Détection de règles différentielles causales dans les graphes de connaissances

Lucas Simonne

▶ **To cite this version:**

Lucas Simonne. Détection de règles différentielles causales dans les graphes de connaissances. Artificial Intelligence [cs.AI]. Université Paris-Saclay, 2023. English. NNT : 2023UPASG008 . tel-04066830

HAL Id: tel-04066830
https://theses.hal.science/tel-04066830

Submitted on 12 Apr 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

université
PARIS-SACLAY

Mining differential causal rules in knowledge graphs

Détection de règles différentielles causales dans les graphes de connaissances

Thèse de doctorat de l'Université Paris-Saclay

École doctorale n° 580, Sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat : Informatique
Référent : Faculté des sciences d'Orsay
Graduate School : Informatique et sciences du numérique

Thèse préparée dans l'unité de recherche Laboratoire interdisciplinaire des sciences du numérique (Université Paris-Saclay, CNRS), sous la direction de Nathalie PERNELLE, Professeure à l'Université Sorbonne Paris Nord, et Fatiha SAÏS, Professeure à l'Université Paris-Saclay.

Thèse soutenue à Paris-Saclay, le 02/02/2023, par

Lucas SIMONNE

Composition du jury
Membres du jury avec voix délibérative

Mustapha LEBBAH Professeur, Université Paris-Saclay	Président
Sébastien FERRÉ Professeur, Université de Rennes	Rapporteur & Examinateur
Bruno CRÉMILLEUX Professeur, Université de Caen Normandie	Rapporteur & Examinateur
Zied BOURAOUI Maître de conférences (HDR), Université d'Artois	Examinateur

Thèse de doctorat

NNT: 2023UPASG008

université PARIS-SACLAY | ÉCOLE DOCTORALE Sciences et technologies de l'information et de la communication (STIC)

Titre: Détection de règles différentielles causales dans les graphes de connaissances
Mots clés: Graphe de Connaissances; Découverte de Connaissances; Causalité; Fouille de Règles

Résumé: La fouille de règles d'association au sein de graphes de connaissances est un domaine de recherche important. En effet, ce type de règle permet de représenter des connaissances, et leur application permet de nettoyer le contenu de graphes en ajoutant des données manquantes ou en supprimant des données éronnées. Cependant, ces règles ne permettent pas d'exprimer des relations causales, dont la sémantique diffère d'une association. Dans un système, un lien de causalité entre une variable A et une variable B est une relation orientée de A vers B et indique qu'un changement dans A cause un changement dans B, les autres variables du système conservant les mêmes valeurs. Plusieurs cadres d'étude existent pour déterminer des relations causales, dont le modèle d'étude des résultats potentiels, qui consiste à apparier des instances similaires ayant des valeurs différentes sur une variable nommée *traitement* pour étudier l'*effet* de ce traitement sur une autre variable nommée *résultat*. Nous proposons dans cette thèse plusieurs approches permettant de définir des *règles* représentant l'effet causal d'un traitement sur un résultat. Cet effet peut être local, *i.e.*, valide pour un sous-ensemble d'instances d'un graphe de connaissances défini par un motif de graphe, ou bien moyen, *i.e.*, valide en moyenne pour l'ensemble d'instances du graphe. La découverte de ces règles se base sur le cadre d'étude des resultats potentiels en appariant des instances similaires, en comparant leurs descriptions RDF au sein du graphe ou bien leurs représentations apprises à travers des modèles de plongements de graphes.

Title: Mining differential causal rules in knowledge graphs
Keywords: Knowledge Graph; Knowledge Discovery; Causality; Rule Mining

Abstract: The mining of association rules within knowledge graphs is an important area of research. Indeed, this type of rule makes it possible to represent knowledge, and their application makes it possible to clean up the contents of knowledge graphs by adding missing triples or by removing erroneous triples. However, these rules express associations and do not allow the expression of causal relations, whose semantics differ from an association. In a system, a causal link between a variable A and a variable B is a directed relationship from A to B and indicates that a change in A causes a change in B, with the other variables in the system maintaining the same values. Several frameworks exist for determining causal relationships, including the potential outcome framework, which involves matching similar instances with different values on a variable named treatment to study the effect of that treatment on an other variable named the outcome. In this thesis, we propose several approaches to define rules representing a causal effect of a treatment on an outcome. This effect can be local, *i.e.*, valid for a subset of instances of a knowledge graph defined by a graph pattern, or average, *i.e.*, valid on average for the whole set of graph instances. The discovery of these rules is based on the framework of studying potential outcomes by matching similar instances, comparing their RDF descriptions or their learned representations through graph embedding models.

Pour papi Michel, qui m'a montré l'amour et la gentillesse d'un grand-père.

Acknowledgments

Je souhaite remercier Fatiha et Nathalie pour la confiance qu'elles m'ont accordée en me permettant de travailler à leurs côtés. Merci pour vos conseils et votre soutien tout au long de ces trois années, ainsi que pour votre temps et votre patience lors de nos réunions et discussions. Pour un ingénieur "agro", se lancer dans ce doctorat était un vrai défi, mais j'ai grandement apprécié cette expérience grâce au cadre que vous m'avez offert et à votre accompagnement. Au-delà de l'aspect professionnel, j'ai apprécié travailler à vos côtés pour vos personnalités, vos valeurs, et ce, dans un cadre toujours jovial.

Je remercie l'institut DataIA pour avoir financé ce doctorat, ainsi que Juliette Dibie pour avoir supporté ma candidature et encadré mon début de doctorat.

Je remercie les membres du jury pour avoir accepté de consacrer une partie de leur temps à mon rapport de thèse et à ma soutenance. Merci à Sébastien Ferré et Bruno Crémilleux d'avoir accepté d'être les rapporteurs de cette thèse, et à Zied Bouraoui et Mustapha Lebbah d'avoir accepté le rôle d'examinateur.

Merci à Arnaud Soulet et Anne Vilnat, membres de mon comité de suivi intermédiaire, pour leurs remarques constructives qui m'ont permis d'orienter mes travaux.

Je tiens à remercier Rallou Thomopoulos pour avoir, dans un premier temps, accepté d'être membre du jury de cette thèse en tant que membre invité, et, surtout, pour les collaborations que nous avons partagées. J'ai beaucoup apprécié travailler sur une partie de ton domaine d'étude. Merci également à Nicolas Salliou pour avoir participé en tant qu'expert, aux côtés de Rallou, dans l'analyse des résultats de certaines approches développées dans cette thèse.

Merci aux différentes personnes m'ayant permis de pratiquer une activité hors-recherche au cours de mon doctorat. Merci à Chantal Reynaud de m'avoir confié le rôle de chargé de TD au cours de ma première année de doctorat, me permettant ainsi de découvrir l'enseignement. Merci à Marie-Aude Richard et à l'équipe de la DiReV de m'avoir fait confiance pour travailler à vos côtés pendant mes 2e et 3e années de doctorat. J'ai été ravi de pouvoir travailler au sein de la DiReV, et notamment avec Marie-Aude, sur des sujets de valorisation de la recherche, et en garderai des souvenirs mémorables. Merci également à Antoine, Cécile, Fanny,

Vincent et Vinicius que j'ai eu la chance de rencontrer à la DiREV.

Je tiens à remercier les différentes parties administratives qui m'ont accompagné au cours de ces trois années. Un grand merci, notamment, à Stéphanie Druetta de l'école doctorale STIC pour sa réactivité, et à Laurent Oria, du LISN.

Je remercie les membres de l'équipe LaHDAK du LISN pour nos échanges et leur soutien. Je suis, en particulier, très heureux d'avoir pu rencontrer mes amis Alexandra, Armita et Taha au LISN. Merci pour tous les moments que l'on a pu passer ensemble, en visio, au labo ou en dehors. Merci également à Joe pour m'avoir chaudement recommandé de faire un doctorat avec Fatiha et Nathalie.

Merci à Mélanie Munch pour ses éclaircissements et nos discussions sur la causalité.

Un immense merci à mes amis pour m'avoir accompagné au cours de ces trois années : Clément G., Alexandre, Marin, Anne, Reynald, Gwendoline, Laura C., Pierre, Théo C., Théo L., Valentin, Laura T., Benoit, Baptiste, Sophie, Léo, Mathieu, Vincent, Cécile, Guillaume R., Guillaume L., Mathilde D., Erwan, Laurence, Mathilde B., Chloé, Louise, PH, Mathilde P., Hippolyte, Julian, Philippe, Salomé, Brieuc, Agnès, Hermine, Britanny, Sylvestre, Aymeric, Amandine et tous les autres.

Je ne saurais trouver les mots pour remercier Axelle, à qui je pense avec tendresse et admiration. Merci Axo pour m'avoir soutenu au cours de ce voyage, j'ai hâte d'entamer le suivant à tes côtés. Je remercie également toute la famille Eberhardt et Marcus.

Enfin, je termine cette liste en remerciant ma famille. Merci à mes parents de m'avoir toujours fait confiance dans mes études, et à mon frère pour son soutien indéfectible.

Contents

1 Introduction **1**
- 1.1 Objectives and Contributions . . . 3
- 1.2 Thesis Outline . . . 4

2 Preliminaries **7**
- 2.1 Knowledge Graph: Data and Ontology . . . 7
  - 2.1.1 RDF Data Graph . . . 7
  - 2.1.2 Ontologies . . . 8
  - 2.1.3 Knowledge Graph Definition . . . 9
- 2.2 Knowledge Graphs Diversity . . . 10

3 State of the Art **15**
- 3.1 Causality . . . 16
  - 3.1.1 A bit of history of causality . . . 16
  - 3.1.2 Causation, Correlation and Association . . . 18
  - 3.1.3 The Fundamental Problem of Causal Inference . . . 19
  - 3.1.4 Potential Outcome Framework . . . 20
  - 3.1.5 Causal Model and Structural Causal Models . . . 26
  - 3.1.6 The Gold Standard: Randomised Experiments . . . 28
  - 3.1.7 Causality in Knowledge Graphs . . . 30
  - 3.1.8 Discussion . . . 31
- 3.2 Rule Mining in Knowledge Graphs . . . 31
  - 3.2.1 Association Rule Mining . . . 32
  - 3.2.2 Action Rules Mining and Counterfactual Explanations . . . 34
  - 3.2.3 Discussion . . . 35
- 3.3 Identity and Similarity of Instances in a Knowledge Graph . . . 36
  - 3.3.1 Identity and Similarity Predicates . . . 36
  - 3.3.2 Mining Similar Instances . . . 37
  - 3.3.3 Similarity between Embedding Vectors . . . 37
  - 3.3.4 Discussion . . . 40

4 Coarsened Matching for Differential Causal Rules Mining 43
4.1 Differential Causal Rule . . . 44
4.1.1 Motivation: Local Effects and Simpson's Paradox . . . 45
4.1.2 Strata Definition . . . 45
4.1.3 Differential Causal Rule Definition . . . 46
4.2 Approaches: *Dicare-S* and *Dicare-C* . . . 48
4.2.1 Adapting the Potential Outcome Framework . . . 48
4.2.2 Abstract Properties Generation . . . 49
4.2.3 Differential Causal Rules Mining . . . 54
4.3 Experiments . . . 61
4.3.1 Datasets Presentation and Settings . . . 61
4.3.2 Communities Detection . . . 63
4.3.3 Mining Evaluation . . . 64
4.4 Comparative Evaluation and Discussion . . . 66
4.4.1 Comparison to Association Rules . . . 66
4.4.2 Comparison to the State of the Art . . . 67
4.4.3 Comparison of *Dicare-S* and *Dicare-C* . . . 68
4.4.4 Expert Assessment . . . 68
4.5 Conclusion . . . 69

5 Counter Effect Rules Mining 71
5.1 Counter Effect Rules . . . 72
5.1.1 Motivation . . . 72
5.1.2 Counter Effect Rules Definition . . . 73
5.2 Approach for Counter Effect Rule Generation . . . 75
5.2.1 Mining Differential Causal Rules . . . 75
5.2.2 Obtaining Opposite Effects . . . 75
5.3 Experiments . . . 76
5.3.1 Dataset Presentation . . . 76
5.3.2 Settings . . . 76
5.3.3 Mined Counter Effect Rules . . . 77
5.4 Discussion . . . 78
5.4.1 Comparison to Association Rules . . . 78
5.4.2 Comparison to Differential Causal Rules . . . 78
5.5 Conclusion . . . 79

6 Knowledge Graph Embeddings for Differential Causal Rule Mining 81
6.1 Average Differential Causal Rules . . . 83
6.1.1 Motivation . . . 83
6.1.2 Average Differential Causal Rules . . . 84

6.2 Approach: *Dicare-E* 84
6.2.1 Overview 84
6.2.2 Using Knowledge Graph Embeddings to Find Similar Instances 85
6.2.3 Generating Pairs of Similar Instances to Study a Treatment Effect 89
6.2.4 Obtaining a Treatment Effect 90
6.3 Experiments 91
6.3.1 Datasets and settings 92
6.3.2 Selection of the Knowledge Graph Embedding Model 92
6.3.3 Effect of the number of differences S on the distance d and the causal effect $causal_r$ 93
6.3.4 Quantitative and Qualitative Evaluation of the Mined Rules 95
6.4 Discussion 97
6.5 Conclusion 99

7 Conclusion and Perspectives **101**
7.1 Summary of the Results 101
7.2 Discussion and Future Work 103
7.2.1 Short-Term Perspectives 103
7.2.2 Long-Term Perspectives 105

8 Résumé en Français **109**
8.1 Introduction 109
8.2 État de l'art 110
8.3 Appariemment tronqué pour la fouille de règles différentielles causales 111
8.4 Fouille de règles à effets opposés 111
8.5 Plongements de graphes pour la fouille de règles différentielles causales 112
8.6 Conclusion 112

List of Figures

2.1 Example of a RDF Graph 8
2.2 Example of an Ontology using OWL 10
2.3 Building a Knowledge Graph [14] 11
2.4 Extract of *YAGO* 13

3.1 Does Nicolas Cage causes people to drown by falling into the pool? 19
3.2 Two variables can be associated through a third variable 19
3.3 Illustration of the fundamental problem of causal inference 20
3.4 Illustration of how model dependence in causal studies is reduced by matching treated (T) and controlled (C) units. 25
3.5 Causal graph with T causing Y 27
3.6 Applying an *intervention* on a causal graph 27
3.7 Design of a randomised experiment 28
3.8 Simplified representation of AMIE [21] 33
3.9 Simplified overview of a knowledge graph embedding model 38
3.10 TransE Entity and Relation Space [8] 39

4.1 Illustration of Simpson's Paradox 46
4.2 Getting communities of property paths from the co-occurrence graph 51
4.3 Updating the description of the instance i with abstract properties from communities 52
4.4 Partitioning instances into described clusters 55
4.5 Generating potential rules from maximally specific described clusters 56
4.6 Schema of *DBpediaW* 61
4.7 Schema of *Vitamin* 63

5.1 Illustration of a counter effect rule 77

6.1 General workflow of Dicare-E with the perturbed-based strategy . . 86
6.2 Generating a perturbed *#Messi* from the original one 88
6.3 Evolution of d between pairs of instances (y-axis) depending on S (x-axis) 94

6.4 $causal_r(R_1)$ (left y-axis, bullets) and number of pairs created (right y-axis, bins) given S values (x-axis). P_T: education (X_1: educated, X_2: not educated) 95

List of Tables

3.1 Instances Values on Treatment and Outcome - Real Data 21
3.2 Instances Values on Treatment and Outcome - Hypothetical Data . 22

4.1 Simpson's Paradox: An Example 60
4.2 Datasets Description 61
4.3 Extract of Communities - *Vitamin* 64
4.4 Mining results using *Dicare-S* and *Dicare-C*. 65

5.1 Extract of Results. X_2 has a higher will to reduce its meat consumption than X_1. 78

6.1 Distance between instances when analysing the categorical treatment *gender*. 90
6.2 Values of instances on an outcome path 91
6.3 Models performance evaluation metrics on *Vitamin* and *DBPediaW* 93
6.4 Results of the evaluation of rules mined using a baseline, *Dicare-C* and *Dicare-E*. 96

Chapter 1

Introduction

According to Oxford's dictionary, the term data refers to *facts or information, especially when examined and used to find out things or to make decisions.* Data as such were collected by humans for centuries, *e.g.*, for constructing buildings, hunting, preparing meals, etc. However, despite their wide use in all types of domains, no norm or standards existed until the end of the 18th century. At that time, the lack of standards resulted in trouble replicating actions, such as collecting fees in different areas. A focus on data was therefore developed, and standards such as the meter, introduced in 1791, or the kilogram in 1799[1], were defined and used to facilitate data-based operations.

Since the end of the 19th century and the construction of the world's first computer by Charles Babbage, computer-based systems have been massively developed and deployed, making it easier to extract insights from data or to realise operations on it. As a result of the combination of these computer based-systems and technologies allowing the representation of information, the amount of observational data created and stored worldwide increased exponentially, and so did the number of applications relying on data. In the 20th century, a new term appeared: Artificial Intelligence (AI). At that time, AI referred to computers that could possibly replicate human reasoning. Nowadays, AI characterises the families of methods designed to analyse the increasing volume of data to understand and extract new knowledge. In 2004, John McCarthy proposed the following definition: *AI is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable.* AI tools are numerous and can be applied to all types of data, *e.g.*, text, images or tabular data. They are implemented in all do-

[1]In case the reader were interested in how such norms were introduced, we strongly advise him/her to visit the Musées des Arts et Métiers `https://www.arts-et-metiers.net/` in Paris

mains, such as agriculture, banking, health, history, etc., where they can be used as a product or to guide decision-makers. Besides their use, AI methods are also an important focus of research in both academia and in industry.

To process and analyse data automatically, data need to be represented and stored. Several data models exist to store data, one of which is knowledge graphs. Thanks to the linked open data (LOD) initiative [6] and to the development of standard languages such as RDF[2] for representing data and OWL[3] for representing knowledge, knowledge graphs have gained importance. Projects like DBpedia [3], and YAGO [84], which led to the creation of big cross-domains RDF knowledge graphs by automatically extracting information from Wikipedia, have pushed the adoption of knowledge graphs as a standard model for representing data and knowledge. While the term *Knowledge Graph* was introduced in 1972 [74], the increase of its renown can be attributed to Google with the introduction of the Google Knowledge Graph [82] in 2012, which was the first big company to announce their use of knowledge graph publicly. Ever since, companies from different businesses developed their own knowledge graphs: in social networks (Facebook [56], LinkedIn [61]), finance (Accenture [57]), e-commerce (Amazon [43]), and other domains. Knowledge graphs are composed of a collection of triples, where subjects are linked to objects through properties, and of an ontology that gathers classes, properties and axioms. The growing interest in knowledge graphs can be explained by the flexibility they propose, as they do not need a defined schema such as in relational databases, which facilitates updating their content. In addition to representing data, knowledge graphs have the ability to represent knowledge in ontologies. This is of interest when collaborating with experts of various domains, who can represent knowledge in a manner that is usable for a reasoning purpose. Another reason for the adoption of knowledge graphs is their ability to ease data sharing and data interoperability. These features enhance the sharing and use of data from different sources. Knowledge graphs are used for many tasks, such as web searches, recommendations, data integration or conversational agents.

An important task on knowledge graphs is rule mining [21, 58], where a rule represents semantic dependencies between properties. An example of such a rule is $bornIn(x, y) \Rightarrow citizenOf(x, y)$. These kinds of rules can be used for link prediction, error detection, schema matching and so on. However, such rules do not express a causal relation, which is needed to explain a phenomenon. A causal relation between A and B is a directed relation from A to B, where a change in variable A causes a change in variable B within a system where all other variables keep the same values. Such relations differ from an association or a correlation, where a relationship between two variables may stand because of randomness or because of a

[2] `https://www.w3.org/RDF/`
[3] `https://www.w3.org/OWL/`

third variable causally linked to both variables. Discovering causal relations is the key to a large spectrum of domains, from chemistry, where the reaction between two chemical compounds can be studied in a lab, to sociology, where experiments can be tested on populations of people. While such relations are important for decision-making and knowledge discovery, they are often complicated to discover. Indeed, the gold standard consists of planning an experiment to measure the effect of a potential cause on an outcome. However, planning such experiments can be difficult for cost or ethical reasons. It is, for instance, impossible to force someone to start smoking. Therefore, causal relations are often studied through observational data rather than data obtained through an experiment. As naively dealing with observational data can lead to biases in the estimation of causal effects, several frameworks exist to tackle this issue, such as the potential outcome framework [31] and the structural causal model [60], which are the two main frameworks that exist to this end. While methods for discovering causal effects exist for relational and column-based data models, few methods focus on causality within knowledge graphs, and none adapt the potential outcome framework. As such, methods are required to discover causal effects explaining an outcome in a knowledge graph. Given an outcome, a set of rules could represent a set of *explanations* as to why the outcome is observed.

1.1 Objectives and Contributions

Discovering causal relations and providing explanations as to why instances of a knowledge graph have different outcome values is the objective of this thesis. More precisely, we developed a novel approach for differential causal rule discovery by adapting the potential outcome framework and using three different methods for matching the instances. The discovered rules allow us to explain a difference in outcome values through a difference in treatment values. It is to note that these rules capture effects that are *potentially* causal. An example of a rule could be that eating a higher-protein diet explains a higher muscle gain. This work does not cover other frameworks in causal discovery, such as the structural causal model [60], and neither does it address the problem of causal knowledge incorporation in machine learning models [5]. In more detail, this thesis presents the following contributions.

The definition of Differential Causal Rules (DCR), published in [81, 78], that are first-order logic rules providing an explanation to a difference of values of a numerical path for two instances of a class in a knowledge graph. An explanation is a difference of values in another path that can either be numerical or categorical. A DCR is composed of an optional component, named *strata*, that expresses the

description of instances on which an effect is valid. An example of DCR is that for people living in cities, being more sensitive to climate change explains a higher will to reduce one's meat consumption.

An algorithm, *Dicare-S* that discovers DCR from a knowledge graph, published in [81]. This algorithm is based on the coarsened matching of instances based on their RDF description, up to instances URI and discretization process. This algorithm is well suited to graphs where the descriptions of instances have low variability.

An extension of *Dicare-S*, *Dicare-C*, that discovers DCR using a higher coarsened level, published in [78]. It is based on a pre-processing step that aims to facilitate the description of instances. More precisely, it generates abstract properties for instances that summarise part of their description using a community detection algorithm and therefore facilitate their matching. This second matching method is adapted to deal with graphs having more diverse and incomplete descriptions.

A third algorithm, *Dicare-E*, that discovers DCR expressing an average effect, published in [79, 80], as opposed to *Dicare-S* and *Dicare-C* that discover rules expressing local effects. This algorithm mines rules iteratively and matches similar instances based on their embedding representations. This algorithm is the most resistant to missing data and discovers rules that are easier to interpret than rules expressing local effects.

A method for processing DCR to facilitate their interpretation, published in [77], especially when they display opposing effects. This method is based on another representation of the rules and is used to rank treatments.

1.2 Thesis Outline

This thesis is organised as follows. Chapter 2 introduces what knowledge graphs are and how they are built. It also presents the diversity of knowledge graphs that exist. Chapter 3 provides the state of the art on causality and presents the potential outcome framework that is adapted in this thesis. It also shows how current rule mining approaches are not suited to express causal relations and presents existing work on instances matching in knowledge graphs. Chapter 4 presents our first contribution with the definition of differential causal rules and by presenting *Dicare-S* and *Dicare-C* that are two algorithms that mine such rules given a knowledge graph. Chapter 5 introduces a type of rule named counter effect rules that is complementary to differential causal rules. Our approach shows how these rules can be used to facilitate the understanding of differential causal rules

and to rank treatments. Chapter 6 presents the third algorithm we developed to mine differential causal rules expressing the average effect of a treatment. These rules are interesting when dealing with a knowledge graph with missing data. Chapter 7 summarises the thesis and discuss the approaches that were developed, and gives perspectives on future work on causality in knowledge graphs.

Chapter 2

Preliminaries

Although all knowledge graphs are represented in the same data model, they can still be very different depending on how they are created, *e.g.*, using extracted knowledge, crowd-sourced. The objectives of this chapter are to provide an overview of what knowledge graphs are, to define the scope of this thesis regarding the type of knowledge graphs it can be applied to, and to provide examples of knowledge graphs.

Section 2.1 defines knowledge graphs and ontologies as well as the standard languages that exist in the semantic web community, such as the resource description framework (RDF) and the web ontology language (OWL).

Section 2.2 presents how knowledge graphs can be constructed, and, the consequences the construction can have on both qualitative and quantitative features of knowledge graphs. It presents popular knowledge graphs studied within the semantic web community. It also gives the scope of this thesis regarding the type of knowledge graphs it can be applied to.

2.1 Knowledge Graph: Data and Ontology

In this section, we first present the two main components that usually compose knowledge graphs, namely the *Resource Description Framework*[1] (RDF) data graph and the ontology. Then, we give our formal definition of a knowledge graph.

2.1.1 RDF Data Graph

The RDF is a standard promoted by the W3C[2]. It is used to describe entities uniformly to facilitate data interchange on the Web.

[1] `https://www.w3.org/RDF/`
[2] `https://www.w3.org/`

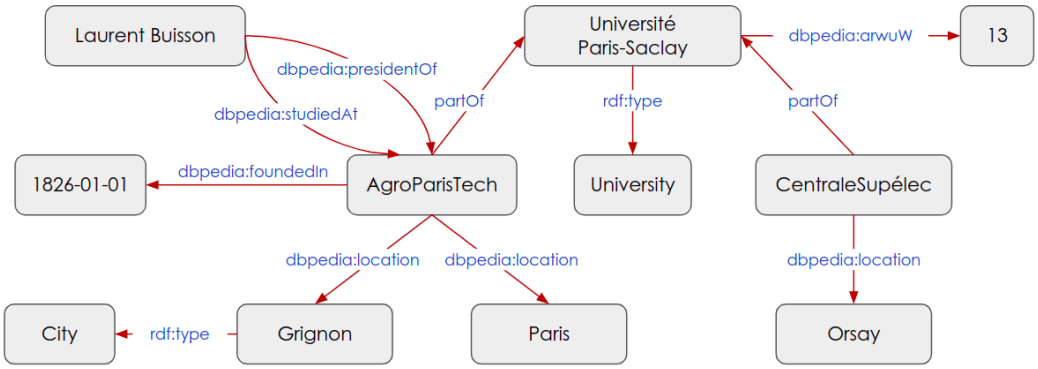

Figure 2.1: Example of a RDF Graph

An RDF data graph is a set of triples that describes entities. The triples are in the form *(subject, property, object)*, where a *subject* and an *object* are represented as nodes in the graph, and a *property* is represented by an arc from *subject* to *object*. A subject is an IRI[3], a property is an IRI, and the object can either be an IRI or a Literal such as a String, an Integer or a Float. Some triples of an RDF Graph may be structured as $(subject, rdf : type, class)$ and indicate that an IRI *subject* is an instance of a *class*.

The use of RDF facilitates the interoperability of a knowledge graph with other knowledge graphs. For instance, if several knowledge graphs constructed by different organisations were to describe a similar domain, using RDF would allow the use of all knowledge graphs combined.

An example of an RDF Graph is presented in Figure 2.1. In this figure, the described entities are French engineering schools that are part of the University Paris-Saclay. As seen in the figure, an RDF Graph is a directed multi-relational graph, and several different properties may exist between two entities. It is to note that beyond this example, entities of a knowledge graph may represent all kinds of types, such as people, buildings, countries, etc.

2.1.2 Ontologies

Ontologies are a core element of the semantic web. They are used in complement to the RDF Data Graph and allow to structure and formalise domain knowledge. More formally, an ontology is defined as *an explicit, formal specification of a shared conceptualisation* [24].

An ontology is usually described using W3C standard languages such as *RDF*

[3]https://www.w3.org/International/O-URL-and-ident.html

Schema (RDFS)[4] or the *Web Ontology Language (OWL)*[5]. RDFS is used to represent a hierarchy of classes and the hierarchy of properties. More precisely, RDFS defines classes with `rdfs:Class`, properties with `rdf:Property`, and their hierarchy with `rdfs:subClassOf` and `rdfs:subPropertyOf`, respectively. Given a property, RDFS also defines its domain and its range with `rdfs:range` and `rdfs:domain` to precise the class, or classes, of its resource and value respectively.

OWL and OWL2, its second version, are more expressive languages that can be used to represent more complex knowledge. An OWL2 ontology is composed of a set of classes and a set of properties that can either express relations between entities, *i.e.*, `owl:ObjectProperty`, or relations between a class and a Literal, *i.e.*, `owl:DataTypeProperty`. A property is defined by its domain and range, which are classes for an `owl:ObjectProperty`, and a class for the domain and a Literal as a range for the `owl:DataTypeProperty`.

In addition to expressing classes and properties, the OWL2 language allows to declare axioms, which are a key component of an ontology. An axiom is a statement that says what is true in a domain [6]. Eight types of axioms exist, including *class axioms*, *object property axioms* and *data property axioms*. For instance, with c_1 and c_2 two classes, the class axiom `owl:subClassOf` indicates a *subsumption* between c_1 and c_2, noted $c_1 \sqsubseteq c_2$. In Figure 2.2, the class *University* is subsumed by the class *Research Institute*. Other class axioms may indicate an *equivalence* or a *disjunction* between c_1 and c_2. An example of an axiom on properties is the *functionality* of a property, which indicates that a functional property is a property that can have only one (unique) value y for each instance x. This can be valid for a datatype property with the axiom `owl:FunctionalDataTypeProperty` or an object property with the axiom `owl:FunctionalObjectProperty`.

Ontologies are therefore used to formalise a vocabulary for a given domain as well as to apply reasoning to infer new knowledge, by making implicit knowledge explicit, and to check the knowledge consistency. It is to note that, more broadly, using an ontology enhances the reasoning in a knowledge graph itself, but also with other knowledge graphs if using the same vocabulary. When ontologies are different, several approaches of ontologies alignment exist to facilitate the use of several graphs by finding the matches between the classes and properties [72].

2.1.3 Knowledge Graph Definition

While the term *Knowledge Graph* was for the first time used in 1972 [74], their study gained attention around 2007 with knowledge graphs being developed in

[4] `https://www.w3.org/TR/rdf-schema/`
[5] `https://www.w3.org/OWL/`
[6] `https://www.w3.org/TR/owl2-syntax/#Axioms`

Figure 2.2: Example of an Ontology using OWL

different projects and all main tech companies such as Google [82], Amazon [43], Facebook [56], Microsoft, Uber, LinkedIn [61], Accenture [57], etc. Academia first contributed to this trend, with researchers increasingly studying various fields of the semantic web. Notably, standard knowledge graphs were created, such as DBpedia [3] and YAGO [84] in 2007.

Despite this craze, a debate still exists to give a formal definition of what a knowledge graph is. We propose to use the definition of *Hogan et al.* [29], *i.e.*, that a knowledge graph is a *graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent relations between these entities.* More precisely, the graph of data conforms to a graph-based data model, and knowledge refers to something known. We retain that knowledge graphs are a multi-relational graph representing data on both instances description, *i.e.*, data graph, and knowledge, *i.e.*, ontology or schema.

Definition 1. ***Knowledge Graph.*** *A* ***knowledge graph*** *KG can be defined by a couple* $(\mathcal{F}, \mathcal{O})$ *with:*

- $\mathcal{O} = (\mathcal{C}, \mathcal{DP}, \mathcal{OP}, \mathcal{A})$ *an* ***ontology*** *defined by a set of classes* $\mathcal{C}$*, a set of owl:DataTypeProperty* $\mathcal{DP}$*, a set of owl:ObjectProperty* $\mathcal{OP}$*, and a set of axioms* $\mathcal{A}$*;*

- $\mathcal{F}$ *a* ***collection of triples*** *(subject, property, object) where subject is an IRI, property is an IRI, and object is either an IRI or a Literal. We note R the set of resources with all IRIs and I the subset of R composed of all IRIs referring to instances. L is the set of Literal values. The triples collection is also referred to as an* ***RDF Data Graph****.*

2.2 Knowledge Graphs Diversity

Knowledge Graphs can be constructed through different processes and from various data sources. As a result, they largely differ on a set of criteria, *e.g.*, their size, completeness, described topic, applications, etc. The objective of this section is

to explain and show this diversity to define the scope of this thesis regarding the knowledge graphs it can be applied to.

