2018

Effectiveness of classification approach in recovering pairwise causal relations from data.

Kristima Guha Thakurta
Iowa State University

Effectiveness of classification approach in recovering causal relations from data

by

Kristima Guha Thakurta

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Science

Program of Study Committee:
Jin Tian, Major Professor
Forrest Sheng Bao
Robyn Lutz

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this thesis. The Graduate College will ensure this thesis is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2018

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

ACKNOWLEDGMENTS

I would like to take this opportunity to thank everyone who helped me with the various aspects of conducting this research and writing the thesis. Firstly, I would like to thank Dr. Jin Tian for his constant guidance, patience, and support throughout the research as well as for the entire duration of my time as a graduate student at ISU. His insights and ideas have always inspired and guided me to keep on working towards the completion of my graduate education. I would also like to thank my committee members for their effort, guidance and their words of motivation: Dr. Forrest Sheng Bao and Dr. Robyn Lutz.

In addition, I would also like to thank my friends, colleagues, the department faculty, and staff for making my time at Iowa State University a wonderful experience.

ABSTRACT

Causal structure discovery is a much-studied topic and a fundamental problem in Machine Learning. Causal inference is the process of recovering cause-effect relationships between the variables in a dataset. In general, causal inference problem is to decide whether X causes Y, Y causes X, or there exists an indirect relationship between X and Y via a confounder. Even under very stringent assumptions, causal structure discovery problems are challenging. Much work has been done on causal discovery methods with two variables in recent years. This thesis extends the bivariate case to the possibility of having at least one confounder between X and Y. Attempts have been made to extend the causal inference process to recover the structure of Bayesian networks from data. The contributions of this thesis include (a) extending causal discovery methods to the networks with exactly one confounder (third variable) ; (b) an algorithm to recover the causal graph between every pair of variables with the presence of a confounder in a large dataset; (c) employing a search algorithm to find the best Bayesian network structure that fits the data . Improved results have been achieved after the introduction of confounders in the bivariate causal graphs. Further attempts have been made to improve the Bayesian network scores for the network structures of some medium to large-sized networks using the standard ordering based search algorithms such as OBS and WINASOBS. Performance of the methods proposed have been tested on the benchmark datasets for cause-effect pairs and from the BLIP library.

CHAPTER 1. OVERVIEW

A very simple example of cause-effect relation one can come across is this: turning the steering wheel of a car towards the left, turns the car to the left. Here, the cause is the process of turning the wheel to the left and the effect is the motion of the car. There exist many such instances in machine learning where we need to understand the cause-effect relationship between multiple variables and take decisions accordingly. Dependence and causation relates one variable to the other and helps us in making important predictions about the values of a variable given the value taken by other variables(Lopez-Paz, 2016). According to Hans Reichenbach(1956), relationships between two variables $\boldsymbol{x}$ and $\boldsymbol{y}$ can assume any of the following: $\boldsymbol{x}$ causes $\boldsymbol{y}$, $\boldsymbol{y}$ causes $\boldsymbol{x}$, or there exists a confounder $\boldsymbol{z}$ that causes an indirect relationship to exist between $\boldsymbol{x}$ and $\boldsymbol{y}$. This thesis focusses on exploring the causal structure that can exist between a pair of variables in a dataset.

The goals of this work are:- a) recovering the causal direction between a pair of variables when confounders are present b) extending the discovery of pairwise causal relations to large directed acyclic graphs to find the direction between every unique pair of variables in the DAG c) generating an order from the obtained causal DAG to recover a Bayesian network from data.

For testing the performance of pairwise causal inference, the 82 pairs of benchmark cause-effect data is used.

For the purposes of testing the Bayesian network recovery algorithms developed here, real-world data sets are used whose Bayesian network structures are known to us. The networks range from very small ones to large network of up to 724 nodes. Apart from these networks, 20 other data sets are used from the BLIP data library to test the performance of the algorithms developed here, in terms of the final Bayesian networks obtained.

1.1 Introduction

The first goal of this work deals with pairwise causal inference from data. Previous work in this field discusses causal inference in networks where two variables are either directly causal or directly anti-causal. On the other hand, this work extends the direct causal and anti-causal relations to indirect ones by introducing confounders and testing the performance of such a model on benchmark cause-effect data.

The next goal of the work introduces algorithms in an attempt to recover Bayesian networks from data. A Bayesian network is a probabilistic graphical model which expresses the relationships existing between every pair of variables through a directed acyclic graph(Pearl et al., 1988). Bayesian networks are used to perform classification, prediction and knowledge discovery in the fields of medicine and engineering. The learning process involves finding the best DAG that fits the data generated from the Bayesian network.

The problem of learning the structure of Bayesian networks have been explored a lot in the past, but research has shown that this problem is an NP-hard problem(Chickering et al., 2003). There exist various methods of structure learning which have been discussed later. The main focus of the thesis is on score-based learning which helps in finding the best structure given the data by the means of maximizing the Bayesian network score. Algorithms have been developed for this purpose, based on shortest-path heuristic(Yuan and Malone, 2012, 2013), dynamic programming(Koivisto and Sood, 2004; Silander and Myllymaki 2006), branch and bound(de Campos et al., 2009; van Beek and Hoffmann, 2015), and other such techniques. This kind of structural learning is completed in two steps: initial causal structure discovery which is called *parent set identification* and the next step is *structure optimization* which employs a greedy algorithm to search for a more optimal structure without introducing cycles.

The parent identification step is usually a polynomial-time algorithm and has been implemented in various ways. Many algorithms restrict the number of possible parents of each node for simplicity and also to reduce the time taken to run these kinds of algorithms. This constraint has been overcome by approximation techniques for the exploration of parent nodes sets, recently(Scanagatta

et al., 2015). Local search has been successful in finding good solutions with simple algorithms. It has been applied to learning Bayesian networks using greedy search over the space of all network structures or over the space of variable orderings(order space). Attempts have been made to improve upon the approach of ordering based search(OBS) by Teyssier and Koller(2005) and window acyclic selection ordering based search(WINASOBS) by Scanagatta, Corani and Zaffalon(2017). Both of these approaches use topological orderings of variables, which are randomly generated, as the search space.

In this thesis, Bayesian network learning is performed in two steps: Parent step identification can be viewed as a step of causal structure discovery. As discussed earlier, a relationship between two variables $\boldsymbol{X}$ and $\boldsymbol{Y}$ can be explained as a direct causal influence from $\boldsymbol{X}$ to $\boldsymbol{Y}$, a direct causal influence from $\boldsymbol{Y}$ to $\boldsymbol{X}$, an indirect relationship between $\boldsymbol{X}$ and $\boldsymbol{Y}$ via another variable $\boldsymbol{Z}$ (which is called the confounder)(Pearl, 2000,) a set of possible combinations of these, or no causal relation between $\boldsymbol{X}$ and $\boldsymbol{Y}$. Once the causal structure is discovered, a total order of this structure is generated and a greedy search is performed in this order space to generate the network structure that maximizes the network score and best fits the data. Experimental results show that the algorithm implemented here gives out score close to the ones generated by the state-of-the-art algorithms on instances with more than 100 variables.

A variety of causal discovery methods have surfaced in the past few years(Friedman and Nachman, 2000; Kano and Shimizu, 2003; Shimizu et al., 2006; Sun et al., 2006, 2008; Hoyer et al., 2009; Mooij et al., 2009, 2010, 2011). All of them tried to solve this task after placing some assumptions and restrictions. Each of the algorithms uses concepts of marginal and conditional probability distributions and factorizes the joint density $p_{C,\,E}(c,\ e)$ of cause C and its effect E into $p_C(c)$ $p_{E\,|\,C}(e|c)$ which yields a model of lower complexity than the alternative factorization of $p_E(e)$ $p_{C\,|\,E}(c|e)$(Mooij et al., 2016). If complexity is measured by Kolmogorov complexity(Janzing and Schölkopf, 2010; Lemeire and Janzing 2013), we can see that p_C contains no information about $p_{E|C}$, which implies that the sum of the complexities of p_C and $p_{E|C}$ cannot be greater than the sum of the complexities of p_E and $p_{C|E}$. There have been instances of some classes of simple conditionals,

e.g., Additive Noise Models(Hoyer et al., 2009) and second-order exponential models(Sun at al., 2006; Janzing et al., 2009), and infer $\boldsymbol{X}$ to be the cause of $\boldsymbol{Y}$ whenever $\boldsymbol{P_{Y|X}}$ is from this class of models and $\boldsymbol{P_{X|Y}}$ is not.

1.2 Main Contributions

The main contributions of this research include:

- Causal discovery methods explored based on *Additive Noise Models*(Hoyer et al., 2009) to recover pairwise causal inference in the presence of confounders.
- Pairwise causal inference extended to recover the causal relations between every pair of variables in a DAG.
- Total order obtained from these causal structures provided as an input to the standard ordering based search algorithms to discover the true underlying Bayesian network structure.
- Experiments performed on benchmark cause-effect pairs data, and a variety of different data sets and networks obtained from the BLIP documentation(Scanagatta et al., 2015).

CHAPTER 2. REVIEW OF LITERATURE

In this chapter, we descrie in some detail certain aspects of pairwise causal inference, Bayesian networks as well as Additve Noise Models. We also descrie in detail some other mathematical and machine learning concepts that are relevant to this research.

2.1 Bayesian Networks

A Bayesian network is a graphical representation of a probability distribution over a set of variables $V = \{ X_1 \ X_2, \ldots, X_n\}$. Each network is a directed acyclic graph(DAG) and a set of conditional probability distributions(CPDs) is associated with every variable, which are conditioned on the possible parents of the variable.

A simple example of a Bayesian network along with its conditional probability distributions is as shown in Fig. 2.1. It depicts a domain of five variables, all of them can take possiblly two values: True or False: B representing Burglary, E representing earthquake, A representing alarm, M representing the occurrence of Mary calling while J represents the occurrence of John calling.

The conditional probability distributions are also shown in the figure. For example, given that burglary has occurred but not earthquake, the probability that the alarm will ring is 0.94 and the probability that the alarm will not ring is 0.06.

The figure shows that A depends on B and E, while B and E are independent of each other. Another statement that can be made from the Bayesian network is that once we know the value of A, the variables B and J are independent.

In general, a Bayesian network consists of a set of conditional independence statements that can be inferred from the structure itself. Under certain assumptions(described later in the chapter), it can also represent dependency and causality between each pair of variables.

