Explainable Methods for Knowledge Graph Refinement and Exploration via Symbolic Reasoning

Mohamed H. Gad-Elrab

Dissertation zur Erlangung des Grades
DOKTOR DER INGENIEURWISSENSCHAFTEN (DR.-ING.)
Fakultät fur Mathematik und Informatikder
Universität des Saarlandes

Saarbrücken, 2021

Day of Colloquium	June 07, 2021
Dean of the Faculty	Univ.-Prof. Dr. Thomas Schuster
Chair of the Committee	Prof. Dr. Kurt Mehlhorn
Reporters	
First Reviewer	Prof. Dr. Gerhard Weikum
Second Reviewer	Prof. Dr. Martin Theobald
Third Reviewer	Dr. Daria Stepanova
Fourth Reviewer	Dr. Simon Razniewski
Academic Assistant	Dr. Erisa Terolli

Abstract

Knowledge Graphs (KGs) have applications in many domains such as Finance, Manufacturing, and Healthcare. While recent efforts have created large KGs, their content is far from complete and sometimes includes invalid statements. Therefore, it is crucial to enhance both the coverage and accuracy of KGs through *KG completion* and *KG validation*, together referred to as *KG refinement*. In this context, it is also vital to provide human-comprehensible explanations for the KG refinement output so that humans have trust in the refined KG quality.

KG exploration, by search and browsing, is essential for users to understand the KG value and limitations towards down-stream applications. However, the large size of KGs makes KG exploration challenging. While the type taxonomy of KGs is a useful asset along these lines, it remains insufficient for deep exploration.

This dissertation tackles the challenges of KG refinement and KG exploration by logical reasoning over the KG in combination with other techniques such as KG embedding models and text mining. We introduce methods for these goals which provide human-understandable output.

Concretely, the dissertation consists of the following contributions:

- To tackle KG incompleteness, we present *ExRuL*, a method for revising Horn rules by adding exceptions (*i.e.*, negated atoms) to their bodies. Learned rules can be used to predict new facts to fill gaps in the KG. Experiments on real-world KGs show that *exception-aware* rules vastly reduce the error rate in fact prediction. Besides, rules provide user-comprehensible explanations for these predictions.

- We also present *RuLES*, a rule learning method that utilizes probabilistic representations of missing facts. The method iteratively extends the rules induced from a KG by incorporating feedback from a precomputed KG embedding combined with text corpora. The method harnesses newly devised measures for rule quality. RuLES improves the quality of the learned rules and their predictions.

- To support KG validation, we propose *ExFaKT*, a framework for constructing human-comprehensible explanations for candidate facts. The method uses rules to rewrite a

candidate fact into a set of related facts that are easier to spot and confirm (or refute). The output of ExFaKT is a set of semantic traces for the candidate facts from both text and the KG. Experiments show that rule-based rewriting significantly improves the recall of the discovered traces while preserving a high precision. Furthermore, the explanations support both manual and automatic KG validation.

- To facilitate KG exploration, we introduce *ExCut*, a method that combines KG embeddings with rule mining to compute informative entity clusters with explanations. Cluster explanation consists of a concise combination of entity relations that distinguish this cluster. ExCut jointly enhances the quality of entity clusters and their explanations by iteratively interleaving the learning of embeddings and rules. Experiments show that ExCut produces high-quality clusters, and the explanations computed for them help humans understand the commonalities among entities within these clusters.

Kurzfassung

Wissensgraphen haben viele Anwendungen in verschiedenen Bereichen, beispielsweise im Finanz- und Gesundheitswesen. Wissensgraphen sind jedoch unvollständig und enthalten auch ungültige Daten. Hohe Abdeckung und Korrektheit erfordern neue Methoden zur *Wissensgraph-Erweiterung* und *Wissensgraph-Validierung*. Beide Aufgaben zusammen werden als *Wissensgraph-Verfeinerung* bezeichnet. Ein wichtiger Aspekt dabei ist die Erklärbarkeit und Verständlichkeit von Wissensgraphinhalten für Nutzer.

In Anwendungen ist darüber hinaus die nutzerseitige *Exploration von Wissensgraphen* von besonderer Bedeutung. Suchen und Navigieren im Graph hilft dem Anwender, die Wissensinhalte und ihre Limitationen besser zu verstehen. Aufgrund der riesigen Menge an vorhandenen Entitäten und Fakten ist die *Wissensgraphen-Exploration* eine Herausforderung. Taxonomische Typsystem helfen dabei, sind jedoch für tiefergehende Exploration nicht ausreichend.

Diese Dissertation adressiert die Herausforderungen der *Wissensgraph-Verfeinerung* und der *Wissensgraph-Exploration* durch algorithmische Inferenz über dem Wissensgraph. Sie erweitert logisches Schlussfolgern und kombiniert es mit anderen Methoden, insbesondere mit neuronalen Wissensgraph-Einbettungen und mit Text-Mining. Diese neuen Methoden liefern Ausgaben mit Erklärungen für Nutzer.

Die Dissertation umfasst folgende Beiträge:
Insbesondere leistet die Dissertation folgende Beiträge:

- Zur Wissensgraph-Erweiterung präsentieren wir *ExRuL*, eine Methode zur Revision von Horn-Regeln durch Hinzufügen von Ausnahmebedingungen zum Rumpf der Regeln. Die erweiterten Regeln können neue Fakten inferieren und somit Lücken im Wissensgraphen schließen. Experimente mit großen Wissensgraphen zeigen, dass diese Methode Fehler in abgeleiteten Fakten erheblich reduziert und nutzerfreundliche Erklärungen liefert.

- Mit *RuLES* stellen wir eine Methode zum Lernen von Regeln vor, die auf probabilistischen Repräsentationen für fehlende Fakten basiert. Das Verfahren erweitert iterativ die aus einem Wissensgraphen induzierten Regeln, indem es neuronale

Wissensgraph-Einbettungen mit Informationen aus Textkorpora kombiniert. Bei der Regelgenerierung werden neue Metriken für die Regelqualität verwendet. Experimente zeigen, dass RuLES die Qualität der gelernten Regeln und ihrer Vorhersagen erheblich verbessert.

- Zur Unterstützung der Wissensgraph-Validierung wird *ExFaKT* vorgestellt, ein Framework zur Konstruktion von Erklärungen für Faktkandidaten. Die Methode transformiert Kandidaten mit Hilfe von Regeln in eine Menge von Aussagen, die leichter zu finden und zu validieren oder widerlegen sind. Die Ausgabe von ExFaKT ist eine Menge semantischer Evidenzen für Faktkandidaten, die aus Textkorpora und dem Wissensgraph extrahiert werden. Experimente zeigen, dass die Transformationen die Ausbeute und Qualität der entdeckten Erklärungen deutlich verbessert. Die generierten unterstützen Erklärungen unterstütze sowohl die manuelle Wissensgraph-Validierung durch Kuratoren als auch die automatische Validierung.

- Zur Unterstützung der Wissensgraph-Exploration wird *ExCut* vorgestellt, eine Methode zur Erzeugung von informativen Entitäts-Clustern mit Erklärungen unter Verwendung von Wissensgraph-Einbettungen und automatisch induzierten Regeln. Eine Cluster-Erklärung besteht aus einer Kombination von Relationen zwischen den Entitäten, die den Cluster identifizieren. ExCut verbessert gleichzeitig die Cluster-Qualität und die Cluster-Erklärbarkeit durch iteratives Verschränken des Lernens von Einbettungen und Regeln. Experimente zeigen, dass ExCut Cluster von hoher Qualität berechnet und dass die Cluster-Erklärungen für Nutzer informativ sind.

Acknowledgments

الحمد لله حمداً كثيراً طيباً..

I would like to express my gratitude to Gerhard Weikum for his continuous support and guidance. Gerhard's guidance helped to sharpen my ideas and elevated the contributions of this thesis. I would also like to gratefully thank Daria Supernova for her professional support throughout the thesis. Working with Daria has been, and continues to be, an enjoyable learning experience. Daria's dedication, determinism, and attention to details taught me a lot. I appreciate being supervised by such elite supervisors and being a member of *MPII* and *IMPRS-CS*.

I would like to thank Martin Theobald ad Simon Razniewsk for accepting to be part of the thesis committee. Also, for the earlier fruitful discussions with them.

I would like to acknowledge the NLP and KRR group led by Jannik Strötgen at *Bosch Center for Artificial Intelligence (BCAI)* for offering me the opportunity to have an exciting PhD sabbatical in their competent group. During my stay, I got the chance to work on challenging problems, which lead to the contributions in Chapter 6.

During this dissertation, I had the opportunity to collaborate with excellent people whom I would like to thank for the great experience. Special thanks to Trung-Kien Tran, Jacopo Urbani, and Vinh Thinh Ho.

Thanks to my friends Akram El-Korashy and Mohamed Alzayat for helping in reviewing this dissertation. Also, I would like to thank Abdalghani Abujabal for the lengthy and fruitful discussions we had.

On the personal side, I am sincerely grateful to my beloved wife Fatma who accepted to join me through the hard Ph.D. journey. Without her support, love, and patience, I cannot imagine being able to continue and overcome the stressful times. I am grateful for having Fatma and our daughter Tuka, who always give my life a greater meaning.

Last but not least, I would like to thank my family: my mother Ebtihal, who believes in me, prays for me and unconditionally supports me in all my choices; my Father Hassan who always encourages me to follow the scientific journey; and my dear siblings Mustafa and Mai for their endless support, help, and encouragement.

Contents

Chapter 1

Introduction

1.1 Motivation

Artificial Intelligence pursues the quest to develop automated agents that are capable of reasoning towards rational decisions. Without sufficient knowledge about the world, automated reasoning is infeasible [Lenat and Feigenbaum, 1991]. The need for such knowledge drives various efforts to represent human knowledge into structured models that are readable by machines. Ideally, the created resources should maintain both high coverage and accuracy.

The need for machine consumable knowledge is manifested in creating several large and structured *knowledge bases* suitable for automated reasoning tasks, which later became known as *knowledge graphs*.

1.1.1 Knowledge Graphs

Knowledge graphs (KGs) are large collections of factual triples $\langle subject\ predicate\ object \rangle$. The *subject* and *object* are entities representing people, places, organizations, etc, and the *predicate* is the relation between them. In addition, some KGs include a *typing system*, indicating the types of KG entities and the relations between these types *taxonomy*. Typing systems are valuable for reasoning and exploration tasks.

Example 1.1. Knowledge about *Albert Einstein* such as being a scientist born in Ulm, Germany who lived in the USA is represented in the triples format as:

$$\langle albert\_einstein\ \textbf{\textit{bornIn}}\ ulm \rangle$$
$$\langle ulm\ \textbf{\textit{isLocatedIn}}\ germany \rangle$$
$$\langle albert\_einstein\ \textbf{\textit{immigratedTo}}\ usa \rangle$$

and the typing system behind the aforementioned entities includes:

$$\langle albert\_einstein \ \textbf{\textit{type}} \ scientits \rangle$$
$$\langle ulm \ \textbf{\textit{type}} \ city \rangle$$
$$\langle germany \ \textbf{\textit{type}} \ country \rangle$$
$$\langle scientits \ \textbf{\textit{subClassOf}} \ person \rangle$$

History. Early KGs were manually crafted; resulting in KGs of high quality but limited scope and scale, for example, Cyc [Lenat, 1995] and WordNet [Miller, 1995]. With the start of the new millennium, the advances in automatic knowledge harvesting led to a leap in the constructed KGs with respect to their size, quality, and coverage. Several large-scale KGs came into existence, including Freebase [Bollacker et al., 2008], and Wikidata [Vrandecic and Krötzsch, 2014], which are manually constructed via online communities. Other KG projects such as DBpedia [Auer et al., 2007], YAGO [Suchanek et al., 2007], BabelNet [Navigli and Ponzetto, 2012], NELL [Mitchell et al., 2015], and KnowledgeVault [Dong et al., 2014] utilize (semi-)automated curation techniques from heterogeneous sources. Simultaneously, efforts to align and interlink existing KGs, *e.g.*, [Saeedi et al., 2018, Raad et al., 2020] came to existence, aiming at creating a unified view, which is referred to as Linked Open Data[1] [Bizer et al., 2011].