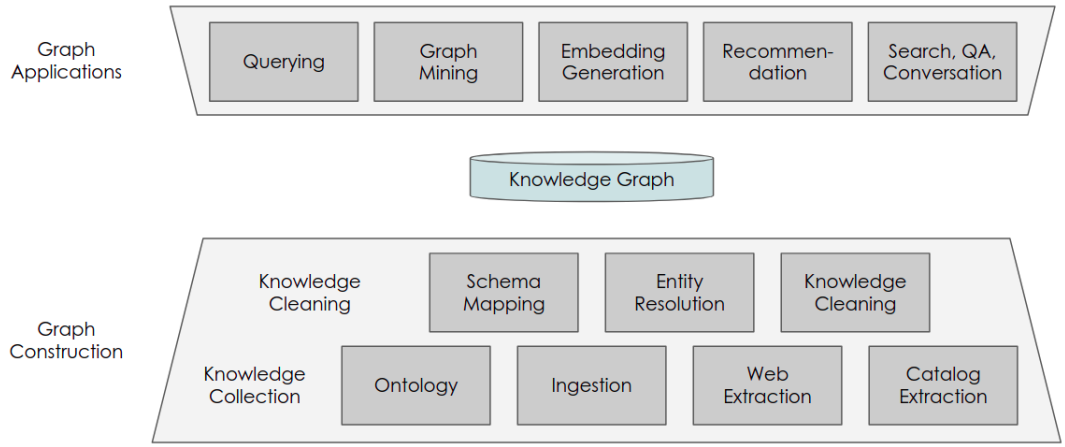

Figure 2.3: Building a Knowledge Graph [14]

Figure 2.3 is a schematic representation of *Dong* [14] on how knowledge graphs are built and their applications. In particular, this figure shows that the construction of a knowledge graph is composed of two parts: first, knowledge is collected from data sources and, secondly, the knowledge is cleaned.

In the first part of the knowledge graph construction, the knowledge collection, knowledge graphs may be built upon automated processes on relational databases, web pages, or a corpus of documents with natural language processing methods. Knowledge graphs can also be built through crowd-sourcing, such as Wikidata [90], where the community can contribute by adding data or editing it.

As a result of this diversity in construction processes, knowledge graphs may differ in size or incompleteness. Let two knowledge graphs, $DrugDiscovery$ and $ChemicalProcess$, illustrate this diversity. In the first example, let a pharmaceutical company that aims to build a knowledge graph, named $DrugDiscovery$. This company may use natural language processing methods over documents such as publications or books on drug testing. As a result, the collection of triples representing the entities and relations extracted from the processing of documents may be missing triples and have some erroneous triples. In the second example, let us consider researchers in chemistry that would be interested in describing their experiments in a knowledge graph. The knowledge graph is named $ChemicalProcess$. The data sources they would use to construct the knowledge graph could be provided by the measuring systems they use, as well as their measures stored in tabular data. In this example, a triples collection would be highly accurate and complete.

As a consequence of the knowledge collection, knowledge graphs are more or less complete and can be composed of erroneous triples depending on their knowledge collection process. Therefore, the second part of the knowledge graph construction is the knowledge-cleaning step. This step consists, through various techniques, of removing erroneous triples, adding missing triples, finding entities that refer to the same one or also finding the correspondence of the schema with the schema of other knowledge graphs. This step is of great interest as it determines the downstream tasks that can be applied to a knowledge graph. For instance, a knowledge graph with missing and erroneous triples will not likely be used for specific queries.

While techniques that address the cleaning of a knowledge graph issue have shown to have high performances, it is, however, almost impossible to have a knowledge graph with no missing or erroneous triples when it has been built with extracted knowledge. Therefore, several assumptions can be made before using a knowledge graph. One assumption is the *Closed World Assumption* (CWA) that states that a missing triple is false. Another, the *Open World Assumption* (OWA), states that a missing fact might be true or false. For instance, in Figure 2.1, the node *AgroParisTech* represents an engineering school. However, this information is not in the graph. In the CWA setting, *AgroParisTech* being an engineering school would be considered false, while this could still be true in the OWA setting. Other assumptions exist and are to be used depending on the application of the knowledge graph.

The diversity of knowledge graphs is also represented through the applications they support. For instance, a knowledge graph such as $ChemicalProcess$ can store data and be the object of queries to retrieve data to train a machine learning model. Knowledge graphs can also be used to learn abstract representations of entities through embedding models [8] to discover new triples or to find similar entities. This could be an application of $DrugDiscovery$, where learning the embedding representation of molecules could lead to the discovery of new treatments.

More generally, the applications of a knowledge graph depend on the domain it describes. Standards knowledge graphs such as YAGO describe general knowledge from several domains. In Figure 2.4, an extract of YAGO shows the description of the singer Elvis Presley. However, YAGO describes not only singers but also people in general, cities, countries, movies, and organisations[7]. It is composed of billions of triples and is linked to other knowledge graphs such as Wikidata, Freebase or DBpedia, therefore facilitating data interoperability. Such knowledge graph can be used for a large set of applications such as question answering or entity recognition [87]. Knowledge graphs with many triples describing general knowledge also serve as standards to test and evaluate the performances of all types of methods, such as in rule mining or key discovery. Knowledge graphs may

[7] `https://yago-knowledge.org/`

Figure 2.4: Extract of *YAGO*

also describe domain-specific knowledge such as industrial processes, human behaviour regarding nutrition [78], historical events [76], or any experiments where observational data are gathered. These knowledge graphs share applications with general knowledge graphs, such as question answering, but can also be the object of other applications that are more domain-specific, for instance, through collaborations with domain experts. These collaborations may lead to the adoption of a common language to describe events of a domain [38] or to the discovery of results that can be used by experts.

Chapter 3

State of the Art

The objective of this thesis is to propose approaches that discover gradual causal rules in knowledge graphs. This chapter presents a background of existing work that has been done on causality and on how causality has been addressed in knowledge graphs.

The concept of causality has been widely studied by philosophers and statisticians. While this concept is rather difficult to apprehend, as well as controversial, the discovery of causal relations has a broad spectrum of applications in many fields, such as life science, chemistry or social science. Consequently, numerous works attempted to tackle causal discovery in statistical and computer science.

We first introduce and discuss the notion of causality to define and scope the content of the thesis and the type of causal discovery it investigates. To this end, we propose an overview of how causality can be apprehended and present the different frameworks that exist to tackle causal relations. One of these frameworks is called the potential outcome framework and aims to estimate the causal effect of a treatment by estimating counterfactuals. This framework is suited to estimate causal effects from observational data when testing causal effects by planning experiments such as A/B tests is impossible. While this framework is widely studied by statisticians, it has never been considered by computer scientists when data are represented as knowledge graphs. A key feature of the potential outcome framework is the counterfactual estimation that can be done using matching techniques. This framework will be detailed since it is used in our approaches. We will also present approaches that estimate the similarity between two individuals in a knowledge graph.

On the other hand, many approaches focus on discovering rules that capture concepts such as associations or correlations. These approaches vary depending on the language expressivity, the search strategy and the quality measure of the rules they generate. In this chapter, we present the different kinds of rules that can be mined in existing approaches in knowledge graphs, and how they differ from the

rules we aim to mine. We explain why such approaches cannot easily be adapted for differential causal rules.

Section 3.1 focuses on causality, describes the main frameworks for its study and introduces the terms used to study causal inference. A deeper analysis of the potential outcome framework is provided.

Section 3.2 is an overview of the different rule types and mining processes, including Association Rules and Action Rules. We show why the semantics of such rules make them unsuitable for a causal study purpose.

Section 3.3 presents several approaches to quantify the similarity between instances of a knowledge graph. More particularly, we present identity and weak-identity links between instances, as well as numerical similarity based on embedding models.

3.1 Causality

This section provides two distinct pieces of content on causality. First, the subsection 3.1.1 gives a brief background on the history of causality and shows how it shifted from a philosophical concept to a key element of artificial intelligence and other scientific fields. Then, the subsections 3.1.2 to 3.1.8 present causality as it is defined by statisticians, the frameworks defined for its study, and how it is addressed for knowledge graphs.

3.1.1 A bit of history of causality

The concept of causality has been widely studied by philosophers throughout the centuries. We first present a selection of philosophical theses discussing the notion of causality, and then discuss how causality is studied within the computer science community nowadays.

The first time the notion of causality was mentioned dates back to the time of Greek philosophers [34], with *Plato* and, significantly, *Aristotle*. Aristotle, in several books such as *Physics*, *Posterior Analytics* and *Metaphysics*, defined four causes during the 4th century before JC: *material*, *formal*, *efficient* and *purpose*. In his mind, causality was not only related to the effect of a thing, but also to its essence itself, *i.e.*, why it had been realised. An example can be provided through the analysis of a house:

1. The *material* cause of the house would be the materials used to build it, such as wood or metal;

2. The *formal* cause is the plan of the house;

3. The *efficient* cause is the people who built the house;

4. The *purpose* cause is to provide shelter to the people living in the house.

Therefore, for Aristotle, four different causes could be defined for an object. This concept evolved over the centuries, and philosophers restrained the study of causality to the efficient cause.

In the middle age, philosophers adapted Aristotle's work in the context of strong Christian culture. For example, *Descartes* makes the distinction between a *general* cause, which is god, and *specific* causes, that mechanical and automatic motion of matter. In his thesis, and the thesis of philosophers after him, such as Spinoza, the concept of causality is materialistic, deterministic, and does not involve the mind.

Leibniz revokes the materialistic view of causation presented by philosophers that preceded him and advocates for the definition of a *reason* that can involve the mind and the soul as final causes.

During the XVIIIth century, *Hume*, in his work *A Treatise of Human Nature* [35], proposes a thesis that had a wide impact on causality. He defines causality as a mental phenomenon. While he agrees that causality should respect (i) contiguity in space and time of the cause and the effect and that (ii) the cause should occur before the effect, it is impossible for him to justify any connection between the cause and the effect. In other words, even given the repetition of an experiment that *could* show a causal effect, it is not enough to guarantee that this causal effect should happen in the following experiment. Therefore, nothing can justify a cause in reality, and causality is a mental phenomenon that people learn by observing things occurring together in time and then making an association between a cause and an effect. Causality, for *Hume*, is a mental construction due to the repeated and constant conjunction between the event we call cause and that we call effect.

In reaction to *Hume*'s work, *Kant* proposes another thesis for causality in his work *Critique of Pure Reason* [40]. While he also believes that causality occurs in the mind, he, however, proposes that causality is a *category* that is innate, rather than something that is learned, that, as a consequence allows people to perceive events happening one after another.

Nowadays, defining causality remains an open question for philosophers and scientists from different domains. The research on causal inference from statisticians and computer scientists increased since the '70s (Rubin [71], Holland [31]), and is nowadays a trending topic (Pearl's Turing Award [60], Bengio [75]). Besides, the number of topics studied by causality is also growing. While most of the existing approaches try to assess causal effects, an increasing number of publications focus on incorporating causal knowledge into machine learning (ML) model training processes, on improving ML's generalisation ability or transfer learning Bengio [75].

3.1.2 Causation, Correlation and Association

This subsection introduces how *association*, *causation*, and *correlation* represent different relations. An *association* is a general concept that indicates that a relation holds between two variables. In other words, two associated variables can be correlated, or one could cause the other. A *correlation* is a specific type of association that indicates a dependence between two variables. It is symmetric and indicates that when one variable changes, the other variable also changes. Two variables can be positively or negatively correlated depending on whether an increase in one variable is associated with an increase or a decrease respectively in the other variable. The correlation r between two random variables X and Y is a value between -1 and 1. It is described in equation 3.1 with $\mathbb{E}(X)$ the expected value of X and σ_X the standard deviation of X.

$$r = \frac{\mathbb{E}[(X - \mathbb{E}(X))(Y - \mathbb{E}(Y))]}{\sigma_X \sigma_Y} \tag{3.1}$$

Correlation is not to be confused with *causation*. Causation is an asymmetric relation between two variables A and B. More precisely, it is a directed relation from A to B, where a change in variable A causes a change in variable B within a system where all other variables keep the same values.

In the following, we present examples illustrating the difference between *causation* and *correlation*. In Figure 3.1, a positive correlation is observed between the number of people who drowned by falling into a pool and the number of films Nicolas Cage appeared in. However, one could doubt that seeing a movie with Nicolas Cage causes people to fall into pools. Another example is the positive correlation between the number of Nobel prize laureates from a country and the average consumption of chocolate by the country's inhabitants. Again, one can doubt that eating chocolate causes winning the Nobel prize. Other examples of spurious correlations can be found on the following website[1].

In the previous examples, we could see that variables that are *a priori* completely independent can be associated. This is merely due to randomness. However, in some cases, an association between two variables can underlie a more complex structure. For instance, the existence of a third variable causal to the two others. Figure 3.2 is an example proposed by Neil [52] to illustrate this type of structure. If looking at the two icons on the bottom, one can see that sleeping while wearing shoes is associated with waking up with a headache: people that slept with their shoes happened to have a headache. However, this association exist due to another variable which is a common cause, *i.e.*, the number of alcoholic beverages that were drunk the night before. Therefore, in addition to randomness, an association can also be due to a causal structure.

[1] `https://www.tylervigen.com/spurious-correlations`

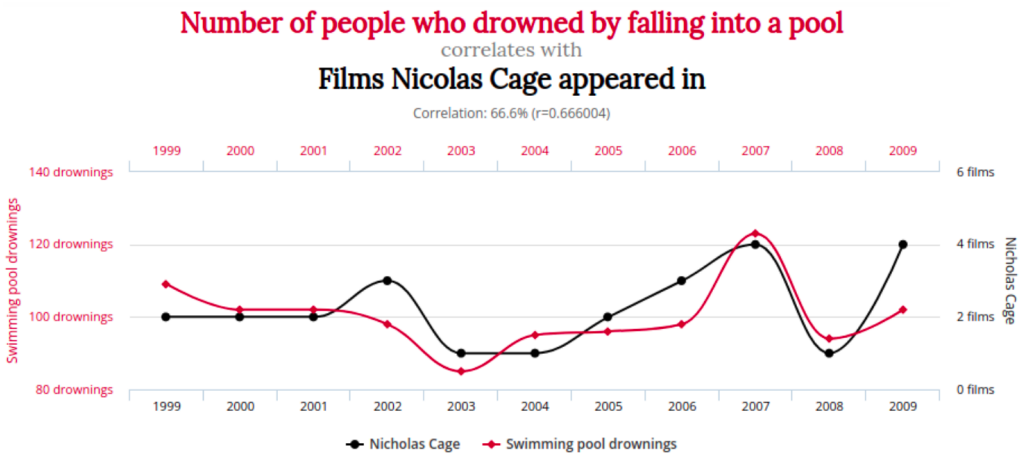

Figure 3.1: Does Nicolas Cage causes people to drown by falling into the pool?

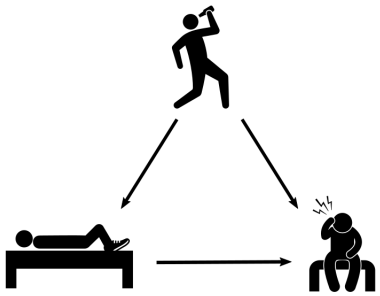

Figure 3.2: Two variables can be associated through a third variable

The examples show that, while an association or a correlation between two variables can be easily measured, demonstrating a causal relation is not trivial. The following section provides a formal introduction to causal inference.

3.1.3 The Fundamental Problem of Causal Inference

This section presents the fundamental problem of causal inference [31] and introduces the vocabulary used within causal inference.

Figure 3.3 presents an example where a person is either suffering from a disease, in which case it is surrounded by a continuous box, or in good health, in which case it is surrounded by a dotted box. Researchers in drug discovery are interested in discovering whether a drug could heal this person, *i.e.*, changing its box from continuous to dotted. To obtain the causal effect of the drug, one would want to compare the health of the sick person after they took the drug to its health if it did not. However, both measures can not be observed, as the person can only take or not take the drug. Therefore, the causal effect of the drug on this person can

not be computed, as taking the drug could heal the person, but the person might have healed without taking it. The problem of not observing both measures is known as the fundamental problem of causal inference.

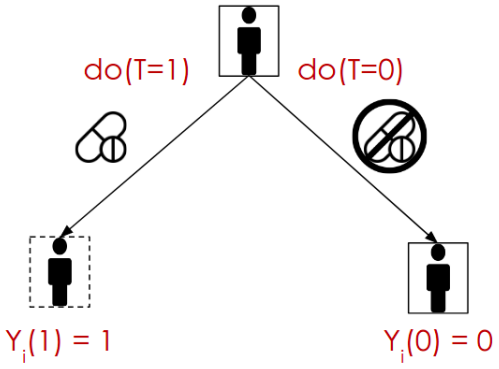

Figure 3.3: Illustration of the fundamental problem of causal inference

More formally, the health Y_i of a person i, named **unit**, is referred to as the **outcome** Y, and the effect of a drug on the person's health is named the **effect**. The drug is referred to as the **treatment** T. $Y_i(do(T = 1))$ is the outcome of i if i takes the treatment, and $Y_i(do(T = 0))$ is the outcome of i if i does not take the treatment. The formula $do(T = 1)$ and $do(T = 0)$ are formally known as an **intervention**. $Y_i(do(T = 1))$ is also written as $Y_i(1)$. The causal effect of the drug on the person is named the **individual treatment effect** TE_i and is the difference between the two outcomes:

$$TE_i = Y_i(1) - Y_i(0) \tag{3.2}$$

Because both outcomes can not be observed, the individual treatment effect can not be computed. The outcome that does not exists is named the **counterfactual**. The notion of counterfactual was first mentioned by the philosopher *Spinoza* during the 17th century.

Several frameworks were introduced to tackle the problem of causal discovery. In particular, two main frameworks are recognised as standards to causal inference: the **potential outcome framework** 3.1.4 and the **structural causal model** 3.1.5. The gold standard of causal discovery is **randomised experiments** that we present in the section 3.1.6.

3.1.4 Potential Outcome Framework

The potential outcome framework is a standard framework for discovering causal relations. The works of Rubin [71, 1, 70] Rosenbaum [68, 69] and Holland [31]

i	$Y(1)$	$Y(0)$	$Y(1) - Y(0)$
1	3	?	?
2	1	?	?
3	?	0	?
4	?	1	?

Table 3.1: Instances Values on Treatment and Outcome - Real Data

are the foundation of this framework. In the following, we present metrics that measure a causal effect in this framework. Then, we present different approaches of the potential outcome framework as well as the hypotheses under which a causal effect can be estimated. Lastly, we introduce matching, one of the most-used methods in the potential outcome framework. It is to note that this framework is used in a large diversity of domains, such as health or political science.

Metrics for Measuring a Causal Effect

The potential outcome framework aims to estimate the counterfactual (3.1.3) to determine a causal effect. Comparing the counterfactual to the observed outcome enables the determination of the average treatment effect ATE. The term *average* is used in opposition to the *individual* effect, which can not be measured, and shows an effect measured on a set of instances. With the notations introduced in 3.1.3, the ATE is defined as follows:

$$ATE = \mathbb{E}[Y(1) - Y(0)] \tag{3.3}$$

Table 3.1 is used to show the computation of the ATE. In this table, computing the ATE as described in equation 3.3 is not possible due to the absence of an outcome, denoted by *"?"*. Complete data as in Table 3.2 would be necessary to compute the ATE. However, it is only presented for an explanatory purpose as it is impossible to obtain. Using the linearity of expectation, the ATE can be derived to $E[Y(1)] - E[Y(0)]$. Thus, the ATE can be computed and equals $ATE = 3.(1/4) + 0.(1/4) + 1.(1/4) + 0.(1/4) = 1$, which would mean that the treatment has an effect.

Another common measure of the treatment effect is the average treatment effect of the treated ATT, described in 3.4, which is similar to the ATE to the difference that it only focuses on treated instances. In the example, the ATT is computed as follows: $ATT = 3.(1/2) + 0.(1/2) = 1.5$.

$$ATT = \mathbb{E}[Y(1) - Y(0)|T = 1] \tag{3.4}$$

i	$Y(1)$	$Y(0)$	$Y(1) - Y(0)$
1	3	0	3
2	1	1	0
3	1	0	1
4	1	1	0

Table 3.2: Instances Values on Treatment and Outcome - Hypothetical Data

As noted in the previous section on the use of observational data rather than data from a randomised experiment, it is to note that both the ATE and the ATT are not equal to the difference between the means of the outcomes of the treated and control sets, $E[Y|T = 1] - E[Y|T = 0]$. There is a bias named the selection bias. Using the linearity of the expectation, it can be demonstrated as follows in equation 3.5:

$$\begin{aligned} E[Y|T=1] - E[Y|T=0] &= E[Y(1)|T=1] - E[Y(0)|T=0] \\ &= E[Y(1)|T=1] - E[Y(0)|T=1] \\ &+ E[Y(0)|T=1] - E[Y(0)|T=0] \\ &= E[Y(1) - Y(0)|T=1] \\ &+ E[Y(0)|T=1] - E[Y(0)|T=0] \\ &= ATT + BIAS \end{aligned} \tag{3.5}$$

The selection bias in equation 3.5 is the difference between the potential outcome of treated instances if they did not take the treatment, which is a counterfactual, and the outcome of the instances that did not take the treatment. In other words, this term should be equal to 0 if the treated and control sets are similar, *i.e.*, with a *covariate balance* between both sets. However, it usually differs from 0 as the treated and control sets often have different distributions. Intuitively, this can be explained by the fact that there is usually a reason why a certain population received a treatment, meaning that it differs from the control population. For instance, one could analyse the effect of a policy aiming to increase the success rate of obtaining a high school degree. To this end, the treated population, *i.e.*, that would be the object of the new policy, for instance, with new lectures, could be students in disadvantaged areas, while the control population would be more privileged students. However, the reason for defining the policy would be that the population have a different success rate, meaning that $E[Y(0)|T = 1] \neq E[Y(0)|T = 0]$. Therefore, bias often exists, and the causal effect is to be computed differently.

Methods and Hypotheses of the Potential Outcome Framework

We showed that a bias could be introduced when estimating the causal effect if comparing the mean outcome of both the treated and control sets. This bias is due to the difference in covariates distribution between the sets. To obtain a non-biased estimation of a causal effect, the potential outcome framework proposes several approaches that are based on the estimation of the counterfactual.

The set of methods can be split into two groups, depending on whether the counterfactual is estimated with an existing unit of the dataset, or with a prediction regarding the other units of the dataset. The first group of methods are matching methods. These approaches estimate the counterfactual of a unit with an existing unit of the dataset. Therefore, it creates matches of instances that can then be compared to estimate the causal effect, which balances the covariate distribution between the treated and control sets. These methods are further investigated in the next section. The second group of families consists of modelling adjustment. They estimate the counterfactual of a unit by making predictions of its potential outcome based on models learned on the units of the dataset. Modelling adjustments might imply different types of models such as linear regression, *i.e.*, named regression adjustment. It has been shown that using both type of models combined lead to better results [22, 28]. Many more methods exist, and some are presented in this tutorial [41], such as natural experiments that aim to *naturally* find sets of instances that form a control and a treated set without attributing the treatment, or instrumental variable.

The methods of the potential outcome framework previously described rely on several assumptions that we present. These assumptions posit an assignment mechanism [83] to ensure a non-biased estimation of the causal effect when creating the control and test sets. The first assumption is the **strongly ignorable treatment assignment** [68]. It stipulates that (i) the treatment assignment should be independent of the potential outcomes given covariates and that (ii) the probability of receiving the treatment should be positive for all instances. The second assumption is called the **Stable Unit Treatment Value Assumption** [70], known as SUTVA. It states that the outcomes of one instance should not be affected by the treatment assignment of any other individuals. This assumption can be more or less difficult to hold depending on the relation between the dataset units, *e.g.*, for people of the same group, like a school or a company. As these assumptions hardly stand in some cases, part of research in causality seeks to define how to relax these assumptions [32].

Matching-based Methodology for Discovering Causal Effects

When dealing with observational data, one of the most used methods for estimating causal effects is matching [83]. Matching can be seen as a pre-processing step that balances the covariates between the treated and control sets. To do so, it sub-samples a dataset by pruning instances that are too dissimilar to the other instances of the dataset. More precisely, it aims to create matches of similar instances that shall be used to measure a causal effect. A causal effect can be assessed after the matching step through various metrics. In this section, we briefly present the steps of matching, *i.e.*, (i) the definition of a distance between instances, (ii) the matching of instances and (iii) the analysis of the outcome.

The first step consists of defining the distance between two instances to determine their similarity. Many distances can be used with tabular data, such as relatively simple ones, *e.g.*, the exact distance, or more complex ones, *e.g.*, propensity score, that we will introduce later. While several distances exist, they are all used for the same task, measuring the similarity between two instances. A distance is chosen depending on the features of the description of the instance. With the instances being described in a low-dimensional space, the euclidean distance can be used to find similar instances. However, such distances become hard to use with higher dimensions, even if they are relaxed using coarsened matching techniques, such as Iacus2012 *et al.* [37], that are doing exact matching on intervals instead of on values. Propensity score [68] facilitates matching when dealing with higher-dimension datasets. Its intuition is to train a model that will predict a score for each instance, representing its probability of being treated. Two instances having the same score can then be considered similar as they share the same probability of being treated. However, limits to propensity score were shown [42], in particular, as it can create many irrelevant matcheIacus2012. Iacus2012e created with the defined distance. Several matching methods exist, such as nearest-neighbour matching or sub-classification. For each of these methods, several parameters are to be defined. In nearest-neighbour matching, a user can decide to match a treated unit to one control unit, or several of them. Also, an instance that has been selected in a pair can either be drawn in another pair or not. In another setting, sub-classification aims to estimate a causal effect in several sub-classes instead of creating pairs through the dataset. The different choices of matches are discussed by *Stuart* [83].

In the third step, a causal effect is estimated given the set of instances selected by the matching step. A causal effect can be directly defined on the selected instances, or on models that represent either the treated or the control units, for instance, by comparing the coefficients of such models. Various metrics exist to measure a causal effect, such as the difference in the potential outcomes between treated or control units, or relative metrics, such as the odds ratio (OR) [86] or

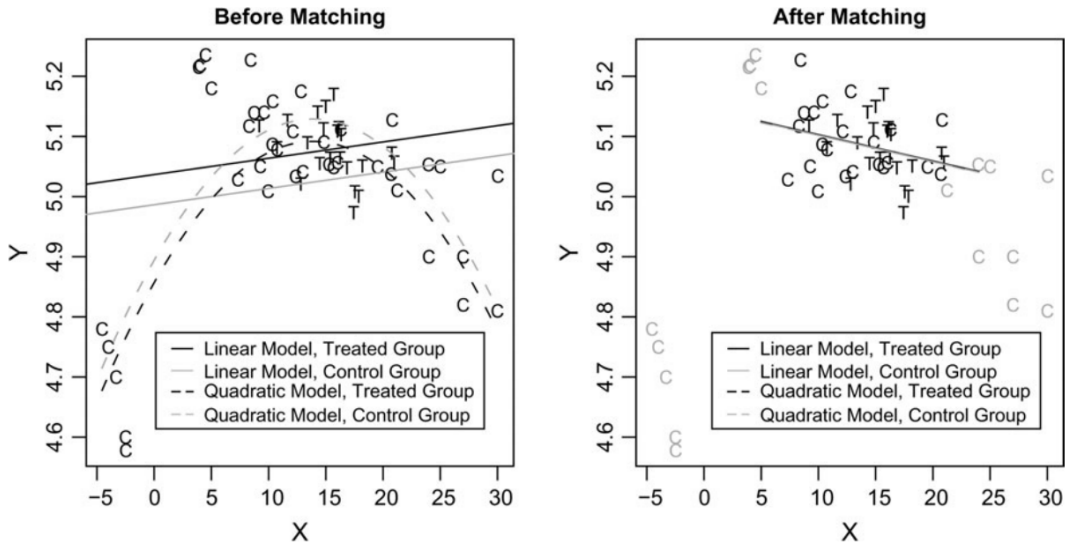

Figure 3.4: Illustration of how model dependence in causal studies is reduced by matching treated (T) and controlled (C) units.

relative risks (RR), that measure how a treatment is associated to an outcome. The choice of the metric depends on the measured effect, *e.g.*, the ATE and the ATT, as well as the type of treatment or the outcome. For instance, with the outcome being a discrete variable, the OR and the RR are better suited.

Figure 3.4, from *Ho* et al. [28], is used to illustrate the steps previously described. Two sets of units are studied: the set of treated units, represented by a T on the figure, and the set of control units, represented by a C. In this example, the outcomes of treated and control units are modelled by a linear or quadratic model. The causal effect is obtained by comparing the model of the treated units to the model of the control units. In the left figure, no matching has been applied, and all instances are considered to estimate the models. In the right figure, only matched instances are selected to estimate the models. As a result, in the left figure, the comparison of the linear models indicates that the treated group has higher outcome values, while the comparison of the quadratic model indicates that the control group has higher outcome values. In the right figure, where matching has been applied, both models show that outcome values are the same for both the treated and control units. This example points out several key messages. The first is that causal analysis is to be proceeded carefully, as naively using different models leads to different estimations of a causal effect. In other words, there exist a model dependence in the estimation of the causal effect. The second is that matching prunes instances dissimilar to others only to compare *comparable* instances and that it reduces model dependence by doing so.

It is to note that the potential outcome framework has been defined for tabular data and that the notion of distance and similarity is essential to the matching process, as estimating a causal effect is done by comparing similar instances. Therefore, approaches focusing on similarity in knowledge graphs are presented in section 3.3.

3.1.5 Causal Model and Structural Causal Models

The structural causal model framework (SCM) is another important framework for studying causality. It consists of modelling a system with equations named *structural equations*. In this framework, such a system can be represented as a *causal graph* that is a directed acyclic graph (DAG) in which nodes represent variables and edges the causal links between the variables they link. This section aims to provide a brief introduction to this framework. For more details, we advise the reader to refer to the work of Pearl [60].

The structural causal model is, to a certain extent, comparable to statistical learning, where the objective is to learn a probabilistic model from observations. A causal model, however, in addition to learning a model from observations, also learns counterfactuals. It does not only learn how to predict an outcome but can also answer the question: *what would have happened to the outcome if a certain variable had a certain value?*. In other words, it is able to model *interventions* that is how the system would react if a variable were to be changed. An example of intervention would be: *what would have happened if this person took the treatment A?* Consequently, causal models have a much bigger explanation power than probabilistic models, which only focus on learning to make predictions.

One component of a structural causal model is a set of *structural equations*, which are functions involving the variables of a system that are linked causally. For instance, let T and Y two variables where T causes Y. The structural equation describing this causal relation is represented as follows:

$$Y := f_Y(T) \tag{3.6}$$

The function f_Y can be any type of function that describes the distribution of Y given T. As a causal relation is not symmetric, the symbol "=" can not be used. Instead, it is replaced by ":=" [60] to indicate the direction of the relation, *e.g.*, from T to Y in this example. In addition to the structural equation, the structural causal model provides a DAG that describes this system. This DAG is depicted in Figure 3.5), where edges represent the direction of the causal relation.

A system can be composed of many more variables and causal relations. For a sake of simplicity, we present another simple system composed of variables X, Y and T with the following structural equations:

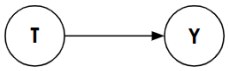

Figure 3.5: Causal graph with T causing Y

$$\begin{aligned} Y &:= f_Y(X, T) \\ T &:= f_T(X) \end{aligned} \tag{3.7}$$

These equations indicate that T is caused by X, Y is caused by X and T, and that T is not caused by any variable. The resulting DAG is presented on the left side of Figure 3.6. As seen in the figure, no edges are directed to the node representing X as it is not caused by T or Y.

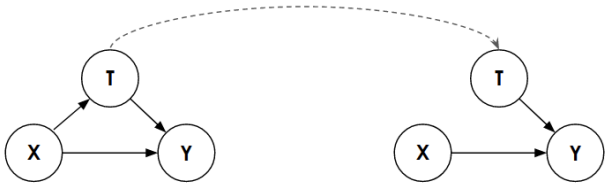

Figure 3.6: Applying an *intervention* on a causal graph

While the cause from T to Y in the previous system (Figure 3.5) is rather simple to study as the system only owns two variables, it is much harder in this system where X causes both T and Y, therefore acting as a *confounder*. Therefore, the causal effect from T to Y should be estimated by considering the confounder X. Interventions were defined for this purpose. An intervention sets the value of a variable, *e.g.*, $T = t$ in our example and, by doing so, removes causal edges in the DAG. It is to note that the intervention on X is possible as the DAG is known by the user, who therefore knows that X is a confounder. The right side of Figure 3.6 shows the DAG after the intervention on T: as the value of T is set, X no longer causes T. The DAG is said to be *mutilated*.