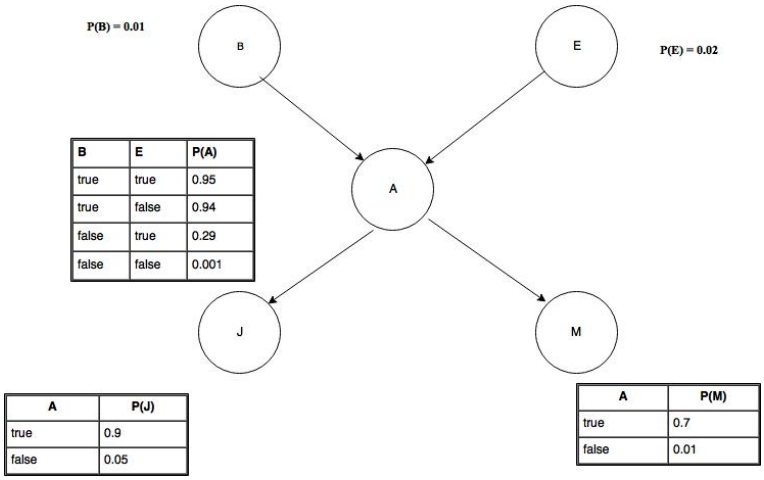

Figure 2.1: A simple Bayesian network modeling the relationship between burglary, earthquake, alarm and whether the neighbors Mary and John will call or not.

2.2 Causal Relationships

Let X, Y be two random variables with joint probability distribution $\mathbf{P}_{X,\ Y}$. An external "intervention" might change some aspects of the system which may influence the joint probability distribution of X and Y. If we set the value of X to be x, then the resulting distribution of Y can be represented as $\mathbf{P}_{Y|X\ =\ x}$(Pearl, 2000). Similarly, we consider that Y takes the value y, the joint distribution becomes $\mathbf{P}_{X|\ Y\ =\ y}$.

The marginal distribution $\mathbf{P}_X$ can be obtained by integrating the joint distribution over all possible values of Y. For such an X and Y, the following definitions have been made by Mooij(2016):

Definition 2.1 *X is said to cause Y if $\boldsymbol{P}_{Y|X\ =\ x} \neq \boldsymbol{P}_{Y|X\ =\ x'}$, for some x, x'.*

Definition 2.2 *X_i is a direct cause of X_j with respect to $X_1, \ldots, X_n$ if Y depends on the value of $X = x$ while fixing all other variables.* The intuition is that a direct causal relation of X on Y is not mediated via the other variables. The direct causal relations can be viewed as a causal graph.

Definition 2.3 *The causal graph G has variables $X_1, \ldots, X_n$ as variable nodes, and a directed edge from X_i to X_j iff X_i is a direct cause of X_j with respect to the rest of the nodes.*

The Figure 2.2 represents the seven different kinds of causal relations that exist and are considered in this research. Although this list of possibilities is not exhaustive and may include (a) feedback

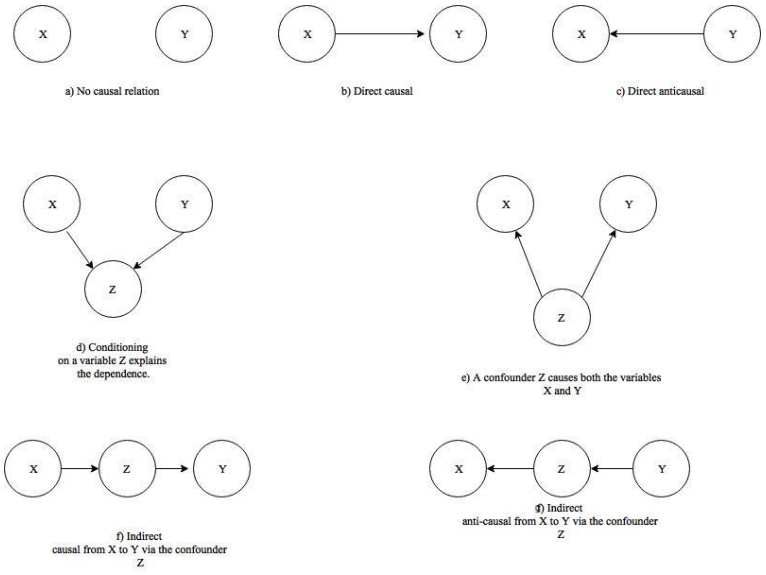

Figure 2.2: The seven different kinds of causal structures that have been used in the research.

relationships with a latent variable, (b) combinations of the various cases shown in Fig 2.2, (c) presence of more than one latent variable. Inferring the causal direction between X and Y using the observational distribution $\mathbf{P}_{X,\ Y}$ is the task that has been considered in this work.

2.3 Machine Learning Concepts

This section deals with a few concepts of learning theory relevant to the research.

2.3.1 Concepts of learning theory

Considering two random variables: an input variable $\boldsymbol{x} \in X$, and an output variable $\boldsymbol{y} \in Y$. The problem of learning is to find a dependence between these two variables that would be able to best predict the values of $\boldsymbol{y}$, provided we have some information about $\boldsymbol{x}$. In order to solve this problem, we can use the following three ways:

1. Given data D, we can sample x and y according to the following rule:

$$D = \{(x_1, y_1), \ldots, (x_n, y_n)\} \sim P^n(x, y), x_i \in X, y_i \in Y. \tag{2.1}$$

2. Using a function $f : X \to Y$ which can be used to map every $x_i \in X$ to a $y_i \in Y$.

3. A *loss function* $l : Y \to Y$, which finds the differences between the predictions made by $f(x)$ and the true and known values of y.

Using these three ways, one can solve the learning problem by finding the function $f \in F$, by minimizing the risk/loss associated with the function.

2.3.2 Cross Validation

In order to select the best model from a set of candidate models, cross-validation splits the data D into two(or more) disjoint subsets, namely the training set and the validation set. Then, cross-validation trains each of the candidate models using the training data and chooses the model with the smallest risk on the validation data.

2.4 Structural equation models

This section defines a multitude of concepts in some detail that is used to describe the structure of directed graphs as well as the set of assumptions that link directed graphs and probability distributions for the nodes of the graph. A generalization of graphical models has also been introduced in this section. Some necessary assumptions to link structural equation models to causal relationships have also been discussed in this section.

2.4.1 Graphs

These definitions have been borrowed from Peters, 2012.

1. A *directed graph* $G = (V, E)$ is a set of nodes $V = \{v_1, \ldots, v_d\}$ and a set of edges $E \subseteq V^2$.

2. For all v_i, v_j, $v_i \neq v_j$, we say that v_i is a parent of v_j if $(i, j) \in E$, and is represented as $v_i \to v_j$. A pair of nodes (v_i, v_j) are *adjacent* if either $v_i \to v_j$, or $v_j \to v_i$, and this is written as $v_i - v_j$.

3. For all $v_i \in V$, $\mathrm{Pa}(v_i) = \{v_j \mid v_j \rightarrow v_i\}$ is the set of parents of v_i.

4. The *skeleton* of G is the set of all edges *(i, j)* such that $v_i \rightarrow v_j$ or $v_j \rightarrow v_i$.

5. A three-node structure forms a *v-structure* if one of them is the child of the two others, which are themselves not adjacent.

6. A *path* in G is a sequence $v_{i_1}, \ldots, v_{i_n}$ such that $v_{i_k} \rightarrow v_{i_{k+1}}$ or $v_{i_{k+1}} \rightarrow v_{i_k}$ for all $1 \leq \mathrm{k} \leq n$ - 1 and $n \geq 2$.

7. G is a directed acyclic graph if it contains no directed path from v_i to itself, for all $v_i \in V$.

8. A path between v_{i_1} and v_{i_n} is *blocked* by $Z \subseteq V \setminus \{ v_{i_1}, v_{i_n}\}$ if

   - $v_{i_k} \in Z$ and
     - $v_{i_{k-1}} \rightarrow v_{i_k} \rightarrow v_{i_{k+1}}$ or
     - $v_{i_{k-1}} \leftarrow v_{i_k} \leftarrow v_{i_{k+1}}$ or
     - $v_{i_{k-1}} \leftarrow v_{i_k} \rightarrow v_{i_{k+1}}$.
   - $v_{i_{k-1}} \rightarrow v_{i_k} \leftarrow v_{i_{k+1}}$ and v_{i_k} and its descendants are not in Z.

9. Given three disjoint subsets $\mathcal{A}$, $\mathcal{B}$, $\mathcal{Z} \subseteq V$, we say $\mathcal{A}$ and $\mathcal{B}$ are *d-separated* by $\mathcal{Z}$ if all the paths between the nodes of $\mathcal{A}$ and the nodes of $\mathcal{B}$ are blocked by $\mathcal{Z}$.

The Figure 2.3 shows a DAG with 5 nodes and 6 edges.

2.4.2 Graphical Models

Before discussing about graphical models, it is important to familiarize ourselves with the formal definition of *d-separation.*

Definition 2.4: Let $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ be three disjoint subsets of variables of a DAG G. Let there be a path between a node in $\mathcal{A}$ to a node in $\mathcal{B}$. Then $\mathcal{C}$ is said to block this path if there is a node v on this path satisfying one of the following two conditions:

- v has converging arrows (along the path) and neither v nor any of its descendants are in $\mathcal{C}$,

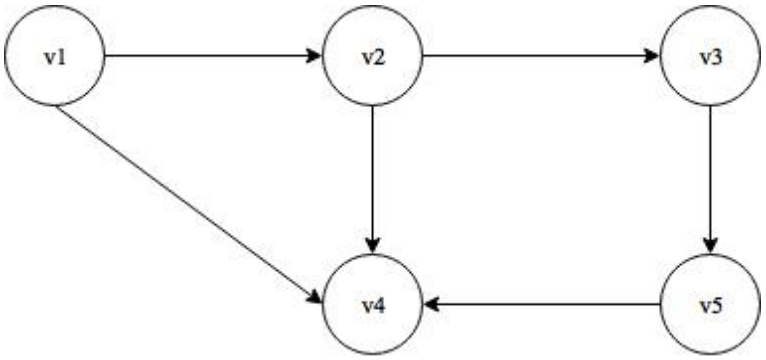

Figure 2.3: A simple DAG

.

- v does not have converging arrows (along the path) and $v \in \mathcal{C}$.