Applications. KGs are used in *information retrieval* tasks like *semantic search* [Haussmann, 2017, Dietz et al., 2018] and *question answering* [Diefenbach et al., 2018, Abujabal, 2019]. They are also crucial for *text analysis* and *language understanding* tasks. In such tasks, KGs provide the required semantics and context to analyze and extract information from noisy sources. Besides, KGs hold implicit patterns describing the dynamics of the real world and human commonsense, which are useful in reasoning [Liu et al., 2020] and data cleaning [Chu et al., 2015] tasks.

The usage of KGs has expanded beyond general-purpose search and text analysis, *e.g.*, [Singhal, 2012], towards more domain-specific applications. KGs are widely used for commercial purposes, *e.g.*, in recommendation systems [Wang et al., 2019b, Guo et al., 2020]. KGs are also utilized in more complex domains such as Academic Literature [Wan et al., 2019], Law [González-Conejero et al., 2018, Junior et al., 2020], Finance [Reuters, 2017, Meij, 2019, Albrecht et al., 2019], and Manufacturing [Bader et al., 2020, Mehdi et al., 2019, Kalayci et al., 2020]. Most notably, KGs are gradually adopted in sensitive domains such as Healthcare [Ernst et al., 2015, Noy et al., 2019, Terolli et al., 2020, Li et al., 2020].

[1] https://lod-cloud.net/

1.1.2 Knowledge Graph Refinement

Coverage and quality. The quality of knowledge graphs is determined by their construction process. For instance, the contributors' expertise and bias determine the quality of manually constructed KGs [Demartini, 2019]. Similarly, automatically curated KGs are affected by the quality of the sources and the effectiveness of the extraction methods [Weikum et al., 2020, Hogan et al., 2020]. Despite the substantial size of existing knowledge graphs, they still have several shortcomings [Darari et al., 2013, Hogan et al., 2020], including (i) the *incompleteness* of both the entities and the facts about them; and sometimes (ii) the *inaccuracy* that appears as either incorrectly curated or outdated statements (*e.g.*, change of marital status).

KG shortcomings directly affect the effectiveness of downstream applications. For instance, in reasoning tasks, both KG coverage and accuracy are essential for deriving valid conclusions. Similarly, in semantic text analysis, missing entities and relations harm the output quality of entity-linking algorithms.

Refinement approaches. Given the above KG limitations, it is crucial to improve the quality of existing KGs, which is the purpose of *KG refinement* [Paulheim, 2017]. Unlike KG construction, *KG refinement* utilizes the existing KG facts, possibly combined with other resources to perform two main tasks: (i) ***KG completion***, which concerns with the prediction of missing relations between KG entities, and (ii) ***KG validation***, which aims at ensuring the correctness and the consistency of the facts in the KG.

Approaches for both KG *completion* and *validation* are divided into two categories: *symbolic (logic-based)* and *sub-symbolic (i.e., statistics-based)* approaches. Symbolic approaches, *e.g.*, [Drabent et al., 2009, Nakashole et al., 2012a, Fierens et al., 2015, Bienvenu et al., 2016], learn inference rules over the KG and utilize these rules to infer new facts or to invalidate existing ones. On the other hand, sub-symbolic approaches learn models of statistical correlations from the KG triples, and, utilize these models to estimate the plausibility of new candidate facts [Paulheim and Bizer, 2014, Nakashole and Mitchell, 2014, Nickel et al., 2016a]. The most prominent statistical approaches are KG embedding models, *e.g.*, [Bordes et al., 2013, Trouillon et al., 2016], where KGs are embedded into a multidimensional continuous space [Nickel et al., 2016a], reflecting the semantic relatedness among KG entities.

Explainability and scalability. KGs have applications in sensitive domains that require human involvement for quality assurance and safety reasons, *e.g.*, Healthcare. Therefore, it is vital to develop refinement methods with human-comprehensible output. Understanding the results helps humans to build trust in the quality of KGs. Explainability also allows overcoming the limitations of KG construction and refinement methods.

For example, explainable results can help in detecting and resolving social bias in KG embedding models [Fisher et al., 2020].

Statistical approaches have the ability to handle noisy and large-scale data [Nickel et al., 2016a] and support the fusion of heterogeneous data modalities, *e.g.*, text with relational tuples [Wang et al., 2014a]. Nevertheless, they suffer from several limitations; in particular, their results are not easily interpretable [Bianchi et al., 2020]. On the other hand, producing explainable results is a core advantage for symbolic approaches [Eiter et al., 2016, Martires et al., 2020]. Yet, logical reasoning alone cannot handle noisy sources [Ji et al., 2011].

Bridging logic-based and statistical methods has the potential for developing large-scale KG refinement approaches that produce human-understandable results. More specifically, rule learning over KGs can offer a good proxy for interpreting obtained results, while statistical techniques can provide logical reasoning with interfaces to collect signals from noisy sources.

1.1.3 Knowledge Graph Exploration

KG exploration is essential for knowledge engineers to understand the KG value and limitations towards downstream applications. However, given the size of existing KGs, exploring the KG is challenging. Moreover, KGs contain heterogeneous data and lack a predefined schema [Mohanty and Ramanath, 2019]; hence, composing explorative queries over the KG is tedious and time-consuming.

Several KG visualization and navigation tools have been developed to facilitate KG exploration [Gómez-Romero et al., 2018]. Other methods support formulating queries via KG-based auto-completion [Mohanty and Ramanath, 2019], query expansion [Lissandrini et al., 2020], or querying by example [Mottin et al., 2016]. Nevertheless, these methods do not provide a holistic view of the KG and still require significant manual effort to grasp the main content of the KG.

An alternative is to summarize the KG to produce human-readable and comprehensive views [Cebiric et al., 2019, Liu et al., 2018]. In some KGs, *e.g.*, YAGO, the type system is a useful asset in that context. However, types in such KGs are still coarse-grained and cannot support deep exploration. Therefore, additional support is needed, which can be achieved by discovering entity clusters of semantically related entities.

1.2 Contributions

The dissertation studies the integration of symbolic reasoning with statistical techniques, like KG embedding and text mining, to address the following challenges:

- ***KG completion***: We aim at improving the precision of rule-based completion by enhancing the quality of the rules learned from the KG.
- ***KG validation***: We investigate the challenge of collecting sufficient evidence supporting (or refuting) candidate facts.
- ***KG exploration***: We study the problem of grouping semantically related KG entities into explainable clusters based on the structure of the KG.

Most importantly, we focus on developing methods that provide human-understandable output. Concretely, we present the following contributions:

ExRuL: Exception-aware rule learning. While KGs are inevitably bound to be incomplete, correlations in the KG can be analyzed to mine inference rules to predict potentially missing facts. Earlier methods, *e.g.*, [Galárraga et al., 2015], focused on learning Horn rules, which do not consider possible exceptions. Therefore, using Horn rules to infer new facts often results in many errors.

To enhance the precision of the rule-based predictions, we present *ExRuL*, a method for effectively revising Horn rules into exception-aware rules (*i.e.*, nonmonotonic rules). We achieve that by adding exceptions (*i.e.*, negated atoms) mined from the KG to the bodies of these rules. Experiments on real-world KGs show that the errors in the revised rules' predictions are vastly reduced compared to Horn rules. Moreover, the revised rules do not just explain the inferred facts, but also indicate when the rules should not infer a triple.

RuLES: Rule learning over Knowledge Graph embedding. Standard rule quality measures, such as confidence, are computed based on the KG alone. Therefore, these measures might be misleading when computed over sparse KGs, preventing rule learning methods from discovering high-quality rules.

We introduce *RuLES*, a rule learning method that utilizes a probabilistic representation of missing facts to address this issue. The method iteratively extends the candidate rules induced from the KG by incorporating feedback from a precomputed KG embedding model combined with text corpora. The method harnesses newly devised measures for rule quality beyond the KG itself, improving the ranking of rules. Experiments demonstrate the effectiveness of our approach, enhancing the quality of the learned rules and their predictions.

ExFaKT: Explainable fact checking. Automatic fact-checking usually starts with collecting evidence for a candidate fact in web sources. This evidence is typically a direct mention of the fact in a supporting or refuting context. Then, the extracted evidence is used to compute a truth score for this fact. This process has two limitations: First, direct mentions are hard to spot and are often not sufficient due to the natural *reporting bias* of the web sources. Second, the computed scores are not sufficient without explanation whenever humans make the final decision.

To better support KG curators in deciding the validity of the candidate facts, we propose *ExFaKT*, a framework for constructing human-comprehensible explanations for candidate facts. *ExFaKT* uses Horn rules to rewrite a candidate fact into a set of other facts that are easier to spot and confirm (or refute). The output is a set of semantic traces (*i.e.*, evidence) for the candidate facts from both web sources and the KG. Experiments show that the rule-based rewriting significantly enhances the recall of the discovered relevant clues while preserving a high precision. Moreover, the experiments show the benefits of the discovered explanations for both manual and automatic fact-checking. Finally, we introduce *Tracy*, a web interface to demonstrate our framework to the end-user.

ExCut: Explainable clustering. KG exploration can be facilitated by *entity clustering*, using unsupervised methods for grouping entities into informative subsets. However, merely clustering the entity set is insufficient. The user also needs to understand the nature of each cluster. Thus, clusters must be explainable in the form of *user-comprehensible labels*. Coarse-grained types available in KGs may not be sufficient to distinguish groups of entities inside individual domains.

To facilitate KG exploration, we introduce *ExCut*, a method that combines KG embedding with rule mining to compute informative clusters with comprehensible explanations. Each explanation consists of a concise combination of entity relations that distinguish the corresponding cluster. Such explainable clusters can help analysts in exploring sets of entities and discovering underlying structures. Furthermore, *ExCut* jointly enhances the quality of entity clusters and their explanations iteratively by interleaving the learning of embeddings and rules. Experiments demonstrate that the iterative process improves the quality of both the clusters and their explanations. Moreover, the user study shows that the produced explanations can help humans in understanding the identified clusters.

1.3 Publications

This section lists the research papers published towards constructing this dissertation. It also indicates the role of *the author* of this dissertation in each publication.

Chapter 3 (Exception-aware rule learning) is based on:

[Gad-Elrab et al., 2016] **Gad-Elrab, M. H.**, Stepanova, D., Urbani, J., and Weikum, G. (2016). Exception-Enriched Rule Learning from Knowledge Graphs. In: *International Semantic Web Conference (ISWC '16)*.

The author has the leading role in the formalization and execution of this research.

[Tran et al., 2016] Tran, D., Stepanova, D., **Gad-Elrab, M. H.**, Lisi, F. A., and Weikum, G. (2016). Towards Nonmonotonic Relational Learning from Knowledge Graphs. In: *International Conference on Inductive Logic Programming (ILP '16)*.

The author made major contributions in formalizing the problem, developing the approach, designing the experiments, and analyzing the results.

Chapter 4 (Rule learning over knowledge graph embedding) is based on:

[Ho et al., 2018] Ho, V. T., Stepanova, D., **Gad-Elrab, M. H.**, Kharlamov, E., and Weikum, G. (2018). Rule Learning from Knowledge Graphs Guided by Embedding Models. In: *International Semantic Web Conference (ISWC '18)*.

The author made major contributions in formalizing the problem, developing the approach, designing the experiments, and analyzing the results.

Chapter 5 (Explainable fact checking) combines the output of both:

[Gad-Elrab et al., 2019] **Gad-Elrab, M. H.**, Stepanova, D., Urbani, J., and Weikum, G. (2019). ExFaKT: A Framework for Explaining Facts over Knowledge Graphs and Text. In: *International Conference on Web Search and Data Mining (WSDM '19)*.

