Big Data Management and Analytics Master Thesis

Ledia Isaj

Combining Bayesian Network and Ontology

prepared at Engie lab CRIGEN
Defended on September 2021

Advisors:	Ahmed Mabrouk	-	Engie lab CRIGEN	ahmed.mabrouk@engie.com
	Sarra Ben Abbes	-	Engie lab CRIGEN	sarra.ben-abbes@external.engie.com
	Lynda Temal	-	Engie lab CRIGEN	lynda.temal@engie.com;
Tutor:	Nacéra Seghouani	-	CentraleSupelec	nacera.seghouani@centralesupelec.fr

Acknowledgments

I would like to express my gratitude to the people who supported me throughout this journey.

First and foremost, I want to express my gratitude to my supervisors Ahmed MABROUK, Sarra BEN ABBES, and Lynda TEMAL for their continuous support during the internship. Their guidance, advice, motivation, and support were key components to the finalization of the thesis. I hold you in great esteem. I would also like to thank Prof. Nacéra SEGHOUANI for supporting us since the beginning, starting with the process of finding the internship, advice on elaborating the master thesis, and insightful feedback.

Secondly, I want to thank all the people that gave me the opportunity to join the Big Data Management and Analytics (BDMA) program and gain this amazing experience. Therefore I would like to show appreciation to the Consortium, the Erasmus+ Program, the jury members, and all affiliated partners for organizing this program, granting me a scholarship, and have the opportunity to obtain knowledge from some of the most qualified professors. It was an honor for me to be a student of BDMA professors. Special thanks go to Charlotte MEURICE, who went above and beyond to welcome us into this master, provide support since the beginning, and most importantly continue to support us throughout this experience, especially during the difficult times of the pandemic, making sure that we were handling it optimistically and encourage not to be down with morale.

I had the chance to meet great friends in this master, whom I would like to thank for making this journey easy and beautiful. It has been a pleasure meeting you and sharing these last two years together. I have great expectations for all of you and I am sure that all of you will accomplish your goals and have a bright future ahead. I would like to give special thanks to one of my fellow colleagues, my study partner, and one of my closest friends, Karim MAATOUK. I am very grateful that I had the opportunity to meet such a genuine friend. Thank you for being there for me in good and bad moments. I am sure that we will continue to keep in touch in the future, even though it might be in distance like the last three semesters.

Last, but not least, I would like to thank my family, to whom I owe all that I am today. Thank you for supporting me throughout all my steps, motivate me to achieve my goals, and always stay strong. Even though virtually, they have lived this experience with me, holding my hand while I take my steps. I would like to give special thanks to my sister, Suela ISAJ, to whom I am indebted for inspiring me to be part of the BDMA program. You are an amazing role model, and an even more amazing and supporting sister.

Ledia ISAJ

Contents

Abstract

Knowledge Representation and Reasoning have long been recognized as the core of Artificial Intelligence. There are several techniques of knowledge representation proposed by different disciples, including Ontologies and Bayesian Networks (BN). Ontologies allow logical reasoning about concepts linked by semantic relations within a knowledge domain. A major limitation of ontologies is their inability to handle uncertainty. Probabilistic Graphical Models are powerful tools for representing and reasoning under uncertainty. BN is a type of Probabilistic Graphical Model, that has emerged as a practically feasible framework of expert knowledge. However, BNs require considerable time for their building phase and it grows more than exponentially as the network size grows. Ontologies and BNs represent two different paradigms, deriving from different disciplines of knowledge representation. Nonetheless, they share a handful of similarities that have motivated several researchers to investigate combining them. Despite significant achievements in this area, it seemed that using the ontology to tackle the major shortcoming of the building phase of BN had not been explored fully. In this thesis, we propose an approach that tries to learn the BN structure using information derived from the ontology. We believe that by using certain non-taxonomic relations in the ontology that encapsulate dependency meaning, we can derive a similarity score between concepts that can be efficiently utilized to derive prior information in the form of constraints or initial structure as input of the BN score-based structure learning algorithm. By doing so, the expert knowledge contributes to the BN construction phase, hence allowing to improve the final result. Experiments were performed on a dataset containing 800000 records. The hill-climbing algorithm was used in this work to learn the structure of the BN, giving as input the prior information derived from the ontology. All of our algorithms were tested in different data sizes. We evaluate the quality of the BN structure learned, by performing cross-validation to obtain unbiased estimates of a model's goodness of fit. We targeted all 28 variables in our model and calculated the expected prediction loss. The algorithms were compared with each other and the Hill-climbing one without prior information. Our results show that the information derived from the ontology is meaningful and can lead to better prediction for some of the variables, as well it can lead to improvement in terms of time.

Keywords: Bayesian Network, Ontology, Structure learning, Hill-Climbing, Non-taxonomic relations

Chapter 1

Introduction

1.1 Context and global problem

Knowledge Representation and Reasoning (KRR) have long been recognized as the core of Artificial Intelligence (AI). The term knowledge representation has been described as "using formal symbols to represent a collection of propositions believed by some putative agent" [Brachman 2004]. The goal is to mimic human knowledge and reasoning, in order to gain intelligent AI agents. It is known that the more information you have, the chances to make accurate decisions and predictions are higher. In a perfect scenario, where all the information needed is available, there would be "perfect" decisions with no risk. However, this is not the case in the real-world applications. Not all the information is available, or accurate, and the problem might be hard to tackle. Naturally, humans process information in a highly complex manner. Apart from straightforward knowledge like facts, general knowledge about objects, concepts, events, people, etc., humans take into consideration intuition, beliefs, common sense, which are not comprehensible by the machines. Knowledge representation models in AI offer a great opportunity to address this problem by allowing to encode human knowledge in a machine-readable format, reason according to this stored information, to derive intelligent decisions.

Several techniques for knowledge representation have been proposed by different disciples. This work is focused on ontologies and probabilistic graphical models. Ontology is well known for representing knowledge in a domain of discourse. An ontology can be viewed as a formal explicit description of concepts in a domain of discourse (classes/-concepts), properties of each concept describing various features and attributes of the concept (properties), and restrictions on them[Noy 2001]. Ontologies allow logical reasoning about concepts linked by semantic relations within a knowledge domain. A major limitation of ontologies is their inability to handle uncertainty. Since the use of ontologies has increased more and more, the demand from the user communities increases for ontology formalisms with the power to express uncertainty [Costa 2006]. Probabilistic Graphical Models (PGMs) are powerful tools for representing and reasoning under uncertainty. Bayesian networks (BN) are a type of PGM, that has emerged as a practically feasible framework of expert knowledge encoding and as a new comprehensive data analysis framework. BN model is represented by a graphical structure that encodes the set of dependencies among random variables. It is also composed of a set of conditional probability tables to represent uncertainties. Both components are jointly used to perform automated reasoning under uncertainty by enabling to answer efficiently various types of probabilistic queries [Pearl 2014]. BN has proved to be a successful tool for quite some time in several real-world applications such as medical diagnosis and prognosis, spam analysis, information retrieval, and natural language processing (cf. [Druzdel 2000], [Heckerman 1994], [Kennett 2001], [Torres-Toledano 1998]). Despite the fact that they are useful in a lot of domains, they require considerable time for their building phase which is considered to be a hard problem to tackle. Given a scoring function, the problem of searching the best BN structure is known to be NP-complete [Chickering 1996].

Ontologies and BNs represent two different paradigms, deriving from different disciplines of knowledge representation. Nonetheless, they share a handful of similarities that have motivated several researchers to investigate combining them. The main directions that have been explored:

- Introducing additional notations to represent probabilistic values in the ontology (e.g. [Yang 2005], [Zhang 2009], [Carvalho 2010], [Mohammed 2016])
- Making use of the semantic richness of the ontology to guide the learning of the BN (e.g. [Fenz 2012], [Jeon 2007], [Ishak 2011a], [Messaoud 2013])
- Using information in BN to enrich the ontology (e.g. [Ishak 2011b], [Wang 2007])

Despite significant achievements in this area, it seemed that using the ontology to tackle the major shortcoming of the building phase of BN had not been explored fully. Some of the previous works in this area use a very specific ontology that can not be extended to other scenarios (e.g. [Helsper 2002], [Bucci 2011]). Some derive the BN from the ontologies rather than learning them. For instance, the concepts are translated to nodes, certain ontology relations are used to link these nodes, and for some proposals, axioms are involved to express nodes or edges or to define the states of variables [Fenz 2012], [Ishak 2011a]. However, in many real-world cases, using object properties as a direct translation is not an option, due to the fact that the ontologies might not manifest such direct dependencies. Moreover, most of these solutions neglect important aspects when they automatically derive the BN structure from the ontology, such as concepts properties, non-taxonomic relations, etc.

In this thesis, we update the state-of-the-art on this area of research and propose an approach that tries to learn the BN structure using information derived from the ontology. We believe that by using certain non-taxonomic relations in the ontology that encapsulate dependency meaning in our use case, we can derive a similarity measure between concepts that can be efficiently used as an input of the BN learning algorithm. By doing so, the expert knowledge contributes to the BN construction phase, hence allowing to improve the final result.

1.2 Objectives and contributions of the work

The main objective of this work consists of using knowledge obtained from the ontology, in scenarios that we don't have prior knowledge of causal dependencies between the variables from the domain expert, or expressed explicitly in the ontology as casual relations. We aim to identify semantic relations in the ontology that encapsulate some dependency nuance between concepts and use the latter to derive a "closeness" metric between them. This metric can be exploited to infer links among the properties (variables in the BN) of the concepts related via these selected relations are set. Thus, we are able to obtain a set of constraints and build an initial structure of the BN. We believe that exploiting this information is a good starting point to learn the BN.

The proposed approach consists of several steps:

1. Dependency extraction from the ontology
2. Subgraph construction and similarity calculation
3. Constraints and initial structure construction
4. Learning the BN using the knowledge derived from the ontology as input

1.3 Research Questions

Therefore, based on the objectives of this thesis, we identify three key research questions of our work:

1. *Research Question (RQ1):* What kind of relations should we use and how can we derive information from them?
2. *Research Question (RQ2):* How to adapt the information derived from the ontology as input for the score-based structure learning algorithms?
3. *Research Question (RQ3):* Does the prior information derived from the ontology improve the learning process in terms of quality of prediction and time?

1.4 Organization of the thesis

The thesis is organized into seven chapters. In Chapter 1 we introduce the problem, the motivation, and our contributions. Chapter 2 presents BN and Ontology, as well as the main remarkable works in the field that try to exploit the benefits of combining these two paradigms, grouped by the direction they follow. In chapter 3 we describe existing algorithms and functions that were used in our approach. Chapter 4 gives a detailed presentation of the approach. In chapter 5 we describe the data we operated on in our use case and experiments. Chapter 6 contains the deployment choices, the attained results, evaluation of our approach, and preliminary conclusions. Finally, we conclude our work and present future directions in Chapter 7.

Chapter 2

Background and State of the art

In this chapter, we start by giving a short overview of the preliminaries that are required to understand the context of this work. We then move on to present the state of art in the area of combining the ontologies and bayesian networks.

2.1 Bayesian Network

Let us denote the finite set of discrete random variables as $U_n = \{X1, X2, ..., Xn\}$, and a generic variable of this set as X_i. The domain of each variable X_i is a finite set $D_i = \{x_{i1}, ..., x_i ri\}$, and x_i will stand for a generic element of D_i.

Definition 1 (Bayesian network) *A Bayesian network (BN) is a graphical representation of a joint probability distribution [Pearl 1988] that includes two components:*

- *A directed acyclic graph (DAG) $G = (U_n, E_G)$. U_n represents the system variables and E_G, the set of arcs, represents direct dependency relationships between variables. Each variable $X_i \in U_n$ has an associated parent set in the graph G, $Par_G(Xi) = \{X_j \in U_n | X_j \to X_i \in E_G\}$. If X_i has no parent (root node), then $Par_G(Xi) = \varnothing$.*
- *A set of numerical parameters,that represent conditional probability distributions. For each variable $X_i \in U_n$, we store a family of conditional distributions depending on its parents, $P(X_i|par_G(X_i))$, one for each possible configuration, $par_G(X_i)$, of the parent set of X_i in the graph. If Xi has no parent, then $P(X_i|par_G(X_i))$ equals $P(X_i)$.*

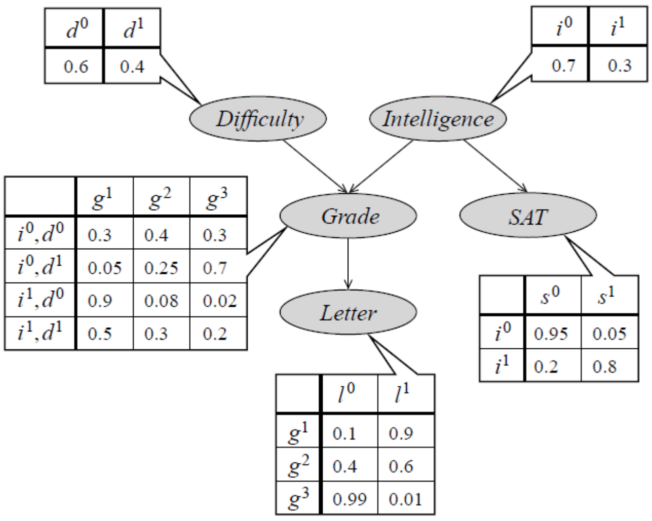
Definition 2 (Markov assumption) *Each variable X_i in G is independent of its non-descendants given its parents [Glymour 2001].*

This assumption allows us to obtain a factorized representation of the joint probability distribution. A Bayesian network(BN) factorizes the joint probability distribution via the following chain rule:

$$P(X_1, X_2, ..., X_n) = \prod_{i=1}^{n} P(X_i|par_G(X_i)) \tag{2.1}$$

Example 1 *An illustrative example of a BN in a discrete domain is shown in figure 2.1 (example taken from [Koller 2009]).*

There are five random variables in this example: the student's intelligence (I), the course difficulty (D), the grade (G), the student's SAT score (S), and the quality of the recommendation letter (L). The domain of random variables are respectively:



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

	g^1	g^2	g^3
i^0,d^0	0.3	0.4	0.3
i^0,d^1	0.05	0.25	0.7
i^1,d^0	0.9	0.08	0.02
i^1,d^1	0.5	0.3	0.2

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^3	0.99	0.01

Figure 2.1: Student bayesian network example [Koller 2009]

$D_I = [low, high]$, $D_D = [easy, hard]$, $D_G = [low, average, high]$, $D_S = [low, high]$, $D_L = [week, strong]$. All of the variables except G are binary-valued, and G is ternary-valued. Therefore, the joint distribution has 48 entries. Figure 2.1 depicts the graphical component of the BN reflecting the dependency relations relative to the modeled domain, as well as the conditional probability distributions of each node. According to the chain rule of BN given in given in equation 2.1, we can denote:

$$P(I, D, G, S, L) = P(I)P(D)P(G|I, D)P(S|I)P(L|G) \tag{2.2}$$

2.1.1 Fundamental Connections

The graphical structure of a BN is bulit upon some fundamential connections:

- Serial: structures of the type X1 $\rightarrow$ X2 $\rightarrow$ X3 .
- Diverging: structures of the type X1 $\leftarrow$ X2 $\rightarrow$ X3 .
- Converging: structures of the type X1 $\rightarrow$ X2 $\leftarrow$ X3 . If there is no connection between X1 and X3, then this structure is known to be a *v-structure*. [Whittaker 2009]

Figures 2.2-2.4 show the possible configurations for three variables and two arc: serial, diverging and converging.