If no path between $\mathcal{A}$ and $\mathcal{B}$ are blocked by $\mathcal{C}$, then we say $\mathcal{A} \perp\!\!\!\perp_d \mathcal{B} \mid \mathcal{C}$, that is $\mathcal{A}$ and $\mathcal{B}$ are *d-separated* given $\mathcal{C}$.

More details on d-separation can be found in Pearl(1995).

Let $G = (V, E)$ be a DAG where $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_d)$ is a random variable with joint probability distribution $P(\boldsymbol{x})$. Every random variale $\boldsymbol{x}_i$ is a representation of the node $v_i \in V$. Then, the follwing hold true:

1. P is *faithful* to G if

$$A \perp\!\!\!\perp_d B \mid Z \iff A \perp\!\!\!\perp B \mid Z \tag{2.2}$$

   for all disjoint sets A, B, $Z \subseteq V$. This condition forces the probability distribution P to not include any other conditional independence statements other than those that are already evident from the d-separations associated with the structure of G.

2. P is *Markov* with repsect to G if

$$A \perp\!\!\!\perp_d B \mid Z \implies A \perp\!\!\!\perp B \mid Z, \tag{2.3}$$

   for all disjoint sets A, B, $Z \subseteq V$. This condition shows that the probability distribution P encodes all the conditional independences in G, that can be obtained from the d-separations

observed in the structure of G. This condition enables the following factorization:

$$p(x) = \prod_{i=1}^{d} p(x_i \mid Pa(x_i)), \tag{2.4}$$

where p is the density function of x, and $\mathrm{Pa}(x_i)$ is the set of parents of $v_i \in V$.

3. The pair *(G, P)* satisfies the minimality condition if it satisfies the Markov condition, but any pair *(G', P)* where *G'* is a graph obtained by removing edges from G, does not satisfy the Markov condition.

4. The *Markov equivalence class* of G is given as:

$$Markov(G) = \{P \mid P\, is\, Markov\, with\, respect\, to\, G\}.$$

We say two DAGs G_1 and G_2 are Markov equivalent if $Markov(G_1) = Markov(G_2)$. Two graphs are Markov equivalent if they have the same skeleton and the same set of immoralities(Verma and Pearl, 1991).

5. If P is Markov with respect to G, the tuple *(G, P)* is called a graphical model.

These graphical models are one step closer to causal models.

2.4.3 Structural Equation Models

A *Structural Equation Model* is a pair $(\mathcal{S}, Q(n))$, where $\mathcal{S} = \{S_1, \ldots, S_d\}$ is a set of equations

$$S_i : x_i = f_i(Pa(x_i), n_i), \tag{2.5}$$

where $n = (n_1, \ldots, n_d)$ is a noise vector, following the probability distribution $Q(n)$. One can map structural equation models (S, Q) to graphical models (G, P) as follows:

1. Construct the graph $G = (V, E)$ by associating the x_i in the equation above to the node $v_i \in V$ and adding an edge $(j, i) \in E$ from each $v_j \in \mathrm{Pa}(x_i)$ to v_i.

2. Construct the probability distribution $P(x)$ by choosing the distributions of each of the noise variables $n_1, \ldots, n_d$. These distributions can be mapped using the equations $\mathcal{S}$ to the random

variables x_i, jointly described by P. This mapping induces a distribution P Markov with respect to the graph G(Pearl, 2009b).

Structural models contain more information than graphical models.

2.4.4 Causation

Given a graphical model (G, P), the DAG G describes the conditional independences that are encoded in the probability distribution P and allows the factorization as has been already mentioned in Eq. 2.4. The causal relationships between a collection of variables $\boldsymbol{x}_1, \ldots \boldsymbol{x}_\mathrm{d}$ are formalized as a DAG by placing two assumptions(Dawid, 2010).

1. The representational assumption, or, the causal structure of $\boldsymbol{x}$ admits a causal DAG $G_0$and does not consider the formation of cyclic graphs.

2. The causal Markov condition, or, the d-separations in G_0 are encoded in the conditional independence statements made for the distribution $P(x)$.

The causal Markov condition states that the edge $x_i \to x_j$ means x_i*causes* x_j. The factorization of Eq 2.4 means that variables are independent when conditioned to their direct causes or their direct parents.

We can also define *causal* structural equation models $(\mathcal{S}, \mathcal{Q})$, with $\mathcal{S} = \{S_1, \ldots, S_d\}$, where the equations

$$S_i : x_i = f_i(Pa(x_i), n_i) \tag{2.6}$$

now mean that the causes of x_i are $\mathrm{Pa}(x_i)$. There are some stronger assumptions(Pearl, 2009b) placed on causal inference from causal graphical models:

1. The causal faithfulness condition states that a causal DAG is faithful to the distribution $P(x)$.

2. The causal minimality condition states that the pair (G, P) satisfies the minimality condition, where G is a causal DAG.

2.5 Observational Causal Inference

2.5.1 Assumptions

The problem of causal inference needs a lot of restrictions with respect to the structural equation models under study. It is a popular fact that the accuracy of the true causal graph is inversely proportional to the number of assumptions that we make. The most popular assumption of observational causal inference is: **Independence of cause and mechanism**. Let us consider two random variables $\boldsymbol{x}$ and $\boldsymbol{y}$ such that the following holds:

$$p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{y} \mid \boldsymbol{x}).p(\boldsymbol{x}) = p(\boldsymbol{x} \mid \boldsymbol{y}).p(\boldsymbol{y}), \tag{2.7}$$

where $p(\boldsymbol{y} \mid \boldsymbol{x})$ can be viewed as a causal mechanism mapping the cause $\boldsymbol{x}$ to an effect $\boldsymbol{y}$, while $p(\boldsymbol{x} \mid \boldsymbol{y})$ can be viewed as a causal mechanism mapping the cause $\boldsymbol{y}$ to its effect $\boldsymbol{x}$.

Another important fact that we should take into consideration is that the conditional decomposition with algorithmically independent factors has a lower Kolmogorov complexity. Hence, we always prefer the causal direction under which the distribution of the cause is independent from the mechanism mapping the cause to the effect. This is the crux of ICM.

The ICM assumption is violated in the incorrect causal direction. In the example above, if the factors $p(\boldsymbol{x})$ and $p(\boldsymbol{y} \mid \boldsymbol{x})$ are independent, then it will not be the case for the factors $p(\boldsymbol{y})$ and $p(\boldsymbol{x} \mid \boldsymbol{y})$. This asymmetry makes causal inference possible.

The other assumptions include:

1. **Causal Sufficiency Assumption:** There exist no common unobserved variables in the domain that are parent of one or more observed variables of the domain.

2. **Markov Assumption:** Given a Bayesian network model B, any variable is independent of all its non-descendants in B, given its parents.

3. **Faithfulness Assumption:** A Bayesian network G and a probability distribution P are faithful to one another iff every one and all independence relations valid in P are those entailed by the Markov assumption on G.

2.5.2 Algorithms

This section provides a quick review of the algorithms that have been used for observational causal inference in other researches so far:

1. **Conditional independence methods:** The Spirtes-Glymour-Scheines(SGS) algorithm(Spirtes et al., 2000) assumes that representational, causal Markov, sufficiency and faithfulness conditions hold, but it does not place any restriction on the relationships of the variables. The algorithm works as follows:

   - Build $G = (V, E)$ with $V = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_d\}$ and $(i, j), (j, i) \in E$, for all $1 \leq i, j \leq d$.
   - For every pair $(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $\exists \mathcal{Z} \subseteq V \setminus \{\boldsymbol{x}_i, \boldsymbol{x}_j\}$ such that $\boldsymbol{x}_i \perp\!\!\!\perp_d \boldsymbol{x}_j \mid \mathcal{Z}$, remove the edge (i, j) from E as they cannot be connected in accordance with the defintion of -separation.
   - For each structure $\boldsymbol{x}_i$ - $\boldsymbol{x}_j$ - $\boldsymbol{x}_k$ with $(i, k) \notin E$ and no $\mathcal{Z} \subseteq \boldsymbol{x}_j \cup V \setminus \{\boldsymbol{x}_i, \boldsymbol{x}_k\}$ such that $\boldsymbol{x}_i \perp\!\!\!\perp_d \boldsymbol{x}_k \mid \mathcal{Z}$, remove the edges (j, i) and (j, k) from E in accordance to the definition of d-separation.
   - Until no more edges get removed from E, repeat
     - if $\boldsymbol{x}_i \rightarrow \boldsymbol{x}_j$ - $\boldsymbol{x}_k$, $\boldsymbol{x}_i$ does not cause $\boldsymbol{x}_k$, and $\boldsymbol{x}_k$ does not cause $\boldsymbol{x}_i$, then remove the edge (k, j) from E.
     - if there is a directed path from $\boldsymbol{x}_i$ to $\boldsymbol{x}_k$, and $\boldsymbol{x}_i \rightarrow \boldsymbol{x}_k$, remove the edge (k, j) from E

   Then the algorithm performs a conditional independence test for all possible conditioning sets $\mathcal{Z} \subseteq V \setminus \{\boldsymbol{x}_i, \boldsymbol{x}_j\}$ and a pair of distinct nodes $(\boldsymbol{x}_i, \boldsymbol{x}_j)$. SGS actually performs 2^{d-2} conditional independence tests and suffers from the curse of dimensionality.

   Thus, SGS algorithm evolved into the PC algorithm, which uses the idea of sorting of the variables to reduce the number of conditional independence tests. But, PC algorithm could not improve the computational complexity of the SGS algorithm. The FCI algorithm deals with *insufficiency* of the SGS/PC algorithm and deals with unobserved confounders too.

2. **Score methods:** Score methods (Heckerman et al., 1997) constructs a mapping from parameter vectors $\theta \in \mathbf{R}^m$ to the set of DAGs on d nodes, and evaluates the score of each candidate with the help of the posterior distribution

$$p(\mathbf{\Theta} = \theta \mid x_1, \ldots, x_n) = \frac{p(x_1, \ldots, x_n \mid \mathbf{\Theta} = \theta) p(\mathbf{\Theta} \mid \theta)}{p(x_1, \ldots, x_n)} \tag{2.8}$$

where the prior distribution p(Θ) is the prior knowledge that would help us distinhguish between the different DAG structures and favour one DAG over the other, and the likelihood distribution $p(x_1, \ldots, x_n)$, measures how well does a DAG explain the data.