The author has the leading role in the formalization and execution of this research.

[Gad-Elrab et al., 2019] **Gad-Elrab, M. H.**, Stepanova, D., Urbani, J., and Weikum, G. (2019). Tracing Facts over Knowledge Graphs and Text. In: *The Web Conference (WWW '19)*.

The author has the leading role in the formalization and execution of this research.

Chapter 6 (Explainable clustering) is based on:

[Gad-Elrab et al., 2020b] **Gad-Elrab, M. H.**, Stepanova, D., Tran, T., Adel, H., and Weikum, G. (2020). ExCut: Explainable Embedding-based Clustering over Knowledge Graph. In: *International Semantic Web Conference (ISWC '20)*.

The author has the leading role in the formalization and execution of this research.

[Gad-Elrab et al., 2020a] **Gad-Elrab, M. H.**, Ho, V. T., Levinkov, E., Tran, T., and Stepanova, D. (2020). Towards Utilizing Knowledge Graph Embedding Models for Conceptual Clustering. In: *International Semantic Web Conference (ISWC '20)*.

The author has the leading role in the formalization and execution of this research.

Chapter 2 (Background) is partially based on:

[Stepanova et al., 2018] Stepanova, D., **Gad-Elrab, M. H.**, and Ho, V. T. (2018). Rule Induction and Reasoning over Knowledge Graphs. In: *Reasoning Web International Summer School (RW '18)*.

The author has a substantial role in preparing the manuscript of this tutorial.

Further publications. The author has also contributed to the following related research, which is not included in the contributions of this dissertation. This work tackles the challenge of *KG validation* by introducing an efficient method for generating explanations for KGs inconsistencies.

[Tran et al., 2020] Tran, T., **Gad-Elrab, M. H.**, Stepanova, D., Kharlamov, E., and Strötgen, J. (2020). Fast Computation of Explanations for Inconsistency in Large-Scale Knowledge Graphs. In: *The Web Conference (WWW '20)*.

1.4 Organization

The remainder of the thesis is organized as follows: Chapter 2 provides the necessary background related to knowledge graphs and symbolic reasoning. Chapters 3 and 4 describe our contributions towards improving rule-based completion by mining exception-aware rules. Chapter 5 describes our contributions in the domain of validating KG facts via collecting complex evidence. Chapter 6 describes our approach for facilitating KG exploration by discovering explainable entity clusters. Finally, Chapter 7 summarizes the contributions of this dissertation and presents possible directions for future work.

Chapter 2

Background

This chapter provides the essential background about knowledge graphs (KGs), KG embedding models and logic rules. We also discuss symbolic (*i.e.*, rule-based) reasoning tasks; namely, deductive and inductive reasoning.

2.1 Knowledge Graphs

A *knowledge graph* (*KG*), or originally known as knowledge base, is a set of interlinked factual information about some domain of knowledge. KGs have been introduced in the Semantic Web community to create the "Web of data" that is readable by machines. They are usually encoded using the *Resource Description Framework (RDF)* data model described by [Klyne and Carroll, 2004]. In the RDF data model, a KG is encoded as a set of triples of the form $\langle$*subject predicate object*$\rangle$ corresponding to positive binary first-order logic (FOL) facts.

The *subject* in the triple format belongs to the finite set of *constants* in the KG such as *e.g.*, *berlin* or *researcher* in Figure 2.1, while the *object* can either be a constant or a literal value in the form of a string, number, or date. Predicates represent a directed relation between the *subject* and the *object*, for example, *livesIn* and *marriedTo*.

Constants in KGs are often classified into two main categories:

- *Entities* representing canonical individuals in the domain of the KG such as persons, location, organizations, or artifacts. For instance, in Figure 2.1, *alice*, *john*, *berlin*, and *beijing* are example entities.

- *Concepts* which are abstractions or entity types in the KG domain. For example, *researcher*, *artist* and *metropolitan* are the concepts in the KG snippet in Figure 2.1. Usually, concepts are interconnected using relations such as *subSetOf*, *disjoint* to form the taxonomy of the KG. An entity can be mapped to a concept

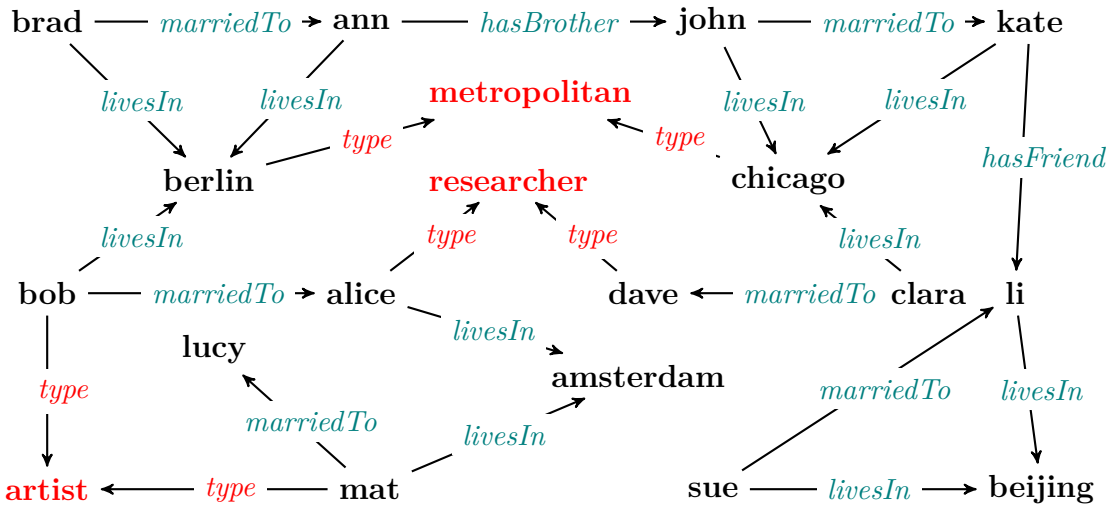


Figure 2.1: Example KG snippet about marriage relations and living places

through the relation *type* (or *isA*); denoting that the entity is an instance of the respective concept.

Knowledge graph representation. For rule learning, we represent knowledge graphs using the semantics of first-order logic (FOL). For that, we define an atom as follows:

Definition 2.1 (Atom)**.** *In first-order logic, an* atomic formula $p(\vec{Y})$ *is a predicate* p *with arguments tuple* $\vec{Y}$*, which is interpreted as "the relation* p *holds between elements in* $\vec{Y}$*". The length of the arguments* $|\vec{Y}|$ *equals to the cardinality of the predicate. In our context, an argument in* $\vec{Y}$ *is either a variable or a constant. An atom with all constant arguments is called a* ground-atom*.*

Based on the definition of an atom, we define the KG under FOL as follows:

Definition 2.2 (Knowledge Graph)**.** *Assuming finite sets of constants* $\mathcal{C}$ *(including entities and concepts) and relations* $\mathcal{R}$ *(i.e., binary predicate names). A knowledge graph* $\mathcal{G}$ *is a finite set of ground-atoms of the form* $p(s, o)$

where $p \in \mathcal{R}$ *and* $s, o \in \mathcal{C}$*.*

The atom signature $p(s, o)$ *represents the existence of the fact* $\langle s\ p\ o \rangle$*.*

As a matter of notation, we donate triples $\langle s\ type\ c \rangle$ or $\langle s\ isA\ c \rangle$ as unary atoms $c(s)$ meaning that s is an instance of the concept/class $c \in \mathcal{C}$.

Example 2.1. Figure 2.1 shows a snippet of a graph about people, family and friendship relations among them as well as their living places and professions. For instance, the upper left part encodes the information that "Ann has a brother John, and lives with

Table 2.1: Examples of real-world KGs and their statistics

Knowledge Graphs	# Entities	# Relations	# Facts
DBpedia [Auer et al., 2007][1]	5M	2.8k	131M
YAGO3 [Rebele et al., 2016]	16M	75	100M
YAGO4-Full [Tanon et al., 2020]	67M	151	343M
Wikidata [Vrandecic and Krötzsch, 2014]	78M	6.7k	974M
Freebase [Bollacker et al., 2008][2]	40M	37k	637M

her husband Brad in Berlin, which is a metropolitan city" represented by the following set of FOL facts:

$$\{\mathit{hasBrother}(\mathit{ann}, \mathit{john}), \mathit{livesIn}(\mathit{ann}, \mathit{berlin}), \mathit{livesIn}(\mathit{brad}, \mathit{berlin}), \\ \mathit{metropolitan}(\mathit{berlin}), \mathit{marriedTo}(\mathit{brad}, \mathit{ann})\}$$

The set $\mathcal{R}$ of relations in the given KG contains the predicates $\mathit{livesIn}, \mathit{marriedTo}$, $\mathit{hasBrother}, \mathit{hasFriend}$, while the set $\mathcal{C}$ of constants comprises the names of people and locations depicted in Figure 2.1. □

2.1.1 Construction

Approaches for constructing knowledge graphs are classified into two main groups: manual and (semi-)automatic. Historically, there were several attempts to construct KGs, yet remarkable attempts started with Cyc Project [Lenat, 1995] and the construction of WordNet [Miller, 1995]. Most of these KGs were limited with respect to their size and domain. Only recently, large-scale, manually constructed KGs started to appear. Prominent examples of manually constructed large-scale KGs include Freebase [Bollacker et al., 2008] and Wikidata [Vrandecic and Krötzsch, 2014], which are constructed collaboratively by volunteers.

Automatic population of KGs from semi-structured resources such as Wikipedia infoboxes using regular expressions and other techniques gave rise to several KG projects, such as YAGO [Suchanek et al., 2007] and DBpedia [Lehmann et al., 2015]. Other projects devoted to the extraction of facts from unstructured resources using information extraction techniques [Niklaus et al., 2018]. For instance, BabelNet [Navigli and Ponzetto, 2012], KnowledgeVault [Dong et al., 2014], and NELL [Mitchell et al., 2015] belong to the latter category. Table 2.1 shows examples of some prominent KGs and their statistics.

[1] As reported by [Tanon et al., 2020] based on DBpedia SPAQL endpoint, March 2020.
[2] Discontinued in 2016. Statistics for Freebase are reported by [Nickel et al., 2016a]

2.1.2 Coverage and Quality

Real-world knowledge graphs contain millions of entities and billions of facts. Consider the example KGs in Table 2.1, Wikidata, for instance, contains around 78M entities and concepts and around 1B statements (*i.e.*, facts). Such a considerable size creates several challenges. First, for humans, exploring large KGs is hard and time-consuming because humans cannot read, understand, or reason about millions or billions of facts. The importance of exploring the KG creates the need for abstracting and summarizing the KG into human-readable and understandable formats [Song et al., 2018, Cebiric et al., 2019, Liu et al., 2018]. Secondly, processing such big graph structures is challenging and requires substantial resources, which motivates creating specific storage and query tools, *e.g.*, [Gonzalez et al., 2014, Urbani et al., 2016, Mami et al., 2019].

Over the years, impressive efforts have been exerted towards creating KGs that comprehensively and accurately reflect the real-world. Nevertheless, existing KGs still face the challenges of (i) incompleteness, (ii) bias; and sometimes (iii) inaccuracy. In the following, we discuss each of these challenges in detail.

Incompleteness. While the existing KGs contain millions of facts, they are still far from being complete. The incompleteness can be observed in the form of missing entities, concepts, or facts. Given the incompleteness of the existing KGs, they cannot be treated under the *Closed World Assumption (CWA)*, which assumes that any fact not known to be true in the KG is indeed false. CWA is commonly adopted while reasoning about probabilistic databases. However, it is not suitable for incomplete real-world KGs.

