using the graphical model and directed acyclic graph (DAG) language. We proposed to model the education system as time-varying treatments, confounders, and time-varying treatments-confounders feedback and derive unbiased causal estimates for the coefficients in fully observed and partially observed confounders. We showed that if we do not control for a sufficient set of confounders, then the causal estimate is biased. We showed that a more intelligent educational system contains feedback between treatments and confounders and provides techniques such as IPTW and g-formula to derive unbiased estimates of the coefficients.

Our model can be implemented in a standard college course by having a pretest exam before starting the course to measure students' prerequisite knowledge and including questions about student family, stress, emotional problems, and other factors such as how much time they can spend for the course per week. The pretest score can determine the initial treatment and study plan for students. Then students continue with the intervention and are tested weekly through quizzes to find the best subsequent interventions, such as talking to the TA, instructor, or the counselor depending on the problem they have. We ask the TAs, instructors, and counselors to ask questions during the meeting with students to collect information about the potential confounders and then try to control for these confounders in our analysis. We then have students take an exam (like a

midterm or final exam) and follow the methods outlined in this article to evaluate the effectiveness of the joint interventions on the exam score.

The causal analysis opens doors to widespread educational studies that can answer a broader set of questions than traditional study designs. In particular, they can provide analysis that applies to individuals and not just cohorts, provided sufficient information on confounding factors is gathered.

Chapter 7

CONCLUSION

In this dissertation, we proposed various algorithms and models for sequential prediction and inference in education. In chapter 3 we modeled and predicted student performance in an educational video game using the hidden Markov model. Our analysis proved the predictive power of the HMM. In chapter 4, we introduced a Python library to generate synthetic time series data according to arbitrary Bayesian network structure. This package is used to generate data in chapter 5. Chapter 5 compares the HMM and LSTM in terms of complexity and number of parameters. Chapter 6 suggests a graphical modeling approach to experimental and quasi-experimental designs. This chapter provides multiple inferences and modeling techniques for quasi-experimental designs with multiple interventions.

The future research is to implement the models and algorithms in this dissertation in standard college courses by having a pretest exam before starting the course to measure students' prerequisite knowledge and including questions about student family, stress, emotional problems, and other factors such as how much time they can spend for the course per week. The pretest score can determine the initial treatment and study plan for students. Then students continue with the intervention and are tested weekly through quizzes to find the best subsequent interventions, such as talking to the TA, instructor, or counselor depending on their problem. We ask the TAs, instructors, and counselors to ask questions during the meeting with students to collect information about the potential confounders and then try to control for these confounders in our analysis. We then have students take an exam (like a midterm or final exam) and follow the methods outlined in this thesis to evaluate the effectiveness of the joint interventions on the exam score. By these means, relatively small changes to standard courses can lead to much richer inferences on what

interventions will best help individual students to succeed

REFERENCES

[1] D. M. Dimitrov and P. D. Rumrill Jr, "Pretest-posttest designs and measurement of change," *Work*, vol. 20, no. 2, pp. 159–165, 2003.

[2] P. L. Bonate, *Analysis of pretest-posttest designs*. Chapman and Hall/CRC, 2000.

[3] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[4] A. Varga and R. Moore, "Hidden markov model decomposition of speech and noise," in *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*, IEEE, 1990, pp. 845–848.

[5] M. R. Hassan and B. Nath, "Stock market forecasting using hidden markov model: A new approach," in *Intelligent Systems Design and Applications, 2005. ISDA'05. Proceedings. 5th International Conference on*, IEEE, 2005, pp. 192–196.

[6] M. Tadayon and G. J. Pottie, "Predicting student performance in an educational game using a hidden markov model," *IEEE Transactions on Education*, vol. 63, no. 4, pp. 299–304, 2020.

[7] Y. Zeng, "Evaluation of physical education teaching quality in colleges based on the hybrid technology of data mining and hidden markov model," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 15, no. 1, pp. 4–15, 2020.

[8] C. Carmona, G. Castillo, and E. Millán, "Designing a dynamic bayesian network for modeling students' learning styles," in *2008 eighth IEEE international conference on advanced learning technologies*, IEEE, 2008, pp. 346–350.

[9] R. Levy, "Dynamic bayesian network modeling of game based diagnostic assessments. cresst report 837.," *National Center for Research on Evaluation, Standards, and Student Testing (CRESST)*, 2014.

[10] J. Sabourin, B. Mott, and J. C. Lester, "Modeling learner affect with theoretically grounded dynamic bayesian networks," in *International Conference on Affective Computing and Intelligent Interaction*, Springer, 2011, pp. 286–295.

[11] H. Ohlsson and K. S. Kendler, "Applying causal inference methods in psychiatric epidemiology: A review," *JAMA psychiatry*, vol. 77, no. 6, pp. 637–644, 2020.

[12] H. R. Varian, "Causal inference in economics and marketing," *Proceedings of the National Academy of Sciences*, vol. 113, no. 27, pp. 7310–7315, 2016.