The idea behind scoring methods is to find the best DAG $G = (V, E)$ where V is the collection of nodes, E is the set of edges and can be defined as the set of parents $\Pi_1, \ldots \Pi_n$ for each of the nodes in V. The most popular scoring metric used is the *BIC* score which is given as follows:

$$BIC(G) = \sum_{i=1}^{n} BIC(X_i, \Pi_i) = \sum_{i=1}^{n} \sum_{\pi \in |\Pi_i|} \sum_{x \in |X_i|} N_{x,\pi} log\theta'_{x|\pi} - \frac{logN}{2}(\mid X_i \mid -1)(\mid \Pi_i \mid), \tag{2.9}$$

where $\theta'_{x|\pi}$ is the maximum likelihood estimate of the conditional probability $P(X_i = x \mid \Pi_i = \pi)$, and $N_{x,\pi}$ is the minimum number of times $(X = x \wedge \Pi_i = \pi)$ appears in the dataset. This process has been used for the purposes of parent set identification for implementing OBS and WINASOBS algorithms.

2.5.3 Additive Noise Models

This sub-section discusses a family of causal discovery methods that exploits *additivity* of the noise.Multivariate cases have been discussed in Hoyer et al.(2009); Peters et al.(2014). This subsection discusses the binary case. A lot of work has been done on causal modeling and causal discovery that assumes that any effect is a linear function of the cause plus an independent Gaussian noise. These models are called the *Structural Equation Models* which have been discussed in great detail above. In the generic sense, *Functional Models* are the models where effects are modeled as nonlinear functions of their causes and latent noise variables(Pearl, 2000).

The family of Additive Noise Models(ANM) assumes the structural equation models $(\mathcal{S}, \mathcal{Q})$ with a set of equations $\mathcal{S} = (S_1, \ldots, S_d)$ of the following form

$$S_i : \boldsymbol{x}_i = f_i(Pa(\boldsymbol{x}_i) + \boldsymbol{n}_i \tag{2.10}$$

where the noise variable $\boldsymbol{n}_i$ and functions f_{i} are continuous for all $1 \leq i \leq d$.

Definition 2.5: *If the joint density p(x, y) satisfies an Additive Noise Model $X \to Y$, but does not satisfy any Additive Noise Model $Y \to X$, then we call the Additive Noise Model* ***identifiable****.*

In general, the following must hold for identifiable additive noise models:

1. The functions f_{i} are linear with nonzero coefficients, and the noise variables $\boldsymbol{n}_{\mathrm{i}}$ are non-Gaussian (Shimizu et al., 2006), or

2. The functions f_{j} are smooth and nonlinear, and the densities of both the equation outputs $\boldsymbol{x}_{\mathrm{i}}$ and the noise variables $\boldsymbol{n}_{\mathrm{i}}$ are strictly postive and smooth (Peters et al., 2014, condition 19).

Apart from these representational, sufficiency and causal Markov assumptions hold for ANMs. The statistical footprint revealing the direction of causation in additive noise models is the dependence structure between the cause and noise variables. Given two random variables $\boldsymbol{x}$ and $\boldsymbol{y}$ with causal relation $\boldsymbol{x} \to \boldsymbol{y}$ and the assumptions described above, there exists an additive noise model

$$\boldsymbol{y} = f(\boldsymbol{x}) + \boldsymbol{n}, \tag{2.11}$$

in the correct causal direction, but there exists no additive noise model

$$\boldsymbol{x} = g(\boldsymbol{y}) + \boldsymbol{n}', \tag{2.12}$$

in the anticausal direction. Due to the definition of the additive noise model, this means that $\boldsymbol{x}$ is independent of $\boldsymbol{n}$, but it cannot be the case that $\boldsymbol{y}$ is independent of $\boldsymbol{n}$'.

Given a consistent nonparametric regression method and a consistent nonparametric independence test, it is possible to decide whether $\boldsymbol{x} \to \boldsymbol{y}$ or $\boldsymbol{x} \leftarrow \boldsymbol{y}$ on the basis of empirical data $\{(x_i, y_i)\}_{i=1}^{n} \sim P^n(\boldsymbol{x}, \boldsymbol{y})$, as n tends to infinity. For both the causal directions, one can proceed by computing a regression function from one variable to the other, and then testing for independence

between the input variable and the obtained regression residuals. The independence tests can be replaced with Gaussianity tests, to discover both linear and nonlinear causal relationships(Hernandez-Lobato et al., 2016).

The additive noise models are not identifiable for structural equations with linear functions and Gaussian noise variables. This can be seen in the following example of two random variables:

- In the Fig 2.4(Lopez-Paz, 2016), we consider a cause variable $\boldsymbol{x} \equiv \mathcal{N}$, a noise variable $\boldsymbol{n} \equiv \mathcal{N}(0, 1)$, and an effect variable $\boldsymbol{y} \leftarrow 2\boldsymbol{x} + \boldsymbol{n}$. We consider the joint distribution $P(\boldsymbol{x}, \boldsymbol{y})$ to be Gaussian. This distribution, being Gaussian, is elliptical and there is no asymmetry that can be used to infer the direction of causation between $\boldsymbol{x}$ and $\boldsymbol{y}$. The regression noise (shown in red) is independent of the cause always.

- In the four plots on the right side of Fig 2.4, we have a cause variable $\boldsymbol{u} \equiv [-1, +1]$, a noise variable $\boldsymbol{e} \equiv [-1, +1]$, and an effect variable $\boldsymbol{v} \leftarrow \boldsymbol{u} + 0.5\ \boldsymbol{e}$. This setup falls under the identifiability conditions of Shimizu et al(2006), since there is no additive noise model in the incorrect causal direction $\boldsymbol{u} \leftarrow \boldsymbol{v}$. This is true because the regression noise(red bars) is dependent on the cause $\boldsymbol{v}$: its variance peaks at $\boldsymbol{v} = 0$, and shrinks as the absolute value of $\boldsymbol{v}$ increases. This asymmetry renders causal inference possible from observing the statistics of the data.

Additive noise models are consistent(Kpotufe at al., 2014), and there exists extensions to discrete variables (Peters et al., 2011), latent variables(Stegle et al., 2010), cyclic graphs (Mooij et al., 2011; Lacerda et al., 2012), and postnonlinear equations $\boldsymbol{x}_i = g_i(f_i(Pa(x_i)) + n_i)$, where $g_{\mathrm{i}} : \mathbf{R} \rightarrow \mathbf{R}$ is an additional monotone function(Zhang and Hyvarinen, 2009).

2.5.4 Information Geometric Causal Inference Models

Additive models rely on the independence between the cause and an extra noise variable. Another model, called the Information Geometric Causal Inference(IGCI) model, exploits Indpendence of Cause and Mechanism(ICM), without introducing any noise. ICM assumption states that the

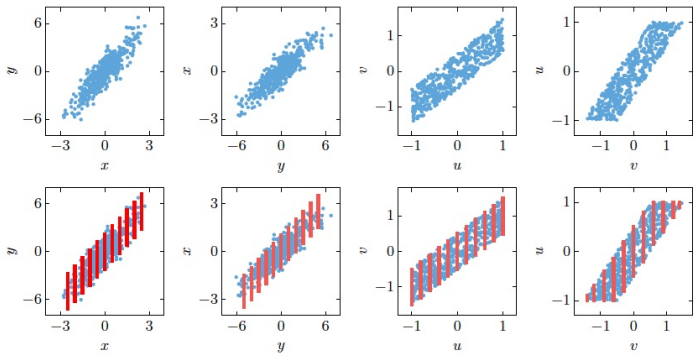

Figure 2.4: Examples of linear additive noise models

.

cause is independent of the mapping of the cause to effect. This assumption can be brought to use in the IGCI methods which prefers the causal direction under which the distribution of the cause is independent from the derivative of the mechanism mapping the cause to effect. Usually, the mechanism used to map cause to effect is an exponentiation function.

2.6 Causal Relation Learning

The method of causal structure discovery proposed by Lopez-Paz talks about the ability to identify $x \rightarrow y$ and $x \leftarrow y$ relations from the data. They have collected the cause-effect pairs from a joint probability distribution:

$$\{(S_i, l_i)\}_{i=1}^{n} \tag{2.13}$$

where

$$S_i = \{(x_{i,j}, y_{i,j})\}_{j=1}^{n_i} \sim P^{n_i}(x_i, y_i) \tag{2.14}$$

where, l_i is either $x_i \rightarrow y_i$ or $x_i \leftarrow y_i$. Featurization has been done to make the variable-length sample S_i, to be of a fixed length. Then, a binary classfier has been trained to distinguish the causal structures.

Much research has been done on data-driven methods such as the ones described above, meanwhile achieving state-of-the-art performance(Guyon, 2013). But the theoretical analysis of these algorithms are difficult because of the quality of the features in these data.

Learning has been proposed from probability distributions too. Muandet et al.(2012) has presented a research on classifying kernel mean embeddings of distributions.

2.6.1 Kernel mean embeddings

In order to featurize probability distributions into a feature vector, kernel mean embeddings have been used(Smola et al., 2007; Muandet, 2015).

Let P be a probability distrubution of a variable z. Then , the kernel mean embedding of P is a continuous, bounded and positive-definite kernel function $k : \mathcal{Z} \times \mathcal{Z} \rightarrow \mathbf{R}$ is defined as in the following equation:

$$\mu_k(P) := \int_{\mathcal{Z}} k(z, .)dP(z), \tag{2.15}$$

such that $\mu_k(P) \in \mathcal{H}_k$ where $\mathcal{H}_k$ is called the Reproducing Kernel Hilbert Space associated with the kernel function k(Schölkopf and Smola, 2001). Usually, k is a characteristic kernel such as the Gaussian kernel, which has been used for their experiments and have defined its form as follows:

$$k(z, z') = exp(-\gamma \mid\mid z - z' \mid\mid^2), \gamma > 0. \tag{2.16}$$

Also, most of the time, one does not have access to the embeddings of the data but instead have access to a sample S from a joint probabillity distribution P as:

$$P_S = \frac{1}{n} \sum_{z_i \in S} \delta_{(z_i)}, \tag{2.17}$$

where $\delta_{(z_i)}$ is said to be a Dirac distribution. The equation above has been approximated by the kernel mean embedding to generate the following(Lopez-Paz, 2016):

$$\mu_k(P_S) = \frac{1}{n} \sum_{i=1}^{n} k(z_i, .) \in \mathcal{H}_k. \tag{2.18}$$

2.6.2 Causal Inference in terms of distribution classification

The learning setup of the research established in Chapter 7 of Lopez-Paz(2016) is organized as follows:

1. There exists a Mother distribution $\mathcal{M}$ defined on $\mathcal{P} \times \mathcal{L}$, where $\mathcal{P}$ is the set of Borel probability measures defined on the space of two causally related variables and $\mathcal{L}$ = {-1, +1}.