More generally, two variables A and B are usually included in a system with other variables. The other system variables may act as confounders to A and B. Consequently, a user should use an *adjustment set* of variables to discard the effect of the confounders and to obtain a non-biased estimation of the causal effect from A to B. The adjustment set defines variables where interventions should be applied and is obtained by analysing the causal graph, *i.e.*, the DAG, of a system. In our example, X is the only variable of the adjustment set. In practice, the adjustment set may be more difficult to define and its construction relies on criteria such as the *backdoor path* [60].

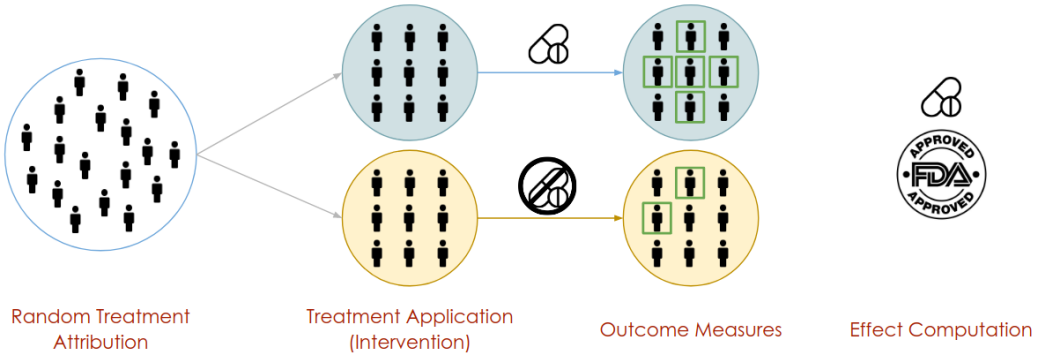

Figure 3.7: Design of a randomised experiment

While a structural causal model is extremely powerful in estimating causal effects, it, however, presents some limitations. Learning the parameters of the structural equations may be complex, and the estimation of causal effects strongly relies on the definition of the causal graph, which is not trivial even with expert knowledge. Furthermore, the estimation of causal effects is highly sensitive to the variables of the adjustment set, as missing or wrongly added variables can lead to biased estimation.

In the next section, we present randomised experiments that are the gold standard for estimating causal effects.

3.1.6 The Gold Standard: Randomised Experiments

Randomised experiments are a design of experiments to estimate the causal effect of a treatment on an outcome. More precisely, as the individual treatment effect can not be computed (section 3.1.3), randomised experiments can be used to estimate several effects such as the ATE (3.3) or the ATT (3.4).

Let a unit, *e.g.* a person or a molecule, that can either take or not take a treatment, *e.g.* a drug or a change of temperature. Given a set of units, a randomised experiment consists of randomly splitting the units in the set of treated that receives the treatment ($T = 1$) and the set of control that does not ($T = 0$). With X the set of covariates and T the treatment value, it results in T being independent of all covariates, $T \perp\!\!\!\perp X$.

For instance, if testing a drug, people from the treated group will receive the drug while people from the control group will receive a placebo. The design of a randomised experiment is schematised in Figure 3.7, where the effect of a drug is assessed[2]

[2]FDA logo: `https://www.istockphoto.com/`.

As a consequence of the random attribution of the treatment, a covariate balance is observed between the set of the treated and the set of control. In other words, the distribution of the covariates is the same in both sets. The conditional distribution of covariates of both sets can be rewritten as:

$$\begin{aligned} P(X|T=1) &= P(X) \\ P(X|T=0) &= P(X) \end{aligned} \tag{3.8}$$

The covariate balance is therefore shown as:

$$P(X|T=1) = P(X|T=0) \tag{3.9}$$

Given the covariate balance, the effect of a treatment is obtained by simply comparing the distribution of the outcomes from both sets. The resulting effect is unbiased.

This result can be compared to the previous frameworks described in 3.1.4 and 3.1.5. Having a covariate balance is the objective of the potential outcome framework 3.1.4, which applies a processing step of instance selection to obtain it. Regarding the structural causal model 3.1.5, it aims to obtain the same unbiased estimate by making all adjustments. From a graphical point of view, applying a random attribution of the treatment removes all edges directed to the outcome, except the edge from the treatment. It is for that reason that the estimated effect is unbiased. Randomised experiments are, therefore, rather simple to design, and are easier to handle to obtain unbiased estimates of causal effects than existing frameworks. These reasons led to using it as a gold standard in causal discovery experiments.

However, applying such experiments can be delicate for several reasons depending on the domain it can be designed for. One of the reasons is the cost of such experiments. Depending on the features of the experiment, such as its duration or the type of instances it is applied on, *e.g.*, people, molecules, a randomised experiment can become expensive. It is for instance, cheaper to test a digital marketing strategy on a social network within a few days than to test a drug on people for several months. Apart from the cost, ethics are another limit of randomised experiments. As an example, it is not possible to force someone to smoke.

Consequently, research in causal discovery is often done on data that already exists, *i.e.*, **observational data**, where a researcher did not interfere before. In such a setting, the independence of the treatment attribution is no longer valid, and comparing the outcomes of the treated and control unit can introduce a bias in the causal effect, as shown in 3.5, as the distribution of covariates in both sets is no longer balanced. For instance, if studying a drug, the treated set could be composed of people that are in general more likely to be sick, or that have a

different diet, etc. As a result, users assessing causal effects in observational data can use approaches within the potential outcome framework or structural causal modelling.

3.1.7 Causality in Knowledge Graphs

To the best of our knowledge, only two papers deal with causality within knowledge graphs [51, 12]. We describe and underlie the limits of these approaches.

In the first paper [51], *Munch et al.* propose a two steps approach to discover causal relations. First, the knowledge graph is converted into a relational database using ontological and expert knowledge. Then, they learn a probabilistic relational model on the relational database. As output, a bayesian network is constructed and provides joint tables of probabilities. These tables describe the likelihood of observing an event given the set of associated variables. For instance, if studying the year when a writer wrote his first book, the table indicates that the likelihood of a writer writing a book before 1980 is 0.01 if the writer is born after 1950 and has studied in a university with a rank above 101. Although such a relationship might be causal, the approach has some limits. Firstly, learning a probabilistic relational model implies discretizing all variables, which is a loss of information for numerical variables. Secondly, it has been shown that learning bayesian networks is a difficult task whenever the database becomes large [53]. Also, this approach has only been evaluated on a small knowledge graph (≈7k triples).

In the second paper, *De Haan et al.* [12] are working on the COoperation DAtabank [3] (CODA) knowledge graph that describes scientific papers in social science, their experimental settings, and the research hypotheses the papers tested. For instance, the description of a paper within the CODA KG may include the effect of a variable, named the Independent Variable (IV), on another variable, named the Dependent Variable (DV). In other words, it describes research hypotheses holding between an IV and a DV, and the relationship between them, such as their strength. Using this knowledge graph, the approach aims to discover new hypotheses for domain experts by predicting new links between an IV and a DV. This link prediction is done by relying on knowledge graph embeddings. While this approach leads to the discovery of new hypotheses, *i.e.*, potentially new causal effects, it, however, relies on exploiting a knowledge graph where effects are already measured and described. It is not designed to be applied on a knowledge graph where effects are to be discovered from scratch. Furthermore, as hypotheses are discovered from a link prediction task, quantifying their causal effect is hardly interpretable.

[3] `http://data.cooperationdatabank.org/`

3.1.8 Discussion

The notion of causality has been introduced and compared to association and correlation. It is to retain that association and causation are different relationships. An association can be observed due to underlying causal structures or randomness and should not be confused with causal relations.

When randomised experiments can not be conducted, the potential outcome framework and the structural causal model are the two standard frameworks for causal inference. On one hand, the potential outcome framework relies on comparing similar instances that differ in treatment value to study the effect of the given treatment. It seems relevant to adapt it to knowledge graphs where the similarity between instances is an important field of research (see section 3.3). On the other hand, the structural causal model relies on tabular data and the definition of a causal graph to learn the parameters of structural equations. Therefore, while there exist approaches to transforming knowledge graphs into relational databases, we believe that adopting the potential outcome framework to the causal discovery problem in knowledge graphs might be more suited.

The study of the state of the art of causality in knowledge graphs showed that only a few approaches focus on causal discovery and that these approaches were either only applicable to small knowledge graphs [51] or not defined to really measure causal effects [12]. Furthermore, none of these approaches was designed to explain why two instances have different values for a numerical property.

In the following section, we present rule mining approaches and study whether they can be used to express a causal relationship or not.

3.2 Rule Mining in Knowledge Graphs

A rule expresses a relation between two graph patterns through a *IF-THEN* structure. Rules are composed of a body $\vec{B}$ and a head $\vec{H}$ such as $\vec{B} \rightarrow \vec{H}$, with $\vec{B}$ and $\vec{H}$ being graph patterns and $\rightarrow$ indicating the implication. Several types of rules exist, such as *association rules* or *action rules*.

The knowledge embedded in a rule can be *declared* or *induced* from observations [29]. Declared knowledge is composed of commonsense knowledge [48], expressed by common people, and of domain knowledge, expressed by domain experts. Such knowledge can be represented in the ontology of a knowledge graph using ontology axioms or additional rules. Conversely, knowledge can also be induced by analysing frequent patterns within a knowledge graph. Resulting rules can also be represented in an ontology.

As we aim to discover causal relations, we shall in this section focus on rules obtained by rule mining approaches that represent *inductive knowledge*. In par-

ticular, we present a standard association rule mining approach, *i.e.*, *AMIE* [20]. In addition to understanding how association and action rules can be mined, we study whether such rules could be suited to express a causal relation.

3.2.1 Association Rule Mining

An association rule AR has a head $\vec{B}$ composed of one atom only, and a body $\vec{H}$ composed by at least one atom, where an atom is a triple (s, p, o) where the object can either be a variable or a constant. An example of an association rule is $hasChild(x, y) \wedge citizenOf(x, z) \Rightarrow citizenOf(y, z)$. Association rules mining approaches are used for knowledge discovery and graph completion, as association rules represent inducted patterns that can be used to remove erroneous triples or discover missing ones.

An association rule is usually assessed by metrics such as the *support*, the *head coverage* and the *confidence*. Given the rule $R : \vec{B} \Rightarrow r(x, y)$, these metrics are described in equations 3.10, 3.11 and 3.12 respectively. The *support* of a rule is the number of correct predictions that can be deduced by the rule. The *head coverage* is a relative value between 0 and 1 that represents the number of instantiations of the rule on the number of instantiations of the head of the rule. The *confidence* is a ratio between the number of instantiations of a rule on the number of instantiations of its body.

$$supp(\vec{B} \Rightarrow r(x, y)) = \#(x, y) : \exists z_1, ..., z_m : \vec{B} \wedge r(x, y) \tag{3.10}$$

$$hc(\vec{B} \Rightarrow r(x, y)) = \frac{supp(\vec{B} \Rightarrow r(x, y))}{\#(x', y') : r(x', y')} \tag{3.11}$$

$$conf(\vec{B} \Rightarrow r(x, y)) = \frac{supp(\vec{B} \Rightarrow r(x, y))}{\#(x, y) : \exists z_1, ..., z_m : \vec{B}} \tag{3.12}$$

As graph completion and refinement is an important topic for knowledge graphs, in particular, due to their creation process (see Chapter 2), a large set of methods exist to mine association rules in knowledge graphs, such as *AMIE* [21] and its improved versions [20, 44], RuDiK [58], and AnyBURL [49]. Such methods are divided into two categories: generate and test techniques and divide and conquer techniques.

We present *AMIE* [21] in a concise manner, which can be considered a standard approach in association rule mining. *AMIE* is a generate and test technique that discovers the complete set of closed and connected horn rules in a knowledge base. Two atoms are *connected* if they share a variable or entity, and a rule is *connected* if every atom is connected transitively to every other atom of the rule [21]. A

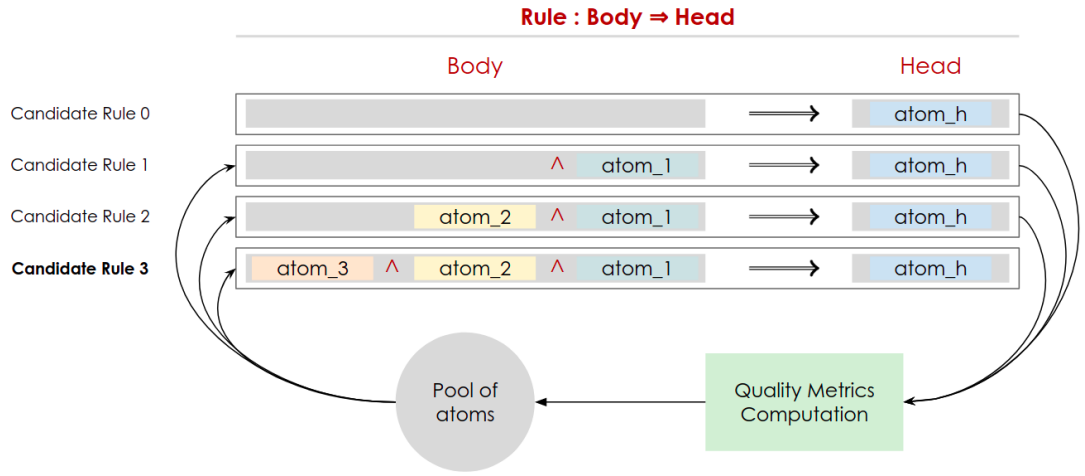

Figure 3.8: Simplified representation of AMIE [21]

rule is *closed* if every variable in the rule appears at least twice. A simplified representation of *AMIE* is presented in Figure 3.8.

The objective of *AMIE* is, given a head defined as a parameter, to find all association rules having this head on a knowledge graph. This task is not trivial as knowledge graphs can be incomplete, erroneous, and might have relations that are not functional, *i.e.*, with several objects given a subject and relation. Moreover, as seen in 2.1, a missing fact in a knowledge graph is not necessarily false. Therefore, metrics such as the confidence (equation 3.12) are not adapted to guide the mining of association rules. AMIE defined the *partial completeness assumption* (*PCA*), which differs from the *OWA* or *CWA* (see Chapter 2). This assumption states that if the knowledge graph owns at least one object for a given entity and relation, it knows all objects of this entity and relation. This assumption enables the computation of the *PCA-confidence* by considering negative examples that do not exist in a KG.

Given the language bias, *AMIE* is described as follows. Given a head defined by the user, *AMIE* iterates between three steps: it (i) considers a candidate rule, (ii) computes quality measures on this rule and (iii) refines the rule to generate new candidate rules. Figure 3.8 is used to illustrate these steps:

(i) The rule *Candidate Rule 0* with a head composed of *atom_h* and an empty body is initialised.

(ii) Several quality metrics are computed and compared to parameters given by a user. Among these metrics are the *support*, the *head coverage* and the *PCA confidence*. A rule is pruned if it does not verify the supports and the head coverage requirements.

(iii) If a rule verifies the requirements, it is extended by adding an atom to the body from the pool of atoms, *e.g.* *Candidate Rule 1* has one atom more than *Candidate Rule 0* with the atom *atom_1*. The choice of the atom is directed by language bias: it can be a dangling atom, an instantiated atom, or a closing atom. Although the figure only shows one rule created from *Candidate Rule 0*, many rules can be created by its extension.

A rule is extended until it no longer verifies the quality measures, *i.e.*, its metrics are below thresholds chosen by the user, or if the length of its *body* reached the maximum number of atoms indicated by the user. An association rule is added to the set of association rules if it verifies the quality measures and the rules extended from it do not verify them. An association rule is evaluated by computing the number of correct and incorrect predictions that can be made on unseen data.

3.2.2 Action Rules Mining and Counterfactual Explanations

Action rules [65, 85] are another type of rule mined over knowledge graphs. Such rules are part of counterfactual explanations. Counterfactuals explanations explain how to obtain an alternative decision from a machine learning model by identifying changes in input features that lead to a desired outcome from the model [66]. Counterfactual explanation approaches can be used for interpretability purposes to understand how a model makes predictions or for a user to know what input to use to obtain a certain outcome.

Action Rules are extensions of classification rules. Therefore, they express a different relation than association rules. An action rule can be presented as in equation 3.13, proposed by *Sýkora et al.*:

$$R : [(\omega) \wedge (\alpha \rightarrow \beta)] \Rightarrow [\phi \rightarrow \psi] \tag{3.13}$$

An action rule expresses that, with a set of attributes that do not change, denoted by ω, an *action*, *i.e.*, a change in a set of attributes denoted by $\alpha \rightarrow \beta$, will change the class of an instance, denoted by $\phi \rightarrow \psi$.

For instance, in the context of a house pricing classifier, an action rule can express that if the number of bathrooms has the value of 3, the class would have been high cost rather than low cost.

The quality of an action rule can be estimated by several metrics, *e.g.*, the support or the confidence. Other quality metrics may focus on the action by measuring how many attributes it changes, how different the values of the attributes are, and whether the changes are coherent or not.

Such rules can be mined through two categories of approaches [11]. The first category of approaches is named *rule-based* and mines action rules by mining association rules in the first step, then generates actions rules as a post-processing

step. The second category of approaches consists of directly discovering action rules from a database.

3.2.3 Discussion

Inductive knowledge represented as rules can be mined from databases and knowledge graphs. These rules represent patterns mined over the database. Among such rules, association rules are a well-known type of rule that indicates an association between atoms, while action rules are a part of counterfactual explanations and provide explanations for how to obtain a different value on an attribute. Section 3.1 emphasises on causal concepts, such as *treatment*, *effect*, or *outcome*, and frameworks on how to discover causal relations, such as the potential outcome framework. These concepts should be represented by rules enlightening a causal effect. However, association and action rules are not defined to express such concepts and are therefore not suited to display causal effects.

Association rule mining approaches aim to find a set of *bodies* for a given *head* to obtain rules to predict new facts in a knowledge graph. Firstly, association rules were not designed to display a causal effect but rather an association, which is different to causation. Indeed, as seen in 3.1, two variables that are not causal can be associated, for example, through a third variable causal to the two variables, or to randomness. Secondly, in such approaches, the head of the rules can be specified by the user and could express an outcome on two instances. However, the language bias does not allow to verify a treatment value for the two instances, neither does it ensures that the instances are similar. For instance, the head of a rule could indicate that two houses have different prices, *i.e.*, the outcome, but in its body one house could be described by its surface and the other by its location, *i.e.*, unrelated treatment. Furthermore, the mining of association rules involves meeting criteria such as *support*, *head coverage*, etc., but not any similarity between instances. Consequently, instances that are used to estimate the quality of the rule could be very different.

Action rules express that a change in a subset of attributes will change the class of an instance. While these rules express the idea of *intervention*, they are not suited for expressing causal relations. Besides, action rules are designed for classification problems and can not be used as such to explain a difference on a numerical property for two specified instances.

Existing rules in knowledge graphs are not suited for expressing a causal effect as they (i) do not explicit concepts as *treatment*, *effect* or *outcome* and (ii) are mined in a way that the relations they describe are hardly causal. Therefore, we will address these limitations and propose rule mining methods that adapt the potential outcome framework. In this framework, matching techniques can be used to estimate causal effects (see section 3.1.4). As matching consists on creating pairs

of similar instances, the following section focuses on various similarity measures that are defined on knowledge graphs.

3.3 Identity and Similarity of Instances in a Knowledge Graph

Knowledge graphs can be generated through various techniques, such as scrapping data from the web or combining existing knowledge graphs (see Chapter 2). As a result, knowledge graphs can own false triples and might be incomplete.

Therefore, identity management in knowledge graphs is an important research theme in the semantic web community. It consists of discovering instances in one or several knowledge graphs that refer to the same entity and has several applications, such as linking instances from different knowledge graphs or deleting false identity links [62]. One common output of such approaches is the generation of more or less strict identity links between instances. These identity links are of great interest in initiatives such as the *Linked Open Data* (*LOD*).

From a less strict perspective, another research theme focuses on similarity, *i.e.*, handling similar instances that do not refer to the same entity. Reasoning on similar instances has different applications, such as knowledge graph completion [18].

As we aim to adapt the potential outcome framework on knowledge graphs, this section emphasises on identity and similarity measures between instances of a knowledge graph. Section 3.3.1 describes identity and similarity links that exist between instances, section 3.3.2 focuses on various approaches aiming to get similar instances, and section 3.3.3 describes knowledge graph embeddings models and how they can be used to measure the similarity between instances.

3.3.1 Identity and Similarity Predicates

Besides predicates describing instances in a knowledge graph, represented as directed edges to other instances or literals, other predicates express the degree of identity or similarity between two instances of the same knowledge graph or different knowledge graphs as for data linking approaches.

Among the identity links, the *owl:sameAs* [59] relation is a widely used relation that states that two instances refer to the same real-world object. However, despite its wide adoption, it has been shown that its use led to erroneous matches [26, 30, 64]; where instances are not described by the same property values.

Less strict approaches were therefore proposed as an alternative to *sameAs*. For instance, instances can be considered identical within a context [4, 63]. In another work, *Halpin et al.* [26] proposed an ontology, *i.e.*, the similarity ontology,

where relations are defined to indicate similarity links or possible identities between instances. However, such links can be interpreted differently depending on the context and the use and should therefore be handled carefully.

3.3.2 Mining Similar Instances

Identity and similarity predicates may be expressed in some knowledge graphs. However, such predicates are often missing and approaches were defined to discover instances that refer to the same entity or, more generally, to similar instances. Discovering similar instances of a knowledge graph may be used for various tasks, such as query relaxation, graph completion, or taxonomy mining. Approaches measuring the similarity and identity between two instances may be separated into two families depending on whether the similarity output is numeric or symbolic.

Approaches that define numerical similarities that can be used on knowledge graphs may exploit instances RDF descriptions, such as in [33, 15], or vector representations learned through embedding models. When considering RDF descriptions, RIBL [33] defines a distance based on edit distances between two graphs. In another approach [15], authors propose a similarity between Horn clauses based on (un-)shared predicates and arguments. Embedding models learn vector representation of instances, and the similarity between two instances is computed as the distance between their embedding vectors. These models are briefly presented in the following section (section 3.3.3).

A similarity between instances of a knowledge graph can also be represented with a symbolic measure. For instance, a similarity can be a relaxed description [36, 17], or a conceptual distance [16, 18]. Both of these similarities rely on queries, either by discovering proper relaxed queries [17], or by analysing the intension of a graph concept [16, 18]. While a numerical similarity is hardly interpretable, these symbolic measures are much easier to understand, with the queries explaining why two instances are similar.

3.3.3 Similarity between Embedding Vectors

This section provides a comprehensive introduction to knowledge graph embeddings. We first introduce what knowledge graph embedding models are and then how they can be used to mine similar instances.

Knowledge Graph Embeddings

Knowledge graph embeddings (KGE) have become a popular research domain. Since Bordes *et al.* proposed *TransE* in 2013 [8], many approaches were developed. An overview of KGE approaches and applications has been proposed by Wang *et al.*

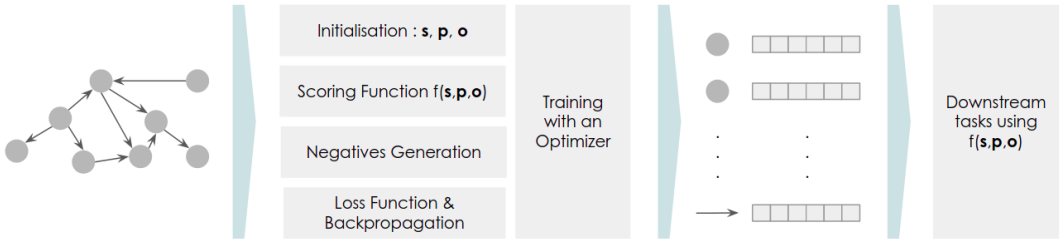

Figure 3.9: Simplified overview of a knowledge graph embedding model

[92]. KGE are part of representation learning, which aims to learn a representation of data to make it easier to extract useful information when building classifiers or other predictors [5]. If focusing on graphs, representation learning has several frameworks, such as Node Representation, with approaches such as *node2vec* [23], or Graph Neural Networks, with approaches such as *graph attention networks* [89]. An overview of representation learning on graphs has been proposed by Hamilton *et al.* [27]. As this thesis focuses on knowledge graphs, this section shall describe KGE only.

A summary of how knowledge graph embedding models are defined, trained and evaluated is provided. Deeper comprehensive work has been realised and can be found in the literature [92, 9]. KGE aim to embed entities and relations of a knowledge graph into low-dimension vectors while preserving the structure of the knowledge graph. To this end, an embedding technique consists of three steps: (i) representing the entities and relations into an embedding space, (ii) defining a scoring function and (iii) learning the representations while minimising a loss function. A simplified overview of embeddings is proposed in Figure 3.9. We propose to study *TransE* [8] as an example.

In the first step (i), entities and relations are defined in an embedding space. For entities, this space can be the vector space but also multivariate Gaussian distributions. Relations usually are operations such as vectors, tensors or mixtures of Gaussians. In *TransE*, as shown in Figure 3.10, both entities, denoted by **s** for the subject and **o** for the object, and relations, denoted by **p**, are defined as vectors in the same vector space.

In the second step (ii), a scoring function is defined to measure the plausibility of a fact. Consequently, facts defined in a knowledge graph should have higher scores than unobserved or false facts. The scoring function of *TransE* is presented in equation 3.14. Intuitively, **p** is interpreted as a translation that connects **s** and **o**, and the score should be high if the fact holds. As its name suggests, *TransE* is part of *Translational Distance Models*, where other models differ to *TransE* with relations represented as hyperplane [94], or distinct spaces for entities and relations [46], or by more or less relaxing the translational requirement. Other

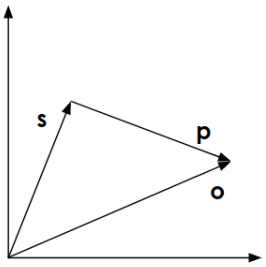

Figure 3.10: TransE Entity and Relation Space [8]

kinds of models exist, such as *Semantic Matching Models* [55, 95], that focus on latent semantics of entities and relations, or deeper models [13, 54].

$$f_{TransE} = -\|(s+p) - o\|_{1/2} \tag{3.14}$$

In the last step (iii), the representation of entities and relations is learned by solving an optimisation problem that maximises the plausibility of observed facts. However, this plausibility should be maximised compared to the plausibility of false facts, which should be minimised. As there are no false or negative facts in a knowledge graph, the training procedure implies generating negative facts using the local closed world assumption (LCWA). With $\mathcal{I}$ the set of individuals, the set of negative facts $\mathcal{C}$ is described in equation 3.15.

$$\mathcal{C} = \{(\hat{s}, p, o) | \hat{s} \in \mathcal{I}\} \cup \{(s, p, \hat{o}) | \hat{o} \in \mathcal{I}\} \tag{3.15}$$

The LCWA states that a knowledge graph is locally complete. Therefore, given a fact, facts considered *negative* can be generated by changing the subject or the object of the fact while the predicate stays unaltered. Given the scoring function and the sets of positive and negative facts, an embedding model can be trained by minimising a loss function $\mathcal{L}$. One example of a loss function, the pairwise margin-based hinge loss, is presented in equation 3.16, where $f(t^-;\theta)$ is the score of a negative fact and $f(t^+;\theta)$ the score of a true fact. While other loss functions exist, such as the binary cross-entropy, all of them aim to maximise the plausibility of a true fact and minimise the plausibility of a negative fact.

$$\mathcal{L}(\theta) = \sum_{t^+ \in \mathcal{F}} \sum_{t^- \in \mathcal{C}} max(0, [\gamma + f(t^-;\theta) - f(t^+;\theta)]) \tag{3.16}$$

Several protocols exist to evaluate the performance of KGE models. Among them, the triple classification and the entity ranking protocols are the most common [93]. In the triple classification protocol, positive and negative facts are classified as correct or false, and the accuracy is used to evaluate the performance

of models. Entity ranking has a different objective, *i.e.*, ranking facts given their score, and therefore has different performance metrics. Given the set of individual rank scores $\mathcal{R}$, this protocol provides several metrics: the *Mean Rank* 3.17, the *Mean Reciprocal Rank* 3.18 and the *Hits@n* 3.19.

$$MR = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} r \tag{3.17}$$

$$MRR = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \frac{1}{r} \tag{3.18}$$

$$Hits@n = \frac{|\{r \in \mathcal{R} | r \leq n\}|}{|\mathcal{R}|} \tag{3.19}$$

The mean rank (MR) computes the mean of all individual ranks of $\mathcal{R}$. It is easily interpretable but sensitive to outliers, *i.e.*, it can be strongly affected by rare high scores. The mean reciprocal rank (MRR) is less sensitive to outliers by computing the mean over all inverse individual ranks of $\mathcal{R}$. The Hits@n computes the number of facts with a rank lower or equal to a value n. Hits@n at several n values are commonly computed as Hits@1, Hits@3 and Hits@10.

KGE are used for various downstream tasks such as link prediction, triple classification and entity resolution. In addition to these tasks, they can also be used for more specific tasks such as taxonomy mining [47] or explaining clusters with association rules [19].

Embeddings Vectors Similarity

In addition to their use within downstream tasks, the representation of entities and relations learned through a KGE can be used to mine similar entities. If the KGE embody the semantic features of entities, similar RDF instances will have similar vectors in the embedding space [91], and similar vectors in the embedding space will represent similar RDF instances [50, 39].

3.3.4 Discussion

The potential outcome framework, particularly matching approaches, states that similar instances should be compared to estimate a causal effect. In the case of matching, the objective of the user is to find the closest matches of instances within a dataset, with the instances composing the matches having different treatment values for its effect to be studied. Consequently, selected pairs of instances should not be identical but rather similar, as they should differ in the treatment value. Therefore, the *owl:sameAs* [59] predicate can not be used, nor methods assessing whether two instances refer to the same entity.

On the other hand, approaches assessing the degree of similarity between instances seem more adapted to the problem of creating matches of similar instances. Still, all approaches focusing on similarity might not be adapted. As the estimation of a causal effect is sensitive to the matches used for its computation [83, 42], explanations of their creation should be provided. Consequently, numerical similarity distances such as [33, 15] are not suited for the problem of creating matches in the potential outcome framework, as they are sensitive to missing data and do not provide explanations for their values. Knowledge graph embeddings as such are not suited for this problem either. While being (i) less sensitive to missing data and (ii) able to create a ranked list of the most similar pairs using a distance in the embedding space, they, however, do not ensure that the created pairs are similar enough for this problem.

Symbolic similarities [17, 18] seem well-suited to create the matches for estimating a causal effect as they are interpretable and give an explanation as to why two instances can be considered similar. However, they may present two challenges. The first is that they are sensitive to missing data. Therefore, instances without a description of some properties may not be used to create matches. The second challenge is that their use might not be trivial with knowledge graphs where the descriptions of instances are complex. In such a knowledge graph, finding similar pairs that only differ on a few properties is impossible, in the same way as it is impossible to find exact matches in tabular data when there are more than a few columns describing instances [1]. Therefore, using such approaches would lead to obtaining pairs where even the most similar ones would differ on a large set of properties. Ranking the similarity of these pairs would be difficult, as it would rely on much expert knowledge to assess which differences are more important.

Chapter 4

Coarsened Matching for Differential Causal Rules Mining

In this chapter, we will present our first contributions to differential causal rules discovery in knowledge graphs. They are based on the potential outcome framework and coarsened matching of instances for causal discovery. These approaches are the objects of the following publications:

1. **Lucas Simonne**, Nathalie Pernelle, and Fatiha Saïs. Fouille de règles différentielles causales dans les graphes de connaissances. *Revue des Nouvelles Technologies de l'Information*, Extraction et Gestion des Connaissances, RNTI-E-37 :293–300, 2021. [81]

2. **Lucas Simonne**, Nathalie Pernelle, Fatiha Saïs, and Rallou Thomopoulos. Differential causal rules mining in knowledge graphs. In *Proceedings of the 11th on Knowledge Capture Conference*, K-CAP '21, page 105–112, New York, NY, USA, 2021. Association for Computing Machinery. [78]

Patterns in knowledge graphs are often represented as rules, such as association and action rules (see section 3.2). However, to our knowledge, no rules expressing a causal effect in knowledge graphs are defined in the literature. Therefore, this chapter defines how causal effects can be represented as rules that we name *differential causal rules*. The causal effect of a treatment can be observed on all instances of a dataset, in which case it has a *general* effect. Still, it can also have effects on subsets of instances only, in which case it has a *local* effect. To define expressive rules, differential causal rules should characterise the set of instances on which a treatment has an effect, whether local or general.