[13] M. Tadayon and G. Pottie, "Causal inference in educational systems: A graphical modeling approach," *arXiv preprint arXiv:2108.00654*, 2021.

[14] J. Pearl, *Causality*. Cambridge university press, 2009.

[15] M. Bessani, J. A. Massignan, T. M. Santos, J. B. London Jr, and C. D. Maciel, "Multiple households very short-term load forecasting using bayesian networks," *Electric Power Systems Research*, vol. 189, p. 106 733, 2020.

[16] C. M. Queen and C. J. Albers, "Intervention and causality: Forecasting traffic flows using a dynamic bayesian network," *Journal of the American Statistical Association*, vol. 104, no. 486, pp. 669–681, 2009.

[17] M. Tadayon and G. Pottie, "Tsbngen: A python library to generate time series data from an arbitrary dynamic bayesian network structure," *arXiv preprint arXiv:2009.04595*, 2020.

[18] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[19] G. D. Forney, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[20] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, no. 02, pp. 107–116, 1998.

[21] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, 2013, pp. 1310–1318.

[22] T. D. Cook, D. T. Campbell, and W. Shadish, *Experimental and quasi-experimental designs for generalized causal inference*. Houghton Mifflin Boston, MA, 2002.

[23] A. T. Corbett and J. R. Anderson, "Knowledge tracing: Modeling the acquisition of procedural knowledge," *User modeling and user-adapted interaction*, vol. 4, no. 4, pp. 253–278, 1994.

[24] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon, "Individualized bayesian knowledge tracing models," in *International conference on artificial intelligence in education*, Springer, 2013, pp. 171–180.

[25] C. Piech, J. Spencer, J. Huang, S. Ganguli, M. Sahami, L. Guibas, and J. Sohl-Dickstein, "Deep knowledge tracing," *arXiv preprint arXiv:1506.05908*, 2015.

[26] X. Xiong, S. Zhao, E. G. Van Inwegen, and J. E. Beck, "Going deeper with deep knowledge tracing.," *International Educational Data Mining Society*, 2016.

[27] J. R. Anderson, C. F. Boyle, and B. J. Reiser, "Intelligent tutoring systems," *Science*, vol. 228, no. 4698, pp. 456–462, 1985.

[28] A. C. Graesser, K. VanLehn, C. P. Rosé, P. W. Jordan, and D. Harter, "Intelligent tutoring systems with conversational dialogue," *AI magazine*, vol. 22, no. 4, pp. 39–39, 2001.

[29] D. Sleeman and J. S. Brown, *Intelligent tutoring systems*. London: Academic Press, 1982.

[30] K. VanLehn, "The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems," *Educational Psychologist*, vol. 46, no. 4, pp. 197–221, 2011.

[31] M. C. Polson and J. J. Richardson, *Foundations of intelligent tutoring systems*. Psychology Press, 2013.

[32] G. Chung and D Kerr, "A primer on data logging to support extraction of meaningful information from educational games: An example from save patch," *CRESST Report*, vol. 814, 2012.

[33] G. K. Chung, K. Choi, E. L. Baker, and L. Cai, "The effects of math video games on learning: A randomized evaluation study with innovative impact estimation techniques. cresst report 841.," *National Center for Research on Evaluation, Standards, and Student Testing (CRESST)*, 2014.

[34] D. Kerr and G. K. Chung, "Identifying key features of student performance in educational video games and simulations through cluster analysis," *JEDM— Journal of Educational Data Mining*, vol. 4, no. 1, pp. 144–182, 2012.

[35] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Oakland, CA, USA, vol. 1, 1967, pp. 281–297.

[36] M. Tadayon and G. Pottie, "Comparative analysis of the hidden markov model and lstm: A simulative approach," *arXiv preprint arXiv:2008.03825*, 2020.

[37] M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "Synthetic data augmentation using gan for improved liver lesion classification," in *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, IEEE, 2018, pp. 289–293.

[38] C. Bowles, L. Chen, R. Guerrero, P. Bentley, R. Gunn, A. Hammers, D. A. Dickie, M. V. Hernández, J. Wardlaw, and D. Rueckert, "Gan augmentation: Augmenting training data using generative adversarial networks," *arXiv preprint arXiv:1810.10863*, 2018.

[39] C. Esteban, S. L. Hyland, and G. Rätsch, "Real-valued (medical) time series generation with recurrent conditional gans," *arXiv preprint arXiv:1706.02633*, 2017.

[40] M. Tadayon and Y. Iwashita, "A clustering approach to time series forecasting using neural networks: A comparative study on distance-based vs. feature-based clustering methods," *arXiv preprint arXiv:2001.09547*, 2020.

[41] ——, "Comprehensive analysis of time series forecasting using neural networks," *arXiv preprint arXiv:2001.09547v1*, 2020.

[42] K.-j. Kim, "Financial time series forecasting using support vector machines," *Neurocomputing*, vol. 55, no. 1-2, pp. 307–319, 2003.

[43] G. P. Zhang, "Time series forecasting using a hybrid arima and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, 2003.