2. A joint distrbution P_i is sampled from the set $\mathcal{P}$ as defined above, which relates two causally related variables via the two possible causal structures as described above(x causes y, and y causes x). The label associated with P_i is l_i. Every $\{(P_i, l_i)\}_{\mathrm{i}\,=\,1}{}^{\mathrm{n}}$ is sample from the Mother distribution $\mathcal{M}$.

3. In practice, access to the joint probabilities are a bit rare. So, the observational samples derived from the probability values are taken into consideration, such that every sample S_i = $\{(x_{i,j}, y_{i,j})\}_{j=1}^{n}$ for all $1 \leq i \leq n$.

4. Every sample is featurized into a kernel mean embedding. If k is a characteristic kernel, then there is no loss of information with the embedding.

5. The high-dimensional embeddings are approximated into a low-dimensional vector, which is actually an $m-$ dimensional vector of fixed length. The conversion requires $O(m)$ computation time and $O(1)$ extra storage.

The low-dimensional vectors, coupled with the labels, form the training data for the learning problem.

2.7 Pairwise Causal Inference

Previous work by Lopez-Paz(2016) introduces a classification approach to identify the direction of causation between a pair of variables, given the data associated with the variables. In order to build a classification model, they have considered only the direct causal and direct anti-causal cases and generated data from the two networks: 1) $\boldsymbol{x} \rightarrow \boldsymbol{y}$ and 2) $\boldsymbol{x} \leftarrow \boldsymbol{y}$.

Training data is of the following form:

$$D = \{(S_i, l_i)\}_{i=1}^{n} \tag{2.19}$$

where

$$S_i = \{(x_i, y_i)\} \tag{2.20}$$

and l_i is 1 or -1 depending on whether the structure is "directly" causal or anti-causal.

This data is used to train a machine learning model and the testing is done on the 82 pairs of cause-effect benchmark data. For each of the cause-effect pairs of data, the model constructed is used to predict the direction of the arrow between the pairs of variables. Accuracy obtained with this dataset is 81.6%.

2.8 Structure Learning

Structure learning is an NP-hard problem. The task is to find the best network structure that maximizes a scoring function. The standard way is to perform heuristic search over some space. Search strategies currently in use are greedy hill-climbing over the space of network structures, modified with a tabu list and random restarts. Recently, some more powerful algorithms have also been proposed.

Teyssier(2005) discusses an algorithm to find a high-scoring network structure. Their approach is based on Cooper(1992) and Buntine(1991) and states that given an ordering of the variables in the network, finding the highest scoring network consistent with the ordering is not NP-hard. If the degree of every node is bound to k, this entire task can be accomplished in $O(n^k)$ time where n is the number of variables. Because of this observation, a search over the space of orderings can be performed rather than in the space of network structures. The score of an ordering is defined as the score of the best network that is consistent with it. A set of local search operators have also been defined. The operator is the flipping of a pair of adjacent nodes in the ordering so that we traverse the order space and use greedy hill-climbing search, with a tabu list and random restarts, to find an ordering that maximizes the score of the network. The idea of using orderings to provide a more global view on structures was used by Larranaga et al.(1996) who used a genetic algorithm to search over structures. Apart from this, Koller(2003) used MCMC over the space of orderings for Bayesian model averaging for structure discovery.

Properties of ordering based search:

1. The space of network structures($2^{\Omega(n^2)}$) is larger than the space of orderings($2^{O(nlogn)}$).
2. Each step in search makes a more global modification to the original ordering.
3. The branching factor in the space of orderings is $O(n)$ rather than $O(n^2)$, reducing the cost of evaluating candidate successors in each step.

The disadvantage of ordering based search is the need to compute a large set of statistics for every variable and for every parent set. This cost can be very high if the number of data instances are large. Steps have been taken to reduce this cost by using AD-trees from Moore and Lee(1997) and by pruning the space of possible parents for each node(Friedman, 1999).

The main goal behind this algorithm is to find the best network structure that is a good predictor for the data. The approach used by their paper is to define a scoring function $score(G : D)$ for the network G and data set D which evaluates different networks relative to D. One of the main scoring functions used is BIC score which has been discussed earlier. Two assumptions are made for using these scoring functions:

- The score is decomposable: The sum of the scores associated with individual nodes and its parents are given as:
$$score(G) = \sum_{i=1}^{n} score(X_i, Pa_G(X_i)) \tag{2.21}$$
- The second property is the reduction of the score to sufficient statistics associated with individual nodes and their parents(Teyessier and Koller).

Given a scoring function, the task is to find

$$argmax_G score(G) \tag{2.22}$$

This kind of problem is also considered to be NP-hard even when the maximum number of parents for every node is restricted to two(Chickering et al., 2003).

2.8.1 Search in Order Space

An idea to reduce the time complexity of the problem is to restrict the number of parents for every node to k. For a given ordering $\mathcal{O}$, the set of possible parents for every node X_i can be given as:

$$\mathcal{U}_{i,\mathcal{O}} = \{U : UprecedesX_i, \mid U \mid \leq k\} \tag{2.23}$$

The optimal parent set for each node X_i is

$$Pa_{\mathcal{O}}(X_i) = argmax_{U \in \mathcal{U}_{i,\mathcal{O}}} score(X_i, U) \tag{2.24}$$

Their paper claims that the optimal parent set for each node is found in $O(nf_{max})$, where f_{max} is the maximum number of possible families per node.f_{max} is given as $O(n^k)$. Search can be performed using greedy hill-climbing with random restarts and a tabu list. A simple operator in use is the swap operator where search is performed by considering all $n-1$ candidate successors of the present ordering. The tabu list is used to prevent the algorithm from reversing a swap. This process continues until a local maxima is reached.

2.8.2 Acyclic Selection OBS(ASOBS)

This has been proposed by Scanagatta(2015). The consistency rule is relaxed in this algorithm to allow the presence of back-arcs in an ordering while keeping the acyclicity constraint intact. For every X_i, this algorithm chooses the best parent set Pa(X_i) that does not create a cycle with the parent sets Pa(X_j) for all X_j that precedes X_i. ASOBS, in general, obtains a higher scoring network than OBS.

2.8.3 Operators for Local Search in the Space of the Orderings

Once the highest scoring network has been found using either OBS or ASOBS, WINASOBS looks for local changes in the order that allows to further increase the score of the network. A greedy hill-climbing search in the space of orderings $\mathcal{O}$ is also performed. An operator receives an ordering and returns a set of successor orderings.

Figure 2.5: Example of the swap operator

Figure 2.6: Example of the insertion operator. The variable F is inserted at the position of variable C in the initial ordering

The different operators discussed in Scanagatta(2017) are as follows:

1. The *swap* operator is defined as follows:

$$Swap(i) : (X_1, \ldots, X_i, X_{i+1}, \ldots, X_n) \rightarrow (X_1, \ldots, X_{i+1}, X_i, \ldots, X_n) \tag{2.25}$$

   Fig. 2.6(Scanagatta, 2017) shows the working of the swap operator. The score of the highest-scoring network given the new order can be computed only by recomputing the parent sets for X_i and X_{i+1}. Further computation can be saved by noting the following:

   - the parent set of X_{i+1} needs to be recomputed only if it contains X_i.
   - the parent set of X_i needs to be recomputed only if X_{i+1} is a *potential parent* for it.

   Using the swap operator, each ordering has $n-1$ candidate successors; thus one evaluates all such successors and selects the one with the highest score. A disadvantage of this operator is the inability to escape local maxima because of the exploration of only the neighborhood space.

2. The *insertion* operator is defined as follows:

$$insertion(i, j) : (X_1, \ldots, X_j, \ldots, X_i, \ldots, X_n) \rightarrow (X_1, \ldots, X_i, X_j, \ldots, X_n) \tag{2.26}$$

   An example of the insertion operator is shown in Figure 2.7(Scanagatta, 2017). The *swap* operator is, thus, a particular case of *insertion* operator, in which it applies to two adjacent nodes. The insertion operator yields $n(n-1)$ candidate successor orderings.

Figure 2.7: Example of the window operator. The variables F, G, H are inserted in the positions of C, D, E

3. The third operator, called the *window* operator, has been proposed by Scanagatta(2017). $window(i, j, w)$ generalizes the insertion operator. It selects the variables at position i in the original ordering and the $w - 1$ following variables. Such variables are inserted at position j in the new order.

$$window(i, j, w) : (X_1, \ldots, X_j, \ldots, X_i, \ldots, X_{i+w}, \ldots, X_n) \rightarrow (X_1, \ldots, X_i, \ldots, X_{i+w}, \ldots, X_j, \ldots, X_n) \tag{2.27}$$

The insertion operator is a special case of the window operator, characterized by $w = 1$. The window operator yields $(n - (w - 1))(n - w)$ candidate successors for a given ordering. One needs to recompute the optimal parent set consistent with the ordering:

- For each of the w variables in the window $\{X_i, \ldots, X_{i+w}\}$, only if their current assigned parent set contains X_{i-1}.
- For X_{i-1}, only if any one of the w variables in the window is a potential parent for it.

Fig. 2.8(Scanagatta, 2017) shows the working of the window operator.

2.8.4 Exploration meta-heuristic functions

Lee and van Beek(2017) have proposed techniques to better the performance of greedy local search via a meta-heuristic. The proposed approaches are as follows:

- **IINOBS:** is a version of iterative local search. An *insertion* operator is employed and the local maxima is perturbed with respect to a perturbation factor until a threshold or termination condition is reached.

- **MINOBS:** is based on genetic algorithms. Orderings generated with insertion operators are mutated or crossover is performed on it.

Scanagatta(2017) explores this idea with the help of their window operator and call their algorithm *WINASOBS*. The algorithm follows the below mentioned steps:

1. After generating a random order, ASOBS is performed to find a good solution or a good network.

2. As mentioned above, ASOBS allows back-arcs as long as cycles are not introduced. So, a *topological ordering* is generated for the learned network.