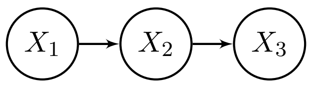


Figure 2.2: Serial connection

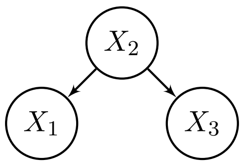


Figure 2.3: Diverging connection

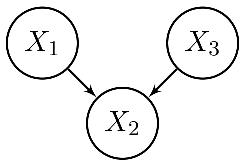


Figure 2.4: Converging connection

Let us analyze the flow of probabilistic information for these three connections to understand their semantics. In converging connections, since there is no arc between X_1 and X_3, it means that they are marginally independent ($X_1 \perp\!\!\!\perp X_3$). However, when conditioning on the X_2 variable, X_1 and X_3 become conditionally dependent. The joint distribution factorizes as $P(X_1, X_2, X_3) = P(X_1)P(X_2)P(X_3|X_1, X_2)$. In serial and diverging connections, the missing arc between the variables corresponds to the conditional independence statement $X_1 \perp\!\!\!\perp X_2|X_3$. For both of them, the joint distribution factorizes as $P(X_1, X_2, X_3) = P(X_1)P(X_3|X_1)P(X_2|X_3) = P(X_3)P(X_1|X_3)P(X_2|X_3)$. Hence, the serial and diverging connections encode the same conditional independence relationships and thus reveal the same factorization for the joint distributions, despite the fact that they encode different graphical structures.

2.1.2 Learning Bayesian Networks

Learning the structure of the BN from the data is considered to be a challenging task. This research space grows exponentially w.r.t. the number of variables in the database. A recursive function was estimated to determine the number of possible DAGs with n variables [Robinson 1977]:

$$f(N) = \sum_{i=1}^{n} (-1)^{i+1} C_n^k 2^{i(n-1)} f(n-1) \tag{2.3}$$

In the literature, there are two big families of methods to learn the structure of a BN from data. Constraints-based algorithms rely on the statistical test to compute the set of conditional independencies between variables and then use this information to construct the BN graph [Shafer 1995][Pearl 2000]. Score-based algorithms formulate the BN learning task as an optimization problem by using a scoring function based on the likelihood of the data given the model. Moreover, a combination between score and constraint-based approaches called hybrid algorithms has been proposed in the literature to extend advantages and reduce the limitation of both classical methods [Tsamardinos 2006] [van Dijk 2003]. Roughly speaking, the hybrid algorithms usually work as follows:

- Restriction step: a constraint algorithm to restrict the search space of graphical solutions.
- Maximization step: among the possible networks that satisfy the constraints from the first step, seeks the one that maximizes a given score function.

Constraint-based methods aspire to build the graph structure This class of methods is aimed at building a graph structure to reflect the dependence and independence relations in the data that match the empirical distribution. However, due to the fact that the number of all conditional independence tests among all nodes and possible relations is exponential, a necessity for some approximations arises [Beretta 2018]. These approaches are usually structured in three steps. In the first step, by making use of independence tests between the variables, a non-oriented graph is constructed. After that, all the independence results are used to derive V-structures (A → B ← C). As discussed previously, V-structure indicates marginal independence between A and C and a conditional dependence between them. In this case, A and C are considered as the causes of

B. After all the V-structures are identified, the rest of the edges are oriented, making sure not to create other v-structures. The most common used algorithms are: PC algorithm [Spirtes 2000], IC algorithm [Pearl 1991] and the Incremental Association Markov Blanket (IAMB) [Tsamardinos 2003].

Score-based algorithms have the intention of maximizing the likelihood L of a set of observed data D, given the model G. This metric is calculated with the following equation:

$$LL(G, D) = \prod_{d \in D} P(d|G) \tag{2.4}$$

Practically, this tends to favor complete graphs (i.e. graphs where each node is linked to the others), which usually result from over-fitting. To overcome this limitation, the likelihood score is almost always combined with a regularization term that penalizes the complexity of the model in favor of sparser solutions. Due to the enormous search space of the valid solutions, the optimization task is often solved with heuristic techniques. Usually, these types of algorithms are composed of two parts: a search algorithm (e.g. Greedy Equivalent Search GES [Chickering 2002]) and an heuristic score (e.g. AIC [Akaike 1974], BIC [Schwarz 1978], BDeu [Buntine 1991]).

2.2 Ontology

There are different definitions that describe the meaning of ontology. The most accepted one states that an ontology is an explicit specification of a conceptualization. The term is derived from philosophy, particularly the branch of metaphysics dealing with the nature of existence. In an artificial intelligence system, what exists is what can be represented in declarative formalism. This set of objects and relationships between them, represent knowledge in a knowledge-based system [Gruber 1995].

An ontology can be viewed as a formal explicit description of concepts in a domain of discourse (classes/concepts), properties of each concept describing various features and attributes of the concept (properties), and restrictions on them. An ontology together with a set of individual instances of classes constitutes a knowledge base [Noy 2001].

Definition 3 (Ontology) *An ontology defines in representational terms:*

- *concepts (classes) $C = C_1, C_2, ..., C_n$ structured in taxonomic (is-a) and partonomic (part-of) hierarchy.*
- *semantic relations between these concepts R_c*
- *properties of concepts*
- *instances I, occurrences of classes and relations*
- *formal axioms*

Ontologies provide the basis for sharing knowledge and as such, they are very useful for a number of reasons:

- Organizing data. Ontologies provide an organization that is flexible and naturally structured in multidimensional ways.
- Improving search. Ontologies are also useful for improving the accuracy of Web searches.
- Data integration. Ontologies can serve as semantic glue between heterogeneous information sources, e.g., sources using different terminologies or languages.

The essential aspect of the ontologies is their inferencing potential based on their logical formal semantics. Languages that are used to express ontologies can be seen as fragments of first-order logic (FOL). Due to the fact that inference in FOL is in general undecidable, fragments of FOL have emerged that are expressive enough to describe the semantics of resources of interest for a certain domain, but limited enough so that inference is decidable and feasible in a reasonable time. [Abiteboul 2011]. One of the decidable fragments of owl that most of these languages are based on is description logic (DL). A description logic knowledge base consists of a Terminological box (TBox), and Knowledge about the objects (ABox).

Example 2 *An illustrative example of a Knowledge Base is shown in 2.5.*

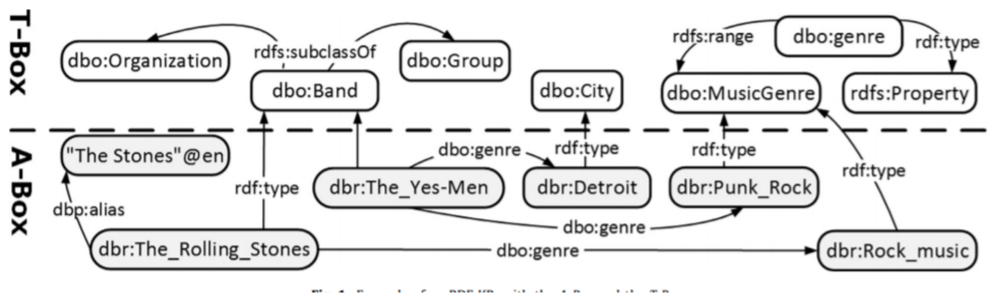


Figure 2.5: Knowledge Base example

2.3 Combining Ontologies and Bayesian Networks

Reasoning in ontology is limited to a logical one. There is a need for handling uncertainty and probabilistic reasoning in ontologies. Bayesian networks have the ability to represent knowledge as well as to do probabilistic reasoning. Both of them are represented by a graph structure. These reasons have intrigued a lot of researchers to propose tailored algorithms allowing to combine the capabilities of ontologies and bayesian networks. In the following subsections, there will be an overview of the previous works that aim to link ontologies and bayesian networks.

2.3.1 Probabilistic Ontologies

Uncertainty is evident in almost all real-world domains since the information is usually incomplete and prone to mistakes and inconsistencies. Therefore, there is high demand to represent uncertainty in ontologies. One way to address this issue is to propose probabilistic extensions to ontologies.

OntoBayes [Yang 2005] describes an ontology-driven model, which integrates Bayesian Networks into the Ontology Web Language (OWL) to preserve the advantages of both. They defined a probabilistic extension of OWL to represent uncertain information in BN structures. These extensions consist of OWL classes to markup probabilities and dependencies in OWL files; “PriorProb”, “CondProb” (both datatype ProbValue), and “FullProbDist”: two disjoint object properties: “hasPrior” or “hasCond”, and a property element to markup dependency between class properties. Bayesian Network in OntoBayes is detailed as a collection of triples, where the predicate is constantly <rdfs:dependsOn> and the subject and object are properties. These extensions enhance knowledge representation in OWL and enable agents to act under uncertainty and complex structured open environments at the same time.

BayesOWL [Zhang 2009] represents the Bayesian network in OWL notation. This framework provides a set of translation rules in order to convert OWL ontologies into a DAG of BN. Each concept class is mapped into a binary variable node, called concept node in BN with states “True” or “False”, denoting whether a randomly selected individual belongs to this concept class or not. If concept class C has a set of subclasses C_i, a subnet is created in the translated BN with a converging connection from each C_i to C. As for logical operators (intersection, union, complement, equivalent, and disjoint), a logic node (L-node) is created in the BN and is connected with the related concept nodes. A BayesOWL prototype was built and it contains a series of APIs, a graphical user interface (GUI), and related documentation. The framework takes 2 input files: an ontology file, and an OWL file with probabilistic information. BayesOWL can be also used for concept mapping, as probabilistic evidential reasoning between two Bayesian networks translated the respective ontologies. As a result, the two BNs are able to exchange beliefs via variables that are similar but not identical.

PR-OWL 1.0 [Da Costa 2006] and PR-OWL 2.0 [Carvalho 2010] were proposed as an extension to the OWL language to define probabilistic ontologies expressed in Multi-Entity Bayesian Networks (MEBN). The basic unit in a MEBN is called M-Fragment(MFrag). These fragments characterize the different pieces of a Bayesian network which, grouped together, constitute an M-Theory. Nodes in an MFrag may be context (must be satisfied for the probability definitions to apply), input (probabilities are defined in other MFrags) or resident (probabilities defined in the MFrag itself). MEBNs extend BNs by integrating first-order logic whose context is the medium. These nodes do not represent random variables but specify the context in which the M-Frag should be placed. (in the example: IsA predicate). PR-OWL is an upper ontology defined in OWL for representing MTheories. PR-OWL defines OWL classes and properties for MEBN terms and restrictions on them. In PR-OWL 1.0, the object property hasPossibleValues links each node with its possible states. Finally, random variables (represented by the class Nodes in PR-OWL 1.0) have unconditional or conditional probability distributions, which are represented by class Probability Distribution. The initial release of PR-OWL had no formal connection between the random variables and properties in the OWL ontology. This limitation was addressed in the second release of PR-OWL. PR-OWL 2.0 formalizes the association between random variables from probabilistic theories with the individuals, classes , and properties from OWL. Moreover, PR-OWL 2.0 allows values of random variables to range over OWL datatypes [Carvalho 2017]. UnBBayes [Matsumoto 2011] is an open-source Java™ application that provides a framework for building probabilistic graphical models and performing plausible reasoning. The project team offers a repository that already includes plug-ins for the Bayesian network (BN), including MEBN and PR-OWL.

HyProb Ontology (hybrid probabilistic ontology) [Mohammed 2016] was proposed to simultaneously handle distributions over discrete and continuous quantities in the ontology. In some scenarios, simply mapping continuous information to discrete states through quantization can cause a great deal of information loss. The semantics of HyProb-Ontology encode uncertainty over properties of instances, and uncertainty over the properties of relations between instances of ontological classes. Properties of classes serve as random variables in this HBN. The semantics requires replacing each continuous variable in the general HBN with a fuzzy discrete variable and extending a link from this fuzzy variable to the continuous variable. They achieved a unified Ground Hybrid Probabilistic Model by conditional Gaussian fuzzification of the distributions of the continuous variables in an ontology.

ByNowLife [Setiawan 2019] integrates BN with OWL by providing an interface for retrieving probabilistic information through SPARQL queries. ByNowLife transforms logical information contained in an ontology into a BN and probabilistic information contained in a BN into an ontology. ByNowLife is useful to integrate different ontologies and perform both logical and probabilistic reasoning through it. This framework consists of three main parts: the Application, the Reasoner, and the Knowledge Base. The application allows querying the Knowledge Base in SPARQL format for logical reasoning and hasProbValueOf special property for probabilistic reasoning. The Reasoner involves two components: the Logical Reasoner, assigned to perform logical reasoning, and the Probabilistic Reasoner, assigned to perform probabilistic reasoning for the Knowledge Base in the form of a Bayesian network, object-oriented BN, and dynamic BN. The knowledge base has 2 subparts: ontology knowledge in OWL/RDF format, and BN knowledge in XDSL format. two forms: ontology knowledge in OWL/RDF format and BN knowledge in XML of Decision Systems Laboratory (XDSL) format. There is a component named the Morpher that conducts data transformation (OWL/RDF) to XDSL (and vice versa). The Morpher also serves to enrich the ontology with the probability values of the nodes (axioms in the ontology) and links in the BN (relations in the ontology).

2.3.2 Construction of Bayesian Networks using ontologies

Using ontologies to learn the structure of Bayesian networks has been explored by several researchers, due to the fact that it is considered to be a good compromise between relaying the learning phase entirely on the data and counting on the domain expert to provide insights. The challenges encountered when constructing BN are:

- identifying the variables that are relevant to the considered domain
- identifying the relationships between the identified variables
- creation of the conditional probability table (CPT) for each variable

In [Helsper 2002],[Bucci 2011], structured methodologies have been presented on to construct BN through ontologies. However, these approaches are designated towards specific ontologies as guideline for the construction of BN, and can not be extended to other ontologies or domains.

Some other work has been done to derive the BN from the ontology, using object properties as a direct translation. The [Fenz 2012] approach is based on the following analogies:

- Concepts → nodes: The classes or individuals that are relevant to the domain are used the populate the BN. If the sub-classes are selected of a specified Node Class, only the subclasses are added to the graph. The same logic is applied to individuals.
- Relations in the ontology → links in the BN: the relations that start and end in the concepts selected in the first step are used to provide links between the BN nodes.
- Axioms → scales and weights: Axioms are inserted in the relevant child/parent node combinations to assign weights, needed for the CPT construction. Moreover, data properties are used to assign numerical values to mutually exclusive, discrete states.
- Instances → findings: Instances are used to derive findings in the BN.

They also provide a CPT construction approach where each node is influenced by (i) the state space of its parent nodes, (ii) the context-specific weight of its parent nodes, and (iii) a distribution function. However, this approach limits the state for parents to 2, except for input nodes. A prototype using the developed method was implemented as a Protege plug-in using the Netica Java API to construct and modify the actual Bayesian network [BnT]. Apart from the boolean node assumption in the CPT construction limitation, another concern is the fact that the structure of the BN is derived from the ontology rather than learned, and not all ontologies manifest such direct dependencies. In addition, the database is not exploited to learn the BN structure.

Similarly, in [Jeon 2007], an approach is described to generate nodes in a BN from of e-health ontology and allows developers to easily establish links among nodes based on a meta-model that represents cause-and-effect relationships among ontologies. They also relate concepts in the ontology to nodes in BN, according to specified rules. If there is a class without subclasses, a BN node with true or false states is constructed. If the node has subclasses, they generate a sub-BN with the nodes from the subclasses.

[Ishak 2011a] shares a similar viewpoint by presuming that the ontology's properties are already causal in order to build an Object-Oriented Bayesian Network (OOBN). They propose to set up a set of mapping rules allowing to generate a prior OOBN structure by morphing an ontology related to the problem to be used as a starting point to the global OOBN building algorithm.

In [Messaoud 2013], they present SemCaDo, an extension of the MyCaDo algorithm for learning Causal Bayesian Networks. MyCaDo (My Causal DiscOvery) is a structure learning algorithm able to select appropriate interventions or experiments. The causal relationships are extracted from the ontology and integrated as constraints (white lists) in the structure learning algorithm to reduce the search task complexity. During the choice of experiment phase, they update the utility function used in the previous work, by incorporating a semantic distance measure from the ontology, aiming to orient the more serendipitous links. After the construction phase, they used the information encoded in the BN to enrich the ontology.