Causal effects can be discovered through several frameworks (see section 3.1) and, in particular, the potential outcome framework and matching techniques. Matching instances of a knowledge graph is a challenging task due to the open-world assumption that is followed in knowledge graphs. To facilitate this task, in addition to defining differential causal rules, this chapter presents an approach that generates abstract properties that summarises part of instances description. Instances can be matched on abstract properties instead of their initial description. The contributions of this chapter are the following:

- The definition of *differential causal rules* that represent the effect of a treatment on a subset of instances of a knowledge graph characterised by a graph pattern named strata.
- The generation of abstract properties based on a community detection algorithm summarising part of instances description.
- An algorithm that mines differential causal rules with a pessimistic heuristic to minimise the number of *false* rules.

The chapter is structured as follows. **Section 4.1** defines differential causal rules and strata, which is a basic graph pattern representing a set of instances.

Section 4.2 presents how the potential outcome framework and matching are adapted to mine differential causal rules in knowledge graphs. More particularly, it expresses the limits of exact instances matching [81] and shows how abstract properties can be generated [78] and used to match instances. Finally, it presents *Dicare-S* and *Dicare-C*, the algorithms that mine differential causal rules in knowledge graphs.

Section 4.3 describes the experiments done with *Dicare-S* and *Dicare-C* and the datasets it was applied on.

Section 4.4 provides a quantitative and qualitative assessment of the rules that were mined. Moreover, it discusses the benefits and limits of *Dicare-S* and *Dicare-C* and compares the semantics of differential causal rules to association rules.

Section 4.5 concludes on the approaches and proposes a set of criteria that future approaches should focus on.

4.1 Differential Causal Rule

This section defines the different terms that compose a *differential causal rule*, that express a causal effect of a treatment on instances of a knowledge graph. Moreover, as effects can vary depending on the set of instances considered, it also

motivates the definition of a *strata*, *i.e.*, a basic graph pattern that specifies the instances on which an effect is observed.

4.1.1 Motivation: Local Effects and Simpson's Paradox

A treatment may have heterogeneous effects depending on the set of units it is applied to. For instance, a drug developed given a disease may only cure women and not men or only young people. In addition, analysing a treatment effect is not trivial as an effect may be subject to paradoxes, therefore expressing a wrong effect. A well-studied paradox is the *Simpson's paradox*, presented in Figure 4.1, where an observed effect might have opposing signs depending on the considered set of variables describing instances. In this example, it is observed that higher X values lead to higher Y values if studying the whole set of instances, as shown with the black line of the figure. However, if the evolution of Y given X is done by considering whether the instances are from subgroup A, represented by triangles, or subgroup B, represented by the plus signs, *e.g.*, men or women, then higher X values lead to smaller Y values. Therefore, the effect has an opposing sign depending on whether the whole set of instances is studied as one or if this set is studied separately by adding a variable.

Several metrics exist to measure a causal effect (see section 3.1.4) that can represent average or local effects. In this chapter, we aim to mine rules that express a causal effect on instances where this effect has been verified only. By observing local effects, we aim to mine rules with local effects that do not present paradoxes. Such specific rules can then be generalised if the effect of a treatment is observed on different subsets of instances. For instance, we shall specify that the effect of a drug is only valid on women rather than expressing an average effect of all instances. Given Figure 4.1, we shall consider that X has a negative effect on instances from both A and B, and, therefore, a negative effect on all instances.

4.1.2 Strata Definition

To describe the set of instances on which a local effect is valid, we define a *strata* as follows.

Definition 2. ***Strata.*** *Let X be a variable that can be instantiated by individuals of the target class C, a strata $ST_i(X)$ is a conjunction of predicates that corresponds to a basic graph pattern in RDF, rooted by X, and such that all the leaves represent literals or class IRIs. i is an index that indicates the basic graph pattern a strata corresponds to.*

The depth of the graph pattern defining a strata or a treatment can be bounded by a given threshold d corresponding to the longest path, one can reach from

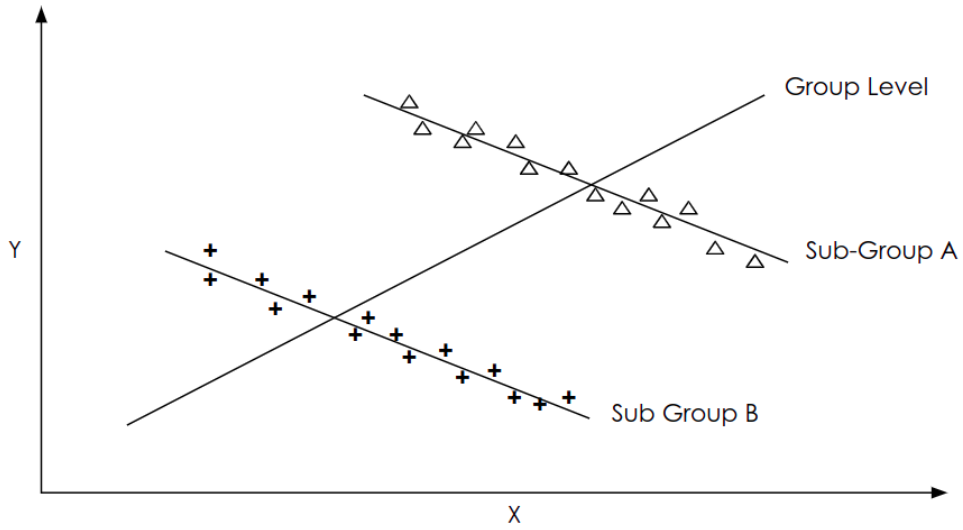

Figure 4.1: Illustration of Simpson's Paradox

X in the graph pattern without cycles. For example, the target class $Writer$, $Novelist(X)$ is a valid strata (*i.e.*, $Novelist \subseteq Writer$). A more specific strata can be: $Novelist(X) \wedge nationality(X, \text{“}France\text{“})$.

4.1.3 Differential Causal Rule Definition

Given the strata definition, we introduce the other components of a differential causal rule, the *treatment* and the *outcome*. As a reminder, we consider a knowledge graph as follows:

Definition 3. ***Knowledge Graph.*** *A knowledge graph KG is defined by a pair $(\mathcal{O}, \mathcal{F})$ where $\mathcal{O}$ is an ontology represented in OWL and $\mathcal{F}$ is a set of RDF triples describing class instances of $\mathcal{O}$.*

Given a knowledge graph, we consider property paths P_T in the form $P_T : p_1(X, Y_1) \wedge p_2(Y_1, Y_2) \wedge \ldots \wedge p_n(Y_{n-1}, Y_n)$. The properties at the end of these paths can have categorical or numerical values and can be single-valued, *i.e.*, functional, or multi-valued. For the sake of understanding, we consider the value(s) of a property path as referring to the value(s) of the property at the end of the path.

We consider that a differential causal rule is applied to two instances (X_1, X_2) of a class of $\mathcal{O}$ and that a treatment T represents a difference in P_T values between X_1 and X_2.

We distinguish two types of treatments: a categorical treatment T_c involving a path of properties whose end is a categorical property,*e.g.*, literal or hierarchical

values, and a numerical treatment T_n involving a path of properties whose end is a numerical property, *e.g.*, integer or real values.

Definition 4. ***Categorical treatment.*** *Let X_1 and X_2 be two instances of a class, a property path P_T and two sets of values V_1 and V_2 of path P_T for X_1 and X_2 respectively. A categorical treatment T_c is defined by:*

$$T_c(X_1, X_2) : P_T(X_1, V_{1,P_T}) \wedge P_T(X_2, V_{2,P_T}) \wedge belongs(v_1, V_{1,P_T}) \wedge belongs(v_2, V_{2,P_T}) \wedge \neg belongs(v_1, V_{2,P_T}) \wedge \neg belongs(v_2, V_{1,P_T}),$$

where $belongs(v, V)$ is a function that verifies that the value v belongs to the set of values V, and $\neg belong$ is used to check that v only belongs to the set of values of one of the two instances.

Definition 5. ***Numerical treatment.*** *Let X_1 and X_2 be two instances of a class, a property path P_T and two sets of values V_{1,P_T} and V_{2,P_T} of path P_T for X_1 and X_2 respectively. A numerical treatment T_n is defined by:*

$$T_n(X_1, X_2) : P_T(X_1, V_{1,P_T}) \wedge P_T(X_2, V_{2,P_T}) \wedge compare_{T_n}(s(V_{1,P_T}), s(V_{2,P_T})),$$

where $compare_{T_n}$ is a function comparing numerical values that can be implemented by $lessThan$ or $greaterThan$ and s is an aggregation function that can be max, min, sum, etc.

Similarly, we consider for the outcome a property path P_O such as $P_O : p_1(X, Z_1) \wedge p_2(Z_1, Z_2) \wedge \ldots \wedge p_m(Z_{m-1}, Z_m)$. For the outcome, we only consider property paths leading to numerical values.

Definition 6. ***Outcome.*** *Let X_1 and X_2 be two instances of a class, a property path P_O, two sets of numerical values U_{1,P_O} and U_{2,P_O} of path P_O for X_1 and X_2 respectively, and s an aggregation function. The outcome O is defined by:*

$$O(X_1, X_2) : P_O(X_1, U_{1,P_O}) \wedge P_O(X_2, U_{2,P_O}) \wedge lessThan(s(U_{1,P_O}), s(U_{2,P_O})),$$

where $lessThan$ is fixed to avoid the creation of equivalent rules due to the permutation of X_1 and X_2.

Example 1. *A categorical treatment T_c could be that two athletes have different manualities: one is right-handed, and the second is left-handed. A numerical treatment T_n could express that the club's budget for one athlete is higher than that of another. An outcome O could be that one athlete has a better ranking than another.*

A differential causal rule (DCR) represents a causal relationship between a treatment T and an outcome O. It expresses that T, *i.e.* a difference of values on a property path P_T, explains O, *i.e.* a difference of values on a property path P_O. As an explanation is to happen before what it explains, P_T is to describe a feature that precedes in times the feature described by P_O.

Definition 7. ***Differential Causal Rule.*** *Given X_1 and X_2 two instances of a target class of $\mathcal{O}$ that belong to the strata ST_i, the property path P_O leading to the result, a treatment $T \in \{T_n(X_1, X_2), T_c(X_1, X_2)$ defined by the property path P_T, and s an aggregation function, a differential causal rule DCR_T is defined as follows:*

$$DCR_T : T \wedge ST_i(X_1) \wedge ST_i(X_2) \wedge P_O(X_1, U_{1,P_O}) \wedge P_O(X_2, U_{2,P_O}) \Rightarrow lessThan(s(U_{1,P_O}), s(U_{2,P_O}))$$

It is to note that the outcome is expressed partly in the body of the rule and partly in its head. We call a rule involving a numerical (resp. categorical) treatment a numerical (resp. categorical) differential causal rule.

Example 2. *Given example 1, a categorical DCR could be that, for athletes playing tennis, being right-handed compared to left-handed explains a better rank. In this example, the strata defines the sports concerned by the rule. A numerical DCR could indicate that, for runners, being younger explains faster running times.*

4.2 Approaches: *Dicare-S* and *Dicare-C*

This section describes how the potential outcome framework, particularly the matching task, is adapted to mine differential causal rules from knowledge graphs. We present two approaches, *Dicare-S* [81] and *Dicare-C* [78], where *Dicare-C* is an extension of *Dicare-S*.

4.2.1 Adapting the Potential Outcome Framework

The matching task consists of creating pairs of similar instances that differ on a treatment value to study the effect of a treatment (see 3.1.4). The matching step can be seen as a pre-processing step that prunes instances that are too different to others in a dataset to balance the covariate distribution between the treated and control sets. We aim to match instances using their RDF description as we deal with knowledge graphs.

To compare instance descriptions, we investigated two different techniques of matching: exact instance matching, applied as a baseline, and coarsened matching used in *Dicare-S* and *Dicare-C*.

Exact Instances Matching

Exact matching consists of matching two instances if they share the same description on a set of paths. In other words, if all paths describing two instances, apart

from the paths leading to the treatment and the outcome, lead to the same values, then the instances can be matched.

However, it has been shown that, in practice, the probability of constructing an exact match decreases with the number of attributes describing the instances [1], to the point that it becomes impossible if considering a big number of attributes. Therefore, the set of attributes on which instances should have the same values is to be restricted to a set of *main* attributes defined by the user.

Regarding the open nature of knowledge graphs, with the existence of functional and not functional properties and missing values, exact matching on property paths would be even more complex and also unfeasible. Exact matching could only be applied if a knowledge graph describes instances with a small set of paths.

As a consequence of the constraints imposed by exact matching, alternatives were proposed.

Coarsened Matching

Iacus *et al.* proposed an alternative to exact matching named *coarsened matching* [37], which basis were already defined by Althauser and Rubin [1]. The principle of coarsened matching is to coarsen each variable *so that substantively indistinguishable values are grouped* [37]. Therefore, while two slightly different values of one variable would be considered different in exact matching, they could be considered identical in coarsened matching. Not only can coarsened matching be used for causal analysis, but it can also reduce bias.

The choice of coarsening is related to the application domain studied within a database. Given a numerical variable, a categorical variable, or ordered values, the coarsened values should be customised based on domain knowledge.

We used coarsened matching in *Dicare-S* [81] and *Dicare-C* [78]. More precisely, while *Dicare-S* is coarsening paths values of a knowledge graph, *Dicare-C* [78] further extends *Dicare-S* by guiding the coarsening step by generating abstract properties where instances should be matched. These abstract properties have semantic meaning and are determined upon a community detection step. They represent a summary of a set of paths and shall replace parts of the initial RDF description of instances, easing the matching step. We present how they are generated in the next section.

4.2.2 Abstract Properties Generation

This section defines abstract properties and how they are generated in *Dicare-C*. These abstract properties ease the matching step between instances by comparing their values on the abstract properties instead of their RDF description.

Definition 8. ***Abstract Property.*** *An abstract property semantically summarises a part of an instance description in a knowledge graph. It is represented as a new path linking an instance of a target class to a community of valued property paths.*

For example, given a target class *Person*, a community might gather property paths such as the place where a person works, which is an IRI, and its way of commute, *e.g.*, by bus or car, which is a literal. Such paths can be of different lengths and lead to IRIs or literal values. The community of valued property paths might be written as $\{commute(X,'bike'), worksAt(X,Y) \wedge city(Y,Paris)\}$, and an abstract property may link an instance of the target class to this community.

Abstract Property Generation

The abstract properties are generated in three main steps: (i) the computation of the similarity between two valued property paths based on co-occurrence, (ii) the detection of communities of valued property paths, and finally, (iii) the generation of abstract properties.

Similarity between valued property paths. In the first step, the similarity between valued property paths is computed. This step is realised on sets of paths indicated by a domain expert that should share semantic proximity to discover meaningful communities. The similarity of two valued property paths is defined as their co-occurrence.

Definition 9. ***Co-occurrence.*** *The co-occurrence between two paths P_i and P_j represents the number of instances having both P_j and P_i in their description, divided by the number of instances having P_i. With D_k the description of an instance k of the target class, i.e., the set of valued paths rooted by k, the co-occurrence is defined as follows:*

$$co - occ(P_i, P_j) = \frac{\#\{k|(P_i \in D_k) \wedge (P_j \in D_k)\}}{\#\{k|P_i \in D_k\}} \tag{4.1}$$

The co-occurrence is asymmetric as only P_i appears in the denominator. Therefore, it is computed for each (P_i, P_j) and (P_j, P_i). Given a set of valued property paths where the community discovery applies, the co-occurrence is computed for each pair of paths. It results in a similarity matrix that can be seen as a directed weighted graph, named *co-occurrence graph*, where each node represents a valued property path P_i, and where each edge from a node P_i to a node P_j is weighted with the similarity $co - occ(P_i, P_j)$.

Example 3. *Let P_1 be the path that indicates if a person watches football. Let P_2 the gender of a person. If 90% of people watching football are men, $co-occ(P_1, P_2)$*

would be equal to 0.9. If 90% of men are watching football, $co - occ(P_2, P_1)$ *would also be equal to 0.9. However, it is less likely than the previous statement. As a result,* $co - occ(P_2, P_1)$ *is likely to be lower than 0.9.*

Communities of valued property paths. The second step consists of detecting communities on the co-occurrence graph. The community detection step is represented in Figure 4.2, where it is shown that communities are discovered from the co-occurrence graph. In this figure, the bold arrows indicate higher weights than the dashed arrows. For a sake of simplicity, only one weight is represented between two edges, however, as shown in example 3, the weights are not necessarily symmetric. This task is complex as the graph's matrix is unsymmetrical. Therefore, spectral clustering algorithms such as Louvain's algorithm [7] can not be used. We have chosen to apply *OSLOM* [2] since it is suitable for detecting communities in directed graphs. As output, *OSLOM* may assign all of the valued property paths into communities. However, some may be isolated without being part of any community. Such valued property paths are interpreted as independent of communities. If no communities are detected, we assume the paths to be unsuitable for explainability as their distribution can be considered random.

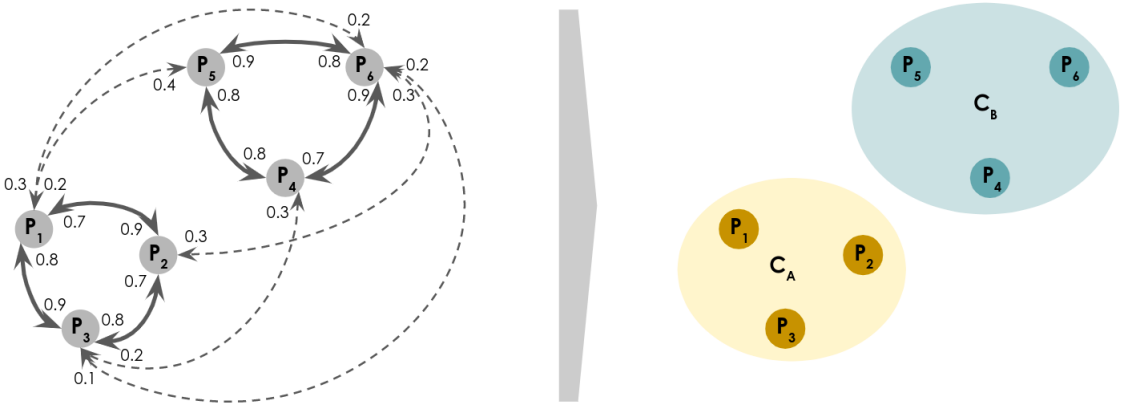

Figure 4.2: Getting communities of property paths from the co-occurrence graph

Generation of abstract properties. In the third step, abstract properties are generated to link instances of the target class to the detected community they are associated with. This step is illustrated in Figure 4.3. It is not trivial to decide whether one instance is associated with a community. Let i_1 and i_2 two instances of a target class and A and B two communities. i_1 has 1 paths related to A and 3 to B, while i_2 has 1 paths related to A and 0 to B. While both instances have the same absolute number of paths associated with A, their proportion of paths to

A (25% for i_1 and 100% for i_2) motivates the generation of an abstract property from i_2 to A only.

Given p_c the proportion of paths an instance of the target class has towards a community c, and T_c a threshold on the proportion given by the expert on the community c, an abstract property linking the instance to c is created if $p_c > T_c$. T_c is to be determined carefully as small (resp. high) values would lead to all (resp. none) instances of the target class having the abstract property in their description, making it not discriminatory and expressive enough to be studied as a treatment.

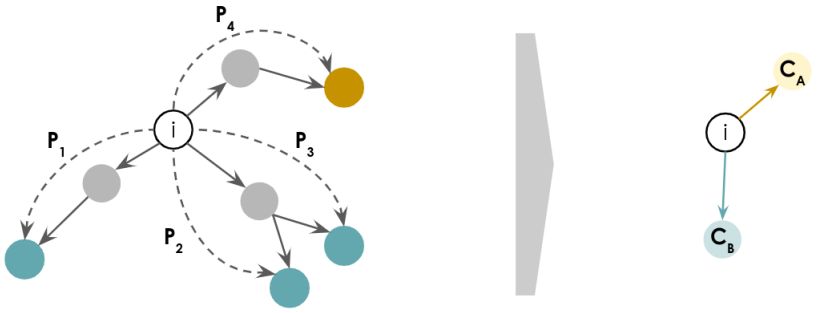

Figure 4.3: Updating the description of the instance i with abstract properties from communities

The abstract properties generation of *Dicare-C* relaxes the exact matching constraint. Indeed, instances having initial RDF descriptions that were different can be matched on newly generated abstract properties if their RDF descriptions were semantically related. Next, we present the algorithm that generates abstract properties.

Abstract Property Algorithm

The algorithm for generating abstract properties is presented in Algorithm 1. It takes as input the KG, the sets of paths $\mathcal{P}$ indicated by the expert where communities are studied, the target class t_c, and the inputs of *OSLOM*, *i.e.*, the number of iterations n, a boolean *isolated* indicating whether all nodes have to be part of a community or not, and a statistical parameter s related to the statistical significance of the discovered communities. High values of s will lead to strong communities, while low values of s lead to higher sensitivity and could detect communities that were not discovered with high values of s. The algorithm returns $KG_{enriched}$, which is the knowledge graph enriched with abstract properties.

$KG_{enriched}$ is created in the initialisation of the algorithm with the *removePaths* function. It is a copy of KG, except that all paths $\mathcal{P}$ related to the community

detection have been removed. After the initialisation, the algorithm iterates on each set of paths $\mathcal{P}_i \in \mathcal{P}$ and applies two steps on each $\mathcal{P}_i$. The first step discovers communities if they exist, while the second step creates abstract properties linking instances to the communities they are linked to. In the following, we present the two steps that are applied to each set of paths.

In the first step, an empty squared matrix of order m is created, where m is the number of paths of $\mathcal{P}_i$. The matrix is named *occmatrix* and is used to discover communities. For each combination of paths of $\mathcal{P}_i$, noted $(\mathcal{P}_{ij}, \mathcal{P}_{ik}) \in \mathcal{P}_i \times \mathcal{P}_i$, the value of co-occurrence is computed as defined in Definition 4.1. This computation is done with the *co-occ* function and has its value stored in *occmatrix*. *OSLOM* is applied to *occmatrix* once it has been filled. It uses the parameters n, *isolated* and s as previously described.

In the second step, abstract properties are generated. If communities are discovered for a set $\mathcal{P}_i$, thresholds T_i for generating the abstract properties are given by the user. In the algorithm, this is done with the *get-thresholds* function. More precisely, T_i contains a set of thresholds T_{i_c} for each community c of the communities that were discovered. Properties are generated by iterating on all instances of the target class t_c. For each instance x of t_c, the proportion of paths p_{x_c} to each community c is computed and compared to the threshold T_c of the community. If p_{x_c} is greater or equal to T_c, then an abstract property is generated from x to c and is added to the output $KG_{enriched}$. $KG_{enriched}$ is used for mining the differential causal rules as presented in the next section 4.2.3.

Algorithm 1 Community Detection and Abstract Properties

Input: KG, $\mathcal{P}$, t_c, *isolated*, s, n
Output: $KG_{enriched}$
$KG_{enriched} \leftarrow KG.removePaths(\mathcal{P})$
for $\mathcal{P}_i \in \mathcal{P}$ **do**
    $occmatrix \leftarrow$ matrix $| \mathcal{P}_i | \times | \mathcal{P}_i |$ ▷ **Part 1**: Discovering Communities
    for $(\mathcal{P}_{ij},\mathcal{P}_{ik}) \in \mathcal{P}_i \times \mathcal{P}_i$ **do**
        $occmatrix[\mathcal{P}_{ij}][\mathcal{P}_{ik}] \leftarrow$ *co-occ* $(\mathcal{P}_{ij},\mathcal{P}_{ik})$
    $comm \leftarrow OSLOM\ (occmatrix, n, isolated, s)$
    if $(comm \neq \emptyset)$ **then** ▷ **Part 2**: Generating Abstract Properties
        $T_i \leftarrow$ get-thresholds$(comm, \mathcal{P}_i)$
        for x in $instances(t_c)$ **do**
            for each community $c \in comm$ **do**
                if $p_{x_c} \geq T_c$ **then**
                    $KG_{enriched} \leftarrow KG_{enriched} \cup abstractProperty(x, c)$
Return $KG_{enriched}$

4.2.3 Differential Causal Rules Mining

This section presents the mining of differential causal rules. Differential causal rules are either mined after the pre-processing step of generating abstract properties for *Dicare-C*, or without a pre-processing step for *Dicare-S*. Apart from this pre-processing step, their mining follows the same procedure for both *Dicare-C* and *Dicare-S* and relies on the potential outcome framework. More precisely, the differential causal rules are mined in three steps. In the first step, potential differential causal rules are generated. In the second step, their effect on an outcome is measured to select the valid rules, *i.e.*, that have an effect on the outcome. Lastly, the third step is a post-processing of the valid rules that aim to facilitate their understanding. The three steps are described in the following sections.

Step 1: Potential Rules Generation

Potential rules are generated in a two steps approach. First, it creates a partition of the instances of the target class. As a result, it obtains clusters of instances where all instances of a cluster are described by the same strata. Then, it generates potential rules by doing a pair-wise comparison of the clusters to verify whether a treatment can be studied or not.

Parameters. The algorithm to generate the potential differential rules is presented in Algorithm 2. It takes as inputs the knowledge graph enriched with the abstract properties $KG_{enriched}$ for *Dicare-C*. For *Dicare-S*, $KG_{enriched}$ is the same knowledge graph as KG. Other parameters are the target class t_c, the property paths Q that will be queried to match instances to clusters, the parameters for the coarsening of paths co, a threshold on the minimum support of instances of a cluster t_{supp}, and a threshold on the maximum number of paths in the treatment of a rule t_{tr}. It is to note that Q is composed of property paths with a depth shorter or equal to d and that it includes the abstract properties previously generated.

Partitioning the instances. In the first step of the potential rules generation, described in lines 4 to 15 in Algorithm 2, instances of the target class t_c are partitioned into clusters that we name described clusters (DC). We provide a definition of described clusters and explain how they are obtained.

Definition 10. ***Described Clusters.*** *A described cluster $DC_i = (ST_i, I_i)$ associates a strata ST_i, which is a graph pattern, to the set of instances I_i of the target class t_c that has the strata ST_i in their description.*

A DC is obtained from an existing DC, apart from the initial described cluster $DC_0 = (ST_0, I_{t_c})$ that associates the empty strata ST_0 to all instances of t_c, noted I_{t_c}. Consequently, the partitioning is initialised with DC_0.

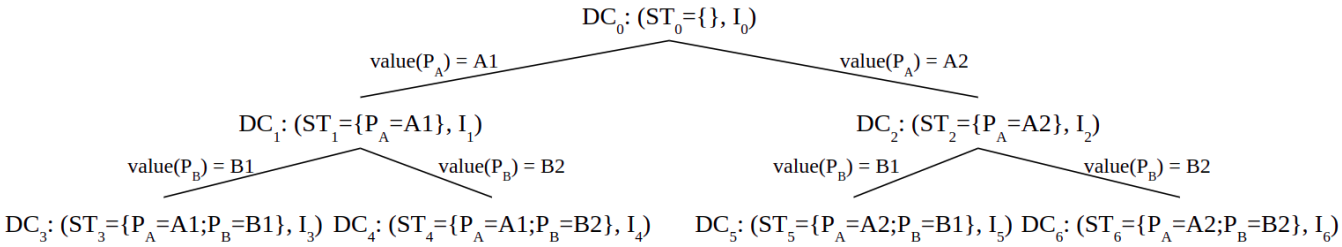

Figure 4.4: Partitioning instances into described clusters

The partitioning of the instances into DCs is done by iteratively querying all paths P that are in Q. For each path P, $co(P)$ is included in co and defines the set of interval values of P. These intervals are used to coarsen the path values. Therefore, with the function $coarsenedQuery$ that we define, instances with the same coarsened value co_{P_j} for path P are obtained. It is to note that the coarsening can be applied for both numerical or categorical values.

Given a described cluster $DC_i = (ST_i, I_i)$, one or several DCs more *specific* than DC_i are obtained by querying P on I_i. More precisely, with n the number of intervals of path P, n more specific DC are obtained from DC_i. Given an interval value co_{P_j} and the instances $I_j \subset I_i$ that have values in co_{P_j}, a more specific described cluster is $DC_j = (ST_j, I_j)$ where ST_l is the strata ST_i enriched with the value co_{P_j} on P.

All property paths of Q are queried until the most specific described clusters are obtained. It results in a partition of the instances. A described cluster DC_i is only queried if its number of instances $n(I_i)$ is higher than the minimum support of instances t_{supp}. We propose example 4 based in Figure 4.4 to illustrate the algorithm. As seen in the algorithm, we propose to use queues to facilitate the construction of described clusters. Using the queues allows to easily select the described clusters that can be extended.

Example 4. *In Figure 4.4, the algorithm is described with $Q = \{P_A, P_B\}$ where each property path in Q can lead to two values. As seen on the root of the proposed tree structure, the partitioning is initialised with DC_0. The nodes at a depth of one represent the DCs obtained after the query on P_A on DC_0, and the nodes at a depth of two represent the DCs obtained after the query on P_B on DC_1 and DC_2.*

Generating Potential Rules. In the tree structure that represents the partitioning of the target class instances into the described clusters, some described clusters are located at the maximum depth.

Definition 11. ***Maximally Specific Described Clusters.*** *A maximally specific described cluster MSC is a described cluster with a strata describing all paths from*

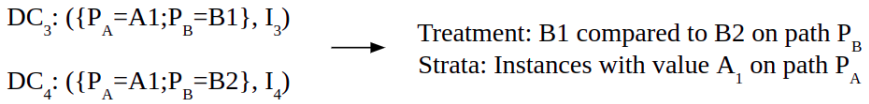

Figure 4.5: Generating potential rules from maximally specific described clusters

Q. They are the leaves of the partitioning tree with a depth equal to the size of Q.

The set of MSC is used to generate the potential rules. To this end, the strata of all pairs of MSC, *i.e.*, $(MSC_i, MSC_j) \in MSC \times MSC$, are compared. With ST_i the strata of MSC_i and ST_j the strata of MSC_j, ST_i and ST_j are compared on each path value. In algorithm 2, this comparison is done by the $getPathDifferences$ function. The number of paths where the strata have different values is obtained and compared to the threshold on the maximum number of paths in a treatment t_{tr}. If it is lower or equal than t_{tr}, then a potential rule $R_{i,j}$ is generated with as treatment the differences of values on the paths that were obtained and as strata the same strata as ST_i or ST_j without the treatment paths. In algorithm 2, the potential rule $R_{i,j}$ is generated by the *builtRule* function. The effect of $R_{i,j}$ is computed using I_i and I_j in the following step. We propose the example 5 to illustrate the generation of a potential rule.