[44] A. A. Ariyo, A. O. Adewumi, and C. K. Ayo, "Stock price prediction using the arima model," in *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, IEEE, 2014, pp. 106–112.

[45] S. Ghosh, O. Vinyals, B. Strope, S. Roy, T. Dean, and L. Heck, "Contextual lstm (clstm) models for large scale nlp tasks," *arXiv preprint arXiv:1602.06291*, 2016.

[46] A. Prakash, S. A. Hasan, K. Lee, V. Datla, A. Qadir, J. Liu, and O. Farri, "Neural paraphrase generation with stacked residual lstm networks," *arXiv preprint arXiv:1610.03098*, 2016.

[47] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[48] F. V. Jensen and F. Jensen, "Optimal junction trees," in *Uncertainty Proceedings 1994*, Elsevier, 1994, pp. 360–366.

[49] K. Murphy, Y. Weiss, and M. I. Jordan, "Loopy belief propagation for approximate inference: An empirical study," *arXiv preprint arXiv:1301.6725*, 2013.

[50] Z. Ghahramani and M. I. Jordan, "Factorial hidden markov models," in *Advances in Neural Information Processing Systems*, 1996, pp. 472–478.

[51] Y. Bengio and P. Frasconi, "An input output hmm architecture," in *Advances in neural information processing systems*, 1995, pp. 427–434.

[52] J. M. Robins, M. A. Hernan, and B. Brumback, *Marginal structural models and causal inference in epidemiology*, 2000.

[53] L. M. Collins, S. A. Murphy, and V. Strecher, "The multiphase optimization strategy (most) and the sequential multiple assignment randomized trial (smart): New methods for more potent ehealth interventions," *American journal of preventive medicine*, vol. 32, no. 5, S112–S118, 2007.

[54] P. R. Rosenbaum, "Observational study," *Encyclopedia of statistics in behavioral science*, 2005.

[55] S. L. Turner, A. B. Forbes, A. Karahalios, M. Taljaard, and J. E. McKenzie, "Evaluation of statistical methods used in the analysis of interrupted time series studies: A simulation study," *medRxiv*, 2020.

[56] A. K. Wagner, S. B. Soumerai, F. Zhang, and D. Ross-Degnan, "Segmented regression analysis of interrupted time series studies in medication use research," *Journal of clinical pharmacy and therapeutics*, vol. 27, no. 4, pp. 299–309, 2002.

[57] M. Taljaard, J. E. McKenzie, C. R. Ramsay, and J. M. Grimshaw, "The use of segmented regression in analysing interrupted time series studies: An example in pre-hospital ambulance care," *Implementation Science*, vol. 9, no. 1, pp. 1–4, 2014.

[58] A. L. Schaffer, T. A. Dobbins, and S.-A. Pearson, "Interrupted time series analysis using autoregressive integrated moving average (arima) models: A guide for evaluating large-scale health interventions," *BMC medical research methodology*, vol. 21, no. 1, pp. 1–12, 2021.

[59] J. M. Gottman, G. V. Glass, and T. Kratochwill, "Analysis of interrupted time-series experiments," *Single subject research: Strategies for evaluating change*, pp. 197–234, 1978.

[60] C. C. Branas, R. A. Cheney, J. M. MacDonald, V. W. Tam, T. D. Jackson, and T. R. Ten Have, "A difference-in-differences analysis of health, safety, and greening vacant urban space," *American journal of epidemiology*, vol. 174, no. 11, pp. 1296–1306, 2011.

[61] S. G. Donald and K. Lang, "Inference with difference-in-differences and other panel data," *The review of Economics and Statistics*, vol. 89, no. 2, pp. 221–233, 2007.

[62] J. B. Dimick and A. M. Ryan, "Methods for evaluating changes in health care policy: The difference-in-differences approach," *Jama*, vol. 312, no. 22, pp. 2401–2402, 2014.

[63] E. P. Martens, W. R. Pestman, A. de Boer, S. V. Belitser, and O. H. Klungel, "Instrumental variables: Application and limitations," *Epidemiology*, pp. 260–267, 2006.

[64] J. D. Angrist, G. W. Imbens, and D. B. Rubin, "Identification of causal effects using instrumental variables," *Journal of the American statistical Association*, vol. 91, no. 434, pp. 444–455, 1996.

[65] S. Greenland, "An introduction to instrumental variables for epidemiologists," *International journal of epidemiology*, vol. 29, no. 4, pp. 722–729, 2000.

[66] R. M. Daniel, S. Cousens, B. De Stavola, M. G. Kenward, and J. Sterne, "Methods for dealing with time-dependent confounding," *Statistics in medicine*, vol. 32, no. 9, pp. 1584–1618, 2013.

[67] T. J. VanderWeele, "Marginal structural models for the estimation of direct and indirect effects," *Epidemiology*, pp. 18–26, 2009.

[68] S. R. Cole and M. A. Hernán, "Constructing inverse probability weights for marginal structural models," *American journal of epidemiology*, vol. 168, no. 6, pp. 656–664, 2008.