2.4 Discussion

Despite significant achievements in this area, it seemed that using the ontology to tackle the major shortcoming of the building phase of BN had not been explored fully. As mentioned before, some of the previous works in this area use a very specific ontology that can not be extended to other scenarios (e.g. [Helsper 2002], [Bucci 2011]). Some derive the BN from the ontologies rather than learning them using direct translation approach [Fenz 2012]. In most cases, this is not applicable to real-world applications as not all ontologies manifest such direct dependencies. Previous works have been done to use the taxonomy of the ontology to derive an initial OOBN, and this can be used as a starting point during the learning phase [Ishak 2011a]. Another approach was proposed to use casual relations in the ontology as a whitelist of constraints in the learning algorithm and incorporate a semantic distance measure from the ontology, aiming to orient arcs that were still undirected [Messaoud 2013]. However, these approaches have overlooked some important aspects and have not exploited fully the semantic richness of the ontology like concepts properties, non-taxonomic relations, etc. Hence, we were motivated to explore the benefits of utilizing them.

Chapter 3

Preliminaries

In this chapter we describe algorithms and functions that were used in our approach.

3.1 Hill-Climbing algorithm

As mentioned earlier, the number of DAGs grows more than exponentially as the number of variables increases. An exhaustive search in the space of all possible DAGs is usually not feasible in practice. In our work, we adopt a heuristic score-based approach algorithm called Hill-Climbing, which is used in many applications. The Hill Climbing algorithm explores the DAGs space starting from an initial network (can be empty if there is no prior information), and then adds, removes, or reverses one edge at a time until the maximum gain in terms of the score is reached (see details in algorithm 1). Hill-Climbing is a greedy algorithm, so it does not guarantee global optimum.

Algorithm 1 Hill Climbing algorithm [Tattar 2018]

```
1: Start from a graph $G$
2: Compute $score_G = score(G)$
3: Set $maxscore = score_G$
4: Perform the following operations as long as $maxscore$ increases:
5: for each possible arc addition, deletion or reversal that results in a DAG do
6:     Compute the score $score(G*)$ of the modified structure $G*$ and set $score_{G*} = score(G*)$
7:     if $score_{G*} > score_G$ then
8:         Update $maxscore = score_G$
9: Return the DAG G.
```

3.2 Score functions

Scored based structure learning algorithms use a scoring function that they try to maximize. Given a training data set $D = u^1, ..., u^n$ of instances of variables U_n, find a DAG $G*$ such that:

$$G* = arg \max_{G \in G_n} g(G : D) \tag{3.1}$$

where $g(G : D)$ is the scoring function measuring the degree of fitness of any candidate DAG G to the data set, and G_n is the family of all the DAGs defined on U_n.

There are many scoring functions used for learning bayesian networks. We used a scoring function based on information theory, more specifically the AIC score. These functions are based on codification and information theory concepts. They seek to reduce the number of elements that are necessary to represent a message. Frequent messages will therefore have shorter codes whereas larger codes will be assigned to the less frequent messages. The minimum description length principle (MDL) selects the coding that requires minimum length to represent the messages. In the case of the Bayesian Networks, the description length includes the length required to represent the network plus the length necessary to represent the data given the network ([Bouckaert 1995]; [Suzuki 1993]). Representing a network means storing its probability values, and this requires a length which is proportional to the number of free parameters of the factorized joint probability distribution [De Campos 2006].This number, called network complexity and denoted as $C(G)$, is:

$$C(G) = \sum_{i=1}^{n}(r_i - 1)q_i \tag{3.2}$$

Usually the proportionality factor is $\frac{1}{2}log(N)$ [Rissanen 1986], therefore the length of the network can be defined as:

$$\frac{1}{2}C(G)log(N) \tag{3.3}$$

As for the description of the data given the model, its length turns out to be the negative of the likelihood function of the data with respect to the network (log-likelihood). The log-likelihood is expressed [Bouckaert 1995]:

$$LL_D(G) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk}log(\frac{N_{ijk}}{N_{ij}}) \tag{3.4}$$

Thus, the MDL scoring function can be defined to deal as a maximization problem as:

$$g_{MDL}(G:D) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk}log(\frac{N_{ijk}}{N_{ij}}) - \frac{1}{2}C(G)log(N) \tag{3.5}$$

The quality of the BN can be measured in terms of measures of information theory. The idea is to select the network structure that best fits the data, penalized by the number of parameters which are necessary to specify the joint distribution:

$$g_{MDL}(G:D) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk}log(\frac{N_{ijk}}{N_{ij}}) - \frac{1}{2}C(G)f(N) \tag{3.6}$$

The $f(N)$ stands for the penalization function.

- $f(N) = 1 \rightarrow$ based on the Akaike information criterion (AIC) [Akaike 1974]
- $f(N) = \frac{1}{2}log(N) \rightarrow$ based on the Schwarz information criterion (BIC) [Schwarz 1978]

3.3 Dijkstra's Shortest Path algorithm

The shortest path problem seeks a path between vertices in a graph such that the total sum of the weights of the edges is minimum. There are many algorithms in the literature that aim to discover such paths. The most distinguished one is Dijkstra's Shortest Path algorithm [Dijkstra 1959]. Many variants of this algorithm exist, but the most common one fixes a single node as the "source" node and finds the shortest paths from the source to all other nodes in the graph [Mehlhorn 2008]. Dijkstra's approach is described in algorithm 2.

Algorithm 2 Dijkstra's Shortest Path Single source

```
Input: Weighted graph $G = (V, E)$, Source vertex $V_s \in V$
Output: Graph $G'$ with all the distances from $V_s$ to all nodes
 1: Create vertex set $Q$ {Vertices not yet processed}
 2: for each $v$ in $V$ do
 3:     $dist[v] \leftarrow \infty$
 4:     $prev[v] \leftarrow undefined$
 5:     Add $v$ to $Q$
 6: $dist[V_s] \leftarrow 0$
 7: while $Q$ not $\emptyset$ do
 8:     $u \leftarrow v$ in $Q$ with min $dist[u]$
 9:     remove $u$ from $Q$
10:     for each neighbour $v$ in $Q$ of $u$ do
11:         $alt \leftarrow dist[u] + length(u, v)$
12:         if $alt < dist[v]$ then
13:             $dist[v] \leftarrow alt$
14: Return $G'$ containing $dist[]$
```

Dijkstra's Shortest Path algorithm starts by first initializing all the distances from the source vertex to all the other vertices with infinity and 0 to itself. Then it finds the lowest-weight relationship from the start node to directly connected nodes. It keeps track of those weights and moves to the "closest" node. On the other steps, it performs the same calculation, but as a cumulative total from the start node. The algorithm continues to do this, always choosing the lowest weighted cumulative path to advance along until it reaches the destination node.

The algorithm to calculate the shortest path between all vertices works the same way and variants for optimization have been introduced. For instance, the distances calculated so far can be kept so that they can be reused when calculating the shortest path to an unseen node, and the nodes can be run in parallel. Let's take a look at the graph G in figure 3.1, and calculate the shortest path from A to the other nodes. In table 3.1 you can see the results after each iteration, with updates shaded. The distances calculated so far are memorized to be used in the next steps, for the other nodes.

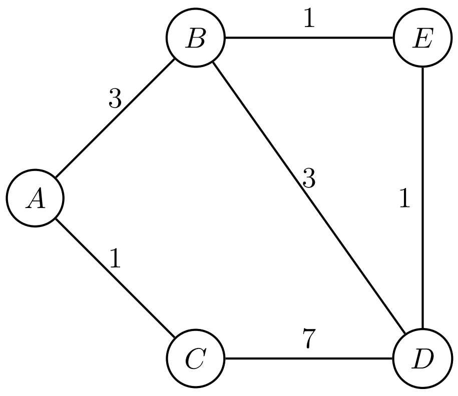


Figure 3.1: Graph G example

Dijkstra shortest path steps for node A							
All nodes start with an ∞ distance and then the start node (A) is set to a 0 distance			1st from A	2nd from A to C to Next	3rd from A To B to Next	4th from A To E to Next	5th from A To D to Next
A	∞	0	0	0	0	0	0
B	∞	∞	3	3	3	3	3
C	∞	∞	1	1	1	1	1
D	∞	∞	∞	8	6	5	5
E	∞	∞	∞	∞	4	4	4

Table 3.1: The steps to calculate the shortest path from node A to all other nodes, with updates shaded

Chapter 4

Proposed approach

In this chapter, we describe the approach we implemented to derive insights from the ontology that can serve as prior information in structure learning algorithms for BN.

4.1 Problem formalization

Learning the structure and the parameters of a BN from the data, as previously mentioned, is a challenging task. Integrating prior knowledge from domain experts to facilitate this task proves to be quite an aid. Many approaches have been proposed to utilize knowledge from the ontologies to guide BN construction. The backbone of such solutions lies in the following direct translations for selected nodes: Concepts → Nodes, Relations in the ontology → links in the BN. The real challenge is what one would do when the ontology does not transcribe such direct dependencies.

To tackle this issue, the approach we develop here relies on the identification of semantic relations in the ontology that encapsulate some dependency nuance between concepts and use these relations to calculate a "closeness" metric between them. We believe that exploiting this information is a good starting point to learn the BN, in scenarios that we don't have prior knowledge from a domain expert. Figure 4.1 outlines the pipeline of the proposed approach, which is explained in detail in the following sections.

- Firstly, using a list of selected relations, we extract a list of triples (concept-relation-concept) from the ontology.
- Secondly, we build a subgraph containing only the extracted triples and calculate a "closeness" score between concepts, using the weighted Dijkstra algorithm.
- Thirdly, taking into account this metric, we derive a list of constraints and an initial BN structure.
- Lastly, learning the BN structure using the derived prior information as input to guide it.

4.2 Dependency extraction

Having two concepts $\langle cp_i, cp_j \rangle \in Cp^2$ in the ontology related by a semantic relation means that there is at least one property of one of them that affects at least one property of the other. Roughly speaking, the definition of one of the concepts depends on the existence of the other. BN allows representing conditional independencies that hold in a probability distribution with a directed acyclic graph (DAG). The domain ontology provided does not

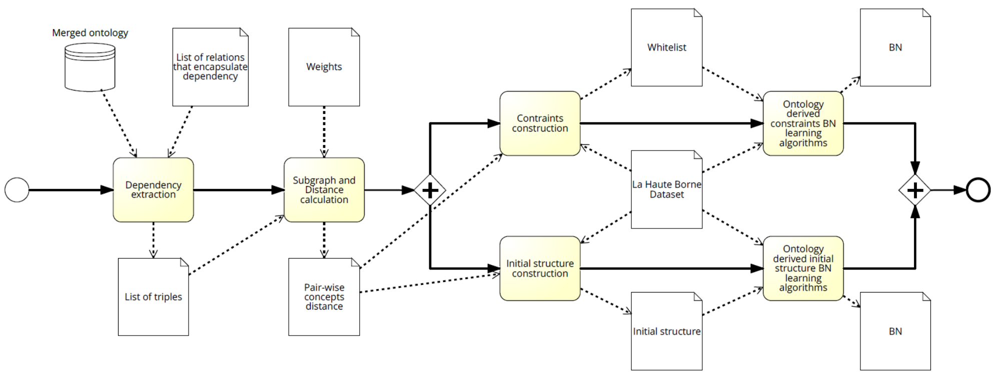

Merged ontology
List of relations that encapsulate dependency
Weights
Dependency extraction
Subgraph and Distance calculation
List of triples
Pair-wise concepts distance
Contraints construction
Initial structure construction
Whitelist
La Haute Borne Dataset
Initial structure
Ontology derived constraints BN learning algorithms
Ontology derived initial structure BN learning algorithms
BN
BN

Figure 4.1: Pipeline of the proposed approach

provide causal/dependence relations between concepts, to be traduced by a directed link between the corresponding BN nodes. Therefore, it was needed to examine the information presented, and select certain relations that fit with the semantic of the dependency in the BN context.

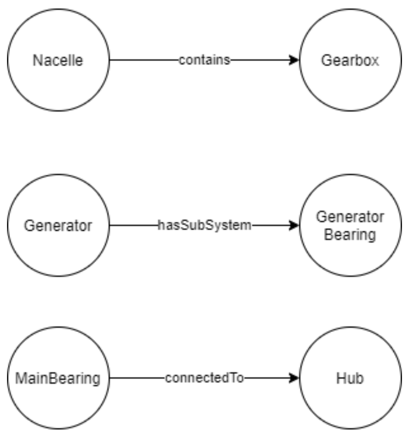


Figure 4.2: Selected relations in the ontology to extract

In figure 4.2 we can see some examples of the selected relations: *contains*, *hasSubSystem* and *connectedTo*. According to our assumption, if a structural component is linked via the selected relations to another structural component, it means that at least one property of one of them affects at least one property of the other. For instance, if *MainBearing* is connected to *Hub*, it is likely that if the temperature of *MainBearing* is high, the temperature of *Hub* will get higher as well, or vice-versa. The same idea applies to the other selected relations.

In this step, SPARQL queries were used to extract the concepts from the ontology related to the selected relations, as described in algorithm 3. Given the domain ontology and a set of selected relations $R = contains, hasSubSystem, connectedTo$, we run a SPARQL query to extract the triples $T = < ci, r, cj >$ with the concepts that are related via these relations.

Algorithm 3 Dependency triples extraction

```
Input: Ontology $O$, Set of selected relations $R$
Output: Set $T$ of extracted triples
1: for each $r$ in $R$ do
2:     Perform SPARQL query:
         SELECT ?c1 ?c2 WHERE {
            ?c1  rdfs:subClassOf ?s .
            ?s a owl:Restriction .
              ?s owl:onProperty  r
              ?s owl:someValuesFrom | owl:allValuesFrom ?c2 .
         }
3:     Add $< c1, r, c2 >$ to $T$
4: return $T$
```

4.3 Subgraph and Distance calculation

In the previous step, a set of concepts and their associated relations were extracted from the ontology. In this step, we want to calculate the path between concepts, to derive how closely connected they are. The term path is defined as a set of edges which connects two concepts. Semantic similarity measures in ontology use the semantic information of the concept such as depth, Information Content (IC), neighborhood, synset, path length, etc., between the two concepts to be matched, to compute the similarity [Sathiya 2019]. Path-based similarity measures in literature like: Rada et al. [Rada 1989] Hirst & Onge et al. [Hirst 1998] , Sussna et al. [Sussna 1993], Wu & Palmer [Wu 1994] use taxonomic information from the ontology to calculate the similarity. If two concepts are linked by parent-child relation then the edge is a *IS-A* relation type. The parent and child concept is called as *hypernym* (a.k.a super concept) and *hyponym* (a.k.a sub-concept) respectively. A concept is said to be at depth k, if it is k edges away from the root node *(owl:Thing)* of the ontology. Let us consider the following example in figure 4.3.

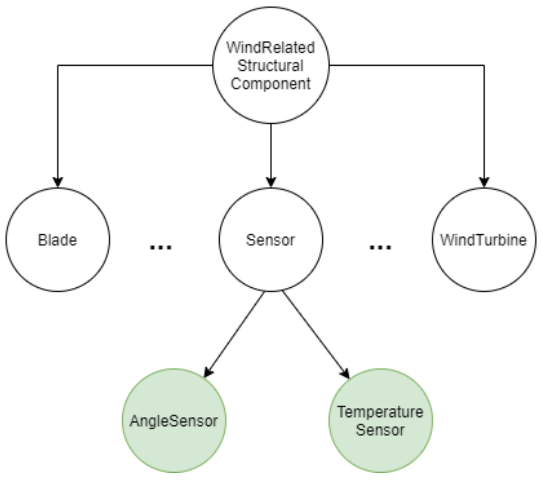


Figure 4.3: Example of semantic similarity measure calculation

Using the subsumption and depth to calculate the similarity, we would get that *AngleSensor* and *TemperatureSensor* have a high similarity score. However, using this kind of information does not fit our purposes, because it does not fit with the semantic of the dependency in the BN context. The type of information we are looking for could be: "*The temperature of Gearbox depends on the temperature of Nacelle, due to the fact Gearbox is contained in Nacelle*".