Example 5. *In Figure 4.5, the MSC DC_3 and DC_4 are presented. Let $t_{tr} = 1$. The comparison of the strata of DC_3 and DC_4 shows that they have the same value on P_A but a different value on P_B. Therefore, they have one path with a different value, which makes it equal to t_{tr}. A potential rule R_1 can therefore be generated. The strata ST_1 of R_1 is a value $A1$ on P_A, and the treatment is a difference of values on P_B that is value $B1$ compared to value $B2$.*

Step 2: Causal Metric Computation for Rule Assessment

The quality of association rules (see section 3.2) is assessed by metrics such as confidence and support, and their mining relies on exploiting properties of these metrics, *e.g.*, the monotony of the support. The support of a rule R is the number of true predictions made by the rule, *i.e.*, the number of times it appears. The confidence of a rule R is the support of R divided by the total number of predictions made by R, *i.e.*, a relative value that is the support of the rule divided by the support of its body. However, these metrics alone are not suited to quantify a causal effect of a rule as they do not measure a treatment effect. For instance, a treatment could appear only a few times, meaning it will have low support, but could still have a substantial effect. Similarly, confidence could indicate that the body and the head of a rule are frequently associated. However, the measure of

Algorithm 2 Generating Potential Rules

Input: $KG_{enriched}$, t_c, Q, co, t_{supp}, t_{tr}
Output: $R_{potential}$
$R_{potential} = []$
$I_0 = instances(t_c)$
$DC_0 = (ST_0, I_0)$
$\mathcal{Q}_{DC_0} = queue(DC_0)$
for i in $range(0, len(Q))$ **do** ▷ **Part 1**: Partitioning the instances of t_c
  $P = Q[i]$
  $\mathcal{Q}_{DC_{i+1}} = queue()$
  while $\neg \mathcal{Q}_{DC_i}.isEmpty()$ **do**
    $DC_l = \mathcal{Q}_{DC_i}.dequeue()$
    for co_{P_j} in $co(P)$ **do**
      $I_j = coarsenedQuery(KG_{enriched}, I_l, P, co_{P_j})$
      if $size(I_j) \geq t_{support}$ **then**
        $DC_j = (ST_l \wedge P = co_{P_j}, I_j)$
        $\mathcal{Q}_{DC_{i+1}}.enqueue(DC_j)$
$MSC = \mathcal{Q}_{DC_{len(Q)+1}}$ ▷ **Part 2**: Generating potential rules
for $(MSC_i, MSC_j) \in MSC \times MSC$ **do**
  if $getPathDifferences(ST_i, ST_j) \leq t_{tr}$ **then**
    $R_{i,j} = builtRule(MSC_i, MSC_j)$
    $R_{potential}.add(R_{i,j})$
Return $R_{potential}$

a treatment effect should also consider the rule's confidence with the opposing head, as it could be higher and suggest an opposing effect of the treatment. More generally, the support and the confidence do not show notions of *comparison*, *e.g.*, as how one treatment performs compared to another. Therefore, another metric is to be used to assess the quality of a differential causal rule.

We propose to use a *causal ratio* inspired by the odds ratio [86] that explicitly shows the strength of the relation between the treatment and the outcome. The causal ratio differs from the odds ratio as it is defined for a knowledge graph and applied to pairs of instances that belong to a defined strata. We give the definition of the causal ratio.

Definition 12. ***Causal Ratio.*** *Let a differential causal rule R with a treatment T such as $R : T \wedge \vec{B} \Rightarrow lessThan(s(U_{1,P_O}), s(U_{2,P_O}))$ where $\vec{B} : ST_i(X_1) \wedge ST_i(X_2) \wedge P_O(X_1, U_{1,P_O}) \wedge P_O(X_2, U_{2,P_O})$. The metric causal ratio, noted $causal_r$, is the following:*

$$causal_r(R) = \frac{supp(T \wedge \vec{B} \Rightarrow lessThan(s(U_{1,P_O}), s(U_{2,P_O})))}{supp(T \wedge \vec{B} \Rightarrow greaterThan(s(U_{1,P_O}), s(U_{2,P_O}))} \tag{4.2}$$

$causal_r$ is the ratio between the support of a differential causal rule and the support of the same rule with the opposing effect. It is a symmetric value, where a value of 1 indicates that the treatment is independent of the outcome, as it equally leads to the outcome *lessThan* or the opposing outcome *greaterThan*. A value higher (resp. lower) than 1, however, means that the treatment has a positive (resp. negative) effect on the outcome, as pairs with the treatment will be more (resp. less) likely to have the outcome than the opposing outcome.

The treatment of numerical rules has been formalised with the $compare_{T_n}$ predicate that can either be *lessThan* or *greaterThan* (see Definition 5). Similarly, a categorical treatment may verify the presence of one value V_1 and the absence of a value V_2, or the inverse. In both cases, a treatment can therefore be studied in two ways. To ease the process of mining the rules, the treatments are initially studied in both ways, and the rule that leads to a positive effect, *i.e.*, a value of $causal_r$ higher than 1, is kept.

The last step to assess the effect of a rule that has a positive effect on the outcome consists in constructing a statistical test to evaluate whether $causal_r$ is significantly higher than 1. It is inspired by *Li* et al. [45] and considers a standard normal deviate $u_{1-\alpha/2}$ to compute a confidence interval with a (1-α) confidence as described in equation 4.3. With $a = supp(T \wedge \vec{B} \Rightarrow lessThan(s(U_{1,P_O}), s(U_{2,P_O}))$ and $b = supp(T \wedge \vec{B} \Rightarrow greaterThan(s(U_{1,P_O}), s(U_{2,P_O})))$:

$$[cr_-, cr_+] = exp(ln(causal_r) \pm u_{1-\alpha/2}\sqrt{\frac{1}{a} + \frac{1}{b}}) \tag{4.3}$$

$causal_r$ is considered significantly higher than 1 if $cr_- > 1$. This confidence interval is computed for all potential causal rules from $R_{potential}$ to select the rules where a treatment has an effect. It is to note that a support of zero is replaced by 1 to avoid a division by 0.

Example 6. *Let us consider two fiction writers who published their first books at ages 18 and 25 and three non-fiction writers who published their first books at ages 17, 31 and 32. The rule explaining that fiction writers publish their first book at a younger age than non-fiction writers would have a* $causal_r$ *value of 2, as four pairs verify the rule, and two pairs where the opposite outcome is observed.*

Step 3: Generalising Consistent Rules

The previous step selected the potential rules that have an effect on the treatment. The selected rules are said to be *specific*, as they built from maximally specific described clusters in the first step. In other words, the selected rules may have strata that are describing a large number of paths. While such strata makes a rule highly expressive by describing the instances on which a treatment has an effect, it may also make the rules less interpretable.

As a treatment effect may appear in different rules, we propose to merge the strata of the rules involving the same treatment to obtain more general rules that are easier to interpret. In addition to being easier to interpret, the number of general rules is also lower than the number of specific rules as the latter are merged. The merging procedure ensures the generation of *consistent* rules only, as defined in definition 13. This procedure is said to be resistant to paradoxes, as described in section 4.1.1, as it verifies local effects and then generalises them in creating more general rules. An example of resistance to paradoxes is presented in example 7.

Definition 13. ***Rule Consistency.*** *Let* $R_1 : S\vec{T}_1 \wedge \vec{T} \Rightarrow lessThan(s(U_1), s(U_2))$. *The rule* R_1 *is consistent iff* $\forall ST_2 \sqsubseteq ST_1, R_2 : S\vec{T}_2 \wedge \vec{T} \Rightarrow lessThan(s(U_1), s(U_2))$ *is also consistent or* ST_1 *is maximally specific and* $causal_r(R_1)$ *is significantly higher than 1. A strata* ST_1 *is said to be maximally specific iff there is no* $ST_2 \sqsubseteq ST_1$ *such that* $depth(ST_2) \leq d$, *where* d *is a parameter setting the maximal depth.*

Example 7. *Table 4.1 describes the continent of writers, the year they were born, and their age when they published their first book. The treatment implying that being born later explains an older age of the first publication is assessed by* $causal_r$. *If matching writers independently of their continent,* $causal_r$ *would equal 0.57. However, if matching writers from Europe only (resp. USA),* $causal_r$ *would equal 5.0 (resp. 4.0). Therefore, the process of generalising consistent rules would first*

Table 4.1: Simpson's Paradox: An Example

Continent	***Birth Date***	***Age First Book***
Europe	1943	35
Europe	1955	40
Europe	1957	39
Europe	1960	46
USA	1970	19
USA	1980	29
USA	1982	27
USA	1983	31
USA	1985	31

lead to the two specific rules that verify the treatment on continents, then to the rule implying that the treatment is valid on both continents.

The process of generalising consistent rules is performed in a recursive merging procedure to find more general rules from the specific rules previously mined.

To do so, the valid specific rules are initially placed in a list named the candidate's list. At each iteration, the first rule of the candidate's list is compared to the other rules of the list to detect rules it could be merged with. Two rules can be merged if they have the same treatment and the paths values of their strata are identical except for one path, *e.g.*, the strata $Genre(X, Fiction) \wedge Continent(X, Europe)$ and $Genre(X, Fiction) \wedge Continent(X, USA)$. We aim to suppress this path to generalise the strata, *e.g.* suppression of Continent. If two rules with the same treatment were to be valid for $Genre(X, NonFiction) \wedge Continent(X, Europe)$ and $Genre(X, Fiction) \wedge Continent(X, USA)$, they could not be merged as they differ on more than one path.

A merged rule is created if there does not exist a rule that uses the same strata with another value for the path to suppress. It will be composed of the treatment and common part of the strata expressed in the rules it is created from. In our example, if the rule $Genre(X, Fiction) \wedge Continent(X, Asia)$ is not valid, the generalised strata $Genre(X, Fiction)$ can not be generated.

The resulting merged rule is then added to the candidate rules. If no rules to be merged with are found, then the rule is at the more general description it can be. In this case, it is removed from the candidate's list and outputted in the set of final differential causal rules. The merging procedure stops when the candidate's list is empty. An example of how rules can be merged is provided in example 7.

4.3 Experiments

The objectives of the experiments are threefold. First, they aim to quantitatively evaluate the results of *Dicare-S* and *Dicare-C* in terms of the number of differential causal rules that can be discovered and the number of outcome differences they can explain. The second objective is to analyse the plausibility and interpretability of the discovered rules qualitatively. This analysis is realised by collaborating with domain experts. Lastly, we aim to compare the discovered rules with both the state-of-the-art approach [51] and association rules.

4.3.1 Datasets Presentation and Settings

Two datasets were used for the experimental evaluation of our approach: *DBpediaW* and *Vitamin*. Some statistics of the datasets are presented in Table 4.2. The datasets can be downloaded at this link[1].

Table 4.2: Datasets Description

	DBpediaW	***Vitamin***
# Facts	6908	86006
# Classes	4	19
# instances t_c	185	1714
# Properties	8	22
# Execution Time (s)	< 1	15.1
# Rules	12	77

DBpediaW

DBpediaW has been built by *Munch* et al. [51]. It describes authors of books and has been obtained by querying *DBpedia* [3]. Its schema is presented in Figure 4.6.

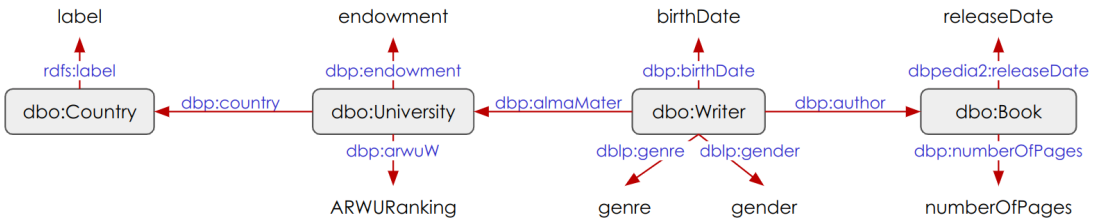

Figure 4.6: Schema of *DBpediaW*

[1]https://github.com/LucasSimonne/kcap2021

The class *Writer* is the main class of *DBpediaW*. It is described by the writers' birth date, genre and gender. This knowledge graph also describes books published by the authors, with their release date and the number of pages. Lastly, it also describes the universities the authors attended, describing their countries and ranking, denoted by *arwuW*. As *DBpediaW* was constructed by querying *DBpedia*, its properties were originally in *DBpedia* and are documented on the DBpedia web-page[2].

The studied outcome path is the age of an author at the time he published his first book. As it is a single value, no aggregation function s is used for the outcome. As a result, the mined rules aim to explain why some authors published their first book at a younger age than others. The discretization has been done similarly as in Munch *et al.* [51] to facilitate the comparison of both approaches.

As *DBpediaW* is a knowledge graph describing a domain relatively simple to understand, the results obtained from *Dicare-S* and *Dicare-C* on it are analysed by people that are not experts of the book publication domain. The evaluation has been realised by researchers in computer science from Université Paris-Saclay.

Vitamin

Vitamin is the second dataset used for the experiment. It is bigger than *DBpediaW* and has a more complex schema described in Figure 4.7. It extensively describes people and their eating habits.

Vitamin has been constructed with data obtained through an online survey by researchers from INRAE[3] [88]. The definition of the schema has been built in collaboration with an INRAE researcher involved in the survey design. The survey is composed of 33 questions, where each question can be answered by one or several answers that can either be numerical or categorical values. The questions cover a wide set of categories that includes criteria in food choice, current and ideal food diet, food preferences, reasons, hindrances and facilitating factors for reducing animal product consumption, socio-demographic information, and agreement with arguments on animal product consumption. The schema has been built to gather information on questions of the same categories under the same or similar concepts. As there were, up to our knowledge, no properties for describing this domain, we also constructed the properties of the schema. Additional information on the survey is available [88, 73].

The studied outcome path is the willingness of people to reduce their meat consumption. In other words, we aim to explain why some people have a higher will to reduce their meat consumption than others. Such explanations could either confirm existing knowledge on this phenomenon or provide new explanations that

[2] `https://dbpedia.org/ontology/`

[3] National Research Institute for Agriculture, Food and Environment

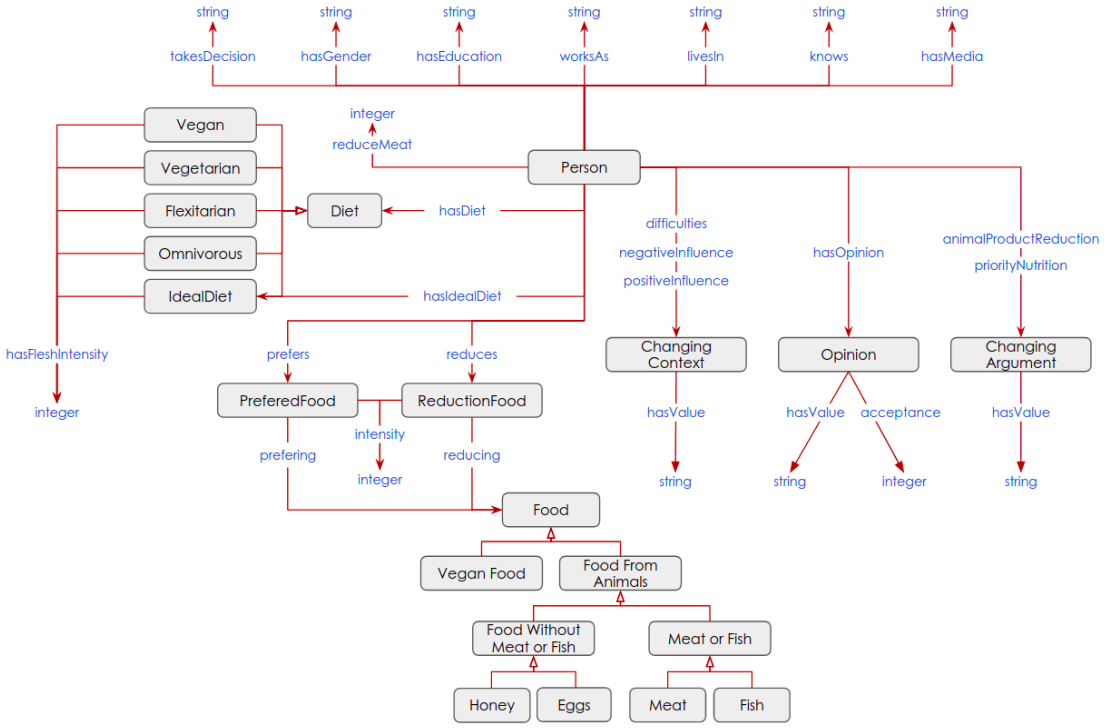

Figure 4.7: Schema of *Vitamin*

could be the object of future experiments. As the outcome is a single value, no aggregation function s is used.

As shown in Figure 4.7, the schema is rather complex. As a consequence, the results of *Dicare-S* and *Dicare-C* over *Vitamin* were submitted to two experts in nutritional behaviour for the qualitative evaluation of the rules.

4.3.2 Communities Detection

The community detection step from *Dicare-C* has only been applied on *Vitamin*. It has not been applied to *DBpediaW* as it is considered simple enough to be exploited directly.

Parameters have been defined for using *OSLOM* [2]. *isolated* has been set to *True* to enable nodes to be single, and the statistical significance of the communities s has been set to 0.9. Three topics were chosen to define the sets of paths $\mathcal{P}$ where communities were searched: food taste, positive or negative opinions on diet-change motivation, opinions on climate change and animal well-being.

The paths related to opinions proved to have four communities used to create abstract properties. An extract of the resulting communities is shown in Table 4.3. Community 1 shows a profile of opinions associated with the importance

Table 4.3: Extract of Communities - *Vitamin*

Community 1
Livestock is a major contributor to global warming
Vegetarian diets are healthier
The problem is not breeding but industrial breeding
Community 2
Vegan and Vegetarian diets are deficient
Human Nature is to eat animals
Eating animal products makes me happy

of animal well-being, the impact of breeding on climate change, and reasonable opinions on meat-free diets. On the contrary, community 2 shows that pro-meat opinions are associated with good opinions on breeding as a natural element unrelated to climate issues, and with bad opinions on meat-free diets. Community 3 gather moderate opinions on climate and animal well-being, while 4 describe various strong but conflicting opinions. The expert assessment is that the resulting communities are well highlighting extreme points of view concerning postures towards food: conservationists or meat lovers. In this sense, the communities were judged accurate, intuitive and useful for explainability purposes.

The other sets of paths concerning food taste and positive or negative influences did not lead to communities. They are, therefore, not used for the explainability purpose in the following steps.

4.3.3 Mining Evaluation

The quantitative and qualitative results of *Dicare-S* and *Dicare-C* on *DBpediaW* and *Vitamin* are presented in Table 4.4. The table presents, for each approach and dataset, the number of rules that are obtained, the average number of rules that can explain the outcome of a pair, the percentage of correct rules as evaluated by experts, and the number of pairs that are explained with at least one rule. The baseline is the exact matching procedure described in 4.2.1, *i.e.*, matching instances if they share the same description on a set of paths.

For each rule, the experts could evaluate a rule with the following choices: a rule (i) *seems relevant*, (ii) *may be relevant*, (iii) the expert *does not know* or (iv) the rule *seems false*.

DBpediaW

Dicare-S and *Dicare-C* have the same results on *DBpediaW* as no community detection step was applied, while the baseline does not mine any rules as exact

Table 4.4: Mining results using *Dicare-S* and *Dicare-C*.

	DBPediaW			***Vitamin***		
	#Rules (Per pair)	%Correct	%Pairs	#Rules (Per pair)	%Correct	%Pairs
Baseline	0 (0)	*NA*	0	0 (0)	*NA*	0.0
Dicare-S	**12** (1.1)	83.3	16.5	0 (0.0)	*NA*	0.0
Dicare-C	**12** (1.1)	83.3	16.5	**44** (1.1)	80.0	52.5

matching is not possible in this dataset. Therefore, we refer to the results of both approaches as the results of *Dicare-S* in the following. *Dicare-S* outputted 22 most specific rules, including 14 gradual and 8 categorical. This set resulted in 12 rules after the generalisation step. The equations 4.4 and 4.5 present a simplified representation of two rules.

$$\begin{gathered} genre(Fiction) \wedge gender(male) \wedge country(US) \wedge \\ ageFirstBook(i_1, u_1) \wedge ageFirstBook(i_2, u_2) \wedge \\ \mathbf{univ.arwuW(i_1, v_1)} \wedge \mathbf{univ.arwuW(i_2, v_2)} \wedge \mathbf{v_1 < v_2} \\ \Rightarrow u_1 < u_2 \end{gathered} \tag{4.4}$$

$$\begin{gathered} genre(Fiction) \wedge country(US) \wedge \\ ageFirstBook(i_1, u_1) \wedge ageFirstBook(i_2, u_2) \wedge \\ \mathbf{birthYear(i_1, v_1)} \wedge \mathbf{birthYear(i_2, v_2)} \wedge \mathbf{v_1 > v_2} \\ \Rightarrow u_1 < u_2 \end{gathered} \tag{4.5}$$

These results show that the generalisation can lead to more general rules even with a drastic consistency constraint. The evaluators, *i.e.*, 5 researchers in computer science, judged that 9 out of the 12 rules (75%) were meaningful and interpretable.

If looking at the rule in equation 4.4, it makes sense that studying at universities of better standing may explain a younger age of first book publication for writers. Such writers could have more credit to editors as their universities are better known. The rule from equation 4.5 indicates that the earlier a writer is born, for instance, a writer born in 1900 compared to another born in 1960, the later he will publish his first book. An explanation could be the democratisation of access to publishing, which is becoming gradually easier. With the years passing, we could imagine new processes to print books and more editors in an extended market.

It is to note that such rules can be considered causal if and only if all properties associated with the outcome are studied. Indeed, a missing property could explain both the outcome and an existing property that explains the outcome. Consequently, the outputted rule could express a correlation between two paths

having the same cause more than causation. The knowledge graph's completeness in terms of paths might also have an effect on the number of pairs explained. For instance, for *DBpediaW*, the rules can explain 16.5% of the differences between ages of the first publication. This suggests that other properties not described in the dataset should be considered to explain the outcome.

Vitamin

Dicare-S did not output any rule on *Vitamin* as matching instances without abstract properties is impossible. The baseline also did not output any rule, as exact matching is not possible either. In total, 119 specific rules were mined and merged into 44 more specific rules by *Dicare-C*. These rules can explain the reduction in meat for 52.5% of the pairs of people. An example of a mined rule is presented in equation 4.6. It expresses that, for women, being part of community 1 compared to community 2 explains a higher meat reduction aim.

$$\begin{gathered} gender(X, woman) \wedge reduction(i_1, v_1) \wedge reduction(i_2, v_2) \\ \mathbf{hasOpinion(i_1, 2)} \wedge \mathbf{hasOpinion(i_2, 1)} \\ \Rightarrow v_1 < v_2 \end{gathered} \tag{4.6}$$

$$\begin{gathered} livesIn(X, countryside) \wedge lowerEducation(X, Bachelor) \wedge hasOpinion(X, 3) \wedge \\ reduction(i_1, u_1) \wedge reduction(i_2, u_2) \wedge \mathbf{gender(i_1, man)} \wedge \mathbf{gender(i_2, woman)} \\ \Rightarrow u_1 < u_2 \end{gathered} \tag{4.7}$$

The experts estimated that 35 of the 44 rules were meaningful and interpretable (80%). In particular, rules involving the mined communities as the treatment were judged very intuitive. It has been noted that a rule's interpretability is dependent on its strata. More precisely, the merged rules are characterised by their length, as their strata can be more or less specific. The more general the rule is, the easier it is to understand. For instance, in the rule described in equation 4.7, the sex treatment has been found in a rule applied to people living in rural areas, having at least a Bachelor, and having the community 3 on opinions. Compared to the previous rule, it is harder to understand and assess.

4.4 Comparative Evaluation and Discussion

4.4.1 Comparison to Association Rules

Association rules were mined with AMIE [20] after adding a predicate in the dataset that compares the outcome of 2 people, named *aimsToChangeComparedTo*.

Without the community step, AMIE discovers 5480 rules that conclude on this predicate. These rules are too numerous and hardly interpretable, also we simplified the graph with the abstract properties. 111 rules were mined with a support of 100 and confidence of 0.1 as parameters. One mined rule is:

$$\begin{aligned} &higherEducation(x, HighSchool) \wedge age(y, [1, 34]) \\ &\Rightarrow aimsToChangeComparedTo(x, y) \end{aligned} \tag{4.8}$$

Such an association rule does not explain the difference in the reduction in meat. In this rule, no treatment clearly appears, as there are no comparisons between x and y on the paths in the association rule. For instance, y could have stopped its studies at high school, in which case education would not be a treatment, but y could also have attended university, in which case education would have been a treatment not described in the rule. More generally, a property can not be used as a treatment if the value is only known for one instance. In addition to this treatment issue, another problem is related to the paths not described in the rule. For instance, an explanation to the head of the rule could be that x is a woman and y is a man, but this is not mentioned in the rule. More generally, even if a treatment were to appear in the association rule, AMIE does not verify the similarity of instances as it does not rely on matching or defines the strata for which a treatment is valid. Therefore, due to the treatment that is not clearly expressed and the missing paths that could explain the outcome, association rules are not suited to explain outcome differences.

4.4.2 Comparison to the State of the Art

The results are compared to Munch *et al.* [51], whose approach based on bayesian networks aims to predict a discretized outcome with discretized attributes. This approach discovers that the year of publishing depends directly on the writers' university and birth date. For instance, a writer born before 1950 who studied in a university with an *arwuW* smaller or equal to 100 has a 0.58 probability of publishing his first book before 1980.

As a reminder, *Dicare-S* and *Dicare-C* aim to explain differences in the outcome rather than making predictions. They discover more rules involving four different treatments, *i.e.*, paths on birthDate, arwuW, genre, and gender, that vary depending on 12 different strata. For instance, one mined rule indicates that the book genre can also explain differences in the age of first publication for a given strata, while such rules were not found by the approach of *Munch et al.* [51]. In addition to discovering more treatments, differential causal rules are more expressive as they express graduality, *e.g.*, with the rank of a university (equation 4.4) or the birth date of an author (equation 4.5), while these paths were discretized

in Munch *et al.*

4.4.3 Comparison of *Dicare-S* and *Dicare-C*

The experiment on *Vitamin* shows an interest in discovering and using abstract properties for matching. Not only the mined communities were judged meaningful and able to represent sets of paths, but their use also made it possible to mine differential causal rules on a complex knowledge graph. While *Dicare-S* can be used on relatively simple knowledge graphs, *Dicare-C* is to be used on more complex knowledge graphs.

4.4.4 Expert Assessment

The aim of *Thomopoulos et al.* [88] was to investigate the relationship between environmental concerns, health and ethical issues and consumer demands, knowing that livestock production has an important ecological footprint. With a random forest model trained on the tabular data used to generate *Vitamin*, they can predict that a person will shift towards fewer animal products in their food diet with a 0.65 accuracy. This relatively low accuracy shows that this data is not easy to exploit for classification purposes. The random forest insights led to the conclusion that: *the willingness to change is stronger for the youngest, hindrances to change are food pleasure, health and to a lesser extent social resistance and animal ethics. The less radical the animal products reduction is, the more environmental concerns are the main motivation.*

Some similarities can be found between the random forest, *Dicare-S* and *Dicare-C*. In all of them, through a more or less complex procedure, the variables can be ordered by discriminating power. For instance, it has been shown in *Dicare-C* that the more general rules had the opinion as treatment, followed by the gender, the place of living and finally the education. In Thomopoulos *et al.* [88], the features that better separate individuals that want to change their diets from individuals that do not can also be retrieved and ranked. However, in our approaches, we do not aim to classify all individuals, but rather to find out all the plausible reasons why one would want to change his diet and not the other. Our approaches guarantee that all the pairs of individuals used to establish the rule quality are identical, *i.e.*, have the same property paths or abstract properties, except for the values that appear in the treatment. This makes our rules adapted to explain differences in the outcome with differences in treatment that can potentially be causal. Furthermore, since the number of treatments and strata can be huge, the community step of *Dicare-C* simplifies the explanations and the discovery process.

4.5 Conclusion

This chapter proposed the first definition of differential causal rules and two approaches, *Dicare-S* and *Dicare-C*, to mine them. The approaches are based on the potential outcome framework and, in particular, on instance matching. As exactly matching instances of a knowledge graph is not feasible, the approaches rely on coarsened matching and generating abstract properties that are a semantic summary of part of the description of the instance.

The discovered rules are expressive and were judged interpretable and mostly correct by experts from the considered application. However, due to the pessimistic heuristic during the rule mining process, the rules cannot provide an explanation for many pairs. Therefore, no explanations are provided for many pairs. A second drawback of the approach is that discovered rules can be more or less general depending on the set of instances they are applied to, which makes them more or less interpretable. In particular, specific rules, *i.e.*, that are applied on precise subsets of graph instances, are harder to interpret.

As a consequence, we highlight several limitations on which *Dicare-S* and *Dicare-C* can be improved in next approaches (see Chapters 5 and 6):

- **Rule Interpretability.** A rule should be easily understandable by an expert to be used as insight. Rules with long strata, as mined through *Dicare-C*, are hardly understandable and are hardly usable. Moreover, it is impossible to rank the rules and the treatment effects.

- **Pairs Explanation.** In terms of explainability, current approaches hardly explain more than half of the pairs having the outcome on the tested datasets. As this can be partly explained by the dataset content, mining more general rules that could explain more pairs is of interest.

- **Instance Matching Criteria.** The comparison of exact matching to coarsened matching with abstract properties showed that matching similar instances is a complex step. Moreover, matching becomes increasingly difficult with the size of a knowledge graph. Alleviating this constraint could allow for mining more general rules.

- **Feasibility on Knowledge Graphs.** Lastly, it is to note that current approaches were developed on knowledge graphs where mining communities is possible. However, all knowledge graphs might not be suited for this task. As a consequence, other approaches should consider matching similar instances without using abstract properties to expand the set of knowledge graphs on which causal discovery can be applied.

In the following chapters, we will propose approaches that aim to satisfy one, or several of the fore-mentioned limitations: Chapter 5 proposes an approach to enable an easier interpretation of differential causal rules, and Chapter 6 proposes an approach that mines a new type of rules expressing an average effect for a treatment to explain more pairs.

Chapter 5

Counter Effect Rules Mining

In this chapter, we will present our contribution to counter effect rule mining in knowledge graphs. These rules are obtained by processing differential causal rules to facilitate their interpretation. This work is the object of the following publication:

1. **Lucas Simonne**, Nathalie Pernelle, and Fatiha Saïs. Counter effect rules mining in knowledge graphs. In *Knowledge Engineering and Knowledge Management*, Lecture Notes in Computer Science, pages 167–173. Springer International Publishing, 2022 [77]

The previous chapter introduced *Dicare-C* and *Dicare-S*, two approaches that mine differential causal rules from a knowledge graph. Such rules express that a treatment has a causal effect on a subset of instances defined by a graph pattern called strata. While these approaches output differential causal rules judged globally of interest and meaningful by experts, some of these rules are hardly interpretable. For instance, one treatment may have opposing effects in some rules depending on the set of instances it is applied to. In addition to their interpretation, the ranking of treatments is not trivial, as the set of rules can be large.

Given these limits of the previous approaches that mine differential causal rules, *Dicare-S* and *Dicare-C*, we propose in this chapter a new type of rule that is complementary to such rules and that facilitate their interpretation. The contributions of this chapter are the following :

- The definition of counter effect rules expressing that a treatment can have opposing effects on two sets of instances given a difference between these sets' descriptions. Such rules can be used to rank treatments and to display the local effects of treatments from differential causal rules.

- An algorithm that mines counter effect rules from previously mined differential causal rules.

The chapter is structured as follows. **Section 5.1** defines counter effect rules, that show opposing causal effects of a treatment on different subsets of instances.

Section 5.2 describes how counter effect rules are mined as a post-processing step of differential causal rules mined using *Dicare-C*.

Section 5.3 presents counter effect rules that were mined with the proposed approach and one dataset, and show how they can be used to rank treatments.

Section 5.4 provides a discussion on counter effect rules and how they can be used to complement differential causal rule mining approaches.

Section 5.5 concludes this chapter on counter effect rules.

5.1 Counter Effect Rules

This section defines the limits of differential causal rules mined by *Dicare-S* and *Dicare-C*, and motivates the need for using a complementary type of rules, named *counter effect rules*, that aim to ease the interpretation of differential causal rules.

5.1.1 Motivation

The previous chapter, Chapter 4, introduced *Dicare-S* and *Dicare-C*, two approaches that were defined to mine differential causal rules from a knowledge graph. Such rules express that a treatment has a causal effect on a specified set of instances described by a strata, *i.e.*, a basic graph pattern.

While differential causal rules were mined and overall judged interpretable by experts, their analysis presents some limitations. More precisely, two issues arise. The first is that, at the scale of one rule, a rule can be difficult to interpret due to its strata length. Indeed, while a general rule, *i.e.*, with few elements in its strata, will be easily interpreted, more specific rules are harder to interpret. The second issue comes at the scale of analysing the whole set of rules and is on treatment effects. Firstly, the rules are numerous, and it might therefore be difficult for a user to go through all of the rules to get insights, for instance, to understand which treatment has the stronger effect. Secondly, a treatment may appear in several rules, sometimes with a *consistent* effect as it can be the same for several strata, but also with *opposing* effects for other strata. Therefore, the effect of one treatment can be ambiguous, and the differential causal rules alone are not suited to explain why the treatment has opposing effects.