On the other hand, under the *Open World Assumption (OWA)*, facts that are not present in the KG are assumed to be unknown (*i.e.*, neither true nor false). For example, given only USA as the living place of Albert Einstein ($livedIn(einstein, usa)$), under CWA, we could infer that $\neg livedIn(einstein, germany)$ holds, which is not true. Nevertheless, we cannot infer whether the Claus $livedIn(einstein, germany)$ is true or false under OWA.

An intermediate assumption for KG completeness is the *Local Closed World Assumption (LCWA)*, under which if a fact about some KG entity is present in the KG, all other unknown alternatives for this fact (*i.e.*, facts with the same subject and predicate but different object) are assumed to be false. For instance, given that the fact $wasBornIn(einstein, germany)$ exists in the KG, we can safely infer under *LCWA* that $\neg wasBornIn(einstein, usa)$ holds. On the other hand, a claim such as $played(einstein, piano)$ remains unknown since the KG does not contain any fact about whether Einstein played any musical instrument or not [Galárraga et al., 2015].

Bias. In addition to incompleteness, both manually and semi-automatically curated

KGs suffer from the problem of reporting bias. Missing information is not uniformly distributed over domains or entities [Janowicz et al., 2018]. Prominent entities such as famous actors, movies, and places are usually complete, while less prominent entities are missing many facts. Similarly, popular domains such as sports are more comprehensive than other domains.

In (semi-)automatic KG construction, KG bias results from the *reporting bias* appearing in the background resources such as news and web articles. Alternatively, it can also result from *systematic bias* in the knowledge extractor design or training data (if any was used). Likewise, KGs that are semi-automatically extracted from Wikipedia infoboxes such as DBPedia and YAGO highly depend on the pre-defined properties that the infoboxes contain [Lajus and Suchanek, 2018].

While crowdsourced KGs (*e.g.*, Wikidata) managed to curate facts from all over the world; they suffer from *demographic bias*. Due to the difference in the distribution of contributors, indeed, facts about some countries are covered better than others. For example, KGs typically store more facts about Austrians than Ghanaians even though there are three times more inhabitants in Ghana than Austria. Moreover, contributors interests vary significantly, *e.g.*, Austrians would add detailed information about music composers, while Ghanaians about national athletes. These differences naturally lead to a cultural bias in KGs.

Inaccuracy. The correctness of the facts in the KG is essential for the integrity of the downstream applications. Still, regardless of the KG construction method, the resulting facts are rarely error-free. Indeed, in the case of manually constructed KGs, human contributors might bring their own opinion on top of the added factual statements (*e.g.*, Catalonia being a part of Spain or an independent country). On the other hand, automatically constructed KGs often contain noisy facts, since information extraction methods are imperfect [Nickel et al., 2016a, Paulheim, 2017].

Additionally, KGs are usually not quickly updated; therefore, some KG facts might be outdated. For example, *isMarriedTo*, *spouse* and *livesIn* relations are subject to updates; and hence, can become outdated as people may get divorced or change their living place. As a canonical example, a KG may still contain $\langle$*angelina_jolie isMarriedTo brad_pitt*$\rangle$, which is not true anymore. Further discussion on the quality of existing KGs can be found in [Paulheim, 2017, Hogan et al., 2020, Weikum et al., 2020].

2.1.3 Related Tasks

Previously described challenges, motivate the need for developing approaches to maintain existing knowledge graphs and facilitate their exploration. In the following, we explain

the tasks facilitating KG refinement and exploration that are relevant to this thesis.

KG completion. The task of *KG completion*, or *link prediction*, is concerned with filling the gaps in the KG through predicting the missing relations among the entities in a KG. In the *KG completion* task, the available KG is assumed to store only a subset of all true facts. In addition, let us assume the existence of an *ideal* KG that contains *all and only* true facts in the world reflecting the relations from $\mathcal{R}$ that hold among the entities in $\mathcal{C}$. Note that is an abstract construct, which is normally unavailable. Therefore, the *KG completion* task concerns the reconstruction of the ideal KG (or its approximation) based on the available KG and possibly other external information sources.

Approaches for addressing *KG completion* can be roughly divided into two groups:

- Symbolic (*i.e.*, logic-based) approaches, *e.g.*, [Drabent et al., 2009, Nakashole et al., 2012a, Fierens et al., 2015, Bienvenu et al., 2016] expect background knowledge in the form of facts and inference rules and use logical reasoning to infer new facts. Inference rules are automatically learned using rule induction methods, such as AIME [Galárraga et al., 2015], RDF2Rules [Wang and Li, 2015], AnyBurl [Meilicke et al., 2019], or Rudik [Ortona et al., 2018]. We discuss rule induction in detail in Section 2.3.

- Sub-symbolic approaches (*i.e.*, statistical) learn models of statistical correlations from the existing triples in the KG and possibly some external sources, *e.g.*, text sources. Learned models, in turn, are used to predict missing facts [Wei et al., 2015, Nickel et al., 2016a]. The most prominent sub-symbolic approaches are KG embedding models, *e.g.*, [Bordes et al., 2013, Trouillon et al., 2016], where KGs are embedded into a multidimensional continuous space [Nickel et al., 2016a] reflecting the semantic relevance among KG entities. Recent years have witnessed the development of several embedding models such as TransE [Bordes et al., 2013], TransH [Wang et al., 2014b], HOLE [Nickel et al., 2016b], and many others as listed in [Nickel et al., 2016a]. More details about KG embedding models can be found in Section 2.1.4.

Some recent attempts were introduced to bridge both approaches, *e.g.*, RLvLR [Omran et al., 2018] and RulEs [Ho et al., 2018], or to compare them as in [Meilicke et al., 2018].

KG validation. *Fact validation*, or *fact-checking*, is the task concerned with evaluating the correctness of the facts, both those existing in the KG and newly added ones. Fact validation in crowd-sourced KGs, such as Wikidata, is usually conducted through collective efforts of human reviewers. Manual reviewing of the KG facts is tedious, time-consuming, and requires wide human knowledge, which is hard to obtain. Therefore, the need for the *Computational Fact Checking* arises [Ciampaglia et al., 2015].

Existing approaches for *computational fact checking* usually start with collecting pieces of evidence from different sources such as databases, news sources, and maybe the whole Web. The collected evidence is used to compute trust scores for the candidate facts [Goasdoué et al., 2013, Hogan et al., 2020]. One possibility for collecting evidence is searching for mentions for the candidate fact in textual sources via *fact-spotting* methods, *e.g.*, [Tylenda et al., 2014b, Tylenda et al., 2014a]. Afterwards, a trust score is estimated based on the reliability of the sources containing these mentions, the language style, and other lexical features [Li et al., 2011, Pasternack and Roth, 2013, Paulheim and Bizer, 2014, Nakashole and Mitchell, 2014, Gerber et al., 2015, Li et al., 2016]. The computed scores help in deciding the correctness of the candidate facts, yet they do not offer human-readable explanations. Some approaches such as [Gerber et al., 2015] provide the text snippets used in computing the scores as an indirect explanation.

Entity clustering. Entity clustering is the task of dividing a given set of entities into semantically related subgroups (*i.e.*, clusters). Clustering the entities helps in summarizing and abstracting large KGs, hence facilitating knowledge graph exploration. In addition, entity clusters form new concepts and types that could enrich the type system of the KG. One important aspect of entity clustering is the definition of the relatedness among entities, which varies according to the target application. In what follows, we provide some commonly used relatedness definitions in the context of KGs:

- In ***entity alignment*** task, which is concerned with merging redundant entities having different identifiers, related items are those instances that represent the same canonical entity. This task can be performed within a single KG to merge duplicates or over different KGs aiming at aligning entities (also known as *sameAs* problem [Raad et al., 2019]), *e.g.*, [Hassanzadeh et al., 2009, Saeedi et al., 2018, Raad et al., 2020].
- In the tasks of ***community detection*** [Vahdati et al., 2018] or *connected subgraphs discovery*, related entities are those that belong to some community or domain. For instance, clustering domain-related entities such as *movies*, *actors*, and *directors* as one cluster representing *domain of art creation*, while *books*, *authors* and *publishers* as another cluster representing *book related domain* is a form of community detection.
- ***Concept discovery***, *type formation* or *conceptual clustering*, is the task of grouping the entities that belong to the same semantic type and creating a label for each group, *e.g.*, [Lisi, 2006, Fanizzi et al., 2008, Fonseca et al., 2011, Dumancic and Blockeel, 2017, Suárez et al., 2019]. For that, relatedness of entities is estimated based on the semantic similarities among them. For example, given a set of entities

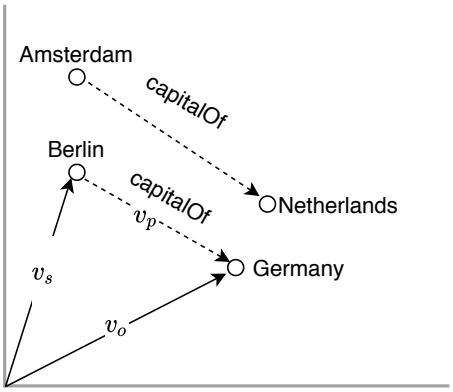


Figure 2.2: Example for translation-based embedding in 2D space

in the movie creation domain, a simplified example would be separating *movies* from *actors*. More complex concepts can also be created via *conceptual clustering*.

In this thesis, we are interested in finding more complex and fine-grained concepts, and hence, we are concerned with the *concept discovery* task in Chapter 6.

2.1.4 Embedding Models

Knowledge graph embedding is the process of representing KG entities and relations as fixed dimension vectors (or matrices). These vectors represent the positioning of the respective entities in the latent space; usually, a low-dimensional space (100 to 1000 dimensions) [Bianchi et al., 2020, Weikum et al., 2020]. Intuitively, entities with similar semantics should have similar vector representations. Then, this similarity can be estimated using vector similarity measures such as *cosine similarity*, in which two similar vectors should have a relatively small angle between them [Bianchi et al., 2020].

Usage. Representing KG entities in a continuous latent space facilitates several tasks that require estimating the relatedness among the entities, for instance, entity clustering [Dumancic et al., 2018] and entity-based recommendation systems [Ristoski et al., 2019, Guo et al., 2020]. More importantly, representing entities as real-value vectors enables integrating KGs into deep learning applications such as *deep entity linking* [Kolitsas et al., 2018]. Nevertheless, the main task for most of the existing KG embedding models is *KG completion* or *link prediction*. For that, KG embedding models are used to produce a ranked list of candidate objects for a given subject and predicate. However, even the most advanced KG embedding models cannot always rank the correct object(s) at the top [Weikum et al., 2020].

Construction. Recent years witnessed many attempts to develop effective embedding

models such as [Bordes et al., 2013, Wang et al., 2014b, Yang et al., 2015, Trouillon et al., 2016, Nickel et al., 2016b, Dettmers et al., 2018]. Developed KG embedding models principally vary concerning the following aspects:

(i) *Entities and relations representation*: While most of the existing models represent both entities and relations as real value vectors, *e.g.*, [Bordes et al., 2013, Nickel et al., 2016b], some represent the relations as matrices *e.g.*, [Nickel et al., 2011, Yang et al., 2015].

(ii) *Plausibility function*: Each model has a scoring function $f : \mathcal{C} \times \mathcal{R} \times \mathcal{C} \rightarrow \mathbb{R}$ that determines the plausibility or the likelihood of a triple. A true triple should have a higher score than other alternative out-of-KG triples sharing the same predicate and subject or object.

(iii) *Objective function*: The position of KG entities and relations in the space is determined by optimizing for a given objective function during embedding training.