For this reason, we decided to calculate the degrees of separation between them via the selected relations, using weighted Dijkstra's Shortest Path algorithm [Dijkstra 1959]. In algorithm 3 we showed how we extracted the triples that contain dependency information. Thenceforth, we built a subgraph using only the concepts and relations in the triples. Weights were given to the relations considering their importance and influence. The relations *isContainedIn*|*contains* transcribe a stronger dependecy between the concepts, as a certain component is contained in another one, followed by *subSystemOf*|*hasSubSystem*, and *connectedTo*. The weights defined: *contains* → 1, *has subSystem* → 1.2 *connected To* → 1.4, as shown in figure 4.4.

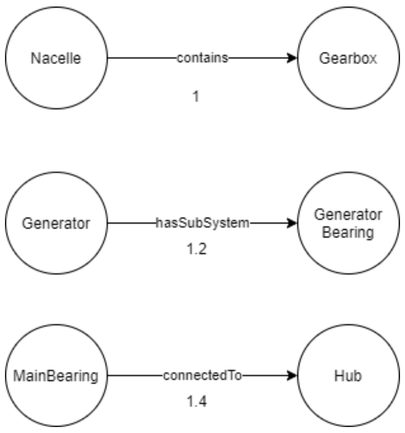


Figure 4.4: Weighted relations

We ran Dijkstra's Shortest Path algorithm for all the concepts to calculate the distance between them. Hence, we can check how "close" these concepts are with each other.

$$Sim_{score} = \frac{1}{distance}$$

Algorithm 4 Similarity score calculation

Input: Subgraph G_s
Output: List L of $< Concept1, Concept2, Sim_{score} >$

```
1: $G'$ {containing distances} ← Call Dijkstra's allShortestPaths()
2: YIELD $L =< Concept1, Concept2, distance >$
3: for each $< Concept1, Concept2, distance >$ in $L$ do
4:     $Sim_{score} = \frac{1}{distance}$
5: return  $L$ of $< Concept1, Concept2, Sim_{score} >$
```

Following the examples presented earlier, now taking into consideration the weights as in figure 4.4, *Nacelle-Gearbox* have a similarity score of 1, *Generator-GeneratorBearing* → 0.83, and *MainBearing-Hub* → 0.71. According to our assumption, at least one property of one of *Nacelle* affects at least one property of *Gearbox*, or vice-versa. Same idea for the other concepts.

4.4 Construction of constraints and initial structure of BN

In the previous steps, we mentioned that, if a structural component is related via one of the relations that encompass certain dependencies to another structural component, it means that at least one property of one of them affects at least one property of the other. Suppose that the property p_k of concept c_i affects the property $p_{k'}$ of concept c_j, then the property that represents p_k in the class c_i, would be considered as input for the property that represents $p_{k'}$ in the class c_j. However, we can not state the orientation. In what follows, we assume that all semantic relations have the same causal orientation, deriving 2 scenarios. Thus, $\forall \{c_i, c_j\} \in C^2$ related by a semantic relation, where c_i is the domain and c_j is the range, p_k of c_i is considered as the cause of p_k of c_j and this latter is the effect, and vise-versa. One might wonder, which properties should we relate to each other? We restricted our solution by linking the properties that possess the same semantics, i.e *Rotor Bearing temperature* with *Hub temperature*. Using the set of similarity scores calculated in the previous step, we derived potential links between variables of the BN, as described in algorithm 5.

Algorithm 5 O_BN linking

```
Input: Ontology $O$, $L =$ of $< c_i, c_j, Sim_{score} >$, threshhold $\tau$
Output: Set $M$ of extracted mappings
 1: Create set $C$ of distinct concepts in $L$
 2: for each $c$ in $C$ do
 3:     $P[c] \leftarrow$ properties of $c$
 4: for each l in $L$ do
 5:     if $Sim_{score} > \tau$ then
 6:         for each $p_k$ in $P[c_i]$ do
 7:             for each $p'_k$ in $P[c_j]$ do
 8:                 if $p_k$ and $p'_k$ have the same semantic then
 9:                     ADD $< p_k$ , $p'_k$ , $Sim_{score} >$ to $M$
10: return $M$
```

4.5 Learning the BN using the knowledge derived from the ontology

The last step of the approach consists of using the information derived from the ontology as prior knowledge for the score-based structure learning algorithms. This information is exploited in 2 different ways. Firstly, the links inferred from the ontology are fixed during the learning process, i.e., the algorithm cannot modify or delete them. The constraints are given in the form of a whitelist (matrix containing the different edges). Whitelisting an arc in one direction (i.e. Yt $\rightarrow$ Gost is in the whitelist) forces that arc to be in the network in that particular direction, and adds the arc in the opposite direction to the blacklist. Secondly, we do not force these restrictions. The information will be used to simply build an initial structure as a starting point for the score-based algorithm.

Chapter 5

Ontology and dataset

This chapter introduces the ontologies and datasets used in our approach during the experimentation and evaluation phase. In the first section, we will present the ontologies utilized in our approach. In the second section, we show the dataset used to learn the Bayesian networks and evaluate the results.

5.1 Ontologies

Three main ontologies were used: *WeatherOntology-1.0.ttl*, *SensorOntology-1.0.ttl* and *WindTurbineOntology-1.0*. Roughly speaking, the wind turbine ontology defines the systems, connections between systems, and connection Points at which systems may be connected. They are usually defined in terms of their properties, which are qualifiable, quantifiable, observable, or operable qualities of the feature of interest. The properties can be evaluated either directly by giving it a constant value or can be qualified evaluations that can evolve in space or time, so the evaluation validity can have a temporal or spatial context. The sensor Ontology mainly defines different types of sensors and the properties they observe. The weather ontology details different properties like temperature, speed, etc.

Ontology	**Classes**	**Object Properties**	**Data Properties**
WindTurbineOntology	159	45	35
SensorOntology	718	1994	85
WeatherOntology	718	2018	85

Table 5.1: Ontlogy details

These ontologies were merged together and the high-level classes and corresponding properties of the merged ontology are shown in figure 5.1. In this schema, we can get information how the wind related structural are conncetd to each other, their main properties, and how these properties are measured. For instance, *HighSpeedShaft*, *Brake*, *Gearbox* and *LowSpeedShaft* are contained in *Nacelle*. *Nacelle* has *Temperature* property that is measured by a *TemperatureSensor*.

5.2 La Haute Borne Datasets

Engie provides open yearly data regarding different attributes of wind related data and structural components. The data can be accessed in: `https://opendata-renewables.engie.com/`. Each year, around 210000 records are stored for the attributes: Pitch angle

(*BA*), Converter torque (*Cm*), Power factor (*Cosphi*), Generator converter speed (*DCs*), Temperature of the first generator bearing (*Db1t*), Temperature of the second generator bearing (*Db2t*), Generator speed (*Ds*), Generator stator temperature (*Dst*), Temperature of the first gearbox bearing (*Gb1t*, Temperature of the second gearbox bearing *Gb2t*, Gearbox inlet temperature (*Git*), Gearbox oil sump temperature (*Gost*), Nacelle angle (*Na_ c*), Grid frequency (*Nf*), Grid voltage (*Nu*), Outdoor temperature (*Ot*), Active power (*P*), Pitch angle setpoint (*Pas*), Reactive power (*Q*), Rotor bearing temperature (*Rbt*), Torque (*Rm*), Rotor speed (*Rs*), Hub temperature (*Rt*), Apparent power (*S*), Vane position (*Va*), Position of the first wind vane on the nacelle (*Va1*), Position of the second wind vane on the nacelle (*Va2*), Absolute wind direction (*Wa*), Absolute wind direction corrected (*Wa_ c*), Average wind speed *Ws*, Wind speed measured on the first anemometer on the nacelle (*Ws1*), Wind speed measured on the second anemometer on the nacelle (*Ws2*), Nacelle angle (*Ya*), Nacelle temperature (*Yt*). The dataset contains metrics for these variables at a 10 minute granularity, throughout the whole year, and average, minimum, maximum ans standard deviation is stored. The averages of these attributes during the 10 minute time frame were used in our analysis.

The attributes are not present in all records. We will describe our dataset in terms of coverage of attributes, which means the percentage of the records where the attribute is present.

$$Coverage\ of\ attr_i\ in\ D = \frac{\sum r_i\ in\ D \mid \exists\ attr_i\ in\ r_i}{|D|} \tag{5.1}$$

In table 5.2, we can see the coverage of each attribute in the *La Haute Borne 2017* dataset, as well as the mean, minimum as maximum. Attributes with coverage less than 40% (Va_avg, Wa_c_avg, Na_c_avg, and Pas_avg) were excluded from the dataset.

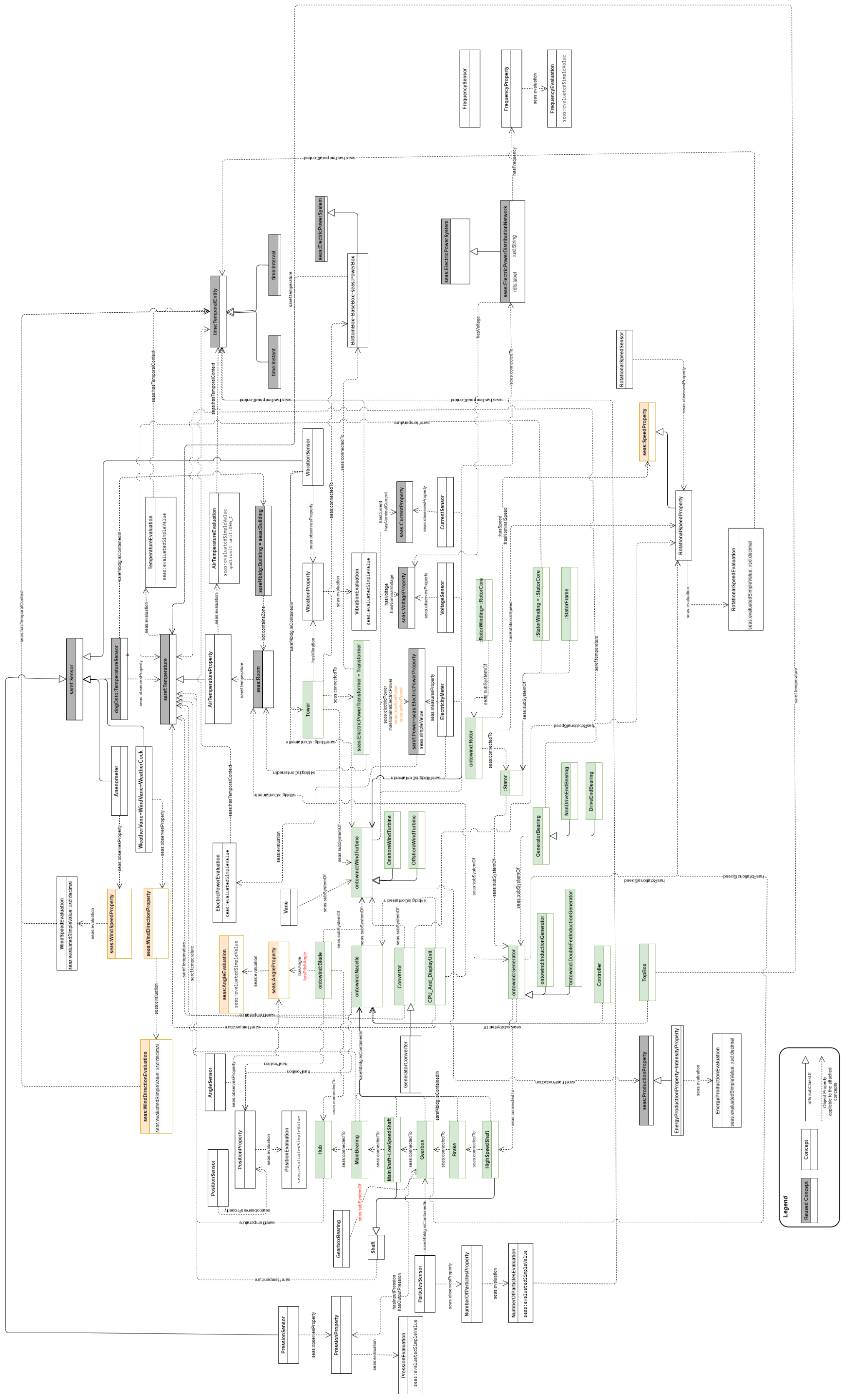


Figure 5.1: Merged Ontologies

La Haute Borne Data 2017					
	Attribute	**Coverage**	**Mean**	**Min**	**Max**
1	Ba_avg	99.1%	12.018	-1.53	132.48
2	Rt_avg	99.1%	19.49	1.78	215
3	DCs_avg	99.1%	1088.096	-574.89	1807.6
4	Cm_avg	99.1%	2252.35	-4375.89	10965.4
5	P_avg	99.2%	360.39	-17.39	2050.78
6	Q_avg	97.7%	27.02	-94.23	266.7
7	S_avg	99.1%	363.89	0	2060.65
8	Cosphi_avg	99.1%	0.99	0.06	1
9	Ds_avg	98.6%	1094.49	-0.18	1803.93
10	Db1t_avg	99.1%	39.27	3.8	81.88
11	Db2t_avg	99.1%	35.78	0.38	276.41
12	Dst_avg	99.1%	55.47	-0.2	101.15
13	Gb1t_avg	99.1%	60.08	8.28	86.21
14	Gb2t_avg	99.1%	60.73	10.2	87.15
15	Git_avg	99.1%	50.79	8.65	72.9
16	Ya_avg	99.2%	189.48	0	360
17	Yt_avg	99.1%	24.59	-1.15	56.09
18	Ws1_avg	98.7%	5.47	0	24.89
19	Ws2_avg	98.7%	5.44	-0.82	23.65
20	Ws_avg	99.2%	5.43	0	24.27
21	Wa_avg	99.2%	187.47	0	360
22	Va1_avg	63.7%	0.64	-179.92	179.94
23	Va2_avg	63.7%	0.64	-179.92	179.94
24	Va_avg	35.4%	2.21	-179.88	179.92
25	Ot_avg	99.1%	12.24	-7.97	70.98
26	Nf_avg	99.1%	49.98	0	50.09
27	Nu_avg	99.1%	700.98	0	729.61
28	Rs_avg	97.9%	10.51	0	17.22
29	Rbt_avg	99.1%	27.46	-0.21	43.07
30	Rm_avg	97.6%	2141.58	-888.08	10875.4
31	Wa_c_avg	15.6%	173.00	0.01	360
32	Na_c_avg	15.6%	173.00	0.01	360
32	Pas_avg	0%	-	-	-

Table 5.2: Summary of the attributes

Chapter 6

Experiments and results

In this chapter, we discuss the results and the evaluation of our approach. Firstly, we explain the experimental setup and our experiments' objective. Secondly, we present the conducted experiments and our evaluation methodology. Finally, we interpret the results and give some preliminary conclusions.

6.1 Experimental setup

In this section, we demonstrate the experimental setup to obtain the results of our implemented approach. In section 6.2 we detail the steps of the approach applied in our use case. In section 6.3 we describe the experiments performed to compare and evaluate our learned bayesian networks. The dataset used contains 800000 records and contains data from the year 2013-2016. The dataset is described in chapter 5. Figure 6.1 illustrates the experimental steps taken.

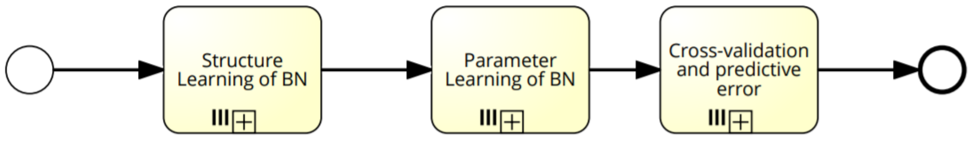


Figure 6.1: Experimental pipeline

Firstly, we run the hill-climbing (HC) algorithm with constraints derived from the ontology. As mentioned earlier we have 12 versions:

- Hill climbing with constraints derived with $\tau = 0.5$, scerario 1 (HC_105)
- Hill climbing with constraints derived with $\tau = 0.5$, scerario 2 (HC_205)
- Hill climbing with constraints derived with $\tau = 0.4$, scerario 1 (HC_104)
- Hill climbing with constraints derived with $\tau = 0.4$, scerario 2 (HC_204)
- Hill climbing with constraints derived with $\tau = 0.2$, scerario 1 (HC_102)
- Hill climbing with constraints derived with $\tau = 0.2$, scerario 2 (HC_202)
- Hill climbing with initial structure derived with $\tau = 0.5$, scerario 1 (Init_HC_105)
- Hill climbing with initial structure derived with $\tau = 0.5$, scerario 2 (Init_HC_205)

- Hill climbing with initial structure derived with $\tau = 0.4$, scerario 1 (Init_HC_104)
- Hill climbing with initial structure derived with $\tau = 0.4$, scerario 2 (Init_HC_204)
- Hill climbing with initial structure derived with $\tau = 0.2$, scerario 1 (Init_HC_102)
- Hill climbing with initial structure derived with $\tau = 0.2$, scerario 2 (Init_HC_202)

Also, the HC algorithm without any prior information was run for comparison purposes. The 12 algorithms were tested in different dataset sizes: 1000,5000,8000,10000,50000,80000,100000,500000, and 800000. Each of these experiments was run 10 times and an average of the results was calculated.