3. This ordering is improved using *window* operator.

The final network obtained has a much better structure that the ones learned using OBS or ASOBS

CHAPTER 3. METHODS USED

This section provides an overview of the algorithm employed in this research that:

1. Improves the performance of pairwise causal inference in the cases where confounding variables are also present.
2. Identifies the causal structures between every pair of variables in a large DAG.
3. Recovers the underlying Bayesian network from the total order of the causal graph obtained in the above step.

3.1 Methodology

The central question of this section is: *given samples from two random variables* $\boldsymbol{x}$ *and* $\boldsymbol{y}$*, does* $\boldsymbol{x} \rightarrow \boldsymbol{y}$ *or* $\boldsymbol{y} \rightarrow \boldsymbol{x}$ *?*

For answering this question, an approach similar to the one discussed in the literature review is used. A learning task is set up for the input-output pairs

$$D = \{(S_i, l_i)\}_{i=1}^{n} \tag{3.1}$$

where each input sample

$$S_i = \{(x_{i,j}, y_{i,j})\}_{j=1}^{n} \sim P^{n_i}(x_i, y_i) \tag{3.2}$$

and each output label l_{i} indicates whether $\boldsymbol{x_i} \rightarrow \boldsymbol{y_i}$ or $\boldsymbol{x_i} \leftarrow \boldsymbol{y_i}$. From these data, we build a model in two steps: First, the variable-length input sample S_i is featurized into a fixed-dimensional vector representation. Then, a classifier is trained on the featurized data and the set of labels to distinguish between causal directions. The label l_i can assume any one of the following values:

- $l_i = 0$ if there is no causal relation between X_i and Y_i,
- $l_i = 1$ if $X_i \rightarrow Y_i$

- $l_i = 2$ if $X_i \leftarrow Y_i$

Using data of this form, multiple models have been trained both from artificially generated data and data generated from known Bayesian networks. The following subsections discusses the methods of generating artificial data, featurization of data and the different kinds of classifier models that have been built during the process of this research.

3.1.1 Synthesis of Artificial data

Synthetic data generated consists of $N = 10000$ samples $S_i = \{(x_{i,j}, y_{i,j})\}_{j=1}^{n_i}$ with $n_i = 1000$ for all $1 \leq i \leq n$. The observational sample S_i has a set of the following hyper-parameters:

1. $k_i \sim$ RandomInteger[1, 10),
2. $m_i \sim$ Uniform[0, 10),
3. $s_i \sim$ Uniform[1, 10),
4. $v_i \sim$ Uniform[0, 10]),
5. $d_i \sim$ RandomInteger[4, 10),
6. $f_i \sim$ RandomSpline(d_i)

where RandomSpline is a smoothing spline with d_i knots sampled from Gaussian(0, 1). After sampling one set of random hyper-parameters, the pairs ($x_{i,j}$, $y_{i,j}$) forming the observational sample S_i follows the following generative model:

$$x_{i,j} \sim GMM(k_i, m_i, s_i) \tag{3.3}$$

$$\epsilon_{i,j} \sim Gaussian(0, v_i), \tag{3.4}$$

$$y_{i,j} \sim f_i(x_{i,j}, \epsilon_{i,j}), \tag{3.5}$$

where $GMM(k, p_1, p_2, v)$ is a Gaussian Mixture Model of k_i components with mixing weights sampled from Uniform[0, 1) and normalized to sum to one, component means sampled from

$Gaussian(0, m_i^2)$, and variance magnitudes samples from $Gaussian(0, s_i^2)$. A collection of $\{S_i\}_{i=1}^N$ observational samples with known causal relationships stored in l_i.

Artificial data has been generated with respect to the seven different causal structures as discussed in Section 2.2. For the two node network structures in Fig 2.2 (a), (b), and (c), data is generated in the following format:

$$S_i = \{(x_{i,j}, y_{i,j}), l_i\} \tag{3.6}$$

where $x_{i,j}$, $y_{i,j}$ and $\epsilon_{i,j}$ are sampled as stated in Eq.(3.3), (3.4) and (3.5). l_i is taken to be either 0, 1 or 2 depending on whether the structure from which data is sampled has no causal, causal or anticausal relations.

For the other network structures, data is sampled according to the equations above but featurization of data is done on a pair of variables only. For example, in Fig 2.2 (f), $x_{i,j}$ is sampled from a Gaussian Mixture Model, while $z_{i,j} = f_i(x_{i,j}, \epsilon_{i,j})$, and $y_{i,j} = f_i(z_{i,j}, \epsilon'_{i,j})$ and l_i is considered to be 1. The same process is applied to the other structures as well.

3.1.2 Featurization of observational samples

After the construction of observational samples $S_i = \{(x_{i,j}, y_{i,j})\}_{j=1}^{n_i}$, the featurized training data is represented as:

$$D = \{(M(\{(x_{i,j}, y_{i,j})\}_{j=1}^{n_i}), l_i)\}_{i=1}^N \tag{3.7}$$

Here, the assumption is made that all observational samples have zero mean and unit variance. Here, the featurization map M accepts an observational sample S_i which takes the following form:

$$M(S_i) = \frac{1}{n}\sum_{j=1}^{n_i}((cos(\langle w_k^x, x_{i,j}\rangle + b_k))_{k=1}^m, (cos(\langle w_k^y, y_{i,j}\rangle + b_k))_{k=1}^m, (cos(\langle w_k^y, w_k^y, (x_{i,j}, y_{i,j})\rangle + b_k))_{k=1}^m) \in \mathbf{R}^{3m}, \tag{3.8}$$

where w_k^x, $w_k^y \sim \mathcal{N}(0, 2\gamma)$, $b_k \sim \mathcal{U}[0, 2\pi]$, and $m = 500$, for all $1 \leq k \leq m$.

The aim is to concatenate the M associated with all bandwidths $\gamma \in \{10^{-2}, \ldots, 10^2\}$, following the multiple kernel learning strategy as described in Lopez-Paz, 2016(Remark 3.3).

The synthetic featurized observational data D contains $3m$ - dimensional real vectors $M(S_i)$ and ternary labels l_i. Now, any kind of standard multi-label classifier can be used to predict the cause-effect relation for a new observational sample S.

3.1.3 Classification Models

A number of different models have been constructed for the purpose of this research.

1. The first model consists of artificial data generated using the principles of Additive Noise Models, but by restricting the set of possible networks to the ones shown in Fig 2.2 b), c), f) and g). The labels included in training is either 1, representing causal and indirect causal direction, or 2, representing anti-causal and indirect anti-causal direction. This is a binary classifier model.

2. The second model is again a binary classifier, but data is generated using a combination of Additive Noise Models and Information Geometric Causal Inference Models. The rest of the procedure is same as in the first model. The networks considered are just direct causal and direct anti-causal. Confounding effects are not taken into consideration. This is again a binary classifier.

3. The third model consists of artifically generated data and labels as its training data using the same procedure as discussed in Section 3.1.1 and the featurization is done as in Section 3.1.2. This data is used for training as well as for validation purposes. This is a ternary classifier.

4. For the fourth model, the training data is generated in the same way as discussed earlier, but for validation purposes, data generated from known Bayesian networks are used. For constructing the validation data, we consider data from the networks as given in Table 3.1: The data is generated from the networks obtained from the Bayesian Network Repository. The number of features is maintained at $n = 1000$ and the featurization is done in the same way as discussed in Section 3.1.2. For every network considered, the Bayesian network is represented as $G' = (V', E')$, where V' is the set of nodes and E' is the set of edges in the

Table 3.1: Validation Data

Dataset	**Number of nodes**
Earthquake	5
Asia	8
Insurance	27
Alarm	37
Hailfinder	56
Andes	223

DAG G'. If $num =\mid V \mid$, which is the number of nodes in the network, then there can be $num(num-1)/2$ causal pairs in a directed graph. This does not represent the number of edges and can include the "no-causal" relationship. For every edge $(i,j) \in E'$ in the network, the causal label for the pair of nodes (i,j) is stored to be 1, and the causal label for the pair of nodes (j,i) is stored to be 2. All no-causal relationships are considered to be causal label 0. In this case, we store $num(num-1)/2$ labels and for those many featurized observational samples from the dataset generated from the Bayesian network.

5. For the fifth model, the same set of networks as in Table 3.1 are used, but this data is used for training the model instead of using synthetic data.

6. For the sixth model, the same set of datasets as in Table 3.1 are used. But, instead of using the data generated from these networks, data is sampled using the details in Section 3.1.1 for the respective DAGs of these datasets. Again, there can be $num(num-1)/2$ causal pairs for which data is sampled artificially but the labels are in accordance with the causal relationship existing between every pair of variables in the DAG.

7. The seventh and final model is a combination of the first and fourth models and the training data is a combination of artificial data sampled from causal structures in Fig. 2.2 and artificial data generated from the DAG structures of the known Bayesian networks.

After the generation of the training as well as validation data, the next step would be to build a predictor model.

3.1.4 Classifiers used

In order to make the classification decisions, Random Forest Classifier is used, which is an ensemble classifier whose concepts have been discussed in the literature review section. The implementation of Random Forest Classifiers is taken from Python's sklearn library. Random forest classifiers performed better than SVMs and neural networks in terms of validation and training accuracies. The number of trees is chosen from {100, 250, 500, 1000, 5000} via cross-validation.

3.1.5 Pairwise Causal Inference

For the purposes of recovering the causal direction between a pair of variables and their data, the first and second models as described in Section 3.1.3 are used for training a classification model. Testing is performed on the 82 pairs of cause-effect benchmark dataset. The accuracy obtained from both the models have been documented in the next chapter.

3.1.6 Causal Inference from a large DAG and Bayesian Network Learning

Once the ternary classification model is ready to use, it has been utilized for predicting the causal relationships between every pair of variables on an unseen observational sample S. The run-time of this algorithm is $O(n^2)$. After generating the required information, it can be seen that the information generated for every pair of variables gives rise to a causal graph. For every unique pair of variables, the model predicts the label that shows the causal structure between those two variables and it can also predict the probability values which shows the probability of the occurrence of every label for those two variables. These probability values are stored for future use.

From the causal graph, a topological or total order of the variables has been generated. Generation of a single topological order of the graph does not give out better results. So, instead of generating just a single order from the causal graph, a variety of different orders have been generated using a variant of *Depth First Search* which takes into account the probability values of whether $X_i \rightarrow X_j$ or whether $X_i \leftarrow X_j$.