Commonly, KG embedding models are categorized with respect to their *scoring function* into the following three categories [Dai et al., 2020]:

- ***Translation*** models such as TransE [Bordes et al., 2013] which embeds entities and relations as vectors assuming that $\mathbf{v_s}+\mathbf{v_r} \approx \mathbf{v_o}$ for true triples, where $\mathbf{v_s}, \mathbf{v_r}, \mathbf{v_o}$ are vector embeddings for subject s, relation r and object o, respectively (as shown in Figure 2.2). Thus, the scoring function is defined as $d(\mathbf{v_s} + \mathbf{v_r} - \mathbf{v_o})$ where d is either L_1 or L_2. During training, the models minimize the loss function which is defined as the aggregation of this score over all positive (*i.e.*, true) and negative training triples. Note that negative examples are synthetically created by corrupting the subject and/or the object of true facts. Similarly, models TransH [Wang et al., 2014b], PTransE [Lin et al., 2015], and HAKE [Zhang et al., 2020] follow the translation-based approach.
- ***Bilinear*** or ***tensor factorization***, models embed entities as vectors and relations as matrices. These models assume that for true triples, the linear mapping $\mathbf{M_r}$ of the subject embedding $\mathbf{v_s}$ is close to the object embedding $\mathbf{v_o}$: $\mathbf{v_s}\mathbf{M_r} \approx \mathbf{v_o}$. RESCAL [Nickel et al., 2011] was an early bilinear model which learned embedding through tensor factorization. Then, several enhanced models followed such as DistMult [Yang et al., 2015], HolE [Nickel et al., 2016b], and ComplEx [Trouillon et al., 2016].
- ***Neural*** models, *e.g.*, [Socher et al., 2013], form an emerging type of KG embedding models. These models utilize Neural Networks techniques to train entity embedding jointly with the task-specific weights and bias of other layers. These

shared parameters enhance the results and allow capturing more patterns, yet they make it harder to train [Rossi et al., 2020]. Neural models are grouped into categories based on the training layers: (i) *Convolution Neural Networks* such as ConvE [Dettmers et al., 2018] and ConvKB [Nguyen et al., 2018]; (ii) *Recurrent Neural Network*, *e.g.*, [Guo et al., 2019]; and (iii) *Capsule Neural Network* such as CapsE [Nguyen et al., 2019].

Both *bilinear* and *neural* models usually show better results than *translation* models on KG completion benchmarks. However, *translation* models, *e.g.*, TransE, are usually easier to train with limited computational power than the others [Rossi et al., 2020].

Previously described models are designed to be trained merely on the structure of the KG. Recently, there have been several attempts to enrich the training data by including more complex data formats and sources, for example, *path-based embedding* [Wang et al., 2014a], *logic-enhanced embedding* [Zhang et al., 2019a, Guo et al., 2016], *schema/type-aware embedding* [Lv et al., 2018], *text-enhanced embedding* [Xiao et al., 2017], and *image-enhanced embedding* [Xie et al., 2017]. More details about these variants can be found in [Bianchi et al., 2020, Rossi et al., 2020].

Limitations. While KG embedding models achieve empirically good results on KG completion benchmarks, they suffer from being limited to seen data, non-deterministic and unexplainable [Bianchi et al., 2020]. Precisely, embedding models cannot predict relations between entities unless being already defined during the training phase. In contrast to that, logical reasoning can be used to reason on new entities if the required knowledge is provided during the prediction. Moreover, the stability of the behavior of statistical models is not granted due to their dependency on big number of parameters that need to be tuned [Huang et al., 2017, Bamler et al., 2019, Martires et al., 2020]. Most critically, it is infeasible to explain the predictions of the statistical models, which makes them hard to debug and verify [Martires et al., 2020].

2.2 Symbolic Reasoning

In this section, we introduce the concepts of rules, logic programs [Eiter et al., 2009], and their usage in reasoning.

2.2.1 Rules

Intuitively, a rule is an *if-then* expression, whose *if-part* may contain several conditions (*i.e.*, atoms), possibly with negation. The *then-part* has one atom (or possibly more)

that has to hold, whenever the *if-part* holds. In general, the then-part can also contain disjunctions, but in this thesis we consider only non-disjunctive rules. More formally, a rule is defined as follows:

Definition 2.3 (Rule)**.** *A* rule r *is an expression of the form*

$$h(\vec{X}) \leftarrow b_1(\vec{Y_1}), \ldots, b_k(\vec{Y_k}), not\ \ b_{k+1}(\vec{Y_{k+1}}), \ldots, not\ \ b_n(\vec{Y_n}) \tag{2.1}$$

where $h(\vec{X}), b_1(\vec{Y_1}), \ldots, b_n(\vec{Y_n})$ *are first-order atoms and the right-hand side of the rule is a conjunction of atoms. Moreover,* $\vec{X}, \vec{Y_1}, \ldots, \vec{Y_n}$ *are tuples of either variables or constants whose length corresponds to the arity of the predicates* $h, b_1, \ldots, b_n$ *respectively.*

The left-hand side of a rule r is referred to as its head, denoted by $head(r)$, while the right-hand side is its body $body(r)$. The positive and negative parts of the body are respectively denoted as $body^+(r)$ and $body^-(r)$. The negation with *not* in $body^-(r)$ is referred to as *default negation* or *negation as failure (NAF)*, *i.e.*, $not\ \ b_n$ is true if either b_n is false or unknown. A rule r is called *positive* or ***Horn*** if $body^-(r) = \emptyset$.

Example 2.2. Consider the following rule

$$r_1 : livesIn(X, Z) \leftarrow isMarriedTo(X, Y), livesIn(Y, Z), not\ type(X, researchers)$$

The head of the rule $head(r_1) = \{livesIn(X, Z)\}$, while the body has a positive part $body^+(r_1) = \{isMarriedTo(X, Y), livesIn(Y, Z)\}$, and a negative part $body^-(r_1) = \{not\ type(X, researchers)\}$. □

A set of rules and background knowledge (*i.e.*, facts) form a *logic program* P that is used for logical reasoning. A logic program with only *Horn rules* is a **monotonic program**, meaning that the inferred clauses (*i.e.*, facts) cannot be invalidated in case new facts become known. On the other hand, a logic program with at least one rule that has *negation as failure (NAF)* atoms is a **non-monotonic program**, implying that previously concluded facts can be invalidated in the light of new knowledge (*i.e.*, facts) [Poole and Mackworth, 2017].

2.2.2 Deductive Reasoning

Deductive reasoning is the task of using some general premises to infer specific conclusions (*i.e.*, facts).

Given a logical program P, deductive reasoning is performed by substituting the variables in the body of every rule in the program with all possible constants from the

matching known facts (*i.e.*, grounding), and hence, instantiating new facts from the head atom of the rule. This process is performed recursively until no more facts can be inferred. The resulting instantiation of the logic program P (*i.e.*, consists of only ground rules) is referred to as a *ground program* $Gr(P)$.

Example 2.3. A possible grounding of the rule r_1 from Example 2.2 is given as follows

$$r_1' : livesIn(dave, chicago) \leftarrow livesIn(clara, chicago), isMarriedTo(clara, dave), \\ not\ \ researcher(dave).$$

□

In this thesis, we adopt the semantics of *Answer Sets Programming* to define the deductive reasoning task for nonmonotonic programs defined above. We start with illustrating the notion of *Interpretation* as follows:

Definition 2.4 (Herbrand Universe, Base, Interpretation)**.** *A* Herbrand universe $HU(P)$ *is a set of all constants occurring in the given program* P*. A* Herbrand base $HB(P)$ *is a set of all possible ground atoms that can be formed with predicates and constants appearing in* P*. A* ***Herbrand interpretation*** *is any subset of* $HB(P)$*.*

We now formally define the satisfaction relation.

Definition 2.5 (Satisfaction, Model)**.** *An interpretation* I satisfies

- *a ground atom* a*, denoted* $I \models a$*, if* $a \in I$*,*
- *a negated ground atom not* a*, denoted* $I \models not\ a$*, if* $I \not\models a$*,*
- *a conjunction* $b_1, \ldots, b_n$ *of ground literals, denoted* $I \models b_1, \ldots, b_n$*, if for each* $i \in \{1, \ldots, n\}$ *it holds that* $I \models b_i$*,*
- *a ground rule* r*, denoted* $I \models r$ *if* $I \models body(r)$ *implies* $I \models head(r)$*, i.e., if all literals in the body hold then the literal in the head also holds.*

An interpretation I is a *model* of a ground program P, if $I \models r$ for each rule $r \in P$. A model I is *minimal* if there is no other model $I' \subset I$. By $MM(P)$ we denote the set-inclusion minimal model of a ground positive program P.

The classical definition of *Answer Sets* is based on the Gelfond-Lifschitz reduct (GL-reduct) [Gelfond and Lifschitz, 1988] (see [Stepanova et al., 2018] for more details). However, for simplicity, we focus on the definition of answer sets based on the FLP-reduct [Faber et al., 2011], as follows:

Definition 2.6 (Faber-Leone-Pfeifer Reduct [Faber et al., 2011], Answer Set)**.** *An interpretation I of P is an* answer set *(or* stable model*) of P iff $I \in MM(fP^I)$, where fP^I is the* Faber-Leone-Pfeifer (FLP) reduct *of P, obtained from $Gr(P)$ by keeping only rules r, whose bodies are satisfied by I, i.e., $fP^I = \{r \in P \mid head(r) \leftarrow body(r), I \models body(r)\}$.*

Example 2.4. Consider the program

$$P = \left\{ \begin{array}{l} (1)\ livesIn(brad, berlin);\ (2)\ isMarriedTo(brad, ann); \\ (3)\ livesIn(Y, Z) \leftarrow isMarriedTo(X, Y), livesIn(X, Z), not\ \ researcher(Y) \end{array} \right\}$$

The relevant part of the ground instantiation $Gr(P)$ of P is obtained by substituting X, Y, Z with *brad*, *ann* and *berlin* respectively. For $I = \{isMarriedTobrad,\ ann)$, $livesIn(ann,\ berlin)$, $livesIn(brad,\ berlin)\}$, FLP-reduct contains the facts (1), (2) and the grounding of (3) $livesIn(ann,berlin) \leftarrow livesIn(brad,\ berlin), isMarriedTo\ (brad,\ ann)$, $not\ researcher(ann)$. As I is a minimal model of these reducts, I is an answer set of P. □

The answer set semantics for nonmonotonic logic programs is based on the *Closed World Assumption* (CWA), under which whatever can not be derived from a program is assumed to be false. nonmonotonic logic programs are widely applied for formalizing common sense deductive reasoning over incomplete information.

2.3 Rule Learning

Broadly speaking, *automatic rule induction*, or *rule learning*, is an important sub-field of machine learning, which focuses on symbolic methods for data analysis, *i.e.*, methods that employ a certain description language in which the learned knowledge is represented.

In the case of propositional logic, rule induction is defined as follows: Given a propositional data set (*i.e.*, single table attributed data), a set of positive examples, and possibly negative examples, rule induction constructs rules that describe the given positive examples and can be used to classify new instances [Fürnkranz et al., 2012]. One prominent example of propositional rule learning is constructing rules to describe a single class in the data, which is known as *Concept Learning* [Raedt, 2008].

Example 2.5. A potential propositional rule learned for class *researcher* could be:

$$r_2 : researcher(X) \leftarrow hasAcademicAdvisor(X), worksAtUniverity(X).$$

where rule r_2 describes the instances of class *researchers* as those with academic advisors who work at a university.

The learned rules are usually referred to as *hypothesis* because they can be falsified if new data is obtained. Furthermore, these rules follow a *hypothesis description language* (also known as *language bias*), defining the structure of the learned rules [Fürnkranz et al., 2012]. More specifically, the *description language* defines the number of atoms and variables in the rule and how they are connected.

Relational learning or *first-order inductive learning* [Raedt, 2008, Dzyuba and van Leeuwen, 2017] concerns inducing rules over more complex data models such as relational data, *i.e.*, data over multiple tables or knowledge graphs. First-order learning approaches are also referred to as *inductive logic programming (ILP)*, since the patterns they discover are expressed in the relational formalism of first-order logic (see [Raedt, 2008] for overview). The goal of ILP is to generalize individual instances/observations in the presence of background knowledge by building hypotheses about yet unseen instances. The most commonly addressed task in ILP is the task of learning logical definitions of relations. From the training tuples, ILP induces a logic program (predicate definition) corresponding to a view that defines the target relation in terms of other relations that are given as background knowledge.