Secondly, after the BN structure is learned, we fitted the parameters of the local distributions, which take the form of conditional probability tables. The experiments were also run for the 12 algorithms, for all dataset sizes mentioned before.

Lastly, we evaluate the quality of the BN structure learned, by performing cross-validation. Cross-validation is a standard way to obtain unbiased estimates of a model's goodness of fit. The data were randomly partitioned into 10 subsets. Each subset was used in turn to validate the model fitted on the remaining 9 subsets. We targeted all 28 variables in our model and calculated the expected prediction loss. The 10-fold cross-validation is performed 10 times for each learning strategy, according to the golden standard originally introduced in [Friedman 2001].

Experiment's Objective: The objective behind the mentioned experimental setup is to explore the performance of the hill-climbing algorithms with prior information derived from ontology, comparing them with each other as well as with the same algorithm without any prior information. The comparison is done in terms of performance and time.

6.2 Implementation in the use case

The first step of our approach is to extract the list of triples with the selected relations: *isContainedIn*|*contains*, *subSystemOf*|*hasSubSystem*, and *connectedTo*. Weights were defined for each relation according to their significance: *isContainedIn*|*contains* → 1, *subSystemOf*|*hasSubSystem* → 1.2 *connected To* → 1.4. Secondly, using this information, we built a subgraph using only the concepts and relations in the triples, while assigning the given weights to the relations. The subgraph is presented in figure 6.2.

The *Similarity score calculation* algorithm (4) was run to investigate how "close" the selected concepts are with each other. For instance, *Gearbox* and *Nacelle* have a similarity score of 1, because *Gearbox* is contained in the *Nacelle*, *Generator* and *Rotor* have a similarity score of 0.83 becasuse *Rotor* is subsystem of *Generator*, *MainBearing* and *Hub* have a similarity score of 0.71 because *MainBearing* is connected to *Hub* etc. After calculating how close the structural components are connected with each other, the next step is to map their properties that are semanticaly the same, as described in algorithm 5. Four threshholds were used as cut-off for the similarity score to restrict the results: $\tau = 0.5, \tau = 0.4, \tau = 0.25, \tau = 0.2$. This way, we were able to construct a set of links between the variables (up to 3 degrees of separation between their nodes). The set of links derived can be found in table A.1. For each τ used, the last paired derived is highlighted in

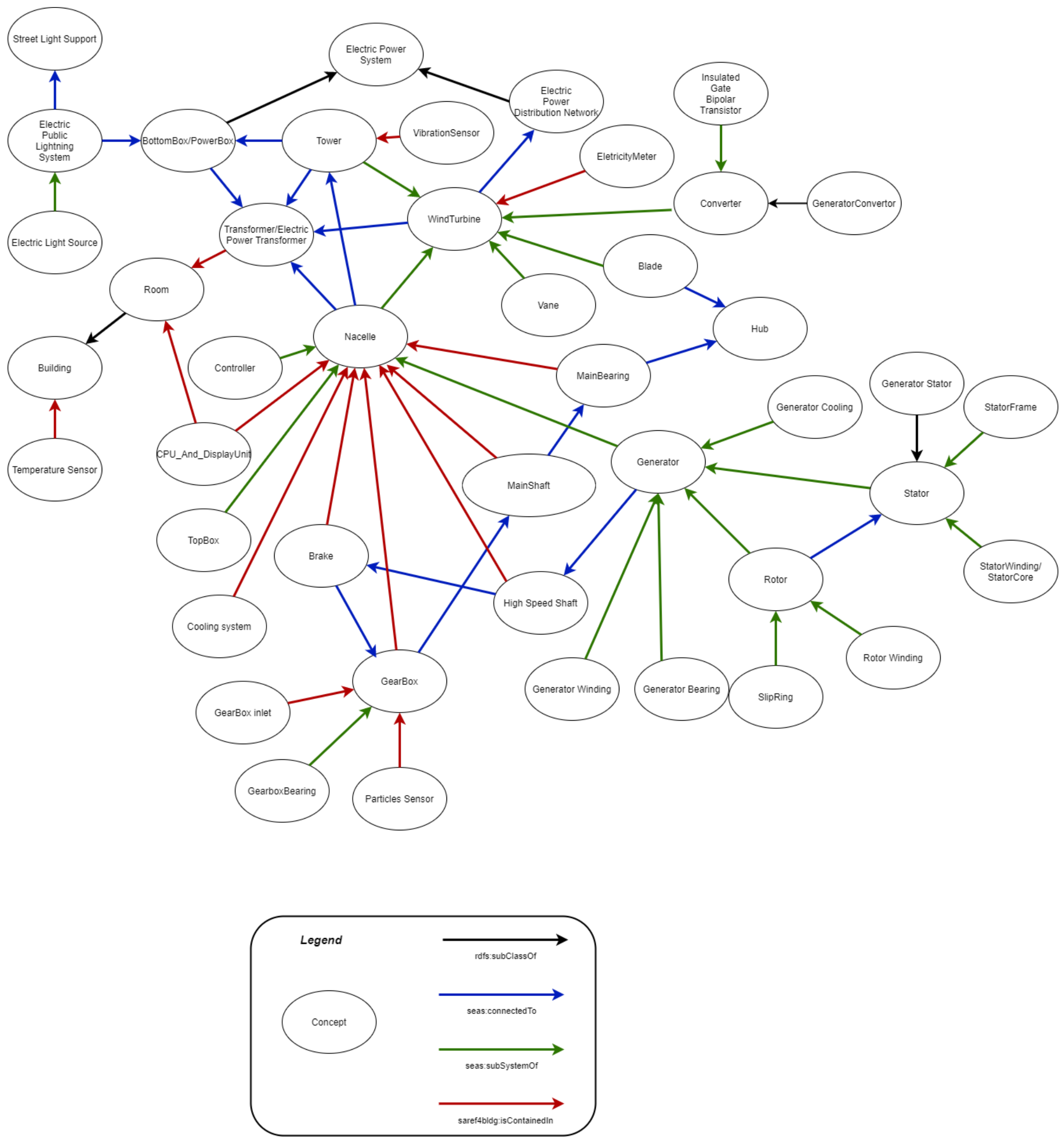


Figure 6.2: Subgraph constructed with the extracted triples

the table. For $\tau = 0.5$, we can derive 10 constraints, $\tau = 0.4 \rightarrow 20$ constraints, $\tau = 0.25 \rightarrow 24$ constraints and $\tau = 0.2 \rightarrow 27$ constrains.

We mentioned earlier that we can not decide the orientation of the links using the dependencies derived from the ontology. Therefore, we assumed that all semantic relations have the same causal orientation, deriving 2 scenarios. Thus, $\forall \{c_i, c_j\} \in C^2$ related by a semantic relation, where c_i is the domain and c_j is the range, $p_k of c_i$ is considered as the cause of $p_k of c_j$ and this latter is the effect, and vise-versa. The oriented links are illustrated in figures 6.3 and 6.4. In both figures, the arcs are color-coded to identify the links derived with a different threshold for the similarity score. They were used to construct a whitelist of constraints for the socred-based learning algorithm, as well as an initial structure that can be used as a starting point.

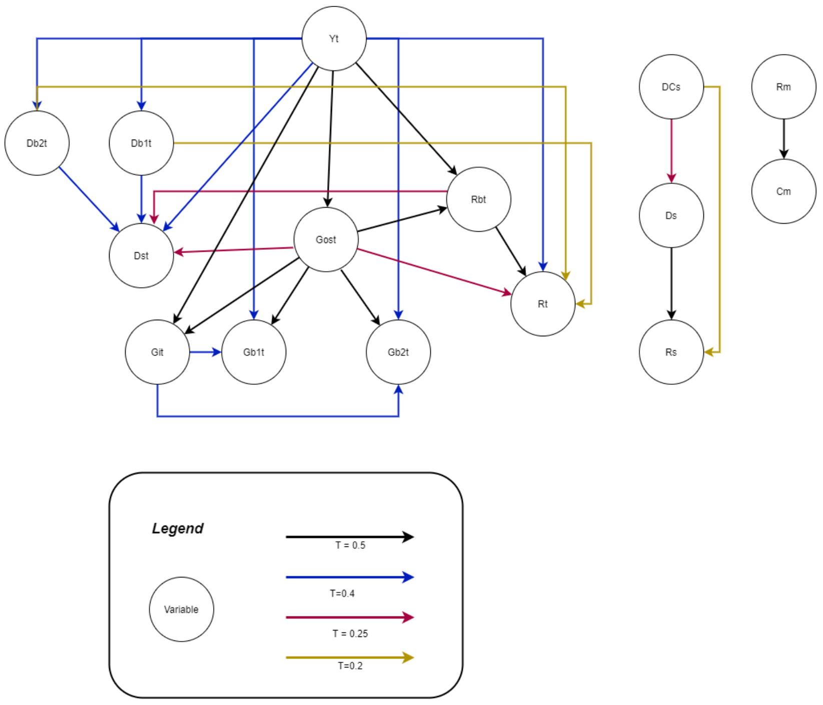


Figure 6.3: Oriented links scenario 1

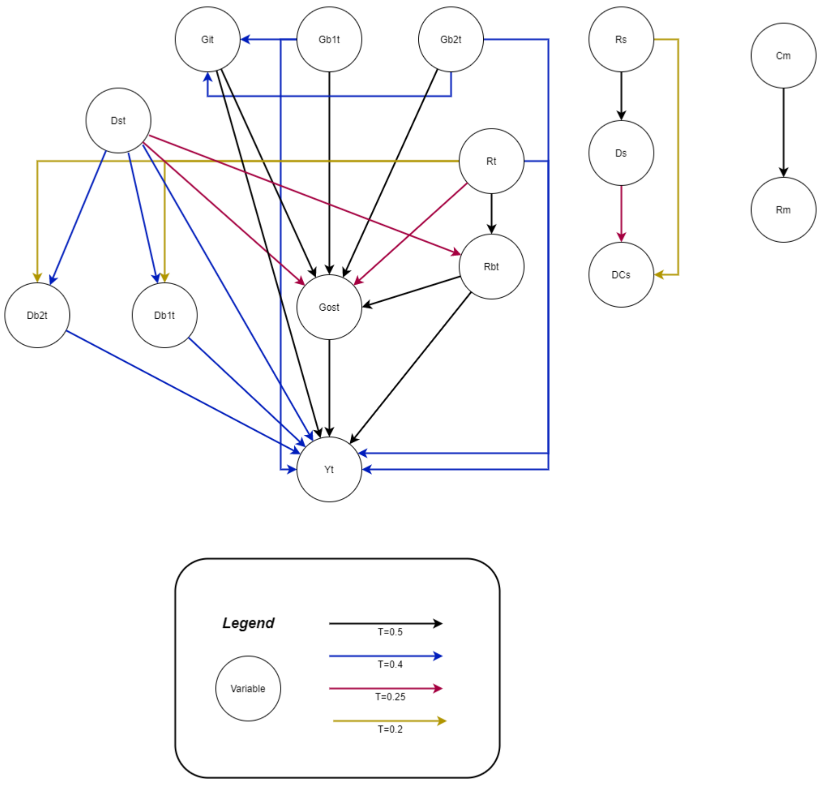


Figure 6.4: Oriented links scenario 2

6.3 Experiments and evaluation of results

In the following subsections, we will present our experiments and evaluate the results for our algorithms with constraints and initial structure.

6.3.1 Experiments and evaluation of HC with constraints

In this subsection, we will detail the experiments with the algorithms that have constraints as input in the hill-climbing algorithm. The structure learning algorithms were tested with different dataset sizes and each of them was run 10 times to get the average results. The quality of results was evaluated via 10-fold cross-validation, each performed 10 times for each learning strategy while measuring the expected loss for all the variables.

Firstly, we wanted to check the overall classification error of our approaches. As mentioned earlier, cross-validation was used to obtain unbiased estimates of a model's goodness of fit. The 10-fold cross-validation is performed 10 times for each approach. In figure 6.5 you can see the average classification errors for hill-climbing algorithms with constraints from the ontology, run on the whole dataset. From the plot, we can see that the HC_102 BN has the best performance as the classification error is the lowest, and HC_204 is the worse. However, the magnitude of the differences is so small as not to be practically significant.

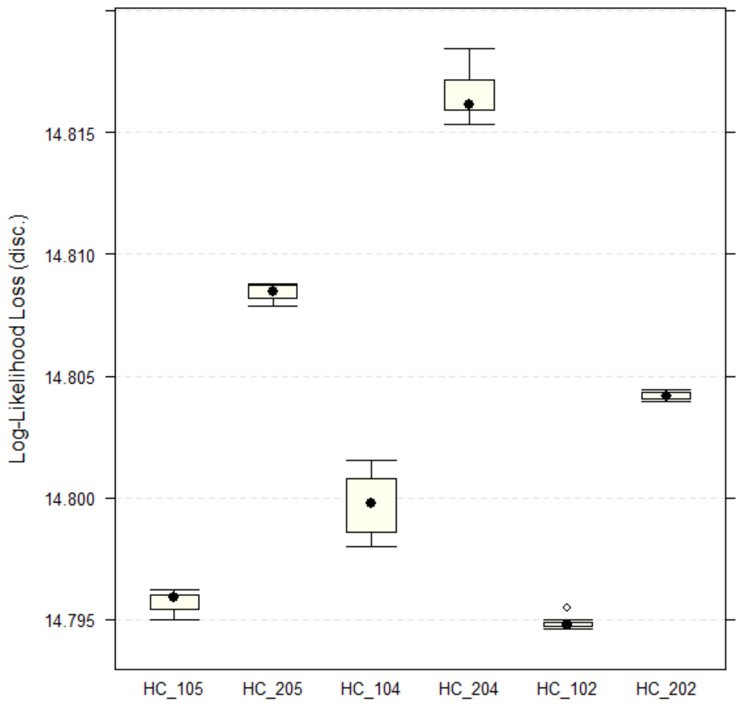


Figure 6.5: Classification error of the networks with constraints

We are also interested in the prediction of each variable in the network. The 10-fold cross-validation performed 10 times was conducted targeting each variable to measure the prediction loss for each algorithm. The whole dataset was used for this experiment as well. In figure 6.6 you can see the results of our HC algorithms with constraints, as well as HC without any prior information. As we can see, some of our approaches have a significant improvement in prediction for some variables compared to the HC without prior information. However, the HC outperforms for some other variables. It is also noticeable that the orientation chosen in our assumption can really affect the results.