The skeleton of the algorithm used for DFS has been documented below.

```
DFS(G, information):
Input: (G, information)
Result: finalOrders
if information is empty then
    p = topologicalSort(G) if p is not in finalOrders then
        add p to finalOrders
    end
end
currentInformation = pop(information)
x = cuurentInformation[0]
y = currentInformation[1]
prob(x, y)[z] = currentInformation[2]
if (x is an ancestor of y) or (y is anancestor of x) then
    DFS(G, information)
end
else
    if prob(x, y)[1] > prob(x, y)[2] then
        DFS(G + x→y, information)
        DFS(G + x←y, information)
    end
    else
        DFS(G + x←y, information)
        DFS(G + x→y, information)
    end
end
```

In the algorithm, the initial G is an empty DAG. The collection of ordered pairs, *information*, such that each element of the set contains the cause variable($\boldsymbol{x}$) at its first index, the effect variable($\boldsymbol{y}$) at its second index and the set of probability values for each possible label between $\boldsymbol{x}$ and $\boldsymbol{y}$. This set of information is sorted with respect to the largest probability value for each $\boldsymbol{x}$ and $\boldsymbol{y}$. We also check that when adding edges to a DAG, we don't add edges between variables who have an ancestor-descendant relationship, in order to avoid the formation of cycles in the DAG.

After the generation of multiple topological orders, these have been tested by running experiments using OBS and WINASOBS algorithms. Instead of generating random numbers as the orders for OBS and WINASOBS, the orders generated from the DFS are provided as an input to these algorithms and the final Bayesian network score generated from our algorithm is compared against the Bayesian network scores generated using random orders.

CHAPTER 4. EXPERIMENTS AND RESULTS

This section reports all the experiments performed and results that have been obtained from the experiments.

4.1 Experiments

In this section, all the experiments that have been performed in this research have been reported. The first subsection discusses the data sets on which these experiments have been performed and the next four subsections discusses the nature of the experiments .

4.1.1 Data sets

For testing whether pairwise causal inference performs well in the presence of confounders or not, the same set of 82 cause-effect pairs have been used as reported in Lopez-Paz(2016).

For the second set of experiments, testing is performed on data generated from a large DAG. Testing data is generated based on the presence of edges between every pair of variables and the confounding effects, if present, between those variables. This data is not generated from the conditional probability distributions provided to us with the network, but is generated using the ANM approach.

The data networks used for this purpose are listed in the following table.

Table 4.1: Data networks used in experiment 2

Network	**Size**
Insurance	27
Alarm	37
Hailfinder	56
Andes	223
Diabetes	413

All the datasets used have 5000 observational samples.

For the third set of experiments, quality of the orders generated from the DFS algorithm described in Section 3.1.6 is compared against the quality of random orders that the standard ordering-based search algorithms use. For this, one needs to know the exact structure of the DAG whose data is used for generation of orders from our algorithm. The networks that have been used are summarized in Table 4.2. The networks are available in Bayesian Network Repository. Some of the datasets are also available in bnlearn package of R. For the others, whose datasets were not available, data has been generated using the bnlearn package.

Table 4.2: Data networks used in experiment 3

Network	**Size**
Earthquake	5
Asia	8
Insurance	27
Alarm	37
Hailfinder	56
Andes	223
Diabetes	413
Pigs	441
Link	724

All the datasets used have 5000 observational samples.

For the fourth set of experiments, we generate the orders from the causal graph generated from the data. These datasets are used by the BLIP software to report the performances of OBS, ASOBS, WINASOBS. The datasets and their sizes are as shown in Table 4.3

4.1.2 Experiment 1

The first set of experiments deals with the performance of pairwise causal inference on the cause-effect benchmark data pairs. The results obtained from both the binary models as described in Section 3.1.3 and listed as the first and second model respectively, have been documented in the Table 4.4

Table 4.3: Data sets used in experiment 4

Network	**Size**
accidents	111
ad	1556
baudio	100
bbc	1058
bnetflix	100
book	500
c20ng	910
cr52	889
cwebkb	839
diabetes	413
dna	180
jester	100
kdd	64
kosarek	190
link	724
msnbc	17
msweb	294
munin	1041
nltcs	16
pigs	441
plants	69
pumsb_star	163
tmovie	500
tretail	135

4.1.3 Experiment 2

These experiments try to expand the horizon of pairwise causal inference to causal inference between every unique pair of variables in a large DAG. For this model, the training data is generated as described in model 3, which is a ternary classifier, of Section 3.1.3. Test data from the networks in Table 4.1 are generated using the ANM principles. The predicted causal relations are checked against the known causal relations from the DAGs. The accuracies obtained are as documented in Table 4.5.

The overall accuracy seemed to be not so good, thus the next experiment was to check whether the top most confident predictions are accurate or not. For this, the accuracy and confusion matrices of the most confident n, $2*n$, $3*n$, ... and so on, predictions are reported, where n is the number of variables in the network. These are named *fraction predictions*. The fraction prediction accuarcies obtained for each of the five networks in Table 4.1 are reported in Tables 4.6, 4.7, 4.8, 4.9 and 4.10.

4.1.4 Experiment 3

The third set of experiments that have been performed is to analyze the quality of the orders that have been generated using the algorithm discussed in the section above. For the purposes of checking the quality of the orders generated, the following metric is used:

$$metric = number\,of\,correct\,causal\,pairs/total\,number\,of\,causal\,pairs \tag{4.1}$$

where the $total\ number\ of\ causal\ pairs = num(num-1)/2$, where $num = \mid V \mid$ for a given DAG $G = (V, E)$. These are the DAGs that are already available to us and can be found in the Bayesian Network Repository.

For counting the number of correct causal pairs, the topological order generated from the causal graph is taken and there can exist $n(n-1)/2$ relationships in the order of length n. As the order is a total order, we can state that for any order $(X_1, \ldots, X_n)$, there might be an edge from X_i to X_j if X_i precedes X_j in the order, but there cannot be an edge from X_i to X_j if X_i succeeds X_j in the order.

Exploiting this fact gives rise to the same number of relationships between every pair of successive variables as there are causal pairs in the original DAG. Now, one can cross-reference these two and check how many causal relations asre correctly predicted by the order. Thus, we can evaluate our defined metric.

To make valid comparisons, the quality of random orders is caluculated using the same metric. As is employed by OBS algorithms, a random order is generated. The order is a collection of numbers, where each number represents a node in the DAG. If $\mid V \mid = n$, then an order will have n

unique numbers in the range $\{0, \ldots, n-1\}$. These random collection of numbers can be compared with the $n(n-1)/2$ causal relations from the DAG to evaluate the same metric as defined above.

To better compare the two, 30 minutes of time has been given to generate orders from our algorithm and 30 minutes of time has been given to generate random orders. To compare the quality of the orders, the minimum, maximum and average of both the metrics have been computed and have been reported for all the 5 different types of classifier models built(as described in Section 3.1.3 as models three through seven) and are summarized in Tables 4.11, 4.12, 4.13, 4.14 and 4.15. These experiments have been performed on the data sets of Table 4.2.

4.1.5 Experiment 4

The fourth set of experiments comprises of the Bayesian network learning algorithms as described earlier. Two main algorithms used for the recovery of the Bayesian network structures are OBS and WINASOBS. The implementations of these are found on the website of the BLIP software. Changes have been made to read the set of fixed orders generated from the causal graph discovery step, instead of generating random orders.

The results section summarizes the Bayesian network scores obtained after running WINASOBS for 1 hour and running OBS for 1 hour, on random orders. To make a valid comparison, orders have been generated using the Depth First Search technique for a time of 50 minutes. The order space is searched for the best Bayesian network using greedy hill climbing for the next 10 minutes. Both OBS and WINASOBS is used to search the order space. All the results have been summarized in the next subsection.

4.2 Results

4.2.1 Results from Experiment 1

The results obtained from experiment 1 on pairwise causal inference have been reported in the Table 4.4 after training the binary classification models using the first two procedures discussed in the Section 3.1.3.

Table 4.4: Results from PairWise Causal Inference

Training model used	**Accuracy**
Classifier 1	83.34%
Classifier 2	73.17%

Here the model 1 represents the data generated using ANM principles after taking confounders into consideration. An accuracy of 83.34% slightly improves the accuracy of 81.6% reported in earlier work which does not consider the presence of confounders.

4.2.2 Results from Experiment 2

The Table 4.5 shows the accuracies of causal inference procedure on DAGs.

Table 4.5: Results from Causal Inference in DAGs

Network	**Accuracy**
Insurance	57.26%
Alarm	44.14%
Hailfinder	51.29%
Andes	45.52%
Diabetes	40.59%

The results obtained here seem to decrease in accuracy as the size of the network gets larger. But, the performances are still better than random guessing, which comes out to be around 33.33%.

The next five tables summarizes the accuarcy of the fraction predictions on each of the DAGs mentioned in Table 4.1

The fraction predictions thus obtained show that the most confident predictions are usually the most accurate ones, especially in the larger networks. This information is partial and can be utilized to generate partial orders instead of total orders in the structure learning step. The intended use of fraction predictions is a part of future work that needs to be addressed.

Table 4.6: Fraction Predictions on Insurance

Fraction	**Accuracy**
n	55.56%
2n	59.26%
3n	51.85%
4n	51.85%
5n	52.6%
6n	53.70%
7n	50.79%
8n	50.92%
9n	52.67%
10n	54.44%

Table 4.7: Fraction Predictions on Alarm

Fraction	**Accuracy**
n	94.59%
2n	90.54%
3n	81.08%
4n	68.91%
5n	63.78%
6n	57.21%
7n	53.28%
8n	52.03%
9n	52.55%
10n	51.08%

4.2.3 Results from Experiment 3

As described in the previous section, experiment 3 deals with the comparison of the orders generated from our algorithm and the random orders after running both of them for 30 minutes each. These experiments are run on the networks from Table 4.2 on all the classifier models as discussed in Section 3.1.3. The results are summarized in Table 4.11, 4.12, 4.13, 4.14 and 4.15 for the last classifier models respectively.

As can be seen, classifier 3 performs the best in terms of generating orders with quality better than random orders. For the next set of experiments, the model from classifier 3 has been used.