Example 2.6. Given the KG in Figure 2.1 as background knowledge, a potential rule that can be produced by relational learning for the target predicate *livesIn* could be

$$r_3 : \mathit{livesIn}(X, Z) \leftarrow \mathit{isMarriedTo}(X, Y), \mathit{livesIn}(Y, Z)$$

where the conjunction of $\mathit{isMarriedTo}(X, Y)$ and $\mathit{livesIn}(Y, Z)$ is the logical definition of this predicate. □

The problem of learning rules efficiently has been tackled in several approaches that can be divided into three main paradigms; namely, *classical inductive logic programing*, *association rule mining*, and most recently *neural rule learning*. In the following, we discuss each of these paradigms.

2.3.1 Classical Inductive Logic Programming

The most prominent setting that has been extensively studied in the context of inductive logic programming concerns the extraction of a hypothesis in the form of a logic program from given sets of positive and negative examples and a logical background theory [Quinlan, 1990, Muggleton and Feng, 1990, Zeng et al., 2014].

More formally, the classical inductive logic programming task of learning from positive and negative examples (also known as learning from entailment) is defined as follows:

Definition 2.7 (Inductive Learning from Examples [Muggleton, 1991])**.**
Given:

- *Positive examples E^+ and negative examples E^- over the target n-ary relation p, i.e., sets of facts;*
- *Background knowledge T, i.e., a set of facts over various relations and possibly rules that can be used to induce the definition of p;*
- *Syntactic restrictions on the definition of p.*

Find: *A hypothesis Hyp that defines the target relation p, which is (i)* complete*, i.e., $\forall e \in E^+$, it holds that $T \cup Hyp \models e$, and (ii)* consistent*, i.e., $\forall e' \in E^-$: $T \cup Hyp \not\models e'$.*

Example 2.7. Suppose that you possess information about some of the relationships between people in your family and their genders. However, you do not know what the relationship *fatherOf* actually means. You might have the following beliefs, *i.e.*, background knowledge.

$$T = \left\{ \begin{array}{l} (1)\ parentOf(john, mary); (2)\ male(john); (3)\ parentOf(david, steve); \\ (4)\ male(david); (5)\ parentOf(kathy, ellen); (6)\ female(kathy); \end{array} \right\}$$

Moreover, you are given the following positive and negative examples.

$$\begin{array}{rl} E^+ = & \{fatherOf(john, mary), fatherOf(david, steve)\} \\ E^- = & \{fatherOf(kathy, ellen), fatherOf(john, steve)\} \end{array}$$

One of the possible hypotheses that can be induced from the above knowledge reflecting the definition of the *fatherOf* relation is given as follows:

$$Hyp : fatherOf(X, Y) \leftarrow parentOf(X, Y), male(X).$$

This hypothesis is consistent with the background theory T, and together with T it entails all of the positive examples, and none of the negative ones. The classical ILP task concerns automatic extraction of such hypothesis. □

Rule construction. The core idea for classical ILP approaches is to attempt constructing a hypothesis that covers the positive examples and does not cover the negative examples, making the rule consistent. ILP approaches start with defining a search space from the most general or universal rule with a body defined as *True* (*i.e.*, applicable

for all positive examples) to the most specific rules with the body as *False* (*i.e.*, not applicable for any example) and other rules are more specific than the universal rule and more general than the most specific ones. To traverse the space, two refinement operators are used: (i) *Add an atom* conjunctively to the body of the rule, which makes it more specific. (ii) *Remove an atom* from the body of the rule to make it more general.

Classical ILP approaches follow one of the two strategies to search for valid rules:

- *General-to-specific* (*Top-down*) strategy: Starting with the most general rule, the procedure recursively *adds atoms* to make it more specific to the positive examples, stopping when none of the negative examples is covered.
- *Specific-to-general* (*Bottom-up*) strategy: The search procedure starts with the most specific rule and relaxes it by removing atoms from it as long as the produced rule still does not cover negative examples.

Rule evaluation. Their coverage of the positive examples determines the quality of the rules. Their consistency is inversely correlated with their coverage of the negative examples. Several measures were introduced to combine both aspects such as *rule precision*, *coverage difference* between positive and negative examples, and *rate difference* between the true positive rate and the false positive rate [Raedt, 2008].

Example approaches. To date, the main tasks considered in the ILP area can be classified based on the following parameters [Sazonau and Sattler, 2017]:

- *type of the data source*, *e.g.*, positive/negative examples, interpretations, text;
- *type of the output knowledge*, *e.g.*, Horn/Nonmonotonic rules over single or multiple predicates, description logic (DL) class descriptions, class inclusions;
- *the way the data is given as input*, *e.g.*, all data at once or incrementally;
- *availability of an oracle*, *e.g.*, involvement of a human expert in the loop;
- *quality of the data source*, *e.g.*, noisy or clean;
- *data (in)completeness assumption*, *e.g.*, OWA, CWA;
- *availability and type of background knowledge*, *e.g.*, DL ontology, set of datalog rules, hybrid theories, etc.; and
- *search strategy*, general-to-specific or specific-to-general [Raedt, 2008].

An overview of some of the systems for Horn and Nonmonotonic (NM) rule induction with their selected properties is presented in Table 2.2.

Limitations. The majority of the existing classical rule induction methods mentioned above assume that the given data from which the rules are induced is complete, accurate, and representative. Therefore, they rely on CWA and aim at extracting rule hypotheses

System	Output rules	Multiple predicates	Increment	Interact	Noise handling
CIGOL [MUGGLETON and BUNTINE, 1988]	Horn	yes	yes	yes	no
FOIL [Quinlan, 1990]	Horn	no	no	no	yes
GOLEM [Muggleton and Feng, 1990]	Horn	no	no	no	yes
LINUS [Dzeroski and Lavrac, 1991]	Horn	no	no	no	yes
CLINT [De Raedt and Bruynooghe, 1991]	Horn	yes	yes	yes	no
MPL [Raedt et al., 1993]	Horn	yes	no	no	no
MOBAL [Morik, 1993]	Horn	yes	yes	no	no
ALEPH [Srinivasan, 2001]	Horn	no	no	yes	yes
σILP [Evans and Grefenstette, 2018]	Horn	no	no	no	yes
DROPS [Corapi et al., 2010]	NM	no	no	no	no
ASPAL [Corapi et al., 2012]	NM	no	no	no	no
XHAIL [Ray, 2009]	NM	no	no	no	no
ILED [Katzouris et al., 2015]	NM	no	yes	no	no

Table 2.2: Overview of classical ILP systems.

that perfectly satisfy the criteria from Definition 2.7. On the other hand, as discussed in Section 2.1.2, knowledge graphs are highly incomplete, noisy, and biased. Moreover, the real world is very complicated, and its exact representation often cannot be acquired from the data, meaning that the task of inducing a perfect rule set from a KG is typically unfeasible. Therefore, in the context of KGs, one normally aims at extracting certain regularities from the data, which are not universally correct, but when seen as rules predict a sufficient portion of true facts.

Moreover, reusing the methods that induce logical theories from a set of positive and negative examples from Definition 2.7 for learning rules for KG completion is not feasible for several reasons: First, the target predicates (*e.g.*, *fatherOf* from Example 2.7) can not be easily identified, since we do not know which parts of the considered KG need to be completed. A standard way of addressing this issue would be just to learn rules for all the different predicate names occurring in the KG. Unfortunately, this is unfeasible given the substantial size of real-world KGs.

Moreover, the negative examples required in most ILP are not available because the CWA cannot be directly applied, and they cannot be easily obtained from domain experts due to the substantial size of KGs.

2.3.2 Relational Association Rule Mining

To overcome the limitations of the classical ILP methods, it is more appropriate to treat the KG completion problem as an unsupervised relational learning task [Galárraga et al., 2015]. Therefore, *relational association rule learning* techniques were adapted

for the extraction of *Horn rules* from incomplete KGs. These concern the discovery of frequent patterns from a data set and their subsequent transformation into rules (see, *e.g.*, [Dehaspe and Raedt, 1997] as the seminal work in this direction).

Association rule mining. An association rule is a rule where certain properties of the data in the body of the rule are related to other properties in its head. For example, consider a database containing transactional data from a store selling computer equipment. From this data, one can extract the association rule stating that 70% of the customers buying a laptop also buy a docking station. The knowledge that such a rule reflects can assist in planning the store layout or deciding which customers are likely to respond to an offer.

Traditionally, the discovery of association rules has been performed on data stored in a single table. Recently, however, many methods for mining relational, *i.e.*, graph-based data have been proposed [Goethals and den Bussche, 2002, Raedt, 2008].

Relational association rules. The notion of multi-relational association rules is heavily based on frequent conjunctive queries and query subsumption [Goethals and den Bussche, 2002]. So, we start with defining the conjunctive query under the semantics of knowledge graphs as follows:

Definition 2.8 (Conjunctive Query)**.** *Given a graph $\mathcal{G}$ under the language signature $\Sigma_{\mathcal{G}} = \langle \mathcal{R}, \mathcal{C} \rangle$, where $\mathcal{R}$, $\mathcal{C}$ are sets of relations and constants respectively. A* conjunctive query *Q over $\mathcal{G}$ is an expression of the form $p_1(\mathbf{X_1}), \ldots, p_m(\mathbf{X_m})$, where $\vec{X_i}$ are symbolic variables or constants and $p_i \in \mathcal{R}$ predicates. The* answer *of Q on $\mathcal{G}$ is the set $Q(\mathcal{G}) = \{(\nu(\mathbf{X_1}), \ldots, \nu(\mathbf{X_m}) \mid \forall i : p_i(\nu(X_i), \nu(Y_i)) \in \mathcal{G}\}$ where ν is a function that maps variables and constants to elements of $\mathcal{C}$.*

The *(absolute) support* of a conjunctive query Q with respect to a KG $\mathcal{G}$, is the number of distinct tuples in the answer of Q on $\mathcal{G}$ [Dehaspe and Raedt, 1997, Goethals and den Bussche, 2002].

Example 2.8. The support of the query

$$Q(X, Y, Z) \text{ :- } \mathit{marriedTo}(X, Y), \mathit{livesIn}(X, Z)$$

over $\mathcal{G}$ in Figure 2.1 asking for people, their spouses and living places is equal to 6, which is the size of the set of tuples $\{\langle \mathit{brad\ ann\ berlin} \rangle, \langle \mathit{john\ kate\ chicago} \rangle, \langle \mathit{bob\ alice\ berlin} \rangle, \langle \mathit{sue\ li\ beijing} \rangle, \langle \mathit{clara\ dave\ chicago} \rangle, \langle \mathit{mat\ lucy\ amsterdam} \rangle\}$ □

Definition 2.9 (Association Rule)**.** *An* association rule *is of the form $Q_1 \Rightarrow Q_2$, such that Q_1 and Q_2 are both conjunctive queries, and the body of Q_1 considered as a set of atoms is included in the body of Q_2, i.e., $Q_1(\mathcal{G}) \subseteq Q_2(\mathcal{G})$ for any possible KG $\mathcal{G}$.*

Example 2.9. For instance, from the above $Q(X, Y, Z)$ and

$$Q'(X, Y, Z) \text{ :- } \mathit{marriedTo}(X,\ Y),\ \mathit{livesIn}(X, Z), \mathit{livesIn}(Y, Z)$$

we can construct the association rule $Q \Rightarrow Q'$. □

Association rules are sometimes exploited for reasoning purposes, and thus (with some abuse of notation) can be treated as logical rules, *i.e.*, for $Q_1 \Rightarrow Q_2$ we write $Q_2 \backslash Q_1 \leftarrow Q_1$, where $Q_2 \backslash Q_1$ refers to the set difference between Q_2 and Q_1 considered as sets of atoms. For example, $Q \Rightarrow Q'$ from above corresponds to r_3 from Example 2.6.