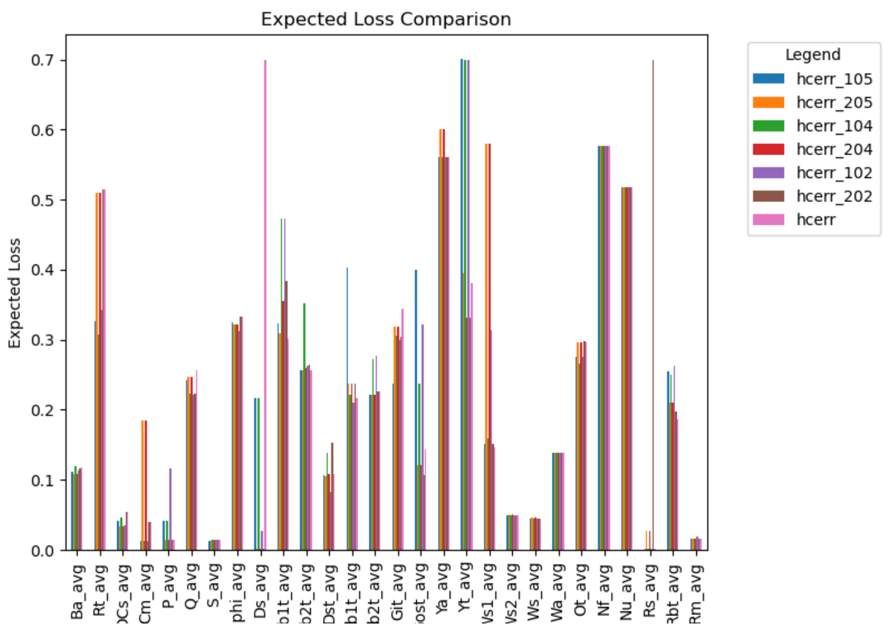


Figure 6.6: Expected Loss Comparison all variables

Let us take a deeper look on the variables we are interested for the prediction: *Hub_temperature*, *Generator_converter_speed*, *Converter_torque*, *Generator_speed*, *Generator_bearing_1_temperature*, *Generator_bearing_2_temperature*, *Gearbox_bearing_1_temperature*, *Gearbox_bearing_2_temperature*, *Generator_stator_temperature*, *Gearbox_inlet_temperature*, *Gearbox_oil_sump_temperature*, *Nacelle_temperature*, *Rotor_speed*, *Rotor_bearing_temperature*. A 10-fold cross-validation is performed 10 times for each algorithm for different dataset sizes.

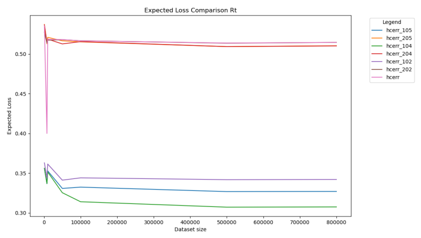


Figure 6.7: *Hub_temperature*

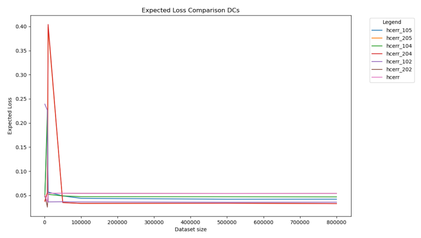


Figure 6.8: *Generator_converter_speed*

In figure 6.7 we can see a significant improvement in prediction of *Hub_temperature* for all dataset sizes for HC_105, HC_104 and HC_102 compared to HC. The HC_104 gets predictions right only ≈ 70% of the time, whereas for HC less than 50%. The second scenario algorithms with constraints, HC_205, HC_204, and HC_202 have the same performance as HC. As for *Generator_ converter_ speed*, all the networks are able to predict it ≈ 50% of the time as presented in figure 6.8. The prediction is a bit better for our approaches with an expected prediction loss of less than 0.05, which is slightly better than the baseline.

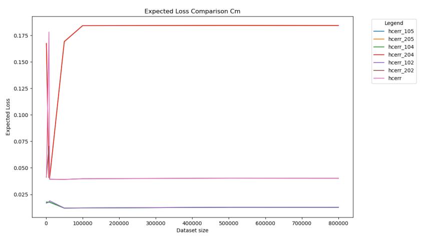


Figure 6.9: *Converter_ torque*

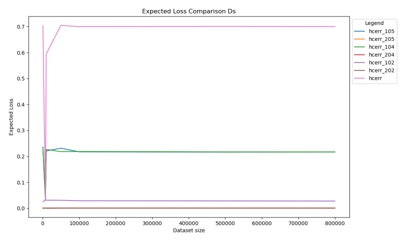
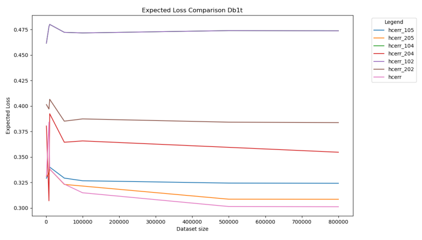


Figure 6.10: *Generator_ speed*

Figure 6.9 shows that the networks have a high accuracy for *Converter_ torque*. HC_105, HC_104, and HC_102 have the best performance with an expected loss of less than 0.025. The other networks, including HC, have a higher loss. HC_204 has the highest loss of ≈ 0.18. In figure 6.10 we can notice the improvement of our approaches compared to HC. HC is able to predict accurately only 30% of the time, whereas the accuracy of the networks learned with prior constraint is ≈ 80% or higher. HC_205, HC_204, and HC_202 have an expected loss of 0, meaning they can predict it correctly 100% of the time.



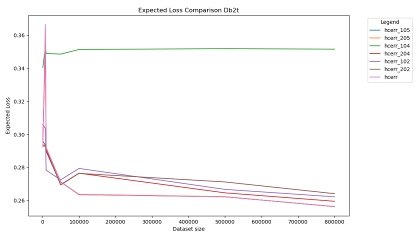


Figure 6.11: *Generator bearing 1 temperature* Figure 6.12: *Generator bearing 2 temperature*

As for *Generator_ bearing_ 1_ temperature* we can observe that HC has a slightly lower expected loss compared to our approaches. However, the expected loss for HC_205 and HC_105 is very close to the HC one, with differences of 0.01 and 0.02 respectively. Due to the very small magnitude of differences, we can say that they have the same performance. The performance of all approaches except HC_104 is similar as well for *Generator_ bearing_ 2_ temperature*. In figure 6.12 we can see that the expected loss is less than 0.28 for these networks.

In figures 6.13 and 6.14 we can see the expected loss for *Gearbox_ bearing_ 1_ temperature*, *Gearbox_ bearing_ 2_ temperature* respectively. HC algorithm performs quite well in both cases. HC_205, HC_104, HC_204, HC_102, and

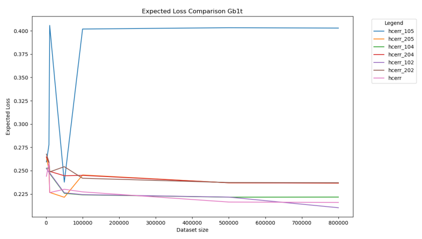


Figure 6.13: *Gearbox bearing 1 temperature*

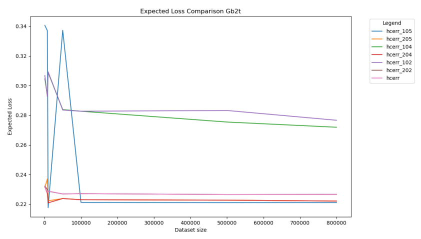


Figure 6.14: *Gearbox bearing 2 temperature*

HC_102 have similar accurancy for *Gearbox_bearing_1_temperature* as well and predict is right in more than 75% of the time. HC_105, HC_205, HC_204, and HC_102 have good accurancy for *Gearbox_bearing_2_temperature* as well.

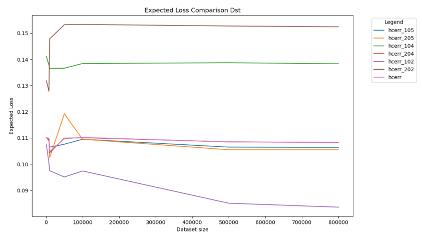


Figure 6.15: *Generator_stator_temperature*

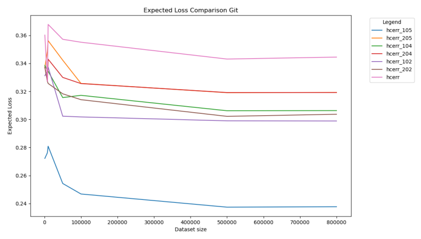


Figure 6.16: *Gearbox_inlet_temperature*

In figure 6.15 you can detect that HC_102 is able to accurately predict the *Generator_stator_temperature* ≈ 91% of the time. HC_205, HC_105, and HC have a good prediction as well, with an expected loss of ≈ 0.11. HC_104 and HC_202 have the highest expected loss of 0.14 and 0.15. However, this score is quite performant as well. In figure 6.16, it is identifiable that all of our algorithms perform better than the one without and prior constraints *Gearbox_inlet_temperature*. HC_105 has an expected loss of ≈ 0.24 whereas HC of ≈ 0.36.

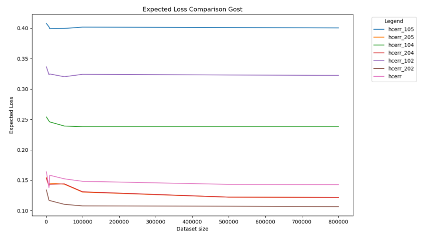


Figure 6.17: *Gearbox_oil_sump_temperature*

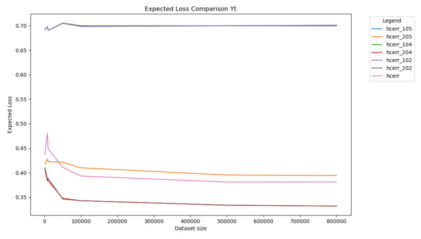


Figure 6.18: *Nacelle_temperature*

Figures 6.17 and 6.18 show the expected loss of the networks for *Gearbox_oil_sump_temperature* and *Nacelle_temperature*. HC and HC_204, HC_205 and HC_202 are able to predict *Gearbox_oil_sump_temperature* quite well with an expected loss of 0.10 - 0.15. As for *Nacelle_temperature* there is an improvement of 5% in prediction with HC_204, HC_202 compared to HC. For both these variables, it is noticeable that the algorithms with constraints oriented in scenario 2 are able to perform quite well. However, if u change the direction of the arcs, the expected loss increases quickly.

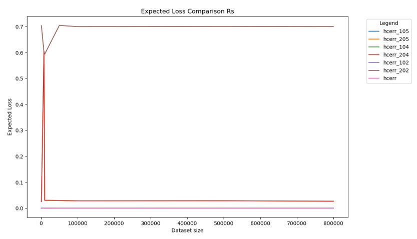


Figure 6.19: *Rotor_ speed*

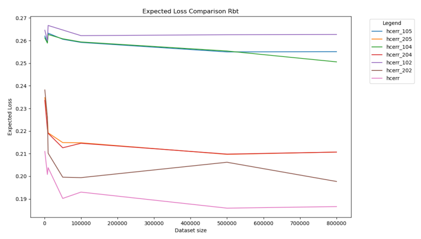


Figure 6.20: *Rotor_ bearing_ temperature*

As for *Rotor_ speed*, you can observe in figure 6.19 that most of our algorithms and HC with an expected loss of ≈ 0 (the lines are overlapping). HC_202 has a really high expected loss for this variable. HC has the lowest expected loss in prediction for *Rotor_ bearing_ temperature* as shown in figure 6.20. Nonetheless, HC_202, HC_204, and HC_205 have a similar performance.

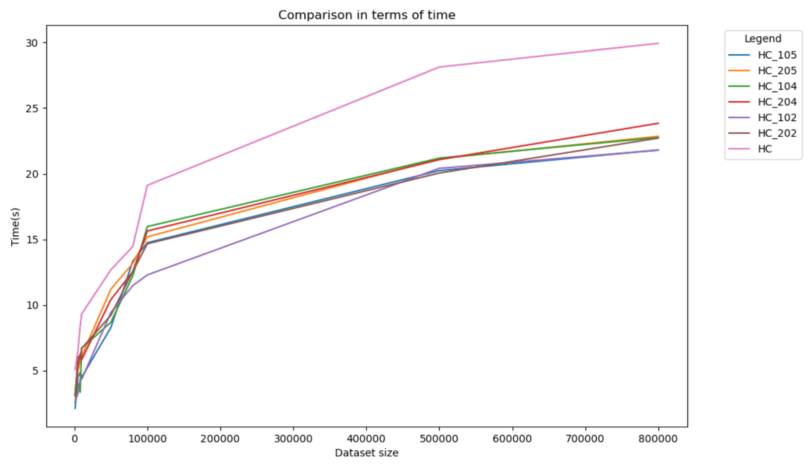


Figure 6.21: Execution time for algorithms with constraints

The last experiment was to check the performance of the algorithms in terms of time. As expected, our algorithms are faster than the HC one without prior constraints. In figure 6.38 you can examine the time acquired by each algorithm for different dataset sizes. Although the difference in time is a maximum of 5 seconds, we believe that in larger networks with more variables and much more data, the difference in time would be quite more significant.

6.3.2 Experiments and evaluation of HC with initial structure

In this subsection, we will detail the experiments with the algorithms that have an initial structure as input in the hill-climbing algorithm. The same steps as described in the previous chapter were conducted for these experiments as well. The structure learning algorithms were tested with different dataset sizes and each of them was run 10 times to get the average results. The quality of results was evaluated via 10-fold cross-validation, each performed 10 times for each learning strategy while measuring the expected loss for all the variables.

In figure 6.22 you can see the average classification errors for hill-climbing algorithms with initial structure derived from the ontology, run on the whole dataset. From the plot, we can see that the Init_HC_204 BN has the best performance as the classification error is the lowest, and Init_HC_102, Init_HC_202 are the worse. However, the magnitude of the differences is so small as not to be practically significant.

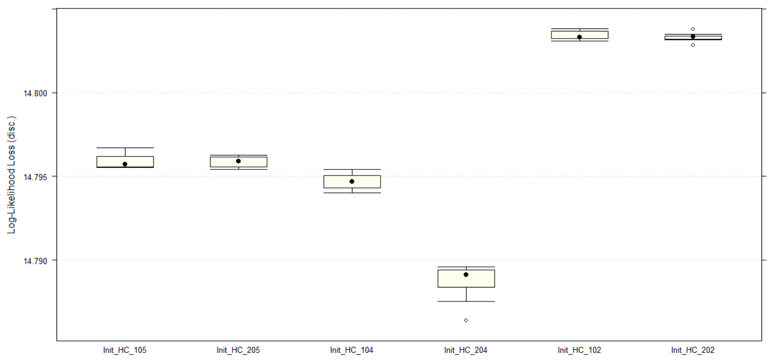


Figure 6.22: Classification error of the networks with initial structure

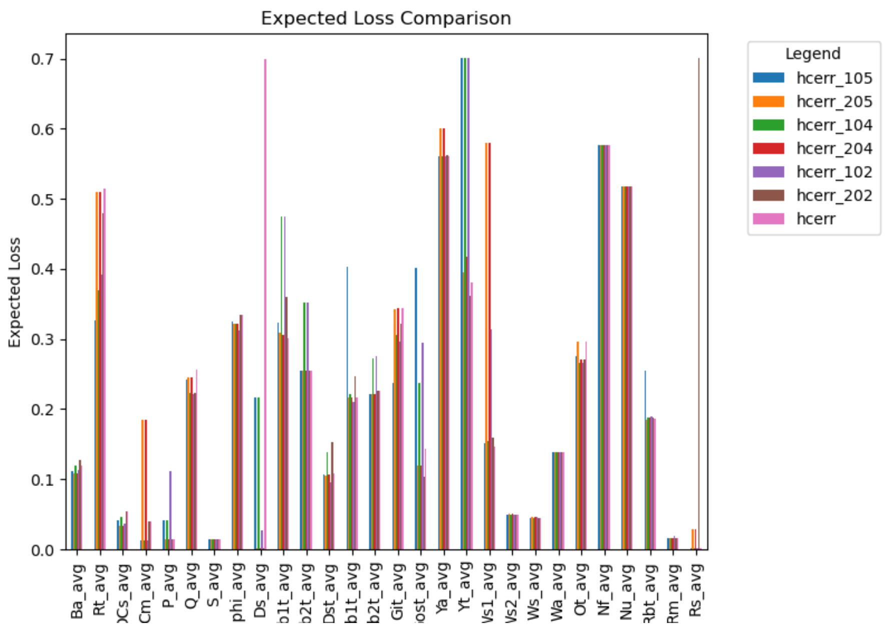


Figure 6.23: Expected Loss Comparison all variables (initial structure)

In figure 6.23 you can see the results of our HC algorithms with initial structure, as well as HC without any prior information. There is some improvement in prediction for some variables in some of our approaches compared to the HC without prior information, and on some others, HC is more performant. Generally, the networks tend to have not as many significant changes as the ones with constraints, due to the fact that the arcs input in the algorithms are not enforced, but a starting point for the algorithms.