This chapter focuses on the second issue, *i.e.*, the treatment effects, and introduces *counter effect rules* to facilitate the interpretability of differential causal

rules. These rules indicate that one treatment can have opposing effects depending on the set of instances it is applied to. By observing the difference between the description of sets of instances where a treatment has opposing effects, it is possible to observe the change of description, *i.e.*, another treatment, that explains the change of effect. This makes it possible to rank treatment effects depending on, first, how consistent the effects of one treatment are, and, secondly, how frequently one treatment can change the effect of another treatment.

5.1.2 Counter Effect Rules Definition

We first propose a definition of a *description change*, *i.e.*, how one graph pattern is modified to obtain another graph pattern. This definition is based on the definition of a strata (see definition 2).

Definition 14. ***Description Change.*** *Let ST_i and ST_j be two strata that describe two sets of instances $\mathcal{I}_i$ and $\mathcal{I}_j$. The description change between ST_i and ST_j is denoted by $\rightarrow$, such that $ST_i \rightarrow ST_j$ indicates that the considered instances changed from $\mathcal{I}_i$ to $\mathcal{I}_j$.*

Let a differential causal rule DCR_1 that expresses the effect of a treatment T on the strata ST_1. Let another strata ST_2. The description change from ST_1 to ST_2, noted $ST_1 \rightarrow ST_2$, can be used to study how the effect of the treatment T evolves when applied to another strata, ST_2.

Example 8. *Let two strata $ST_1 : Researcher(X) \wedge gender(X, \text{“}Man\text{“})$ and $ST_2 : Researcher(X) \wedge gender(X, \text{“}Woman\text{“})$. Let T be a treatment whose effect is studied on ST_1 and ST_2. The description change $ST_1 \rightarrow ST_2$ can be used to study how the effect of T changes depending on whether it is applied to researchers that are men or women.*

Given the definition of description change, we propose a definition for *counter effect rules.*

Definition 15. ***Counter Effect Rule.*** *Let two instances X_1 and X_2 of a class $C \in \mathcal{O}$, and a treatment T that explains an outcome O when X_1 and X_2 belong to a strata ST_i. A counter effect rule CER_T expresses that, if X_1 and X_2 belong to another strata $ST_j \neq ST_i$, then T will explain the opposite outcome $\overline{O}$. A description change from ST_i to ST_j explains an opposite effect of a treatment. More formally, a CER_T is defined as follows:*

$$\begin{aligned} CER_T(X_1, X_2) : C(X_1) \wedge C(X_2) \wedge T(X_1, X_2) \wedge P_O(X_1, U_1) \wedge P_O(X_2, U_2) \wedge \\ [ST_i(X_1) \wedge ST_i(X_2) \rightarrow ST_j(X_1) \wedge ST_j(X_2)] \Rightarrow [O(X_1, X_2) \rightarrow \overline{O}(X_1, X_2)] \end{aligned} \tag{5.1}$$

It is to note that such a rule is built from the combination of two differential causal rules. It is not a first-order logic rule, but a representation of how one treatment may have opposing effects depending on the instances it is applied on. Such rules are of interest to experts as they ease the understanding of the differential causal rules for two reasons. First, they explicitly show that a treatment may have opposing effects and why. Secondly, such rules can be used to show the consistency of treatments, which can be a criterion to rank them. We discuss these criteria in section 5.3.

Example 9. *Let two strata* $ST_1 : Researcher(X) \wedge gender(X, \text{“}Man\text{“})$ *and* $ST_2 : Researcher(X) \wedge gender(X, \text{“}Woman\text{“})$*. Given two instances* X_1 *and* X_2*, a treatment* T *indicates that* X_1 *lives in the countryside and* X_2 *lives in a city. The outcome* O *indicates that* X_1 *has a higher will to reduce its meat consumption than* X_2*. Let* DCR_1 *and* DCR_2 *be two differential causal rules as defined in Definition 7.* DCR_1 *express that, for men researchers, living in the countryside compared to living in cities, i.e.,* T*, explains a higher will to reduce one's meat consumption.* DCR_2 *express that treatment* T *has the opposite effect of* O *for women researchers, as for them living in cities compared to living in the countryside explains a higher will to reduce one's meat consumption.*

The resulting CER is presented as follows (equation 5.2). For the sake of understanding, its representation has been simplified. The first line indicates the treatment, *i.e.*, living in the countryside compared to living in cities. The second line expresses a part of the outcome with the paths leading to the values of meat reduction. The third line is composed of two parts: the first indicates the common part of ST_1 and ST_2, $Researcher(X)$ in this example, and the second, in bold, shows the description change, *i.e.*, what explains the opposing outcome of the treatment. The fourth and last line is the head of the rule and shows the change of effect of the treatment that results from the description change.

$$\begin{gathered} livesIn(X_1, countryside) \wedge livesIn(X_2, city) \wedge \\ reduceMeat(X_1, u_1) \wedge reduceMeat(X_2, u_2) \wedge \\ Researcher(X) \wedge [\mathbf{gender}(\mathbf{X}, \text{“}\mathbf{Man}\text{“}) \rightarrow \mathbf{gender}(\mathbf{X}, \text{“}\mathbf{Woman}\text{“})] \\ \Rightarrow [\mathbf{higherThan}(\mathbf{u_1}, \mathbf{u_2}) \rightarrow \mathbf{lessThan}(\mathbf{u_1}, \mathbf{u_2})] \end{gathered} \tag{5.2}$$

It is to note that the length of the description change plays an essential role in the understanding of a counter effect rule. Indeed, a small description change such as presented in equation 5.2 is easy to understand. However, the longer the change, the harder it is to understand. Therefore, this length should be controlled as a parameter in the mining of the counter effect rules.

5.2 Approach for Counter Effect Rule Generation

This section describes the approach to mine counter effect rules. This approach is constructed in two steps and consists in the post-processing of differential causal rules that can be mined by *Dicare-C*. In the first step, differential causal rules are discovered by using *Dicare-C*, *i.e.*, the approach described in Chapter 4. In the second step, the differential causal rules are processed to obtain counter effects rules.

5.2.1 Mining Differential Causal Rules

In the first step, DCRs are mined with *Dicare-C*. As a reminder, the algorithm takes as input a KG, a target class c, a set of property paths $\mathcal{P}$ that lead to possible treatments P_T, the property path leading to the outcome P_O, and expert knowledge to guide the mining. It outputs the DCRs.

Dicare-C relies on a bottom-up strategy. First, it generates potential DCRs and evaluates them using a metric inspired by the odds ratio. The obtained DCRs are said to be *specific* as they express treatments on stratas composed of several predicates. Secondly, the strata of the DCRs are merged to obtain more general rules if they share the same treatment and outcome. While the effect of a specific rule is quantified by the odds ratio, a rule obtained from a merge is not quantified. Therefore, merged rules can not be ranked. Regarding the mining of counter effect rules, only the specific DCRs are considered to ease their discovery.

5.2.2 Obtaining Opposite Effects

In the second step of the algorithm, the set of CERs, $\mathcal{D}$, is obtained from the DCRs previously mined.

As shown in equations 5.1 and 5.2, a CER expresses that a change from one strata to another explains an opposite effect of the same treatment. It is therefore obtained by comparing two differential causal rules if they have the same treatment but opposite outcomes. However, as discussed in section 5.1.2, the generation of all CERs on this criterion alone is not desirable, as some CERs may have a description change too long to be understandable. As a consequence, instead of outputting all CERs, we introduce a threshold, d_c, that is the maximum number of changes a rule can have, *i.e.*, the number of differences between the two strata it is built from. For instance, the CER in equation 5.2 has $d_c = 1$. The use of a threshold ensures the generation of a set of rules that can be interpreted.

In the algorithm, the DCRs are first sorted by the treatment they express. Then, for each unique treatment T, two sets of rules are obtained: $\mathcal{R}_{(T,O)}$ and $\mathcal{R}_{(T,\overline{O})}$ depending on the rules outcome, where $\mathcal{R}_{(T,O)}$ is the set of rules where T

explains T, and $\mathcal{R}_{(T,\overline{O})}$ is the set of rules where T explains the opposing outcome. A potential counter effect rule per pair $(R_i, R_j) \in \mathcal{R}_{(T,O)} \times \mathcal{R}_{(T,\overline{O})}$ is constructed. With ST_i the strata of the rule R_i, the potential counter effect constructed from R_i and R_j is added to $\mathcal{D}$ if its number of changes $n(ST_i, ST_j)$ is lower or equal than d_c.

Example 10. *Let us consider DCR_1 and DCR_2 the DCR given in section 5.1.2. DCR_1 and DCR_2 have the same treatment but opposite outcomes and can therefore be used to generate the potential CER described in equation 5.1. With d_c set to 1, as $n(ST_1, ST_2) = 1$, this CER can be added to $\mathcal{D}$.*

A final procedure is applied on the CERs of $\mathcal{D}$. Similarly, as for the mining of differential causal rules, the CERs are merged to obtain more general rules.

5.3 Experiments

We conducted experiments in order to show the relevance of the CERs that can be mined over a knowledge graph. The obtained CERs were then compared to the DCRs mined using *Dicare-C* (see Chapter 4) and to association rules mined using *AMIE* [21].

5.3.1 Dataset Presentation

The dataset *Vitamin* has been used for the experiment. It is extensively described in section 4.3.1. As a reminder, it describes people and their socioeconomic characteristics, such as age, gender, current and ideal diet, opinions on animal welfare facts and climate change. The outcome is the difference between a person's current and ideal diet, *i.e.*, their willingness to reduce their meat consumption. It is denoted by $P_O = reduceMeat$ having values $\in \mathbb{N}$.

In addition to mine treatments that may explain differences in one's meat reduction willingness, we are interested in observing whether such treatments are consistent or may have opposite effects. Also, we aim to rank them based on their consistency and ability to change the effect of other treatments.

It is to note that *DBpediaW* has not been used in this experiment as no counter effect rules could be mined over it.

5.3.2 Settings

The DCRs mined by *Dicare-C* on *Vitamin* are the same as the DCRs obtained in section 4.3. The mining of the CERs has been done with d_c set to 1 to obtain rules with a description change of one element only.

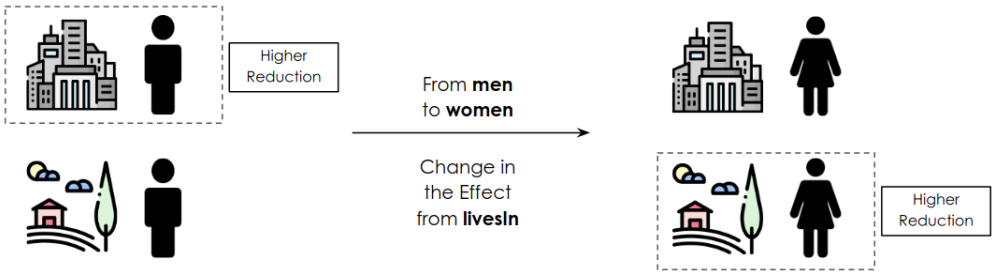

Figure 5.1: Illustration of a counter effect rule

5.3.3 Mined Counter Effect Rules

In total, 23 CERs with a change of one element were discovered using the set of 119 specific DCRs previously mined. As a reminder, the DCRs used to create the CER were overall judged accurate and meaningful by experts in the field. An example of a CER, CER_1, is presented in equation 5.3[1], where living in cities compared to living on the countryside explains a higher will to reduce one's meat consumption for men that attended high school, but a lower will if they were women. This rule is also illustrated in Figure 5.1.

$$\begin{aligned} CER_1 : livesIn(X_1, city) \wedge livesIn(X_2, countryside) \wedge reduceMeat(X_1, u_1) \wedge \\ reduceMeat(X_2, u_2) \wedge [education(X, high\_school) \wedge hasOpinion(X, 1)] \wedge \\ [\mathbf{gender(X, man)} \rightarrow \mathbf{gender(X, woman)}] \\ \Rightarrow [\mathbf{greaterThan(u_1, u_2)} \rightarrow \mathbf{greaterThan(u_2, u_1)}] \end{aligned} \tag{5.3}$$

Table 5.1 presents an extract of treatments involved in the mined DCRs (*Treatment*), the number of DCRs they are involved in (*#DCR*), their consistency (*% same O*), *i.e.*, the percentage of DCRs they are implied on and where they have the same outcome, the number of CERs (*#CER*), and the most common treatments paths that explain their counter effect (*Counter Treatment*).

Two observations are presented in this table. The first observation is on the ranking of treatments based on their consistency value. This ranking shows that two treatments are strong: the first is the sensitivity to climate change and animal well-being, and the second is gender, as women seem more willing to reduce their meat consumption than men. Both of these treatments have a 100% consistency, meaning they always explain the same outcome on strata where they have an effect. On the contrary, the consistency metric also shows that treatments may have less consistent effects, such as the treatment on education that equally leads

[1]The value 1 for predicate *hasOpinion* represents a person with pro-meat opinions who believes that breeding is unrelated to climate issues

Table 5.1: Extract of Results. X_2 has a higher will to reduce its meat consumption than X_1.

Treatment	***#DCR (% same O)***	***#CER***	***Counter Treatment***
$hasOpinion(X_1, 2) \wedge hasOpinion(X_2, 1)$	14 (100%)	0	*NA*
$gender(X_1, man) \wedge gender(X_2, woman)$	10 (100%)	0	*NA*
$livesIn(X_1, city) \wedge livesIn(X_2, campaign)$	18 (88%)	2	gender,education
$education(X_1, university) \wedge education(X_2, high\_school)$	16 (50%)	7	gender,opinion

to opposing effects. The second observation is on *counter treatments*, which figure on the last column of the table. Counter treatments exist only for treatments that are not consistent, *e.g.*, the difference of education or the place of living. As such treatments change the effects of the latter ones, they can be considered stronger. It appears that counter treatments that are often found are also the treatments that are the most consistent.

5.4 Discussion

5.4.1 Comparison to Association Rules

A set of association rules has been mined using AMIE [21] with a support of 100 and confidence of 0.1. As in section 4.3, the target head of the association rules has been set by introducing a new predicate that compares the difference in the *reduceMeat* predicates. A set of 111 rules were mined using AMIE. However, none of them could explain a difference in the $reduceMeat$ predicate between the two instances. More precisely, while the body of the mined rules could contain predicates on both instances, they do not compare the same ones for the two instances. Therefore, as described in section 4.3, association rules are not suited for such explainability purposes. Consequently, they are also not suited to show one treatment can have different effects depending on the set of instances it is applied on.

5.4.2 Comparison to Differential Causal Rules

Counter effect rules are obtained after processing a set of differential causal rules. We aim to emphasise the gains of this processing compared to the analysis of the differential causal rules alone.

As discussed in section 5.1.1, counter effect rules were defined in response to the limits of differential causal rules regarding interpretability. More precisely, the differential causal rules are numerous and may express opposing effects for the same

treatment, making it difficult for an expert to analyse them. The results of the mining of counter effect rules, presented in section 5.3, lead to several observations. First, it showed that treatments may lead to different outcomes and are not always the same. Secondly, the counter effect rules can be used to rank the treatments based on their consistency, and also underlie the stronger effects of treatments on other treatments. As a consequence, they allow separate treatments based on whether they have general or more local effects. This can be of great interest to an expert as they discovered general effects are usually already known and published in the literature, as in *Thomopoulos et al.* [88], while more local effects may be leading to deeper analysis or the design of new experiments.

In other words, while some differential causal rules may be easy to interpret, especially the more general ones, using the counter effect rules allows to get a faster and wide analysis of the treatments than studying the differential causal rules alone, therefore proposing an approach to some of the limits of the latter.

5.5 Conclusion

In this chapter we presented a new approach for generating counter effect rules that express that a treatment effect can have opposing effects on different sets of instances of a knowledge graph. These rules are generated as a post-processing step from a set of differential causal rules.

Counter effect rules have shown to be of interest to facilitate the **interpretation of differential causal rules**. Indeed, while a differential causal rule studies a local effect, counter effect rules provide a deeper analysis of a treatment in terms of consistency and comparison to other treatments. As a consequence, the mining of both types of rules should be used complementary.

While this chapter provided an approach that increases the interpretability of previous results, thus improving a limit enunciated in the conclusion of Chapter 4, it, however, does not improve other limits such as the number of **explained pairs** or the **instance matching criteria**. The next chapter will address these limits by introducing *Dicare-E*, an embedding-based approach.

Chapter 6

Knowledge Graph Embeddings for Differential Causal Rule Mining

Chapter 6 presents our contributions to differential causal rules mining using knowledge graph embeddings. While previous approaches presented in Chapter 4 rely on comparing RDF descriptions of instances, we show that rules can also be obtained by comparing instances embedding representations. The obtained rules express an average effect compared to the local effect of the rules expressed in Chapter 4, making them easier to interpret. This chapter is based on the following publications:

1. **Lucas Simonne**, Nathalie Pernelle, Fatiha Saïs, and Rallou Thomopoulos. Découverte de règles causales dans les graphes de connaissances à l'aide de plongements dans les graphes. In *IC 2022: 33es Journées francophones d'Ingénierie des Connaissances (Proceedings of the 33nd French Knowledge Engineering Conference), Saint-Etienne, France, June 29 - July 1, 2022*, pages xx–xx, 2022. [80]

2. **Lucas Simonne**, Nathalie Pernelle, Fatiha Saïs, and Rallou Thomopoulos. Discovering causal rules in knowledge graphs using graph embeddings. In *WI-IAT'22: International Conference on Web Intelligence, Niagara Falls Canada, November 17-20, 2022*, pages xx–xx, 2022. [79]

The mining of differential causal rules and counter effect rules has been shown useful in discovering causal effects of treatments on instances of a knowledge graph. However, the rules that are mined with *Dicare-C* and *Dicare-S* only express local effects. They are, therefore, sometimes difficult to interpret, and the number of

instance pairs for which differences in the outcome are provided with an explanation needs to be improved. Furthermore, while matching RDF instances with complex descriptions has been addressed by a community detection algorithm, this coarsened matching process still remains difficult as it relies on an expert to guide the community detection step.

Consequently, we developed a new approach named *Dicare-E* that we present in this chapter. The objectives of this new approach are threefold. First, we aim to mine causal rules expressing an average causal effect on all instances of a knowledge graph, in opposition to the local effects expressed by rules mined in *Dicare-C* and *Dicare-S*. These average rules should be easier to interpret. Secondly, we aim to explain more pairs that have the outcome. Lastly, we aim to facilitate the matching of instances to have an approach adapted to more complex knowledge graphs that may have missing data. More precisely, *Dicare-E* is a hybrid algorithm that combines numerical and symbolic methods to discover causal rules. While *Dicare-C* and *Dicare-S* use instances descriptions to match similar instances, *Dicare-E* uses instances representation in a low dimensional vector space to measure the similarity of instances and to determine if they can be matched.

The contributions of this chapter are the following:

- A method to match similar instances based on their embedding representations. This method defines a threshold in the embedding space to ensure the matching of similar instances only.

- The definition of average causal rules, which are differential causal rules expressing an average effect of a treatment across all instances of a target class in a knowledge graph.

- An algorithm, *Dicare-E*, that mines average causal rules and an analysis of its results that shows that *Dicare-E* explains a higher number of pairs compared to *Dicare-C* and is more resistant to missing data.

The chapter is structured as follows. **Section 6.1** presents the motivation of *Dicare-E* by underlying the limits of *Dicare-C* and *Dicare-S* presented in chapter 4. It also shows how we propose to use graph embeddings in an attempt to solve such limits.

Section 6.2 presents *Dicare-E*, the algorithm that we developed to discover differential causal rules by relying on graph embeddings.

Section 6.3 presents the results of the experiments we realised using *Dicare-E*.

Section 6.4 concludes the chapter.

6.1 Average Differential Causal Rules

This section emphasises on the motivation of both defining a new type of rules that are easier to interpret and easing the matching of instances of a knowledge graph. It defines the new type of rules, named *average differential causal rules*.

6.1.1 Motivation

The approaches presented in the previous chapter, *Dicare-C* and *Dicare-S*, define differential causal rules and how they adapt the potential outcome framework to mine them in a knowledge graph. These rules express the local effects of treatments on a subset of instances. They are mined by matching similar instances based on their RDF description. However, these approaches presented several limits as exposed in section 4.5, namely the **interpretability** of the rules, the number of **explained pairs**, and the **instance matching** criteria.

The first limit is the interpretability of differential causal rules. While counter effect rules, introduced in Chapter 5, enhance the interpretation of differential causal rules when studied as a whole, it does not ease the interpretation of a rule by itself. The expert assessment of the rules indicates that the more specific the strata of a rule is, the more difficult the rule is to be interpreted. Moreover, the community detection step used to simplify the matching step reduces the granularity in the treatment's precision. Indeed, the effect of a particular path within a community used as a treatment can not be assessed. Finally, as the obtained rules are discovered by generating specific ones, a rule's treatment effect is no longer measured by the metric, and thus, rules can not be ranked on how strong their effects are.

The second limit is the number of pairs these rules can explain. Indeed, as differential causal rules express local effects, many pairs are not provided with an explanation for why they have the outcome.

The third limit is that *Dicare-C* and *Dicare-S* use the RDF description of instances to match similar instances, making it not trivial to create matches, especially in case of incompleteness. *Dicare-C* and *Dicare-S* measure the similarity between two instances as the number of paths they have in common, with the paths being original RDF descriptions or abstract paths generated from a community detection step. However, using this similarity measure becomes increasingly difficult with the number of paths describing instances. Indeed, the higher the number of elements where instances need to be similar to be matched, the more difficult it is to find matches within a database [1]. This statement is even stronger with small databases where it is less likely to find a match for a given instance. Finding similar instances is even more difficult with knowledge graphs where parts of instances description can be missing, and instances may be described by non-

functional properties. While *Dicare-C* already addressed these problems by reducing the number of paths to study by using a community detection step, finding two instances with the same descriptions in a knowledge graph is complex and even impossible in some cases.

This chapter aims to answer these limits by introducing an approach that mines rules expressing the average effects of treatments. These rules are discovered by using knowledge graph embeddings.

6.1.2 Average Differential Causal Rules

We propose the definition of *average differential causal rules*. They are differential causal rules that do not express any constraint on the instance pairs through a specified strata. These rules express an average effect of a treatment over all instances of a target class in a knowledge graph.

Given the definitions of categorical treatment T_c (see Definition 4), numerical treatment T_n (see Definition 5) and outcome O (see Definition 6), an average differential causal rule $ADCR$ is defined as follows:

Definition 16. ***Average Differential Causal Rule (ADCR).*** *Given X_1 and X_2 two instances of a target class of an ontology $\mathcal{O}$, P_O the property path leading to the outcome, a treatment $T \in \{T_n(X_1, X_2), T_c(X_1, X_2)\}$ defined by the property path P_T, and s an aggregation function, an average differential causal rule DCR_T is defined as follows:*

$$ADCR_T : T \wedge P_O(X_1, U_{1,P_O}) \wedge P_O(X_2, U_{2,P_O}) \Rightarrow lessThan(s(U_{1,P_O}), s(U_{2,P_O}))$$

An average differential causal rule expresses the average effect of a treatment over the instances of a target class in a knowledge graph. It is to note, as for differential causal rules, that an average differential causal rule can be categorical or numerical depending on the type of treatment path.

Example 11. *A categorical ADCR could be that, on average, being a woman compared to being a man explains a higher life expectancy. A numerical ADCR could indicate that, on average, being younger explains a better athlete rank.*

6.2 Approach: *Dicare-E*

6.2.1 Overview

This section presents *Dicare-E*, an approach that mines average differential causal rules from a knowledge graph. *Dicare-E* is composed of three main parts. In the first part, similar instances across the knowledge graph are matched on their

representation as embedding vectors. The matching is realised given a distance threshold that is estimated. Although two strategies were proposed to estimate this threshold (see [80, 79] for more details), we present the strategy developed in [79] as it is more interpretable and has a smaller complexity in time. In the second part, matches of similar instances are created using the estimated distance threshold and are used to study the effect of treatments on an outcome path. In the last part, the effect of treatments is estimated by computing a causal metric on the set of similar instances.

The workflow of *Dicare-E* is presented in Figure 6.1.

6.2.2 Using Knowledge Graph Embeddings to Find Similar Instances

Problem Statement

The first part of *Dicare-E* consists of defining the method to match similar instances of a knowledge graph. To this end, we propose to use embedding models to match instances based on their vector representation. However, matching instances based on their distance in the embedding space is challenging.

The objective of knowledge graph embedding approaches is to represent entities and relations from a knowledge graph in low-dimensional vectors (see section 3.3.3). As a reminder, the output of embedding models are vectors representing entities and relations of a knowledge graph. Depending on a dataset, a model or another might perform better. While it can be difficult to compare and represent instances with their RDF descriptions, it is much easier with vectors. Indeed, vectors can be compared through various functions such as euclidean or cosine distances. Vectors representation has been studied in many works, including in Wang *et al.* [91] where authors show that similar RDF instances have similar vectors in the embedding space. In Moon *et al.* [50] and Jain *et al.* [39], authors found that similar vectors in the embedding space will represent similar RDF instances. These works motivate the use of embedding approaches to match similar instances based on their embeddings.

However, given a set of instances of a knowledge graph whose vectorial representation has been learned, matching similar instances is not trivial. For example, to find an instance similar to an instance i, by using a distance such as the euclidean distance, one may consider the closer instance in the embedding space, the instance j. While this distance could indicate that j is the closer instance to i in the embedding space, it does not guarantee that they are similar. However, the potential outcome framework indicates that only pairs of similar instances are to be used to compute a treatment effect and that pairs of instances that are not similar should be pruned. Therefore, estimating a threshold d_{tr} is a key element.

The threshold is used to create a set of pairs of similar instances, that is, with a distance d such as $d < d_{tr}$.

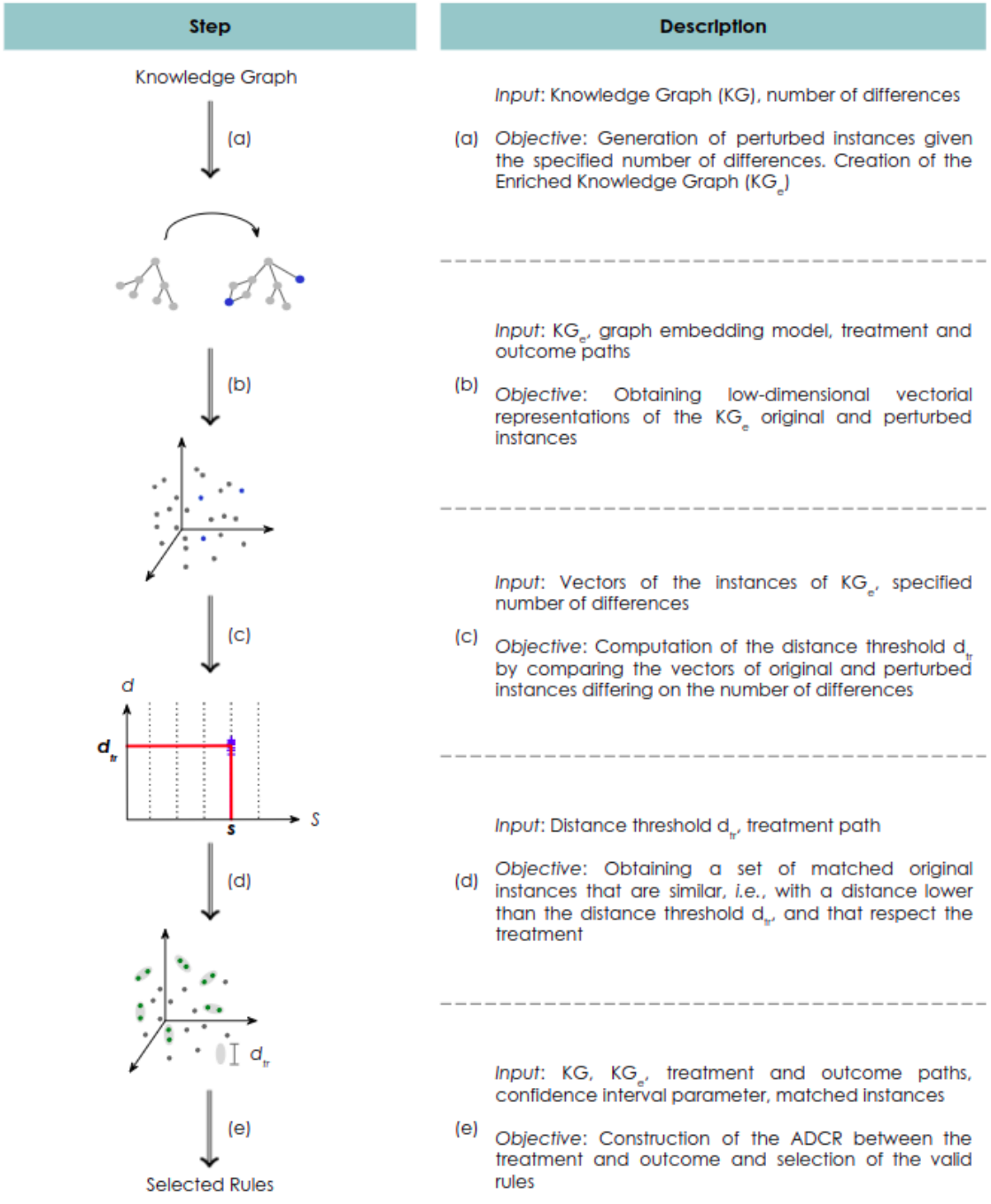

Figure 6.1: General workflow of Dicare-E with the perturbed-based strategy

Estimation of the Distance Threshold

In this section, we propose a method to address the distance threshold estimation problem. As the distance in the embedding space is not interpretable, this method generates a ground truth in the embedding space to make the distance more interpretable. More precisely, the method generates perturbed instances from existing instances given a number of differences. By computing the distance in the embedding space between the existing instances and their perturbed versions, it generates an interpretable ground truth on the distance. The distance threshold can then be estimated in an interpretable way.

Perturbed instances are RDF copies of original instances of the knowledge graphs, except that they are modified in a controlled manner. The perturbed instances are considered as part of ground truth, as their differences from the original instances of the KG are known. Computing the distance between the vector representation of original and perturbed instances within the embedding space allows the estimation of the distance threshold corresponding to a chosen number of differences. The workflow of this method is presented in Figure 6.1, where the threshold d_{tr} is estimated with steps (a), (b), and (c). In the following, we present how we generate perturbed instances and how they are used to define d_{tr}.

In the first step, step (a) in Fig. 6.1, perturbed instances are generated for a subset of class instances that are randomly selected. For each class instance, a specified number of perturbed instances are created. A perturbed instance i' is generated from an existing instance i with the parameter S that expresses the number of differences.

Definition 17. ***Number of differences on a property path.*** *Let P be a property path, and O_i (resp. $O_{i'}$) the set of literal values of P describing i (resp. i'). Let $s_P(i, i')$ be the number of differences on P between i and i'. $s_P(i, i')$ is defined as follows:*

$$s_P(i, i') = max(|O_i|, |O_{i'}|) - |O_i \cap O_{i'}| \tag{6.1}$$

Definition 18. ***Number of differences.*** *Let $\mathcal{P}$ be the whole set of property paths considered for the differences between two instances i and i'. The number of differences between i and i', noted $s(i, i')$, is computed over all property paths of $\mathcal{P}$ as follows:*

$$s(i, i') = \sum_{P \in \mathcal{P}} s_P(i, i') \tag{6.2}$$

It is to note that, in some cases, several URIs may share the same literal values. Therefore, URIs are not considered in the number of differences as instances are

compared by relying on literal values only. For instance, the URIs *#France* and *#france* may be described on one property, the property related to their number of inhabitants, and have the same literal value. Comparing *#France* and *#france* would not make a difference as they have the same literal values. Also, existing identity or difference links between URIs are not considered.

Given the number of differences we defined, perturbed instances are generated so that their number of differences with the original instance equals the parameter S. The perturbation process can modify a literal value and add or suppress a data property, and new literal values are selected among existing values.