Rule construction. Given a KG $\mathcal{G}$, *query description language* $\mathcal{L}$, and a monotonic query evaluation function q (*e.g.*, frequency of the query), relational association rule mining proceeds in two steps:

- First, it starts with frequent query mining similar to Apriori Algorithm [Agrawal et al., 1996]. Queries are iteratively extended with all possible extensions from the given dataset until they reach some length. After each expansion, the generated query candidates are evaluated against q and if they are less than some pre-set threshold they are pruned.
- Second, association rules are generated from the resulting queries by choosing one of the atoms in the query as the head of the rule and the other atoms as its body.

Rule evaluation. A large number of measures for evaluating the quality of association rules and their subsequent ranking have been proposed. Measures of *support, confidence* are the most prominent and basic ones.

For $r :\ H \leftarrow B, \mathit{not}\ \ E$, with $H = h(X,\ Y),\ B = \mathit{body}^+(r), E = \mathit{body}^-(r)$ involving variables from $\vec{Z} \supseteq \{X, Y\}$ and a KG $\mathcal{G}$, the *(standard) confidence* is given as:

$$\mathit{conf}(r, \mathcal{G}) = \frac{\mathit{r\text{-}supp}(r, \mathcal{G})}{\mathit{b\text{-}supp}(r, \mathcal{G})}$$

where $\mathit{r\text{-}supp}(r, \mathcal{G})$ and $\mathit{b\text{-}supp}(r, \mathcal{G})$ are the *rule support* and *body support*, respectively, which are defined as follows:

$$\mathit{r\text{-}supp}(r, \mathcal{G}) = \#(x, y) : h(x, y) \in \mathcal{G}, \exists \vec{z}\ B \in \mathcal{G}, E \notin \mathcal{G}$$
$$\mathit{b\text{-}supp}(r, \mathcal{G}) = \#(x, y) : \exists \vec{z}\ B \in \mathcal{G}, E \notin \mathcal{G}$$

where $x, y \in \mathcal{C}$ are constants (*i.e.*, entities) substituting the variable X and Y. Similarly, $\vec{z} \subseteq \mathcal{C}$ is a tuple of entities substituting other variables in $\vec{Z} \setminus X, Y$ respectively.

Example 2.10. Consider the previously demonstrated rules:

$$r_1 : \mathit{livesIn}(X, Z) \leftarrow \mathit{isMarriedTo}(X, Y), \mathit{livesIn}(Y, Z), \mathit{not\ researcher}(X)$$
$$r_3 : \mathit{livesIn}(X, Z) \leftarrow \mathit{isMarriedTo}(X, Y), \mathit{livesIn}(Y, Z)$$

and the KG $\mathcal{G}$ in Figure 2.1, we have $\mathit{r\text{-}supp}(r_1, \mathcal{G}) = \mathit{r\text{-}supp}(r_3, \mathcal{G}) = 3$, $\mathit{b\text{-}supp}(r_1, \mathcal{G}) = 4$, and $\mathit{b\text{-}supp}(r_3, \mathcal{G}) = 6$. Hence, $\mathit{conf}(r_1, \mathcal{G}) = \frac{3}{4}$ and $\mathit{conf}(r_3, \mathcal{G}) = \frac{3}{6}$. □

Example approaches. Several relational association rule mining approaches starting with WARMeR [Goethals and den Bussche, 2002] attempted to enhance the performance and quality of the discovered rules over real-world knowledge graphs. The main differences between these approaches are the imposed description language, the quality evaluation function, and accordingly search heuristics and search space pruning methods. The most prominent examples of systems that are specifically tailored towards inducing Horn rules from KGs are:

- **AMIE** [Galárraga et al., 2015] followed a similar procedure to the described above. Yet, they adopt a variety of techniques from the database area and search heuristics; allowing for parallel processing and better scalability. Furthermore, AIME mines *closed* Horn rules, in which every variable appears at least twice, *e.g.*, r_2 from 2.6. Restricting to closed rules ensures predicting actual fact by the generated rule, but not just its existence [Galárraga et al., 2015]. For that, they defined a set of refinement operators used to extend the query:
  - *add dangling atom*: add a new positive atom with one fresh variable, *i.e.*, variable not appearing elsewhere in the rule;
  - *add an instantiated atom*: add a positive atom with one argument being a constant and the other one being a shared variable, *i.e.*, variable already present in another rule atom;
  - *add closing atom*: add a positive atom with both of its arguments as shared variables.

  Additionally, AMIE introduced an evaluation function for the produced rules under the *Partial Completeness Assumption (PCA)* (see Section 2.1.2). PCA-based confidence was shown effective in ranking rules based on their predictive quality.

- **RDF2Rules** [Wang and Li, 2015] parallelizes the process of the query construction process by extracting *frequent predicate cycles* (FPCs) of a certain length k, which have the following form:

$$(X_1, p_1^{d_1}, X_2, p_2^{d_2}, \ldots, X_k, p_k^{d_k}, X_1)$$

where, X_is are variables to appear in the extracted rules, $p_{i=1,\dots,k}$ are predicates connecting these variables, and $d_{i=1,\dots,k} \in \{0, 1\}$ reflect the direction of the respective edges in the KG labeled by the respective predicates. Then, the rules are extracted from them by choosing a single predicate to be in the head of the rule, and collecting the rest into its body. RDF2Rules introduced *soft confidence* as a scoring function to rank the learned rules under OWA.

RDF2Rules is capable of accounting for unary predicates (*i.e.*, types of entities), which are avoided in AMIE for scalability reasons. Unary predicates are added to the constructed rule at the final stage after analyzing the frequent types for FPCs corresponding to a given rule. While RDF2Rules performs the rule extraction faster than AMIE due to their effective pruning strategy, the supported rule patterns are more restrictive.

Other recent approaches have managed to get better results with respect to the structure of the rules and the ranking function, *e.g.*, Rudik [Ortona et al., 2018] and CARL [Tanon et al., 2017]. Further approaches focus on enhancing the mining computational efficiency, such as Anyburl [Meilicke et al., 2019] and *ontological path-finding* (OP) [Chen et al., 2016].

2.3.3 Neural Rule Learning

To the best of our knowledge, rule induction via sub-symbolic reasoning, *i.e.*, Neural Networks, was introduced in RuleNet [Mcmillan et al., 1992]. RuleNet proposed a framework for integrating sub-symbolic classification of words and rule learning in one process. Nevertheless, the neural approach was not adapted for rule induction over KGs until recently.

The rise of knowledge graph embedding models (see Section 2.1.4) motivated several attempts for learning rules on the sub-symbolic level. One seminal work in this direction is Neural-LP [Yang et al., 2017] based on the idea of modeling rules as differential operations introduced by TensorLog [Cohen, 2016]. More specifically, Neural-LP proposes to utilize Neural Networks to learn an estimation for such differential operators.

Neural-LP was followed by several other attempts to learn logical rules that can be used for link predictions via sub-symbolic reasoning, for example, ReINN [Pandey et al., 2018], RLvLR [Omran et al., 2018], Iter [Zhang et al., 2019a], DRUM [Sadeghian et al., 2019], and Neural-LP-N [Wang et al., 2020].

The core advantage of the Neural-based learning approaches is their ability to capture hidden correlations that are not easily captured by symbolic learning due to the

incompleteness and bias of the existing KGs. Also, they help to scale the learning process compared to the approaches of ILP and association rule mining. Additionally, the latent representation of the KG supports including other sources, *e.g.*, text, during the rule learning process. Most importantly, modeling rule learning as neural network layers enables the integration of rule learning in other neural-based tasks. Such integration can offer a good proxy to explain the results of these systems. For example, CrossE [Zhang et al., 2019b] is a KG embedding model designed to produce rule-like explanations for the predicted triples.

Despite the advantages of Neural-based learning, the learned rules are still limited to simple and short structures. Existing approaches usually produce rules with at most two conjunctive atoms, which usually express simple transitivity.

Chapter 3

ExRuL: Exception-aware Rule Learning

While real-world knowledge graphs (KGs) contain billions of facts, they are far from being complete or ideal. Manually curating the knowledge is tedious and time-consuming. Meanwhile, knowledge graphs encode many implicit correlations. These correlations can be analyzed to infer Horn rules and to predict new facts. However, Horn rules do not consider possible exceptions, so predicting facts via such rules introduces errors.

To overcome this problem, we present *ExRuL*, a method for effective revision of Horn rules into *exception-aware rules* (*i.e.*, nonmonotonic rules). We achieve that by adding exceptions (*i.e.*, negated atoms) mined from the KG to the bodies of the rules. Experiments on real-world KGs demonstrate the effectiveness of the *exception-aware* rules reducing the prediction errors compared to Horn rules. Moreover, the revised rules do not just explain the inferred facts but also indicate when the rules should not infer certain triples.

3.1 Introduction

Motivation and problem. While recent advances in information extraction have led to the construction of large-scale knowledge graphs (KGs) , KGs are still incomplete and contain errors. To complete and curate a KG, inductive logic programming and data mining techniques (*e.g.*, [Wang and Li, 2015, Galárraga et al., 2015, Chen et al., 2016]) have been used to identify prominent patterns, such as *"Married people live in the same place"*, and cast them in the form of Horn rules, such as:

$$r_1 : livesIn(Y, Z) \leftarrow isMarriedTo(X, Y), livesIn(X, Z).$$

This has twofold benefits. First, since KGs operate under the Open World Assumption (OWA), the rules can be used to derive additional facts. For example, applying the rule

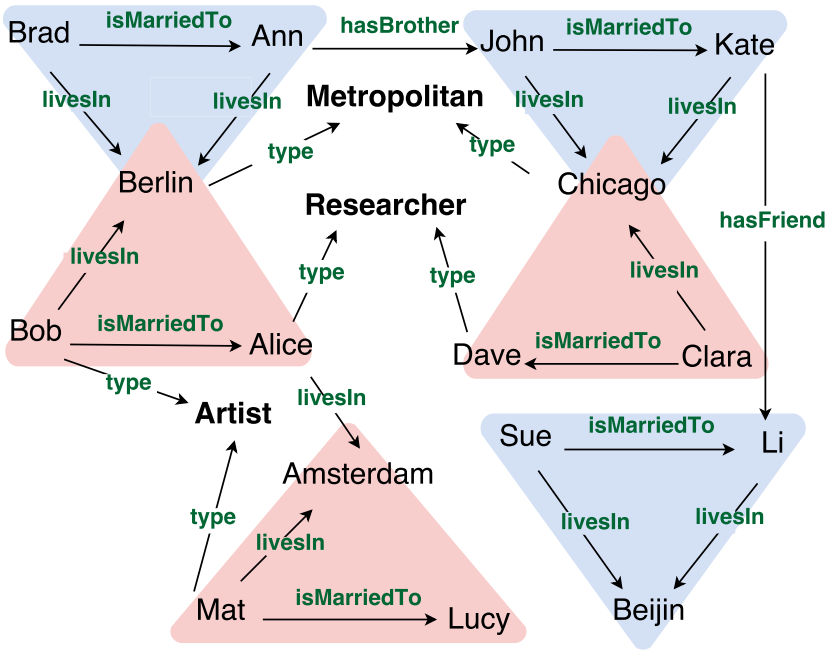


Figure 3.1: Sample KG with exceptions

r_1 mined from the graph in Figure 3.1, the missing living place of Dave can be deduced based on the data about his wife Clara. Second, rules can be used to eliminate erroneous facts in the KG. For example, assuming that *livesIn* is a functional relation, Amsterdam as a living place of Alice could be questioned as it differs from her husband's.