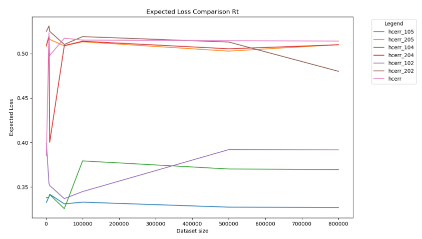


Figure 6.24: *Hub_temperature* (Initial)

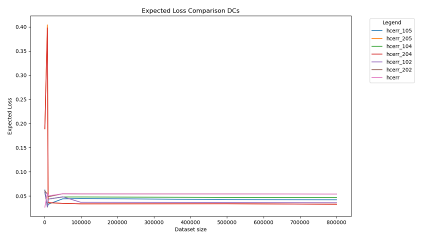


Figure 6.25: *Generator_converter_speed* (Initial)

In figure 6.24 we can see that Init_HC_105, Init_HC_104, Init_HC_102 have and improvement of ≈ 15% compared to HC, wheras Init_HC_205, Init_HC_204, Init_HC_202 have a similar one. As for *Generator_converter_speed*, all algorithms perform quite well, predictiong this varriable correctly ≈ 95% of the times.

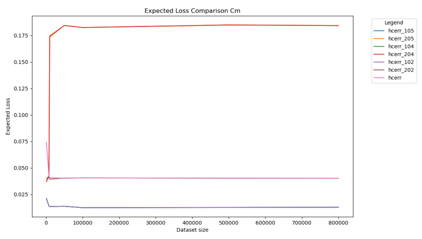


Figure 6.26: *Converter_torque* (Initial)

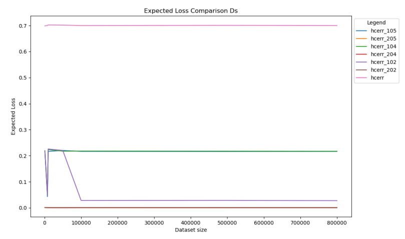


Figure 6.27: *Generator_speed* (Initial)

The networks have a good performance for *Converter_torque* as well. However, in figure 6.26 we see a slight improvement for Init_HC_105, Init_HC_104, Init_HC_102 with an expected prediction loss of less then 0.025. Our algorithms have a big improvement for *Generator_speed* compared to HC. In figure 6.27 you can see that HC has an expected prediction loss of 0.7, meaning that it is predicting this variable accurately on 30% of the time, while for our approaches is ≈ 0.2, 0.1 or even close to 0.

In figures 6.28 and 6.29 you can witness the results for *Generator_bearing_1_temperature* and *Generator_bearing_2_temperature*. HC and most of our approaches have similar expected prediction loss for *Generator_bearing_1_temperature*, except for Init_HC_104 and Init_HC_102. The Init_HC_105 and HC have are able to predict accurately *Generator_bearing_2_temperature* ≈ 75% of the times.

In figure 6.30 we can notice that HC and most of our approaches except Init_HC_105 have good performance for *Gearbox_bearing_1_temperature*. As for *Gearbox_bearing_2_temperature*, the algorithms with initial structure, scenario 2, and HC are able to perform quite well. We can see in figure 6.31 that algorithms with initial structure,

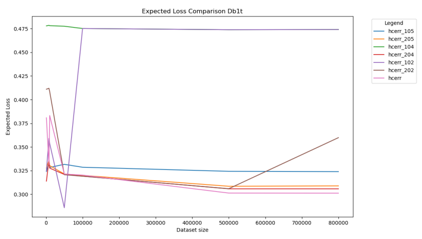


Figure 6.28: *Generator bearing 1 temperature* (Initial)

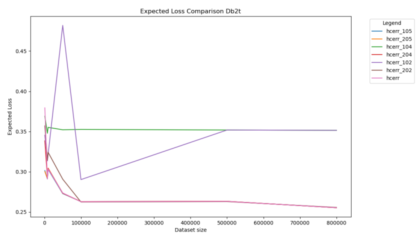


Figure 6.29: *Generator bearing 2 temperature* (Initial)

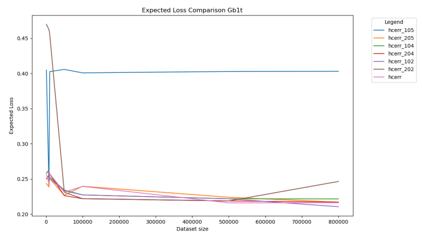


Figure 6.30: *Gearbox bearing 1 temperature* (Initial)

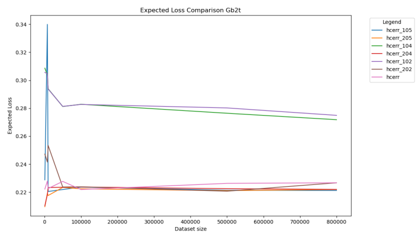


Figure 6.31: *Gearbox bearing 2 temperature* (Initial)

scenario 1 have an increase in expected prediction loss.

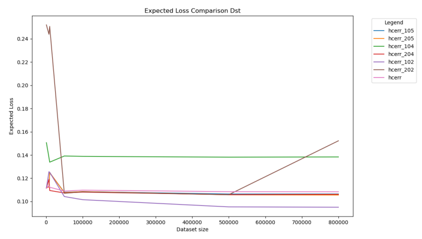


Figure 6.32: *Generator_stator_temperature* (Initial)

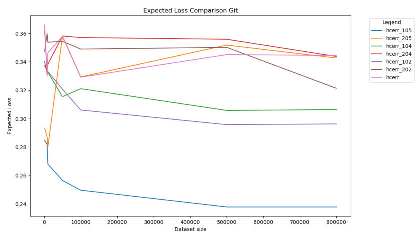


Figure 6.33: *Gearbox_inlet_temperature* (Initial)

Figure 6.32 shows that most networks perform quite well for this *Generator_stator_temperature*. As for *Gearbox_inlet_temperature*, we can see in figure 6.33 that Init_HC_105, Init_HC_104, Init_HC_102 have an improvement of up to 10% compared to HC.

Figure 6.34 6.34 and prove as well that the correct orientation of the arcs can change the quality of prediction significantly, as Init_HC_205, Init_HC_204, Init_HC_202 are $\approx 25\%$ and $\approx 30\%$ more accurate than Init_HC_105, Init_HC_104, Init_HC_102 with regard to *Gearbox_oil_sump_temperature* respectively. HC has a good performance as well.

As for *Rotor_speed*, you can observe in figure 6.36 that most of our algorithms and HC with an expected loss of ≈ 0 (the lines are overlapping). Init_HC_202 has a high expected loss for this variable. In figure 6.37 we can observe some differences in the prediction for

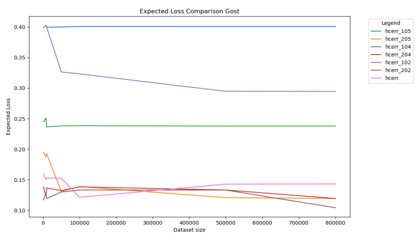


Figure 6.34: *Gear-box_oil_sump_temperature* (Initial)

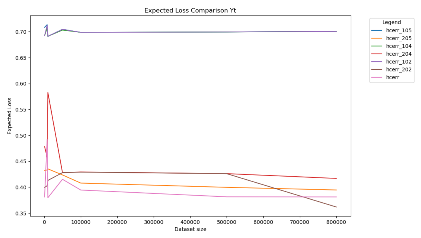


Figure 6.35: *Nacelle_temperature* (Initial)

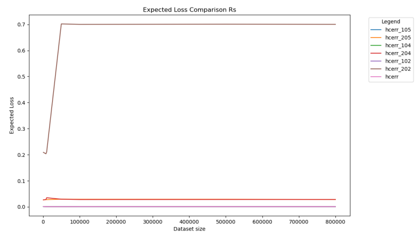


Figure 6.36: *Rotor_speed* (Initial)

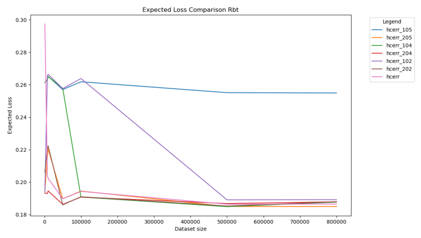


Figure 6.37: *Rotor_bearing_temperature* (Initial)

different data sizes. Init_HC_202, Init_HC_204, Init_HC_205, and HC have a similar performance with an expected prediction loss of ≈ 0.2.

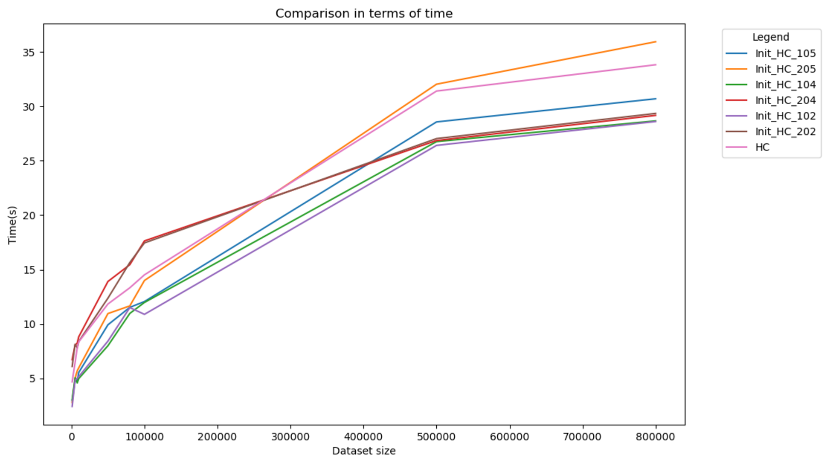


Figure 6.38: Execution time for algorithms with initial structure

As for execution time, the HC algorithm and Init_HC_205 have the highest time as the dataset increases. There is some improvement for the rest of the algorithms compared to HC, but the difference seems small.

6.4 Discussion

The results have proven that deriving information from the ontology from non-taxonomic relations and use it as input for the structure learning algorithms could be quite beneficial. Some of the variables had an improvement in the prediction of more than 50%. However, it was noticeable that the arbitrary orientation we did for the links leaves room for improvement. We assumed that all semantic relations have the same causal orientation, deriving 2 scenarios. Thus, for all concepts ($\{c_i, c_j\} \in C^2$) related by a semantic relation r, , where c_i is the domain and c_j is the range, property p_k of c_i is considered as the cause of p_k of c_j and this latter is the effect, and vise-versa. This restriction is noticeable for some of the variables in our algorithms. In some cases, the algorithms of scenario 1 have a high prediction, while the algorithms of scenario 2 have a lower accuracy for the same variable and vise-versa. Therefore, we were not able to conclude which algorithm performed better overall, as the overall expected prediction loss was quite similar.

If we compare the approaches with constraints and initial structure, they both have good results. However, the magnitude of differences in prediction with comparison to each other and HC is lower than the ones with constraints. The algorithms with constraints tend to have a higher improvement in prediction for some variables, or it worsens quicker. The ones that use an initial structure witness differences in prediction as well, but the gap is tighter, due to the fact that the constraints are not enforced, but used as a starting point.

As for the execution in time, we observed relatively significant improvement for the algorithms with constraints compared to HC, due to the reduction in complexity. We believe that in larger networks with more variables and a bigger dataset, the difference in time would be higher. Moreover, the ratio of input arcs and overall arcs in the network is quite small as well. If more information can be derived from ontology, the differences would be higher as well.

Chapter 7

Conclusions and Future Work

This chapter will highlight the conclusions of our work in solving the problem of learning the structure of Bayesian Networks, our experiments, and our results. In section 7.1, we outline a summary of the main contributions of this research. In section 7.2, we discuss the limitations of the proposed approach and possible future work needed in this research area.

7.1 Summary of Contributions

In this thesis, we researched the possibility of utilizing non-taxonomic relations in the ontology to derive information that can be used as input in the structure learning phase of BN. Therefore, our research addressed three main research questions:

Research Question (RQ1): What kind of relations should we use and how can we derive information from them?

There are a lot of relations in the ontology apart from taxonomic ones that can be utilized. In our approach, we selected specific non-taxonomic relations that encapsulate "dependency" nuance. We believe that if a concept is related via one of the relations that encompass certain dependencies to another concept, it means that at least one property of one of them affects at least one property of the other in the BN as well. A similarity score based on *Dijkstra's shortest path* algorithm was used to calculate the degree of separation between the concepts in terms of the selected relations.

Research Question (RQ2): How to adapt the information derived from the ontology as input for the score-based structure learning algorithms?

Using the similarity score, we derived links between the properties (variables in the BN) that have similar semantics. As for the orientation of the links, we assumed that all semantic relations have the same causal orientation, deriving 2 scenarios. Thus, $\forall$ concepts $\{c_i, c_j\} \in C^2$ related by a semantic relation, where c_i is the domain and c_j is the range, $p_k of c_i$ is considered as the cause of p_k of c_j and this latter is the effect, and vise-versa. This information was exploited in 2 different ways. Firstly, the links inferred from the ontology were fixed during the learning process in the form of constraints. Secondly, rather than forcing these restrictions, the information will be used to simply build an initial structure as a starting point for the score-based algorithm.

Research Question (RQ3): Does the prior information derived from the ontology improve the learning process in terms of quality of prediction and time?

Different experiments with various data sizes were conducted. The quality of the BN structure learned was evaluated via cross-validation. The data were randomly partitioned into 10 subsets. Each subset was used in turn to validate the model fitted on the remaining 9 subsets. We targeted all 28 variables in our model and calculated the expected prediction loss. Each experiment was performed 10 times for each learning strategy. Our results prove that deriving information from the ontology and use it as input for the structure learning algorithms could be quite beneficial. Some of the variables had an improvement in the prediction of more than 50%. If we compare the approaches with constraints and initial structure, they both have good results. However, the magnitude of differences in prediction with comparison to each other and HC is lower than the ones with constraints. The algorithms with constraints tend to have a higher improvement in prediction for some variables, or it worsens quicker. The ones that use an initial structure witness differences in prediction as well, but the gap is tighter, due to the fact that the constraints are not enforced, but used as a starting point. As for the execution in time, we observed relatively significant improvement for the algorithms with constraints compared to HC, due to the reduction in complexity.

7.2 Limitations and future work

During the work on this research, we encountered several limitations and considered some assumptions due to the limited time available for implementation. These limitations are listed below with possible further future work.

- **Arbitrary orientation of the links:** In this work, we assumed that all semantic relations have the same causal orientation, restricting the search space with 2 scenarios derived from the ontology, from domain to range, and vice versa. This assumption caused some limitations of the overall performance of the BN. In some cases, the algorithms of scenario 1 have a high prediction, while the algorithms of scenario 2 have a lower accuracy for the same variable and vise-versa. Therefore, we were not able to conclude which algorithm performed better overall.

- **Evatuating the approach on larger networks:** As mentioned earlier, we noticed significant improvement for some of the variables, but the magnitude of the improvement in the whole network is smaller. Nonetheless, having a similar expected prediction loss while improving the execution time is quite beneficial. We observed a relatively significant improvement for the algorithms with constraints compared to HC, due to the reduction in complexity. We believe that in larger networks with more variables and a bigger dataset, the difference in time would be much higher. Moreover, the ratio of input arcs and overall arcs in the network is quite small as well. If more information can be derived from ontology, the differences would be higher as well.

- **Determining the optimal threshold:** Another possible path to explore is to perform multiple tests in order to determine if there is an optimal configuration for the threshold we use to restrict the amount of information derived from the ontology. Perhaps, after a certain degree of separation between the nodes the prediction does not improve or worsens.