Table 4.8: Fraction Predictions on Hailfinder

Fraction	**Accuracy**
n	100%
2n	71.43%
3n	61.31%
4n	48.66%
5n	40%
6n	35.41%
7n	31.63%
8n	29.24%
9n	30.16%
10n	31.78%

Table 4.9: Fraction Predictions on Andes

Fraction	**Accuracy**
n	99.51%
2n	99.51%
3n	98.54%
4n	95.09%
5n	89.49%
8n	76.54%
16n	51.75%
24n	43.18%
32n	40.08%
64n	35.81%

4.2.4 Results from Experiment 4

The following two tables summarizes the comparison of OBS and WINASOBS after running both the ordering based algorithms on random orders and on orders generated from the DFS.

Table 4.17 summarizes the results of running OBS for and WINASOBS for 1 hour each on orders generated from the DFS described in the previous chapter.

Table 4.10: Fraction Predictions on Diabetes

Fraction	**Accuracy**
n	100%
2n	99.78%
3n	99.40%
4n	99.33%
5n	98.83%
8n	97.19%
16n	86.29%
24n	69.57%
32n	61.31%
64n	52.07%

Table 4.11: Results on Classifier 3

Networks	Orders from DFS			Random orders		
	Min	Max	Mean	Min	Max	Mean
Earthquake	0.0	0.8	0.4	0.0	0.8	0.4
Asia	0.0	0.64	0.32	0.0	0.64	0.32
Insurance	0.19	0.41	0.408	0.05	0.42	0.239
Alarm	0.189	0.546	0.473	0.05	0.282	0.162
Hailfinder	0.09	0.647	0.422	0.039	0.209	0.128
Andes	0.198	0.431	0.413	0.137	0.255	0.195
Diabetes	0.271	0.40	0.342	0.257	0.357	0.308
Pigs	0.003	0.254	0.131	0.006	0.014	0.01
Link	0.003	0.198	0.140	0.023	0.039	0.031

Table 4.12: Results on Classifier 4

Networks	Orders from DFS			Random orders		
	Min	Max	Mean	Min	Max	Mean
Earthquake	0.0	0.8	0.4	0.0	0.8	0.4
Asia	0.0	0.64	0.32	0.0	0.64	0.32
Insurance	0.17	0.37	0.252	0.05	0.42	0.239
Alarm	0.19	0.301	0.22	0.05	0.282	0.162
Hailfinder	0.05	0.343	0.212	0.039	0.209	0.128
Andes	0.03	0.212	0.109	0.137	0.255	0.195
Diabetes	0.005	0.10	0.007	0.257	0.357	0.308
Pigs	0.01	0.272	0.115	0.006	0.014	0.01
Link	0.002	0.020	0.017	0.023	0.039	0.031

Table 4.13: Results on Classifier 5

Networks	Orders from DFS			Random orders		
	Min	Max	Mean	Min	Max	Mean
Earthquake	0.0	0.8	0.4	0.0	0.8	0.4
Asia	0.0	0.64	0.32	0.0	0.64	0.32
Insurance	0.17	0.41	0.29	0.05	0.42	0.239
Alarm	0.21	0.32	0.29	0.05	0.282	0.162
Hailfinder	0.02	0.23	0.219	0.039	0.209	0.128
Andes	0.03	0.212	0.109	0.137	0.255	0.195
Diabetes	0.005	0.10	0.007	0.257	0.357	0.308
Pigs	0.01	0.272	0.115	0.006	0.014	0.01
Link	0.002	0.020	0.017	0.023	0.039	0.031

Table 4.14: Results on Classifier 6

Networks	Orders from DFS			Random orders		
	Min	Max	Mean	Min	Max	Mean
Earthquake	0.0	0.8	0.4	0.0	0.8	0.4
Asia	0.0	0.64	0.32	0.0	0.64	0.32
Insurance	0.19	0.41	0.408	0.05	0.42	0.239
Alarm	0.189	0.546	0.473	0.05	0.282	0.162
Hailfinder	0.09	0.647	0.422	0.039	0.209	0.128
Andes	0.22	0.31	0.25	0.137	0.255	0.195
Diabetes	0.04	0.212	0.119	0.257	0.357	0.308
Pigs	0.001	0.05	0.023	0.006	0.014	0.01
Link	0.002	0.022	0.019	0.023	0.039	0.031

Table 4.15: Results on Classifier 7

Networks	Orders from DFS			Random orders		
	Min	Max	Mean	Min	Max	Mean
Earthquake	0.0	0.8	0.4	0.0	0.8	0.4
Asia	0.0	0.64	0.32	0.0	0.64	0.32
Insurance	0.19	0.41	0.408	0.05	0.42	0.239
Alarm	0.189	0.546	0.473	0.05	0.282	0.162
Hailfinder	0.09	0.647	0.422	0.039	0.209	0.128
Andes	0.178	0.226	0.201	0.137	0.255	0.195
Diabetes	0.18	0.21	0.207	0.257	0.357	0.308
Pigs	0.001	0.004	0.002	0.006	0.014	0.01
Link	0.05	0.10	0.08	0.023	0.039	0.031

Table 4.16: Bayesian networks scores from OBS with random orders versus generated orders

Network	**Random**	**Generated from DFS**
accidents	-10459.540	-11860.401
ad	-10722.270	-10751.798
baudio	-37909.214	-37911.351
bbc	-45020.845	-32977.215
bnetflix	-14574.756	-14668.426
book	-37949.899	-37630.639
c20ng	-124506.819	-155388.328
cr52	-30941.078	-30827.039
cwebkb	-39015.112	-39034.799
diabetes	-2269454.885	-2319157.760
dna	-36013.085	-36218.709
jester	-46858.324	-45557.820
kdd	-59008.437	-60822.601
kosarek	-34575.824	-39096.619
link	-157673.413	-144462.516
msnbc	-151624.681	-151624.681
msweb	-29536.963	-28542.545
munin	-1611981.270	-1616805.482
nltcs	-5836.562	-5836.562
pigs	-1070522.140	-1014322.357
plants	-18106.365	-22570.593
pumsb_star	-11631.515	-11636.230
tmovie	-9454.046	-10506.751
tretail	-24356.053	-24252.306

Table 4.17: Bayesian networks scores from WINASOBS with random orders versus generated orders

Network	**Random**	**Generated from DFS**
accidents	-856.304	-891.802
ad	-7834.330	-9628.437
baudio	-33754.757	-33913.995
bbc	-25303.285	-30889.077
bnetflix	-12285.907	-12284.019
book	–30669.928	-30802.724
c20ng	-69801.484	-71084.460
cr52	-13299.655	-13851.880
cwebkb	-18017.408	-18221.050
diabetes	-1913081.630	-1915905.858
dna	-18461.574	-18484.449
jester	-45557.820	-37212.597
kdd	-57216.396	-57227.539
kosarek	-24814.180	-24552.242
link	-36973.546	-38605.601
msnbc	-151624.681	-151624.681
msweb	-25795.179	-25880.024
munin	-1031449.910	-1034520.173
nltcs	-5836.562	-5836.562
pigs	-800684.722	-812524.905
plants	-14745.945	-14748.302
pumsb_star	-3068.467	-3068.960
tmovie	-4970.129	-5229.745
tretail	-20214.213	-20278.973

CHAPTER 5. SUMMARY AND CONCLUSION

This research project focusses on recovering the causal direction between a pair of variables, extending this inference to every pair of variables in a DAG and finally attempting to find the best fit Bayesian network from the data. For that purpose, we have used a causal discovery step by modeling it as a classification problem. Score-and-search based methods employed performs a search over the space of total orders of nodes in the network and recovers the best fit networks using OBS and WINASOBS searching algorithms.

The first few set of experiments dealt with pairwise causal inference in the presence of confounders. The accuracy obtained on the benchmark cause-effect data slightly improves the previously obtained results whose classification models do not consider confounders.

The next step is to discover a causal structure between every pair of variables. The algorithm implemented is a simple classification problem and runs in $O(n^2)$ time.

The results obtained show that the accuracies obtained from fraction predictions are higher, suggesting that the most confident predictions can be used to generate a partial order instead of total orders. Future work comprises of utilizing these partial orders to generate a network structure.

Another contribution made is the way in which multiple topological orders have been generated from the causal information generated in the previous step. The results obtained achieves Bayesian network score close to the ones obtained using WINASOBS for the medium-sized networks, using the *window* operator with a window of size 5. The experiments have been reported on the 24 different benchmark datasets from the BLIP library. The results also suggest that there have not been much significant improvement in the Bayesian network scores due to the underlying assumption that all the Bayesian networks are Additive Noise Models. In practice, Bayesian networks do not follow the principles of ANMs. More work can be done to explore different models of data that do not restrict the data generation procedure.

REFERENCES

Mooij, J. M., Peters, J., Janzing, D., Zscheischler, J., Scholkopf, B. (2016). Distinguishing Cause from Effect Using Observational Data: Methods and Benchmarks *Journal of Machine Learning Research*

Scanagatta, M., Corani, G., P.de Campos, C. and Zaffalon, M. (2017).Learning Bayesian Networks with Thousands of Variables.

Scanagatta, M., Corani, G, P.de Campos, C. and Zaffalon, M. (2017). Improved Local Search in Bayesian Network Structure Learning. *Proceedings of Machine Learning Research.*

Lee, C. and van Beek, P. (2016). Metaheuristics for Score-and-Search Bayesian Network Structure Learning.

Bartlett, M. and Cussens, J.(2015) Integer linear programming for the Bayesian network structure learning problem

Chickering,D. M. , Meek, C. and D. Heckerman, Kaufmann, Morgan(2003). Large-sample learning of Bayesian networks is hard. *Proceedings of the 19st Conference on Uncertainty in Artificial Intelligence, UAI- 03*

Cooper, G. F. and Herskovits, E. (1992)A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*

Cussens, J., Malone, B. and Yuan, C.(2013) . IJCAI 2013 tutorial on optimal algorithms for learning Bayesian networks.

I. Alonso-Barba, J., de la Ossa, L., and M. Puerta, Jose(2011). Structural learning of Bayesian networks using local algorithms based on the space of orderings. *Soft Computing* 15(10): 18811895, 2011.

Teyssier, M., Koller, D.(2005) Ordering-based search: A simple and effective algorithm for learning Bayesian networks. *Proc. of UAI.* pp. 548 —549

Pearl, J.(1988) Probabilistic reasoning in Intelligent Systems. Morgan Kaufmann.

Lopez-Paz, D., Muandet, K., Schölkopf, B. and Tolstikhin, I. (2015) Towards a Learning Theory of Cause-Effect Inference

Lopez-Paz, D., Nishihara, R., Chintala, S., Schölkopf, B. and Bottou, L. (2017) Discovering Causal Signals in Images.