Example 12. *In Fig. 6.2, #Fake_ Messi is a perturbed instance of #Messi with $S = 1$, where the difference is on $P = hasAge$. Another #Fake_ Messi with $S = 1$ could be generated by generating a new club, e.g., #Fake_ PSG, with owner #QSI and president #Kahn.*

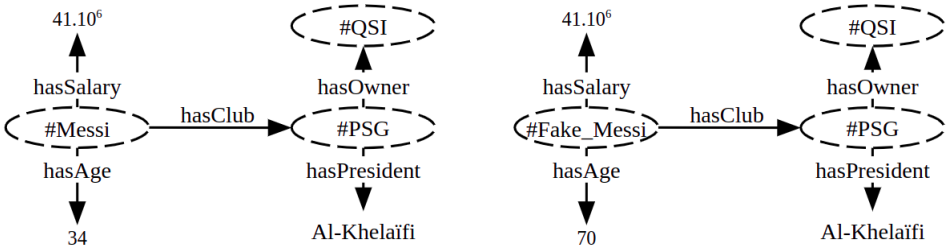

Figure 6.2: Generating a perturbed *#Messi* from the original one

In step (b), an embedding model is trained on the knowledge graph enriched with the perturbed instances, named KG_e. The vector representations of the original and perturbed instances are learned and are used to estimate the distance threshold. The distance threshold is defined as the mean distance between the vector of an original instance and the vector of a perturbed instance that has been created from the original instance using a number of differences. This estimation is performed in step (c).

More precisely, given P_T the property path associated with the treatment T to analyse, and P_O the property path associated with the outcome O, P_T and P_O are removed from KG_e for the embedding model training as the matching of instances is to be carried out independently to T and O. The resulting knowledge graph is denoted by KG_f. An embedding model f is then trained on KG_f. After the training, the embedding vectors of the perturbed instances $\mathcal{G}_S$ and the corresponding original instances I are obtained. A distance threshold d_{tr} is then computed as the mean distance between pairs of instances (i, i') having a number of differences S, where i is an instance of I and i' is an instance of $\mathcal{G}_{i,S}$.

Definition 19. ***Distance Threshold.*** *Given the number of differences S, the euclidean distance d, the sets of original instances I and their perturbed versions $\mathcal{G}_{i,S}$ and $\vec{i}$ (resp. $\vec{i'}$) the embedding vector representing i (resp. i'), the distance threshold d_{tr} is defined as follows:*

$$d_{tr} = \frac{1}{\mid I \mid} \sum_{i \in I} \frac{1}{\mid \mathcal{G}_{i,S} \mid} \sum_{i' \in \mathcal{G}_{i,S}} d(\vec{i}, \vec{i'}) \tag{6.3}$$

6.2.3 Generating Pairs of Similar Instances to Study a Treatment Effect

The effect of a treatment on an outcome is represented by an average differential causal rule (see Definition 16). Obtaining an average differential causal rule is done in two steps. In the first step, the average differential causal rule is created given the studied treatment. In this step, such a rule is qualified as *potential* as its effect has not yet been measured. This step, therefore, consists of obtaining the pairs that verify the treatment. It is represented in step (d) of Figure 6.1. In the second step, the effect of the treatment is computed by using the obtained pairs and is presented in the next section.

Depending on whether a property path $P_T \in \mathcal{P}$ is numerical or categorical, the process of generating a potential rule and the pairs of similar instances used to evaluate it differs. We present the processes for generating potential rules with a categorical or a numerical path as follows.

Categorical Treatment. When P_T leads to categorical values, one or several combinations of values can be built depending on the number of categorical values of P_T. Consequently, one potential rule $ADCR_T$ is generated for each combination (V_i, V_j) of different values of P_T. In the sequel, we will refer to each combination of this form as a treatment T. To create the pairs of similar instances that verify T, a set of instances $\mathcal{I}_{V_i}$ (resp. $\mathcal{I}_{V_j}$) having the value V_i (resp. V_j) is obtained from KG. Then, a distance matrix $\mathcal{D}$ is constructed by computing the distance d between the embedding vectors of all pairs $(i, j) \in \mathcal{I}_{V_i} \times \mathcal{I}_{V_j}$. Given d_{tr}, pairs of instances are added to the set of similar pairs $\mathcal{M}$ if their distance d verifies $d < d_{tr}$. This process is done with a greedy strategy without replacement, as we aim to construct pairs as diverse as possible. In other words, an instance can only appear in one pair at maximum.

Numerical Treatment. When P_T leads to numerical values, one potential rule DCR_T can be generated. The process for obtaining $\mathcal{M}$ is similar to a categorical treatment, except that $\mathcal{D}$ is a squared matrix where each instance with a value for

Table 6.1: Distance between instances when analysing the categorical treatment *gender*.

		$P_P = Man$				
		j_1	j_2	j_3	j_4	j_5
$P_P = Woman$	i_1	**0.7**	1.4	1.2	1.9	1.8
	i_2	1.7	1.4	1.5	**0.8**	1.2
	i_3	3.2	**0.6**	1.0	0.9	1.8
	i_4	0.8	1.2	2.7	1.4	1.5

P_T is both in a row and in a column. Pairs are then drawn without replacement and added to $\mathcal{M}$ if they verify T and if their distance d is lower than d_{tr}.

Example 13. *In table 6.1, we assume that $d_{tr} = 1$. The treatment path is "hasGender" and is, therefore, a categorical path. In this example, pairs are built to study the treatment "T: being a man compared to being a woman", where women are the rows of the table and men are the columns. Three pairs verifying the treatment can be created: (i_1, j_1), (i_2, j_4) and (i_3, j_2). The instances i_4, j_3 and j_5 are not matched as no available instances were similar enough to them.*

6.2.4 Obtaining a Treatment Effect

In the last step of *Dicare-E*, step (e), the effects of the treatments are measured. The metric $causal_r$, defined in Definition 12, is used to assess the effect of the treatment T on the outcome and is therefore used to select the rules from the set of potential rules built in the previous step.

The procedure to measure the effect of a rule is the same as described in 4.2.3, to the difference that an average effect is measured instead of a local effect on a strata. Let a potential rule $ADCR_T$ and a set of similar pairs $\mathcal{M}$ that verify the treatment T. As a reminder, $causal_r$ is defined in Definition 12. It is computed as follows:

$$causal_r(R) = \frac{supp(T \wedge \vec{B} \Rightarrow lessThan(s(U_{1,P_O}), s(U_{2,P_O})))}{supp(T \wedge \vec{B} \Rightarrow greaterThan(s(U_{1,P_O}), s(U_{2,P_O}))} \tag{6.4}$$

$causal_r$ compares the number of pairs of $\mathcal{M}$ that verify the outcome to the number of pairs of $\mathcal{M}$ that verify the opposite of the outcome, *i.e.*, $greaterThan$ instead of $lessThan$.

It has values in $[0, +\infty[$ and measures the strength of the relation between the treatment and the outcome. The treatment and the outcome are considered independent if $causal_r(ADCR_T) = 1$. On the contrary, the more distant to 1, the stronger the strength of the relation, *i.e.*, the higher the effect of the treatment on

Table 6.2: Values of instances on an outcome path

Instance	Value on P_O
i_1	0
i_2	1
i_3	1
j_1	1
j_2	0
j_4	0

the outcome. If higher than 1, the treatment has an effect on the outcome, while a value lower than 1 indicates that the treatment has the opposite effect on the outcome (*i.e.* $greaterThan$).

Example 14. *In table 6.2, by using the pairs created in example 13, the treatment T: "being a man compared to being a woman" has an estimated effect:* $causal_r(ADCR_T) = \frac{2}{1}$.

To select the rules expressing an effect of a treatment on the outcome, a confidence interval is built on the causal ratio. A rule with a lower bound of the confidence interval higher than 1 is selected as expressing an effect.

When a user wants to test other treatments or study other outcomes, the embedding model f is to be retrained as pairs are to be created without considering the treatment and outcome.

6.3 Experiments

The experiments aim to show the relevance of the discovered rules, meaning (i) whether they were judged meaningful by domain experts and (ii) their ability to explain differences in outcome values, *i.e.*, the number of pairs having the outcome that can be explained by at least one rule. As *Dicare-E* relies on graph embeddings, we also evaluated how robust *Dicare-E* is to missing data. Moreover, we compared the results of *Dicare-E* to both a baseline and to *Dicare-C* [78]. The baseline we considered is a naive approach that discovers DCRs with strict literal-based matching (see section 4.3).

An extract of the code can be found on this link[1].

[1]https://github.com/WebIntelligence2022DCRMining/Submission

6.3.1 Datasets and settings

Datasets

The datasets *Vitamin* and *DBpediaW* have been used for the experiment. These datasets have been used in the previous chapters and are described in section 4.3.1.

Vitamin describes people and their socioeconomic characteristics such as age, gender, current and ideal diet, opinions on animal welfare facts and climate change. We are interested in explaining the difference between a person's current and ideal diet, *i.e.*, their willingness to reduce their meat consumption denoted by *reduceMeat* having values $\in \mathbb{N}$.

DBpediaW describes authors and books they wrote. We try to explain why some writers publish their first book at a younger age than others. The treatment path is denoted by the predicate *publishedAt* that expresses the difference between a writer's birth date and his first book publication date.

As the outcomes of both datasets are single values, no aggregation functions s are used.

Experiments Settings

The parameters for the perturbed instances generation are $S = 15$, $n = 150$ and $m = 3$ for *Vitamin* and $S = 2$, $n = 50$ and $m = 3$ for *DBpediaW*. The values for S were defined to select pairs with a number of differences lower than one-third of their number of literals. The quality metric $causal_r$ is computed with the confidence of 90% for both methods.

The treatment paths that were studied for *Vitamin* include people's gender, age, type of habitation, work, education, source of media, current diet, and sets of various opinions on climate change and animal welfare. The treatment paths that were studied for *DBpediaW* include people's birth date, gender, writing genre, and the universities they studied at.

For the baseline and *Dicare-C* [78], we defined the maximal length of paths to explore as the length of the longest path of an instance description of the target class, *i.e.*, a $Person$ for *Vitamin* and a $Writer$ for *DBpediaW*.

To select the most suitable embedding model for each dataset, we trained embedding models using the python library *AmpliGraph* [10].

6.3.2 Selection of the Knowledge Graph Embedding Model

Table 6.3 presents the performances of the embedding models trained over *Vitamin* and *DBpediaW* on a link prediction task. The metrics described in this table are the mean reciprocal rank (equation 3.18) and the Hits@n (equation 3.19). $ConvE$ performs the best on *Vitamin*: it obtains the highest *MRR* and *Hits@n* values,

and is therefore used in the following. *DistMult* has the best performances on *DBpediaW* and is used.

Table 6.3: Models performance evaluation metrics on *Vitamin* and *DBPediaW*

	DBPediaW				***Vitamin***			
	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
ConvE	0.2292	0.1881	0.2598	0.2844	**0.3384**	**0.2498**	**0.3969**	**0.4850**
DistMult	**0.2736**	**0.2004**	**0.3169**	**0.3705**	0.2167	0.1341	0.2451	0.3675
ComplEx	*NA*	*NA*	*NA*	*NA*	0.1732	0.1011	0.1867	0.3113
TransE	0.2088	0.1737	0.2099	0.2786	0.1470	0.1341	0.1563	0.2474
Baseline	0.0007	0.00014	0.0029	0.0081	0.0057	0.0009	0.0023	0.0081

The performances were obtained by training the models with the original knowledge graphs. We assume that the models with the best performances on the whole knowledge graphs will also perform best when removing the treatment and outcome paths.

The runtime of *Dicare-E* for running the 7 treatment paths that were tested is around 9 hours for *Vitamin* and around 10 minutes for *DBpediaW*.

6.3.3 Effect of the number of differences S on the distance d and the causal effect $causal_r$

A key parameter in the generation of perturbed instances is the number of differences S. This section provides an analysis of this parameter. It first focuses on the correlation between S and the distance d, and then on the effect of S on the causal metric $causal_r$.

Correlation between the number of differences S and the distance d

The variation of the euclidean distance d with the number of differences S is studied. To this end, an experiment is designed as follows: 100 instances of $Person$ in *Vitamin* are randomly selected, and, for each of them, one perturbed instance is generated per number of differences S ranging from 0 to S_{max}. S_{max} is set to 24 for this study, *i.e.*, in average half of the literals of instances of $Person$ can be different.

After the generation step, an embedding model is trained over *Vitamin* enriched with the perturbed instances. The euclidean distance d is computed between the selected instances of $Person$ and the perturbed instances generated from them given S. Fig. 6.3 is a violin plot showing the distribution of d between pairs of instances given each number of differences S from 0 to S_{max}. The wider line in

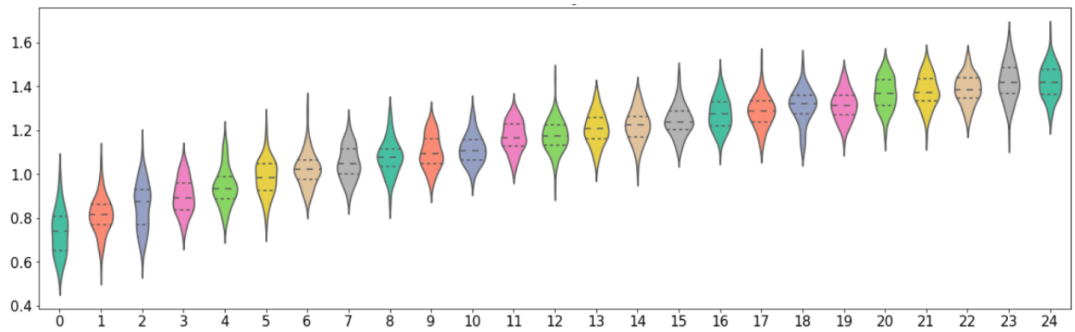

Figure 6.3: Evolution of d between pairs of instances (y-axis) depending on S (x-axis)

the middle of each violin indicates its median value. Similarly, as for the first method, Fig. 6.3 shows that, as expected, higher S values lead to higher euclidean distances.

Effect of the number of differences S on the causal effect $causal_r$

The previous section showed that different values of S will lead to different estimations of the distance threshold. As a consequence, the estimated effect of a treatment may be different depending on the parameter S since the set of selected pairs will be different.

To underline the importance of S, the rule R_1 : *being educated instead of being not educated explains a higher will to reduce one's meat consumption* is studied, where *educated* stands for possessing a high school degree or a higher degree. Several metrics on R_1 are described in Fig. 6.4. In this figure, the x-bar indicates the values of S. The left y-bar indicates the values of $causal_r(R_1)$, and the right y-bar indicates the number of pairs created to estimate $causal_r(R_1)$. The black circles show the values of $causal_r(R_1)$, and the vertical bars, named error bars, are the corresponding confidence intervals of $causal_r(R_1)$. The histogram indicates the number of pairs created to estimate $causal_r(R_1)$.

A rule is valid if the error bar does not cross the horizontal red bar set at 1. In this example, R_1 is not valid for $S = 10$ as, although its $causal_r$ value is lower than 1, its confidence interval ranges from 0.5 to 1.1 and, therefore, includes the value 1, meaning it is not significantly different from 1. R_1 is valid with $S = 20$ since the confidence interval is lower than 1. Moreover, since $causal_r(R_1) < 1$, the rule is changed to *being not educated instead of being educated explains a higher will to reduce its meat consumption.*

Smaller S values, *i.e.*, lower d_{tr} values, lead to smaller sets of pairs. Such sets are composed of the most similar instances. As their number is small, the

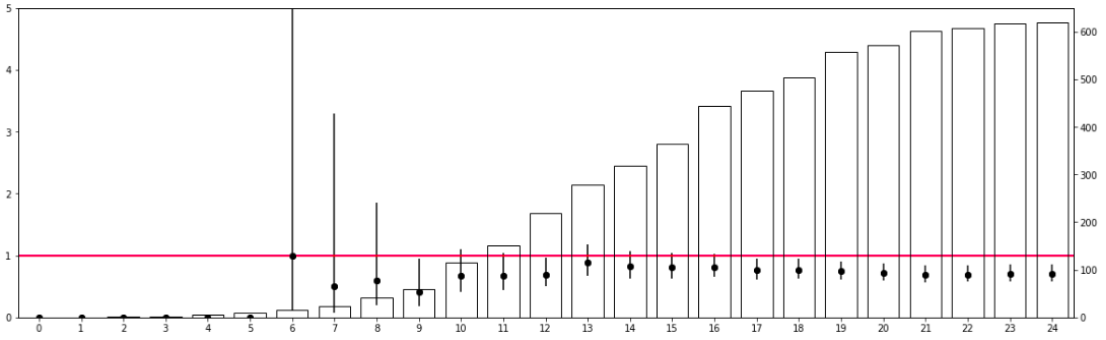

Figure 6.4: $causal_r(R_1)$ (left y-axis, bullets) and number of pairs created (right y-axis, bins) given S values (x-axis). P_T: education (X_1: educated, X_2: not educated)

estimation of $causal_r$ has a large variance, *i.e*, large confidence intervals. Conversely, higher S values lead to bigger sets of pairs that, however, are less similar. The estimation of $causal_r(R_1)$ is more likely to be biased but has a smaller variance. Therefore, S allows controlling the bias-variance trade-off of the estimation of $causal_r$. The more the dataset is large and representative of the real-world distribution, the more likely it is to find similar instances with lower S values. The quality of the discovered rules would therefore be higher.

6.3.4 Quantitative and Qualitative Evaluation of the Mined Rules

22 rules were discovered on *Vitamin* and 3 rules on *DBpediaW*. Two rules, one rule for each dataset, are presented: R_2 (equation 6.5) for *Vitamin*, and R_3 (equation 6.6) for DBpediaW [2]. R_2 states that being omnivorous compared to being vegetarian is an explanation for a higher will to reduce one's meat consumption. R_3 states that being born after another author, *e.g.*, in 1980 compared to 1900, is an explanation for a younger age of first book publication.

$$\begin{aligned} R_2 : \mathbf{hasDiet}(\mathbf{i_1}, \mathbf{vegetarian}) \wedge \mathbf{hasDiet}(\mathbf{i_2}, \mathbf{omnivorous}) \wedge \\ reduction(i_1, v_1) \wedge reduction(i_2, v_2) \Rightarrow v_1 < v_2 \end{aligned} \tag{6.5}$$

$$\begin{aligned} R_3 : ageFirstBook(i_1, u_1) \wedge ageFirstBook(i_2, u_2) \wedge \\ \mathbf{birthYear}(\mathbf{i_1}, \mathbf{v_1}) \wedge \mathbf{birthYear}(\mathbf{i_2}, \mathbf{v_2}) \wedge \mathbf{v_1} > \mathbf{v_2} \\ \Rightarrow u_1 < u_2 \end{aligned} \tag{6.6}$$

[2]The predicates *belong* have been omitted to simplify the reading.

The following section provides an analysis of the mined rules. The summary of the results is presented in Table 6.4. In this table, we give for each KG and each approach tested, that is the baseline, *Dicare-C* and *Dicare-E*, the number of discovered rules, the average number of rules per pair of instances, the percentage of pairs evaluated as correct by the experts, and the number of explained pairs.

Table 6.4: Results of the evaluation of rules mined using a baseline, *Dicare-C* and *Dicare-E*.

	DBPediaW			***Vitamin***		
	#Rules (Per pair)	%Correct	%Pairs	#Rules (Per pair)	%Correct	%Pairs
Baseline	0 (0)	*NA*	0	0 (0)	*NA*	0
Dicare-C	**12** (1.1)	83.3	16.5	**44** (1.1)	80.0	52.5
Dicare-E	3 (**1.4**)	**100**	**84.3**	22 (**2.5**)	**100**	**85.7**

Vitamin. Two researchers in nutrition and human behaviour evaluated the 22 rules from *Dicare-E* and the 30 best rules from *Dicare-C*. This evaluation has been done by following the same protocol as for *Dicare-C* (see section 4.3), where the choices for a rule are (i) *seems relevant*, (ii) *may be relevant*, (iii) *does not know* or (iv) *seems false*. No rules were mined with the baseline, as no exact matches could be found. The rules of *Dicare-E* were all judged relevant or potentially relevant, while 5 rules of *Dicare-C* had an effect judged unsure or false. The researchers mentioned that the more specific the strata of the rules from *Dicare-C*, the harder they were to understand. Moreover, *Dicare-C* discovers rules that involve computed abstract properties, while *Dicare-E* approach involves precise property paths that belong to the original KG. For instance, *Dicare-C* considers only one abstract treatment that represents a set of paths on animal welfare and global warming, while *Dicare-C* mines distinct rules that involve distinct treatments corresponding to the aforementioned paths.

Besides the quality of the rules, *Dicare-E* explains much more pairs of instances having the outcome (85.7% vs 52.5% for *Dicare-C*), and, for each pair, *Dicare-E* provides in average twice as many explanations (2.5 rules vs 1.1 rules per pair). Moreover, *Dicare-E* is more robust as removing 15% of the triples reduces the percentage of pairs explained from 85.7% to 83.2% (22 to 17 rules, with the 17 rules being included in the 22 original rules) in our approach and from 52.5% to 35.8% for *Dicare-C*.

DBPediaW. As this dataset is rather simple, a group of 5 researchers in computer science has been selected to review the rules. No rules were mined with

the baseline. By considering the votes of the reviewers, the percentage of rules considered as could be relevant or relevant is 83.3% for *Dicare-C* and 100% for *Dicare-E*. There was no disagreement in the expert's judgements. As for *Vitamin*, the percentage of pairs explained is higher with our approach and is more robust to missing data, *i.e.*, removing 15% of the triples, as it decreases from 84.3% to 33.5% for *Dicare-E* (from 3 to 1 rule) and from 16.5% to 3.7% for *Dicare-C*.

6.4 Discussion

Dicare-E has been developed as an attempt to solve some of the limits raised for *Dicare-C*, namely the interpretability of the rules it mines, the number of pairs explained and the instance matching criteria. This section compares both approaches on a set of criteria, including the aforementioned ones.

Rule Interpretability. *Dicare-C* mines differential causal rules that express a local effect of a treatment on a subset of instances. In the obtained set of rules, the same treatment may appear in several rules, making it possibly difficult to interpret. To overcome this limit, we developed an approach that generates counter effect rules in Chapter 5. However, a differential causal rule alone remains difficult to interpret. *Dicare-E* discovers *average* differential causal rules, which were judged more interpretable by experts than differential causal rules.

Rule Expressiveness. While discovering average effects lead to more interpretable rules, it however implies a loss of expressiveness in the treatment effects. Indeed, an average effect may hide local effects, such as different effects of one treatment. A trade-off exists between interpretability and expressiveness. It is to note that, however, the average differential causal is more expressive when it comes to the treatment they express. *Dicare-C* relies on a community detection step, which results in gathering paths under abstract properties that will compose some of the treatments. Such abstract properties, while having a meaningful semantics, can only express the combined effects of the properties they are composed of and not the precise effect of one property in particular. By matching instances on their embedding representations, *Dicare-E* can assess any path as a treatment.

Number of Pairs Explained. As a consequence of their average effects, average differential causal rules can be applied to much more pairs of instances than differential causal rules. Therefore, they provide many more possible explanations as to why the outcome can be observed. Not only do the rules explain more pairs, but they also result in providing more explanations for one pair having the outcome.

Instance Matching Criteria. *Dicare-E* uses instances vector representations to match similar instances, while *Dicare-C* relies on instances RDF descriptions. Using the embeddings presents two features of interest. First, it allows matching instances even if they have missing values, which is not possible in *Dicare-C*. As a consequence, *Dicare-E* is more robust to missing data. Secondly, it enables an easy ranking of best matches with the distance between embedding vectors. This ranking makes it possible to create the best matches with a greedy matching strategy, which has been shown to perform well [25]. In *Dicare-C*, pairs are created from sets of similar instances based on their values on a given set of paths only, making it not possible to rank potential pairs.

Runtime. Training an embedding model is computationally high. Consequently, the runtime of *Dicare-E* is higher than the runtime of *Dicare-C*. Training a model only once instead of training it for each treatment before creating pairs of similar instances would reduce the runtime. However, this setting has been tested and led to a different set of constructed pairs. Consequently, using the once-trained model could lead to biased results, and the models are retrained for each treatment.

Feasibility on Knowledge Graphs. *Dicare-C* relies on a pre-processing step that aims to discover communities within a knowledge graph. Such communities lead to the creation of abstract properties that are used to simplify the matching procedure. However, all knowledge graphs might not be suited for community detection. *Dicare-E*, on the contrary, does not rely on a pre-processing step and seems therefore better suited to be applied to a larger diversity of knowledge graphs. However, as it relies on embeddings, it can only be applied to knowledge graphs in which entities and relations representations can be correctly learned by an embedding model.

It appears that *Dicare-C* and *Dicare-E* seem complementary and may be adapted to different use cases and knowledge graphs. *Dicare-C* discovers expressive rules displaying local effects, whose interpretation can be rather difficult, but that is rather fast to obtain. Such local effects may be of interest to users that may investigate them. On the other hand, *Dicare-E* has a higher runtime but discovers rules displaying average effects, on more precise treatments, that are easier to interpret. *Dicare-E* seems adapted to be used on any knowledge graphs that can be represented in an embedding space, while *Dicare-C* is suited for small knowledge graphs and bigger knowledge graphs where communities can be discovered.

6.5 Conclusion

In this chapter, we defined a type of causal rule, *i.e.*, average differential causal rules, that express the average effect of a treatment on an outcome. We also proposed *Dicare-E*, an approach for mining such rules from a knowledge graph.

Dicare-E is based on the potential outcome framework. More particularly, it matches similar instances by using their embedding vector representations. As defining the similarity of two vectors on their distance alone in the embedding space is not trivial, we proposed two methods to define a threshold in this space to ensure the creation of similar pairs only. The created pairs are used to mine average differential causal rules that can both describe a numerical or a categorical treatment. Such rules were evaluated as interpretable by experts and are able to explain many pairs having the studied outcome.

The comparison of *Dicare-E* with the previous approach, *Dicare-C* [78], shows that it improves the results regarding some limits raised in section 4.5. More precisely, the average differential causal rules are **more interpretable** and **explain more pairs** having the studied outcome. Also, as the **matching relies on embeddings**, it is possible to match instances whose description is partly missing and ease the matching process by finding the more similar instances. *Dicare-E* is more time-consuming than *Dicare-C* as it must train an embedding model. Lastly, in terms of **feasibility on knowledge graphs**, *Dicare-E* can be used on any knowledge graphs that can be represented in an embedding space, while *Dicare-C* is suited for small knowledge graphs and knowledge graphs where communities can be mined.

Although *Dicare-E* improves several criteria compared to the previous approach, there are still limitations to this work. The first concerns the evaluation of the rules. The experiments were applied on the same knowledge graphs as *Dicare-C* where no ground truth is available, and there are up-to-date no knowledge graphs where the ground truth is known. The second is on the sensibility of *Dicare-E* to its parameters. It has been shown that changing the distance threshold with different parameters for the matching step may lead to various results in terms of rules. The definition of the parameters used to estimate the distance threshold should therefore be done before running the approach to control the similarity of the selected pairs to respect the potential outcome framework.

Chapter 7

Conclusion and Perspectives

To conclude the thesis, this chapter provides both the highlights of the results of the work that has been conducted, as well as possible directions for future work.

7.1 Summary of the Results

The objective of this thesis is to define how causal relations can be discovered within a knowledge graph and represented as causal rules. To this end, several approaches were developed and evaluated and are discussed in this section.

In Chapters 2 and 3, we defined what knowledge graphs are and presented a bibliographical study of the problem of causality in general and, more specifically, in knowledge graphs. The study focused on (i) the definition of a causal relationship and the frameworks for discovering this type of relationship, including the potential outcome framework; (ii) the study of the different rules that can be discovered from a knowledge graph, including association rules, and the demonstration of why these rules are not suitable for the expression of causal relations; (iii) the study of approaches allowing the determination of similar instances within a knowledge graph, constituting the basis of the potential outcome framework, for the study of the potential results that one seeks to apply to knowledge graphs.

State-of-the-art has made it possible to highlight the following elements:

- **Several frameworks exist to discover causal relations**. The definition of a causal relationship is the subject of research in philosophy and statistics. Several frameworks exist for determining a causal relationship, including the potential outcomes framework, based on comparing similar instances to which a different treatment is applied.

- **Current rule mining approaches, such as association rule mining, are not suited for causal discovery.** Although association rule mining

is an important topic, this type of rule only allows the expression of an association because of its syntax and the way these rules are mined.

- **Both symbolic and numerical approaches are used to determine similar instances within a knowledge graph.** In particular, graph concepts allow one to define a conceptual distance, and graph embeddings allow one to learn vector representations of instances between which a distance can be computed. However, these distances alone do not determine whether two instances are sufficiently similar for a causal discovery problem.

In Chapter 4, we presented two approaches [81, 78] that define and mine differential causal rules. These rules express the effect of a cause variable, named treatment, on a variable named result, for a set of instances defined by a graph pattern named strata. These rules thus provide explanations as to why two instances have different results. This chapter shows that such rules expressing a causal relationship can be obtained by adapting the potential outcome framework. The developed approaches have been tested and evaluated by experts on two datasets, including a dataset named *Vitamin*. *Vitamin* describes people, their eating habits, and their opinions on food and climate change issues. Experts have found the rules determined on Vitamin to be broadly interpretable and meaningful. However, these rules have some limitations, including that (i) some rules are difficult to interpret, as one treatment can have opposite effects, (ii) their ordering is not clearly defined, and (iii) the rules only explain a small proportion of instances pairs with a different outcome.

In Chapter 5, we described an approach [77] that defines and mines counter effects rules. The definition of these rules follows the limitation expressed by the causal differential rules of chapter 4 on the interpretation of the rules. It allows for facilitating the interpretation of the rules. In particular, counter effects rules show that one treatment can have opposite effects depending on the sets of instances to which it is applied. They also show the differences between these sets of instances that explain the opposing effect. The search for counter effects rules allows us to (i) clarify the interpretation of causal differential rules and (ii) rank treatments by looking at, first, how consistent their effects are in the rules where they appear and, secondly, their propensity to change the meaning of other treatments.

In Chapter 6, we presented a new type of rule named average differential causal rules [80, 79]. Unlike differential causal rules, which express the effect of a treatment on a subset of instances, average differential causal rules express the average effect of a treatment on all instances of a knowledge graph. We developed a hybrid approach that mines such rules. The mined rules are (i) simpler to interpret and (ii) explain significantly more differences in the outcome than differential causal rules. This hybrid approach is based on the potential outcome framework and

knowledge graph embeddings. Learning graph embedding vectors representing the instances of a knowledge graph allows us to determine similar instances across the entire graph, even if the descriptions of the instances are incomplete. An important result was raised from the experimental evaluation, as this approach is more robust to missing data than previous approaches.

To conclude, we showed in this thesis that causal relations in knowledge graphs can be represented as rules that we named differential causal rules. These rules provide explanations as to why instances of a target class of a knowledge graph have different outcome values and are obtained by adapting the potential outcome framework. The potential outcome framework is a standard framework to discover causal effects. The method of this framework that has been adapted is matching, which consists of comparing similar instances that differ in the treatment path to study. We showed that discovering rules using matching could be done by several methods, *i.e.*, symbolic or hybrid methods, that are more or less resistant to missing data in a knowledge graph. The causal effects expressed in the rules are explained by treatments. They may express a local effect, *i.e.*, over a subset of instances of the target class described by a graph pattern, or an average effect, *i.e.*, over all instances of the target class. A trade-off exists between rules expressing a local or an average effect. Rules with local effects are more expressive but may be harder to interpret and provide explanations to a small number of pairs having different outcome values. On the other hand, rules expressing average effects are easier to interpret and explain more pairs but are less expressive.

7.2 Discussion and Future Work

In this section, in regard to the limits of our contributions, we propose perspectives for future work. We structure these directions in two categories: short-term perspectives and long-term perspectives.

7.2.1 Short-Term Perspectives

Rules Generalisation Heuristic. In *Dicare-S* and *Dicare-C*, presented in Chapter 4, differential causal rules are obtained in several steps, including the discovery of specific differential causal rules and the generalisation of specific rules to more general ones. This generalisation is made by verifying a consistency constraint, *i.e.*, that states that two specific rules can be merged only if they both verify the same treatment and effect. However, in practice, there are many cases where the effect of a treatment on a strata can be unknown. This can be explained by a lack of data or by a strata where no instances are found, as it could lead to inconsistent descriptions. Consequently, the effect of a treatment on such strata can not be

verified, and these strata can not be part of more general rules due to the strict generalisation process. This results in rules that can not be generalised due to missing information and that are, therefore, more difficult to interpret than more general rules. While this generalisation process ensures the discovery of rules with strong confidence, we believe another generalisation process could be tested. In such a process, a less strict heuristic could be defined, such as generalising all rules except the ones where the treatment has been shown to have no effect.