- **Experimenting on other structure learning algorithms:** In this work, we tried to explore the benefits of the prior information from the ontology as inputs in score-based learning algorithms, focusing on the Hill-Climbing one. Other experiments can be conducted on other structure learning algorithms as well.

- **Combining with previous works:** It would be interesting to combine our approach with previous works that try to derive the BN from ontology. As detailed earlier, to derive the BN from the ontology is not feasible in most real-world scenarios, as not all the ontologies have such direct information. However, they could be utilized to derive an initial small substructure that can be used as a starting point, and use this structure and the constraints derived following the methodology of our approach to learning the BN.

Appendix A

List of links derived from the ontology

In table A.1 we show the list of links between the variables of the network inferred from the ontology. Different threshold values were used: $\tau = 0.5, 0.4 and 0.2$. The last link obtained with each threshold is highlighted in the table.

Var1	**Var2**	**Var1 name**	**Var2 name**	**Score**
Yt	Gost	Nacelle_temperature	Gearbox_oil_sump_temperature	1
Git	Gost	Gearbox_inlet_temperature	Gearbox_oil_sump_temperature	1
Rbt	Yt	Rotor_bearing_temperature	Nacelle_temperature	1
Cm	Rm	Converter_torque	Torque	0.83
Gb1t	Gost	Gearbox_bearing_1_temperature	Gearbox_oil_sump_temperature	0.83
Gb2t	Gost	Gearbox_bearing_2_temperature	Gearbox_oil_sump_temperature	0.83
Ds	Rs	Generator_speed	Rotor_speed	0.83
Rt	Rbt	Hub_temperature	Rotor_bearing_temperature	0.71
Git	Yt	Gearbox_inlet_temperature	Nacelle_temperature	0.5
Rbt	Gost	Rotor_bearing_temperature	Gearbox_oil_sump_temperature	0.5
Git	Gb1t	Gearbox_inlet_temperature	Gearbox_bearing_1_temperature	0.45
Git	Gb2t	Gearbox_inlet_temperature	Gearbox_bearing_2_temperature	0.45
Yt	Gb1t	Nacelle_temperature	Gearbox_bearing_1_temperature	0.45
Yt	Gb2t	Nacelle_temperature	Gearbox_bearing_2_temperature	0.45
Dst	Db1t	Generator_stator_temperature	Generator_bearing_1_temperature	0.42
Dst	Db2t	Generator_stator_temperature	Generator_bearing_2_temperature	0.42
Dst	Yt	Generator_stator_temperature	Nacelle_temperature	0.42
Db1t	Yt	Generator_bearing_1_temperature	Nacelle_temperature	0.42
Db2t	Yt	Generator_bearing_2_temperature	Nacelle_temperature	0.42
Yt	Rt	Nacelle_temperature	Hub_temperature	0.42
Gost	Rt	Gearbox_oil_sump_temperature	Hub_temperature	0.29
Gost	Dst	Gearbox_oil_sump_temperature	Generator_stator_temperature	0.29
Dst	Rbt	Generator_stator_temperature	Rotor_bearing_temperature	0.29
DCs	Ds	Generator_converter_speed	Generator_speed	0.28
DCs	Rs	Generator_converter_speed	Rotor_speed	0.21
Db1t	Rt	Generator_bearing_1_temperature	Hub_temperature	0.2
Db2t	Rt	Generator_bearing_2_temperature	Hub_temperature	0.2

Table A.1: Links between the variables of the BN

List of Figures

List of Tables

Bibliography

[Abiteboul 2011] Serge Abiteboul, Ioana Manolescu, Philippe Rigaux, Marie-Christine Rousset and Pierre Senellart. Web data management. Cambridge University Press, 2011. (Cited on page 13.)

[Akaike 1974] Hirotugu Akaike. A new look at the statistical model identification. IEEE transactions on automatic control, vol. 19, no. 6, pages 716–723, 1974. (Cited on pages 12 and 20.)

[Beretta 2018] Stefano Beretta, Mauro Castelli, Ivo Gonçalves, Roberto Henriques and Daniele Ramazzotti. Learning the structure of Bayesian Networks: A quantitative assessment of the effect of different algorithmic schemes. Complexity, vol. 2018, 2018. (Cited on page 11.)

[BnT] Bayesian Network Tab (BNTab) 1.1.3. `https://protegewiki.stanford.edu/wiki/Bayesian_Network_Tab_(BNTab)_1.1.3`. Accessed: 2021-07-11. (Cited on page 16.)

[Bouckaert 1995] Remco Ronaldus Bouckaert. Bayesian belief networks: from construction to inference. PhD thesis, 1995. (Cited on page 20.)

[Brachman 2004] RJ Brachman and HJ Levesque. Expressing knowledge. Knowledge Representation and, 2004. (Cited on page 5.)

[Bucci 2011] Giacomo Bucci, Valeriano Sandrucci and Enrico Vicario. Ontologies and Bayesian networks in medical diagnosis. In 2011 44th Hawaii International Conference on System Sciences, pages 1–8. IEEE, 2011. (Cited on pages 6, 16 and 17.)

[Buntine 1991] Wray Buntine. Theory refinement on Bayesian networks. In Uncertainty proceedings 1991, pages 52–60. Elsevier, 1991. (Cited on page 12.)

[Carvalho 2010] Rommel N Carvalho, Kathryn B Laskey and Paulo CG Costa. PR-OWL 2.0–bridging the gap to OWL semantics. Uncertainty reasoning for the semantic web II, pages 1–18, 2010. (Cited on pages 6 and 14.)

[Carvalho 2017] Rommel N Carvalho, Kathryn B Laskey and Paulo CG Costa. PR-OWL–a language for defining probabilistic ontologies. International Journal of Approximate Reasoning, vol. 91, pages 56–79, 2017. (Cited on page 15.)

[Chickering 1996] David Maxwell Chickering. Learning Bayesian networks is NP-complete. In Learning from data, pages 121–130. Springer, 1996. (Cited on page 6.)

[Chickering 2002] David Maxwell Chickering. Optimal structure identification with greedy search. Journal of machine learning research, vol. 3, no. Nov, pages 507–554, 2002. (Cited on page 12.)

[Costa 2006] Paulo CG Costa, Kathryn B Laskey and Ghazi AlGhamdi. Bayesian ontologies in AI systems. 2006. (Cited on page 5.)

[Da Costa 2006] Paulo Cesar G Da Costa, Kathryn B Laskey and Kenneth J Laskey. PR-OWL: A Bayesian ontology language for the semantic web. In Uncertainty Reasoning for the Semantic Web I, pages 88–107. Springer, 2006. (Cited on page 14.)

[De Campos 2006] Luis M De Campos and Nir Friedman. A scoring function for learning Bayesian networks based on mutual information and conditional independence tests. Journal of Machine Learning Research, vol. 7, no. 10, 2006. (Cited on page 20.)

[Dijkstra 1959] Edsger W Dijkstraet al. A note on two problems in connexion with graphs. Numerische mathematik, vol. 1, no. 1, pages 269–271, 1959. (Cited on pages 21 and 26.)

[Druzdel 2000] Marek J Druzdel and Linda C Van Der Gaag. Building probabilistic networks:" Where do the numbers come from?". IEEE Transactions on knowledge and data engineering, vol. 12, no. 4, pages 481–486, 2000. (Cited on page 5.)

[Fenz 2012] Stefan Fenz. An ontology-based approach for constructing Bayesian networks. Data & Knowledge Engineering, vol. 73, pages 73–88, 2012. (Cited on pages 6, 16 and 17.)

[Friedman 2001] Jerome Friedman, Trevor Hastie, Robert Tibshiraniet al. The elements of statistical learning, volume 1. Springer series in statistics New York, 2001. (Cited on page 34.)

[Glymour 2001] Clark Glymour, Richard Scheines and Peter Spirtes. Causation, prediction, and search. MIT Press, 2001. (Cited on page 9.)

[Gruber 1995] Thomas R Gruber. Toward principles for the design of ontologies used for knowledge sharing? International journal of human-computer studies, vol. 43, no. 5-6, pages 907–928, 1995. (Cited on page 12.)

[Heckerman 1994] David Heckerman, John Breese and Koos Rommelse. Troubleshooting under uncertainty. Rapport technique, Technical Report MSR-TR-94-07, Microsoft Research, 1994. (Cited on page 5.)

[Helsper 2002] Eveline M Helsper and Linda C Van Der Gaag. Building Bayesian networks through ontologies. In ECAI, volume 2002, page 15th, 2002. (Cited on pages 6, 16 and 17.)

[Hirst 1998] Graeme Hirst, David St-Ongeet al. Lexical chains as representations of context for the detection and correction of malapropisms. WordNet: An electronic lexical database, vol. 305, pages 305–332, 1998. (Cited on page 25.)

[Ishak 2011a] Mouna Ben Ishak, Philippe Leray and Nahla Ben Amor. Ontology-based generation of object oriented bayesian networks. In BMAW 2011, pages 9–17, 2011. (Cited on pages 6, 16 and 17.)

[Ishak 2011b] Mouna Ben Ishak, Philippe Leray and Nahla Ben Amor. A two-way approach for probabilistic graphical models structure learning and ontology enrichment. In KEOD 2011, pages 189–194, 2011. (Cited on page 6.)

[Jeon 2007] Beom-Jun Jeon and In-Young Ko. Ontology-based semi-automatic construction of bayesian network models for diagnosing diseases in e-health applications. In 2007 Frontiers in the Convergence of Bioscience and Information Technologies, pages 595–602. IEEE, 2007. (Cited on pages 6 and 16.)

[Kennett 2001] Russell J Kennett, Kevin B Korb and Ann E Nicholson. Seabreeze prediction using Bayesian networks. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages 148–153. Springer, 2001. (Cited on page 5.)

[Koller 2009] Daphne Koller and Nir Friedman. Probabilistic graphical models: principles and techniques. MIT press, 2009. (Cited on pages 9, 10 and 53.)

[Matsumoto 2011] Shou Matsumoto, Rommel Novaes Carvalho, Marcelo Ladeira, Paulo CG Costa, Laecio Lima Santos, Danilo Silva, Michael Onishi, Emerson Machado and Ke Cai. UnBBayes: a java framework for probabilistic models in AI. Java in academia and research, page 34, 2011. (Cited on page 15.)

[Mehlhorn 2008] Kurt Mehlhorn and Peter Sanders. Algorithms and data structures: The basic toolbox. Springer Science & Business Media, 2008. (Cited on page 21.)

[Messaoud 2013] Montassar Ben Messaoud, Philippe Leray and Nahla Ben Amor. Active learning of causal bayesian networks using ontologies: a case study. In The 2013 International Joint Conference on Neural Networks (IJCNN), pages 1–8. IEEE, 2013. (Cited on pages 6, 16 and 17.)

[Mohammed 2016] Abdul-Wahid Mohammed, Yang Xu and Ming Liu. Knowledge-oriented semantics modelling towards uncertainty reasoning. SpringerPlus, vol. 5, no. 1, pages 1–27, 2016. (Cited on pages 6 and 15.)

[Noy 2001] Natalya F Noy, Deborah L McGuinnesset al. Ontology development 101: A guide to creating your first ontology, 2001. (Cited on pages 5 and 12.)

[Pearl 1988] Judea Pearl. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, 1988. (Cited on page 9.)

[Pearl 1991] Judea Pearl and TS Verma. A Theory of Inferred Causation. In Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference (KR91), page 441. Morgan Kaufmann Pub, 1991. (Cited on page 12.)

[Pearl 2000] Judea Pearlet al. Models, reasoning and inference. Cambridge, UK: CambridgeUniversityPress, vol. 19, 2000. (Cited on page 11.)

[Pearl 2014] Judea Pearl. Probabilistic reasoning in intelligent systems: networks of plausible inference. Elsevier, 2014. (Cited on page 5.)

[Rada 1989] Roy Rada, Hafedh Mili, Ellen Bicknell and Maria Blettner. Development and application of a metric on semantic nets. IEEE transactions on systems, man, and cybernetics, vol. 19, no. 1, pages 17–30, 1989. (Cited on page 25.)

[Rissanen 1986] Jorma Rissanen. Stochastic complexity and modeling. The annals of statistics, pages 1080–1100, 1986. (Cited on page 20.)

[Robinson 1977] Robert W Robinson. Counting unlabeled acyclic digraphs. In Combinatorial mathematics V, pages 28–43. Springer, 1977. (Cited on page 11.)

[Sathiya 2019] B Sathiya and TV Geetha. A review on semantic similarity measures for ontology. Journal of Intelligent & Fuzzy Systems, vol. 36, no. 4, pages 3045–3059, 2019. (Cited on page 25.)

[Schwarz 1978] Gideon Schwarz. Estimating the dimension of a model. The annals of statistics, pages 461–464, 1978. (Cited on pages 12 and 20.)

[Setiawan 2019] Foni A Setiawan, Eko K Budiardjo and Wahyu C Wibowo. ByNowLife: a novel framework for OWL and bayesian network integration. Information, vol. 10, no. 3, page 95, 2019. (Cited on page 15.)

[Shafer 1995] Glenn Shafer. P. Spirtes, C. Glymour and R. Scheines," Causation, Prediction and Search"(Book Review). Synthese, vol. 104, no. 1, pages 161–176, 1995. (Cited on page 11.)

[Spirtes 2000] Peter Spirtes, Clark N Glymour, Richard Scheines and David Heckerman. Causation, prediction, and search. MIT press, 2000. (Cited on page 12.)

[Sussna 1993] Michael Sussna. Word sense disambiguation for free-text indexing using a massive semantic network. In Proceedings of the second international conference on Information and knowledge management, pages 67–74, 1993. (Cited on page 25.)

[Suzuki 1993] Joe Suzuki. A construction of Bayesian networks from databases based on an MDL principle. In Uncertainty in Artificial Intelligence, pages 266–273. Elsevier, 1993. (Cited on page 20.)

[Tattar 2018] Prabhanjan Tattar. Marco Scutari and Jean-Baptiste Denis. Bayesian Networks: With Examples in R. Boca Raton, CRC Press., 2018. (Cited on page 19.)

[Torres-Toledano 1998] José Gerardo Torres-Toledano and Luis Enrique Sucar. Bayesian networks for reliability analysis of complex systems. In Ibero-American Conference on Artificial Intelligence, pages 195–206. Springer, 1998. (Cited on page 5.)

[Tsamardinos 2003] Ioannis Tsamardinos, Constantin F Aliferis, Alexander R Statnikov and Er Statnikov. Algorithms for large scale Markov blanket discovery. In FLAIRS conference, volume 2, pages 376–380, 2003. (Cited on page 12.)

[Tsamardinos 2006] Ioannis Tsamardinos, Laura E Brown and Constantin F Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. Machine learning, vol. 65, no. 1, pages 31–78, 2006. (Cited on page 11.)

[van Dijk 2003] Steven van Dijk, Linda C Van Der Gaag and Dirk Thierens. A skeleton-based approach to learning Bayesian networks from data. In European Conference on Principles of Data Mining and Knowledge Discovery, pages 132–143. Springer, 2003. (Cited on page 11.)

[Wang 2007] Ying Wang. Integrating uncertainty into ontology mapping. In The Semantic Web, pages 961–965. Springer, 2007. (Cited on page 6.)

[Whittaker 2009] Joe Whittaker. Graphical models in applied multivariate statistics. Wiley Publishing, 2009. (Cited on page 10.)

[Wu 1994] Z Wu and M Palmer. V Verbs semantics and lexical selection. In Proceedings of the 32nd annual meeting on Association for Computational Linguistics, pages 133–138, 1994. (Cited on page 25.)

[Yang 2005] Yi Yang and Jacques Calmet. Ontobayes: An ontology-driven uncertainty model. In International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06), volume 1, pages 457–463. IEEE, 2005. (Cited on pages 6 and 14.)

[Zhang 2009] Shenyong Zhang, Yi Sun, Yun Peng, Xiaopu Wanget al. BayesOWL: A prototype system for uncertainty in semantic web. In Proceedings of the International Conference on Artificial Intelligence, 2009. (Cited on pages 6 and 14.)