Rules Analysis. Both differential causal rules and average differential causal rules can be used to explain the outcome of pairs of instances. Also, several rules can provide an explanation for one pair. In this case, either all rules explain the same outcome of the pair, *i.e.*, given the treatment of the rule that is verified by the pair, the pair also verifies the outcome of the rule, or rules may provide opposing explanations, *i.e.*, the treatment of the rules are verified by the pair, but the outcome of the pair is not always the outcome of the rules. In *Dicare-C* and *Dicare-E*, a pair is considered to be explained if it verifies at least the treatment and the outcome of one rule. However, a deeper analysis of the rules may be provided. Chapter 5 proposes a first analysis of differential causal rules obtained in Chapter 4 when treatments may lead to opposing effects depending on the strata they are applied to. We believe such an analysis could also be done on the rules obtained in Chapter 6, to analyse whether relationships between treatments can be obtained, *e.g.*, if they lead to the same or opposing outcomes. Such relations could be used to propose an ordering between the rules in addition to their support and their causal metric.

Concepts of Nearest Neighbours. Our approaches adapted the potential outcome framework, especially the matching method. Matching consists of creating pairs of similar instances. As the literature mentioned [1, 28] and as our work confirmed [78, 79], matching becomes increasingly difficult with the number of properties or paths that describe the instances. The literature indicates that the set of key properties that should be controlled is to be defined carefully. In our symbolic approaches [81, 78], this set is defined by an expert. In our hybrid approaches [80, 79], the best matches are obtained using embedding vectors that capture a representation of the instances. However, while the embedding-based distance is easy to use to select the closest matches in a knowledge graph, it is however hardly interpretable. Although the quality of a matching process is hardly measurable without ground truth, we believe other matching strategies could be applied. For instance, we believe that using the concept of nearest neighbours [18] can be an interesting lead to match similar instances in an interpretable way. Moreover, it does not require a training phase as embeddings do.

7.2.2 Long-Term Perspectives

Approaches Generality. In order to show the generality of our approaches, it would be interesting to apply them to knowledge graphs other than *Vitamin* or *DBpediaW*. However, the approaches developed in this thesis can not be used for all knowledge graphs. As discussed in Chapter 2, knowledge graphs diversity can be observed by considering their content, which can describe expert or general knowledge, or their size. We list the features a knowledge graph should possess to apply our approaches. Firstly, such a knowledge graph should describe instances of a target class with an outcome path of interest, *i.e.*, a property path to explain. Indeed, many knowledge graphs, such as wikidata, provide general descriptions of instances and rarely the same paths for all instances. In this case, applying our approaches would not be relevant as there would not necessarily be a path of interest to explain. More generally, we believe that our approaches are to be used on domain-specific knowledge graphs. Secondly, as our approaches rely on the potential outcome framework, the estimations of causal effects represented in causal rules are computed by comparing similar instances. While assessing this similarity becomes difficult with the increasing number of paths describing instances, it is also difficult to measure when only a few describe instances. Indeed, in this situation, creating pairs could be rather easy. However, a low number of paths may not discriminate enough, as the instances of created pairs could still be rather different. Therefore, our approaches should be applied to knowledge graphs with a minimum number of paths describing their instances. Lastly, regarding the size of the knowledge graph, our approaches have been tested on small and medium-sized knowledge graphs. We believe that both *Dicare-C* and *Dicare-E* could be applied to larger knowledge graphs by adapting their settings.

Rules Validation. While our approaches rely on experts' validation to judge and select the causal rules from the set of discovered rules, we believe there are other ways to select the causal rules. More precisely, at least two sets of methods could be used. First, statistical methods such as refuting methods could be adapted. Their objective is to discard rules that have no effects. An example of such a method consists of changing the values of one variable, such as the treatment or the outcome, to random values and observing whether an effect is still measured or not. While such methods could discard rules that have no effects, they are, however, not able to make the distinction between rules expressing a causal relation or a correlation. Consequently, the second set of methods to adapt would be methods selecting causal relations. To this end, however, no statistical methods exist but only expert knowledge, as done in this thesis. For instance, when learning a structural causal model, which is the main framework in causal discovery with the potential outcome framework, the estimation of causal effects is computed on

treatments selected by the user in a pre-processing step. Independence between a variable and the outcome is also to be indicated by the user. Therefore, future approaches should (i) discard correlations between property paths before running the approaches, as already defined in our approaches, and (ii) ensure a rigorous validation process of the rules by the experts. While a validation process has already been defined in this thesis, it could be improved by using more experts or by applying consensus ranking techniques to select the most valid rules.

Rules Interpretability. Two types of rules expressing a causal relationship were defined in this work, namely differential causal rules and average differential causal rules. Given instances of a target class, the first underlies the causal effect of a treatment on a subset of instances defined by a basic graph pattern, while the second focus on an average effect of a treatment on all instances of the target class. Both rules are used as explanations as to why two instances may have a different result for a given outcome property path. The evaluation of both types of rules showed that the rules expressing average effects were easier to interpret and could explain more differences in the result. However, they are also less expressive as they display an average effect which can hide heterogeneous local effects. As a conclusion to rules interpretability, we believe that our approaches can be extended to provide even more interpretable rules. It would be possible, for instance, to target strata where a treatment has a particularly strong effect instead of outputting all possible rules. This could lead to displaying only the stronger treatment effects to experts instead of all possibilities, easing the interpretation of the rules. Such rules could be obtained in a post-processing step of the set of average differential causal rules, for instance, by analysing the frequent descriptions of instances where a rule is valid. To give even more insights to the experts, we believe that another future work on the rules definition could be, in opposition to focusing on treatment effects, to display strata where treatments have no effects. In other words, such strata would not be sensitive to interventions, and this information would be useful to experts in experimental or policy design, for instance.

Semantic Representation of Causal Relations. In this thesis, causal relations are represented as rules which are one possible way to represent this kind of knowledge. Besides, rule and knowledge can be represented as other components, such as axioms or relations defined within an ontology. While mining rules are of interest to a user as they can be easily interpreted, other forms of knowledge representation could also be relevant to the representation of causal relations. One possible example of other causal relations approaches outputs could be predicates that specify that one relation causally links two or more predicates and under which conditions. One recent work [67] proposed the first version of an ontology

that could include such knowledge. Overall, developing this ontology could lead to the ability to express causal relations in a more precise manner in addition to rules.

Multiple Knowledge Graphs. In the context of Linked Open Data (*LOD*), knowledge graphs can be linked if they describe the same or similar content. Therefore, a research question of interest for future work could investigate how several knowledge graphs can be dealt jointly to discover causal relations. While extensive work can be found in the literature on the discovery and management of identity across knowledge graphs, this work would rather focus on discovering similar instances across knowledge graphs. This task is not trivial, given differences that may exist between different knowledge graphs. It would likely adapt ontology alignment methods to compare instances in the same referential.

Chapter 8

Résumé en Français

8.1 Introduction

Les graphes de connaissances permettent de stocker des données decrivant le monde, ses individus et les relations qui les lient. Depuis la fin des années 2010, l'étude et l'utilisation des graphes de connaissances connait un essor grandissant, aussi bien dans le milieu académique, *i.e.*, chercheurs en université, que de la part des entreprises privées, *e.g.*, Google, Microsoft, etc. Les données decrites par les graphes de connaissances peuvent être des connaissances encyclopédiques, à l'image de *Yago* ou de *DBpedia*, ou bien des connaissances de domaines spécifiques, tels la musique ou l'agronomie.

Plusieurs thématiques d'étude des graphes de connaissances existent. Parmi celles-ci, l'on trouve notamment la fouille de règles d'association, qui, à partir d'un graphe de connaissances, permet d'obtenir un ensemble de règles permettant à la fois la découverte de nouvelles connaissances, ainsi que le nettoyage d'un graphe de connaissance en supprimant des données erronnées et en ajoutant des données manquantes. Les connaissances découvertes par des règles d'association relèvent, comme leur nom le suggère, d'associations. De manière générale, une association indique l'existence d'un lien statistique entre deux variables. Un tel lien peut indiquer une relation de corrélation, de causalité, ou bien résulter d'un fait aléatoire. En conséquence, en terme de découverte de connaissances, une association exprimée par une règle d'association peut aussi bien indiquer une relation qui aura une valeur ajoutée pour l'utilisateur d'un graphe de connaissance, ou bien n'aura pas de sémantique exploitable car relevant d'une simple association.

Une association regroupe donc un ensemble de relations, dont les relations causales. De façon informelle, la causalité est l'influence par laquelle un événement, *i.e.*, une cause, contribue à la production d'un autre événement, *i.e.*, un effet, où la cause est responsable de l'effet, et l'effet est dépendant de la cause. La recherche

de relations causales est l'objet de nombreuses expériences dans une diversité de domaines. Par exemple, en santé, un médicament est étudié afin de déterminer s'il permet de soigner ou non une maladie. En marketing, des tests sont réalisés afin de décider quelle campagne de publicité est à privilégier pour maximiser les ventes d'un produit.

L'étude de relations causales est donc un sujet de grande importance. Cependant, la fouille de ce type de relation au sein de graphes de connaissances est un sujet très peu étudié. De ce fait, l'objectif de cette thèse est de découvrir des relations causales au sein de graphes de connaissances. Dans cette thèse, nous nous intéressons plus particulièrement à développer de nouvelles approches de découverte de relations causales en appliquant le cadre d'étude des résultats potentiels, *potential outcome framework* en anglais, qui cherche à estimer l'effet d'une variable sur une autre. Ainsi, d'autres cadres d'études, comme les modèles causaux structuraux, ne sont pas étudiés dans cette thèse. L'incorporation de représentations causales dans l'apprentissage de modèles d'intelligence artificielle est un sujet qui n'est également pas adressé par cette thèse.

8.2 État de l'art

Le chapitre 3 fournit une étude de l'état de l'art concernant la causalité au sein des graphes de connaissances. L'étude peut se diviser en plusieurs parties, dont (i) la définition d'une relation causale et les cadres d'études de la causalité permettant de découvrir ce type de relation; (ii) l'étude des différents types de règles pouvant être découvertes depuis un graphe de connaissances, dont les règles d'association, et la démonstration de pourquoi ces règles ne sont pas adaptées à l'expression de relations causales; (iii) l'étude des approches permettant la détermination d'instances similaires au sein d'un graphe de connaissances, cet élément constituant la base du cadre d'étude des résultats potentiels que l'on cherche à appliquer aux graphes de connaissances. L'état de l'art a permis de mettre en évidence les éléments suivants :

- **Plusieurs cadres d'études existent pour découvrir des relations causales.** La définition même d'une relation causale est l'objet de recherches en philosophie et en statistiques. Plusieurs cadres d'études existent pour déterminer une relation causale, dont le cadre d'étude des résultats potentiels qui est basé sur la comparaison d'instances similaires sur lesquelles un traitement différent est appliqué.

- **Les règles d'association ne permettent pas d'exprimer une relation causale.** Bien que la fouille de règles d'association soit un sujet important,

ce type de règles permet seulement d'exprimer une association du fait de sa syntaxe et de la façon dont ces règles sont obtenues.

- **Des approches symboliques et numériques permettent de déterminer des instances similaires au sein d'un graphe de connaissance.** En particulier, les concepts de graphes permettent de définir une distance conceptuelle, et les plongements de graphes permettent d'apprendre des représentations vectorielles d'instances entre lesquelles il est possible de calculer une distance. Cependant, ces distances seules ne permettent pas de déterminer si deux instances sont suffisamment similaires pour un problème de découverte causale.

8.3 Appariemment tronqué pour la fouille de règles différentielles causales

Le chapitre 4 présente des contributions publiées dans des conférences nationales et internationales [81, 78]. Ces travaux ont porté sur la définition de règles différentielles causales. Ces règles expriment l'effet d'une cause nommée traitement sur une variable nommée résultat, et ce pour un ensemble d'instances défini par un motif de graphe nommé strate. Ces règles permettent ainsi de fournir des explications à pourquoi deux instances ont un résultat différent. Ce chapitre montre que de telles règles exprimant une relation causale peuvent être obtenues en adaptant le cadre d'étude des résultats potentiels.

Les approches développées [81, 78] ont été testées et évaluées par des experts sur deux jeux de données, dont un jeu de données nommé *Vitamin*. *Vitamin* décrit des personnes, leurs habitudes alimentaires et leur avis sur des questions d'alimentation et de changement climatique. Les experts ont jugé les règles découvertes sur *Vitamin* comme étant globalement interprétables et ayant du sens. Cependant, ces règles présentent certaines limites, dont (i) leur ordre qui n'est pas clairement défini, (ii) certaines règles sont difficiles à interpréter, un traitement pouvant avoir des effets opposés, et (iii) les règles expliquent une faible proportion des paires ayant un résultat différent.

8.4 Fouille de règles à effets opposés

Dans le chapitre 5, nous proposons une contribution publiée dans une conférence internationale [77] définissant des règles à effets opposés. La définition de ces règles fait suite aux limites exprimées par les règles différentielles causales du chapitre 4, et permet de faciliter l'interprétation de ces règles. Plus particulièrement, les règles

à effets opposés montrent qu'un même traitement peut avoir des effets opposés en fonction des ensembles d'instances auxquels il est appliqué, et explicitent les différences entre ces ensembles d'instances.

La fouille des règles à effets opposés permet de (i) clarifier l'interprétation des règles différentielles causales et de (ii) hiérarchiser les traitements en regardant le sens de leurs effets dans les règles où ils apparaissent et leur tendance à changer le sens d'autres traitements.

8.5 Plongements de graphes pour la fouille de règles différentielles causales

Dans le chapitre 6, nous proposons des contributions publiées dans des conférences nationales et internationales [79, 80] définissant des règles causales à effets moyens. À la différence des règles différentielles causales qui expriment l'effet d'un traitement sur un sous-ensemble d'instances, les règles causales à effets moyens expriment l'effet moyen d'un traitement sur l'ensemble des instances d'un graphe de connaissance. Cette approche permet d'obtenir des règles (i) plus simples à interpréter et (ii) qui expliquent significativement davantage de différences sur le résultat par rapport aux règles différentielles causales.

Cette approche est fondée sur le cadre d'étude des résultats potentiels et exploite les plongements de graphes pour la comparaison des instances. L'apprentissage des plongements de graphes représentant les instances d'un graphe de connaissances permet de déterminer des instances similares à travers l'ensemble du graphe, et ce même si les descriptions des instances sont incomplètes. En conséquence, cette approche est également davantage résiliante aux données manquantes que les précédentes approches.

8.6 Conclusion

Nous avons montré dans cette thèse que les relations causales dans les graphes de connaissances peuvent être représentées par des règles que nous avons nommées règles causales différentielles. Ces règles fournissent des explications à pourquoi des instances d'une classe cible d'un graphe de connaissances ont des valeurs différentes sur un chemin de propriétés, et sont obtenues en adaptant le cadre des résultats potentiels. Le cadre des résultats potentiels est un cadre standard pour découvrir des effets causaux. La méthode de ce cadre qui a été adaptée est l'appariement, qui consiste à comparer des instances similaires qui diffèrent par le chemin de traitement à étudier. Nous avons montré que la découverte de règles à l'aide de l'appariement pouvait se faire par plusieurs méthodes, *i.e.*, symboliques ou

hybrides, qui sont plus ou moins résistantes à l'incomplétude de l'information. Les effets causaux exprimés dans les règles sont expliqués par des traitements. Ils peuvent exprimer un effet local, *i.e.*, sur un sous-ensemble d'instances de la classe cible décrit par un motif de graphe, ou un effet moyen, *i.e.*, sur toutes les instances de la classe cible. Il existe un compromis entre l'utilisation des règles exprimant un effet local ou un effet moyen. Les règles exprimant un effet local sont plus expressives mais peuvent être plus difficiles à interpréter, et fournissent des explications à un petit nombre de paires ayant des valeurs de résultat différentes. D'autre part, les règles exprimant des effets moyens sont plus faciles à interpréter et expliquent un plus grand nombre de paires, mais sont moins expressives.

Bibliography

[1] Robert P. Althauser and Donald Rubin. The computerized construction of a matched sample. *American Journal of Sociology*, 76(2):325–346, 1970.

[2] José J. Ramasco Andrea Lancichinetti, Filippo Radicchi and Santo Fortunato. Finding statistically significant communities in networks. *PLoS One*, 2011.

[3] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web*, pages 722–735, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[4] Wouter Beek, Stefan Schlobach, and Frank van Harmelen. A contextualised semantics for owl:sameas. In Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Simone Paolo Ponzetto, and Christoph Lange, editors, *The Semantic Web. Latest Advances and New Domains*, pages 405–419, Cham, 2016. Springer International Publishing.

[5] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[6] Tim Berners-Lee. *Linked Data - Design Issues*. 2006.

[7] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, Oct 2008.

[8] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 2787–2795, Red Hook, NY, USA, 2013. Curran Associates Inc.

[9] Luca Costabello, Sumit Pai, Nicholas McCarthy, and Adrianna Janik. Knowledge graph embeddings tutorial: From theory to practice, September 2020. https://kge-tutorial-ecai2020.github.io/.

[10] Luca Costabello, Sumit Pai, Chan Le Van, Rory McGrath, Nicholas McCarthy, and Pedro Tabacof. AmpliGraph: a Library for Representation Learning on Knowledge Graphs, March 2019.

[11] Agnieszka Dardzinska. *Action rules mining*, volume 468. Springer, 2012.

[12] Rosaline de Haan, Ilaria Tiddi, and Wouter Beek. Discovering Research Hypotheses in Social Science Using Knowledge Graph Embeddings. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12731 LNCS:477–494, 2021.

[13] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings, 2017.

[14] Xin Luna Dong. Challenges and innovations in building a product knowledge graph. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining*, KDD '18, page 2869, New York, NY, USA, 2018. Association for Computing Machinery.

[15] S. Ferilli, T. M. A. Basile, M. Biba, N. Di Mauro, and F. Esposito. A general similarity framework for horn clause logic. *Fundam. Inf.*, 90(1–2):43–66, 2009.

[16] Sébastien Ferré. Concepts de plus proches voisins dans des graphes de connaissances. In Catherine Roussey, editor, *IC 2017 : 28es Journées francophones d'Ingénierie des Connaissances (Proceedings of the 28th French Knowledge Engineering Conference), Caen, France, July 3-7, 2017*, pages 163–174, 2017.

[17] Sébastien Ferré. Answers Partitioning and Lazy Joins for Efficient Query Relaxation and Application to Similarity Search. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10843 LNCS:209–224, 2018.

[18] Sébastien Ferré. Link prediction in knowledge graphs with concepts of nearest neighbours. In Pascal Hitzler, Miriam Fernández, Krzysztof Janowicz, Amrapali Zaveri, Alasdair J. G. Gray, Vanessa López, Armin Haller, and Karl Hammar, editors, *The Semantic Web - 16th International Conference, ESWC 2019, Portorož, Slovenia, June 2-6, 2019, Proceedings*, volume 11503 of *Lecture Notes in Computer Science*, pages 84–100. Springer, 2019.

[19] Mohamed H. Gad-Elrab, Daria Stepanova, Trung Kien Tran, Heike Adel, and Gerhard Weikum. ExCut: Explainable Embedding-Based Clustering over Knowledge Graphs. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12506 LNCS(October):218–237, 2020.

[20] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. Fast rule mining in ontological knowledge bases with AMIE+. *VLDB J.*, 24(6):707–730, 2015.

[21] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. Amie: Association rule mining under incomplete evidence in ontological knowledge bases. pages 413–422, 05 2013.

[22] Steven Glazerman, Dan M. Levy, and David Myers. Nonexperimental versus experimental estimates of earnings impacts. Technical report, 2003.

[23] Aditya Grover and Jure Leskovec. Node2vec: Scalable feature learning for networks. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 13-17-August-2016:855–864, 2016.

[24] Thomas R. Gruber. Toward principles for the design of ontologies used for knowledge sharing? *International Journal of Human-Computer Studies*, 43(5):907–928, 1995.

[25] Xing Sam Gu and Paul R. Rosenbaum. Comparison of multivariate matching methods: Structures, distances, and algorithms. *Journal of Computational and Graphical Statistics*, 2(4):405–420, 1993.

[26] Harry Halpin, Patrick J. Hayes, James P. McCusker, Deborah L. McGuinness, and Henry S. Thompson. When owl:sameAs isn't the same: An analysis of identity in linked data. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6496 LNCS(PART 1):305–320, 2010.

[27] William L. Hamilton, Rex Ying, and Jure Leskovec. Representation Learning on Graphs: Methods and Applications. pages 1–24, 2017.

[28] Daniel E. Ho, Kosuke Imai, Gary King, and Elizabeth A. Stuart. Matching as nonparametric preprocessing for reducing model dependence in parametric causal inference. *Political Analysis*, 15(3):199–236, 2007.

[29] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia D'amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto

Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge graphs. *ACM Computing Surveys*, 54(4):1–37, may 2022.

[30] Aidan Hogan, Jürgen Umbrich, Andreas Harth, Richard Cyganiak, Axel Polleres, and Stefan Decker. An empirical survey of linked data conformance. *J. Web Semant.*, 14:14–44, 2012.

[31] Paul W. Holland. Statistics and causal inference. *Journal of the American Statistical Association*, 81(396):945–960, 1986.

[32] Guanglei Hong and Stephen W Raudenbush. Evaluating kindergarten retention policy. *Journal of the American Statistical Association*, 101(475):901–910, 2006.

[33] Tamás Horváth, Stefan Wrobel, and Uta Bohnebeck. Relational instance-based learning with lists and terms. *Mach. Learn.*, 43(1/2):53–80, 2001.

[34] M. Hulswit. *From Cause to Causation a Peircean Perspective*. Springer Science & Business Media, 2002.

[35] David Hume. *A Treatise of Human Nature*. Clarendon Press, 1739.

[36] Carlos A. Hurtado, Alexandra Poulovassilis, and Peter T. Wood. Query relaxation in rdf. In Stefano Spaccapietra, editor, *Journal on Data Semantics X*, pages 31–61, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[37] Stefano M. Iacus, Gary King, and Giuseppe Porro. Causal inference without balance checking: Coarsened exact matching. *Political Analysis*, 20(1):1–24, 2012.

[38] Liliana Ibanescu, Juliette Dibie, Stéphane Dervaux, Elisabeth Guichard, and Joe Raad. Po^2 - A process and observation ontology in food science. application to dairy gels. In Emmanouel Garoufallou, Imma Subirats Coll, Armando Stellato, and Jane Greenberg, editors, *Metadata and Semantics Research - 10th International Conference, MTSR 2016, Göttingen, Germany, November 22-25, 2016, Proceedings*, volume 672 of *Communications in Computer and Information Science*, pages 155–165, 2016.

[39] Nitisha Jain, Jan Christoph Kalo, Wolf Tilo Balke, and Ralf Krestel. Do Embeddings Actually Capture Knowledge Graph Semantics? *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12731 LNCS:143–159, 2021.

[40] Immanuel Kant. *Critique of Pure Reason*. 1781.

[41] Emre Kiciman and Amit Sharma. Tutorial on causal inference and counterfactual reasoning. In *ACM KDD International Conference on Knowledge Discovery and Data Mining*, August 2018.

[42] Gary King and Richard Nielsen. Why Propensity Scores Should Not Be Used for Matching. *Political Analysis*, 27(4):435–454, 2019.

[43] Arun Krishnan. Making search easier: How amazon's product graph is helping customers and products more easily. 2019.

[44] Jonathan Lajus, Luis Galárraga, and Fabian M. Suchanek. Fast and exact rule mining with amie 3. *The Semantic Web*, 12123:36 – 52, 2020.

[45] Jiuyong Li, Thuc Duy Le, Lin Liu, Jixue Liu, Zhou Jin, and Bingyu Sun. Mining causal association rules. *Proceedings - IEEE 13th International Conference on Data Mining Workshops, ICDMW 2013*, pages 114–123, 2013.

[46] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. *Proceedings of the AAAI Conference on Artificial Intelligence*, 29(1), Feb. 2015.

[47] Félix Martel and Amal Zouaq. Taxonomy extraction using knowledge graph embeddings and hierarchical clustering. *Proceedings of the ACM Symposium on Applied Computing*, pages 836–844, 2021.

[48] John McCarthy and Vladimir Lifschitz. *Formalizing Commonsense: Papers by John McCarthy*. Greenwood Publishing Group Inc., USA, 1990.

[49] Christian Meilicke, Melisachew Wudage Chekol, Daniel Ruffinelli, and Heiner Stuckenschmidt. Anytime bottom-up rule learning for knowledge graph completion. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 3137–3143. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[50] Changsung Moon, Paul Jones, and Nagiza F. Samatova. Learning entity type embeddings for knowledge graph completion. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 2215–2218, New York, NY, USA, 2017. Association for Computing Machinery.

[51] Melanie Munch, Juliette Dibie, Pierre Henri Wuillemin, and Cristina Manfredotti. Interactive causal discovery in knowledge graphs. *CEUR Workshop Proceedings*, 2465:78–93, 2019.

[52] Brady Neal. *Introduction to causal inference from a machine learning perspective*. Course Lecture Notes (draft), 2020.

[53] Richard E. Neapolitan. Learning Bayesian networks witn integration of indirect prior knowledge. *Prentice Hall*, 2003.

[54] Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. A novel embedding model for knowledge base completion based on convolutional neural network. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics, 2018.

[55] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, page 809–816, Madison, WI, USA, 2011. Omnipress.

[56] Natasha Noy, Yuqing Gao, Anshu Jain, Anant Narayanan, Alan Patterson, and Jamie Taylor. Industry-scale knowledge graphs: Lessons and challenges. *Communications of the ACM*, 62 (8):36–43, 2019.

[57] Ekpe Okorafor and Atish Ray. The path from data to knowledge. 2019.

[58] Stefano Ortona, Venkata Vamsikrishna Meduri, and Paolo Papotti. RuDiK: Rule discovery in knowledge bases. *Proceedings of the VLDB Endowment*, 11(12):1946–1949, 2018.

[59] Peter F. Patel-Schneider, Patrick Hayes, and Ian Horrocks. Owl web ontology language semantics and abstract syntax section 5. rdf-compatible model-theoretic semantics. *Technical Report. W3C.*, 2004.

[60] Judea Pearl. Causal inference in statistics: An overview. *Statistics Surveys*, 3(September):96–146, 2009.

[61] Bee-Chung Chen Qi He and Deepak Agarwal. Building the linkedin knowledge graph. 2016.

[62] Joe Raad, Wouter Beek, Frank van Harmelen, Nathalie Pernelle, and Fatiha Saïs. Detecting erroneous identity links on the web using network metrics. In *The Semantic Web – ISWC 2018*, pages 391–407, Cham, 2018. Springer International Publishing.

[63] Joe Raad, Nathalie Pernelle, and Fatiha Saïs. Detection of contextual identity links in a knowledge base. In *Proceedings of the Knowledge Capture Conference, K-CAP 2017*, 2017.

[64] Joe Raad, Nathalie Pernelle, Fatiha Saïs, Wouter Beek, and Frank van Harmelen. The sameas problem: A survey on identity management in the web of data, 2019.

[65] Zbigniew W. Ras and Alicja Wieczorkowska. Action-rules: How to increase profit of a company. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, PKDD '00, page 587–592, 2000.

[66] Peyman Rasouli and Ingrid Chieh Yu. Care: Coherent actionable recourse based on sound counterfactual explanations, 2021.

[67] Youssra Rebboud, Pasquale Lisena, and Raphael Troncy. Beyond Causality: Representing Event Relations in Knowledge Graphs. In *EKAW, 23rd International Conference on Knowledge Engineering and Knowledge Management*, Bolzano, Italy, September 2022.

[68] Paul R. Rosenbaum and Donald B. Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 79(1):41–55, 1983.

[69] Paul R. Rosenbaum and Donald B. Rubin. Reducing bias in observational studies using subclassification on the propensity score. *Journal of the American Statistical Association*, 79(387):516–524, 1984.

[70] Donald B. Rubin. Bias reduction using mahalanobis-metric matching. *Biometrics*, 36(2):293–298, 1980.

[71] Rubin D. B. Estimating causal effects of treatment in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66(5):688–701, 1974.

[72] Marta Sabou, Elodie Thiéblin, Ollivier Haemmerlé, Nathalie Hernandez, and Cassia Trojahn. Survey on complex ontology matching. *Semant. Web*, 11(4):689–727, jan 2020.

[73] Nicolas Salliou and Rallou Thomopoulos. Adopting, adhering to or abandoning veganism: insights on vegan diet transitions from life story accounts. (June), 2020.

[74] Edward W. Schneider. Course modularization applied: The interface system and its implications for sequence control and data analysis. 1973.

[75] Bernhard Scholkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward Causal Representation Learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

[76] Stijn Schouten, Victor de Boer, Lodewijk Petram, and Marieke van Erp. The wind in our sails: Developing a reusable and maintainable dutch maritime history knowledge graph. In *Proceedings of the 11th on Knowledge Capture Conference*, K-CAP '21, page 97–104, New York, NY, USA, 2021. Association for Computing Machinery.

[77] Lucas Simonne, Nathalie Pernelle, and Fatiha Saïs. Counter effect rules mining in knowledge graphs. In *Knowledge Engineering and Knowledge Management*, Lecture Notes in Computer Science, pages 167–173. Springer International Publishing, 2022.

[78] Lucas Simonne, Nathalie Pernelle, Fatiha Saïs, and Rallou Thomopoulos. Differential causal rules mining in knowledge graphs. In *Proceedings of the 11th on Knowledge Capture Conference*, K-CAP '21, page 105–112, New York, NY, USA, 2021. Association for Computing Machinery.

[79] Lucas Simonne, Nathalie Pernelle, Fatiha Saïs, and Rallou Thomopoulos. Discovering causal rules in knowledge graphs using graph embeddings. In *WI-IAT '22: IEEE/WIC/ACM International Conference on Web Intelligence, Niagara Falls Canada, November 17-20, 2022*. ACM, 2022.

[80] Lucas Simonne, Nathalie Pernelle, Fatiha Saïs, and Rallou Thomopoulos. Découverte de règles causales dans les graphes de connaissances à l'aide de plongements dans les graphes. In *IC 2022 : 33es Journées francophones d'Ingénierie des Connaissances (Proceedings of the 33nd French Knowledge Engineering Conference), Saint-Etienne, France, June 29 - July 1, 2022*, 2022.

[81] Lucas Simonne, Nathalie Pernelle, and Fatiha Saïs. Fouille de règles différentielles causales dans les graphes de connaissances. *Revue des Nouvelles Technologies de l'Information*, Extraction et Gestion des Connaissances, RNTI-E-37:293–300, 2021.

[82] Amit Singhal. Introducing the knowledge graph: things, not strings. google. 2012.

[83] Elizabeth A. Stuart. Matching methods for causal inference: A review and a look forward. *Statistical science : a review journal of the Institute of Mathematical Statistics*, 25(1):1–21, 2010.

[84] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, page 697–706, New York, NY, USA, 2007. Association for Computing Machinery.

[85] Lukáš Sýkora and Tomáš Kliegr. Action rules: Counterfactual explanations in python. *CEUR Workshop Proceedings*, 2644:28–41, 2020.

[86] Magdalena Szumilas. Explaining odds ratios. *Journal of the Canadian Academy of Child and Adolescent Psychiatry l'adolescent*, 19:227, 2010.

[87] Thomas Pellissier Tanon, Gerhard Weikum, and Fabian M. Suchanek. Yago 4: A reason-able knowledge base. *The Semantic Web*, 12123:583 – 596, 2020.

[88] Rallou Thomopoulos, Nicolas Salliou, Patrick Taillandier, and Alberto Tonda. Consumers' Motivations towards Environment-Friendly Dietary Changes: An Assessment of Trends Related to the Consumption of Animal Products. In *Handbook of Climate Change Across the Food Supply Chain*. 2020.

[89] Petar Velicković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *arXiv*, pages 1–12, 2017.

[90] Denny Vrandečić and Markus Krötzsch. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85, sep 2014.

[91] Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. Attributed graph clustering: A deep attentional embedding approach. *IJCAI International Joint Conference on Artificial Intelligence*, 2019-Augus:3670–3676, 2019.

[92] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, 2017.

[93] Yanjie Wang, Daniel Ruffinelli, Samuel Broscheit, and Rainer Gemulla. On evaluating embedding models for knowledge base completion. *CoRR*, abs/1810.07180, 2018.

[94] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by transvlating on hyperplanes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 28(1), Jun. 2014.

[95] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases, 2014.