State-of-the-art and its limitations. Methods for learning rules from KGs are typically based on inductive logic programming or association rule mining (as discussed in Chapter 2). However, these methods are limited to Horn rules, *i.e.*, all predicates in the rule body are positive. This is insufficient to capture rules that have exceptions, such as *"Married people live in the same place unless one is a researcher"*:

$$r_2{:}livesIn(Y,Z) \leftarrow isMarriedTo(X,Y), livesIn(X,Z), not\ \ researcher(Y).$$

This additional knowledge along with the KG in Figure 3.1 could offer an explanation for Alice living in an unexpected place. If r_2 often holds, then one can no longer complete the missing living place for Dave by assuming that he lives with his wife Clara. Thus, understanding exceptions is crucial for KG completion and curation.

Our goal is to learn rules with exceptions, also known as *nonmonotonic rules*. Learning nonmonotonic rules under the Closed World Assumption (CWA) is a well-studied problem that lies at the intersection of inductive and abductive logic programming (*e.g.*,

[Sakama, 2005, Ray, 2009]). However, these methods cannot be applied to KGs treated under the OWA.

Approach and contribution. We present **ExRuL**, a novel method that takes a KG and a set of Horn rules as input and yields a set of exception-aware rules as output. The output rules are no longer necessarily Horn clauses (*e.g.*, rule r_2 above could be in our output). So we essentially tackle a variant of a *theory revision* problem [Wrobel, 1996] under OWA.

ExRuL proceeds in four steps: First, we compute what we call "exception witnesses" predicates that are potentially involved in explaining exceptions (*e.g.*, *researcher* in our example). Second, we generate nonmonotonic rule candidates that we could possibly add to our KG rules. Third, we devise quality measures for nonmonotonic rules to quantify their strength with respect to the KG. In contrast to prior work, we do not merely give measures for individual rules in isolation, but also consider their cross-talk through a new technique that we call "partial materialization". Fourth and last, we rank the nonmonotonic rules by their strengths and choose a cut-off point such that the obtained rules describe the KG's content as well as possible with awareness of exceptions.

The salient contributions of this chapter are:

- We introduce *ExRuL*, a framework for nonmonotonic rule mining as a knowledge revision task to capture exceptions from Horn rules and overcome the limitations of prior work on KG rule mining.
- We developed an algorithm for computing exception candidates, measuring their quality, and ranking them based on a novel technique that considers *partial materialization* of judiciously selected rules.
- We proposed a generalization of *ExRuL* to revise Horn rule learned from multi-relational data.
- We evaluate our approach on real-world KGs, namely YAGO3 and IMDB KGs. Experimental results show the gains of our method for rule quality as well as fact quality when performing KG completion.

3.2 Preliminaries

Propositionalized knowledge graph. Propositionalization is the process of transforming a relational dataset into a propositional one. Relations between different tables

	bornInUS	*livesInUS*	*stateless*	*immigrant*	*singer*	*poet*	*hasUSPass*
p1	✓	✓			✓		✓
p2	✓	✓					✓
p3	✓	✓			✓	✓	✓
p4	✓	✓					✓
p5	✓	✓	✓				
p6	✓		✓				
p7	✓		✓				
p8	✓		✓	✓			
p9	✓			✓		✓	
p10	✓			✓	✓	✓	
p11	✓				✓	✓	✓

Figure 3.2: Example of a propositional KG about some USA inhabitants

(or entities) are converted to *attribute-value* pairs or unary predicates *in first-order logics* [Kramer et al., 2001, Ristoski and Paulheim, 2014].

In this chapter, we first describe our method using rules mined over unary predicates. For that, we apply a simple form of relation propositionalization on the given KG to obtain a new version of the KG with only unary predicates. More specifically, we project binary relations into multiple unary ones by concatenating the predicate with one of its arguments, *e.g.*, the predicate *livesIn* in Figure 3.1 can be translated into three unary ones *livesInAmsterdam, livesInBerlin, livesInChicago* when concatenated to the respective objects. The produced KG can then be represented in a single transactional table where the rows are the subjects and columns are the unary predicates or attributes, similar to the table shown in Figure 3.2.

Nonmonotonic logic programs. Recalling the general definition for rules from Chapter 2, a *(nonmonotonic) logic program* P is a set of *rules* of the form

$$H \leftarrow B, not \;\; E \tag{3.1}$$

where H is an atom of the form $a(\vec{X})$ and is called the rule head $Head(r)$, B is a conjunction of positive atoms of the form $b_1(\vec{Y_1}), \ldots, b_k(\vec{Y_k})$ to form $Body^+(r)$, and $not \;\; E$, denotes the conjunction of atoms $not \;\; b_{k+1}(\vec{Y_{k+1}}), \ldots, not \;\; b_n(\vec{Y_n})$. Here, not is the so-called *negation as failure (NAF)* or *default negation*. The negated part of the body is

denoted as $Body^{-}(r)$.

Rule-based KG completion. The task of *KG completion* is discussed in Section 2.1.3. For this Chapter, we formally define the task of rule-based completion as follows:

Definition 3.1 (Rule-based KG Completion)**.** *Let $\mathcal{G}$ be a KG over the signature $\Sigma_{\mathcal{G}} = \langle \mathcal{R}, \mathcal{C} \rangle$. Let, moreover, R be a set of rules with predicates from $\mathcal{R}$ induced from $\mathcal{G}$. Then* rule-based completion of $\mathcal{G}$ with respect to *R is a graph $\mathcal{G}_R$ constructed from any answer set $\mathcal{G}_R \in AS(R \cup \mathcal{G})$.*

Example 3.1. Consider the propositional KG $\mathcal{G}$ given in a tabular form in Figure 3.2, where a tick appears in an intersection of a row s and a column o, if $o(s) \in \mathcal{G}$ (*i.e.*, $\langle s \ \ isA \ \ o \rangle \in \mathcal{G}$). Suppose we are given a set of rules $\mathcal{R} = \{r_1, r_2\}$, where

$$r_1 : livesInUS(X) \leftarrow bornInUS(X), not \ \ immigrant(X);$$
$$r_2 : livesInUS(X) \leftarrow hasUSPass(X).$$

The program $\mathcal{G} \cup \mathcal{R}$ has a single answer set $\mathcal{G}_{\mathcal{R}}$=$\mathcal{G} \cup \{livesInUS(p_i) \mid i$=$6, 7, 11\}$, from which the completion $\mathcal{G}_{\mathcal{R}}$ of $\mathcal{G}$ can be reconstructed. □

Example 3.2. Similarly for relational data, given the KG in Figure 3.1 as $\mathcal{G}$ and the rule set $R = \{livesIn(Y, Z) \leftarrow marriedTo(X, Y), livesIn(X, Z)\}$, we have $\mathcal{G}_R = \mathcal{G} \cup \{livesIn(lucy, amsterdam)\}$. □

3.3 Learning Exception-aware Rules

Horn rule revision. Before we formally define our problem, we introduce the notion of an incomplete data source following [Darari et al., 2013].

Definition 3.2 (Incomplete Data Source)**.** *An incomplete data source is a pair $G = (\mathcal{G}^a, \mathcal{G}^i)$ of two KGs, where $\mathcal{G}^a \subseteq \mathcal{G}^i$ and $\Sigma_{\mathcal{G}^a} = \Sigma_{\mathcal{G}^i}$. We call $\mathcal{G}^a$ the available graph and $\mathcal{G}^i$ the ideal graph.*

The graph $\mathcal{G}^a$ is the graph that we have available as input. The ideal graph $\mathcal{G}^i$ is the perfect completion of $\mathcal{G}^a$, which is supposed to contain all correct facts with entities and relations from $\Sigma_{\mathcal{G}^a}$ that hold in the current state of the world.

Given a potentially incomplete graph $\mathcal{G}^a$ and a set of Horn rules $\mathcal{R}_H$ mined from $\mathcal{G}^a$, our goal is to add default negated atoms (*i.e.*, exceptions) to the rules in $\mathcal{R}_H$ and obtain a revised ruleset $\mathcal{R}_{NM}$ such that the set difference between $\mathcal{G}^a_{\mathcal{R}_{NM}}$ and $\mathcal{G}^i$ (the red area in Figure 3.3) is smaller than between $\mathcal{G}^a_{\mathcal{R}_H}$ and $\mathcal{G}^i$ (the gray area in Figure 3.3). If in

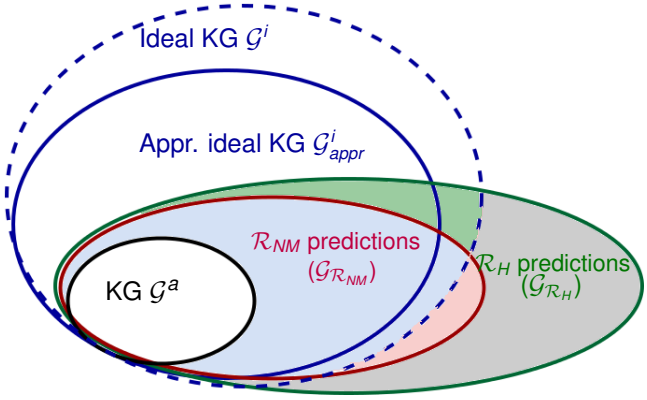


Figure 3.3: Relations between the ideal, approximated and available slices of a KG

addition the set difference between $\mathcal{G}^a_{\mathcal{R}_{NM}}$ and $\mathcal{G}^i$ is the smallest among the ones produced by other revisions $\mathcal{R}'_{NM}$ of $\mathcal{R}_H$, then we call $\mathcal{R}_{NM}$ an *ideal nonmonotonic* revision. For single rules such revision is defined as follows:

Definition 3.3 (Ideal Nonmonotonic Revision)**.** *Let $G = (\mathcal{G}^a, \mathcal{G}^i)$ be an incomplete data source. Moreover, let $r: \ a \leftarrow b_1, \ldots, b_k$ be a Horn rule mined from $\mathcal{G}^a$. An* ideal nonmonotonic revision *of r with respect to $\mathcal{G}$ is any rule*

$$r': \ a \leftarrow b_1, \ldots, b_k, not \ \ b_{k+1}, not \ \ b_n, \tag{3.2}$$

such that $\mathcal{G}^i \triangle \mathcal{G}^a_{r'} \subset \mathcal{G}^i \triangle \mathcal{G}^a_r$ [1], i.e., the completion of $\mathcal{G}^a$ based on r' is closer to $\mathcal{G}^i$ than the completion of $\mathcal{G}^a$ based on r, and $\mathcal{G}^a_{r''} \triangle \mathcal{G}^i \subset \mathcal{G}^a_{r'} \triangle \mathcal{G}^i$ for no other nonmonotonic revision $r'' \neq r'$ of r. If k=n, then the revision coincides with the original rule.

In this work, we assume that the ideal graph $\mathcal{G}^i$ is not available. Therefore, we cannot verify whether a revision is ideal for $\mathcal{R}_H$. However, we estimate based on some quality functions whether a given revision produces an approximation of $\mathcal{G}^i$ that is better than the approximation produced by the original Horn ruleset. For this purpose, we introduce a generic quality function q which receives as input a revision $\mathcal{R}_{NM}$ of the ruleset $\mathcal{R}_H$ and a graph $\mathcal{G}$, and returns a real value that reflects the quality of the revised set $\mathcal{R}_{NM}$.

We can now formally define our problem as follows:

Definition 3.4 (Quality-based Horn Rule Revision)**.**
Given: *KG $\mathcal{G}$, set of non-ground Horn rules $\mathcal{R}_H$ mined from $\mathcal{G}$, and quality function q.*
Find: *a set of rules $\mathcal{R}_{NM}$ obtained by adding default negated atoms to $Body^-(r)$ for some $r \in \mathcal{R}_H$, such that $q(\mathcal{R}_{NM}, \mathcal{G})$ is maximal.*

[1] $\mathcal{G}_1 \triangle \mathcal{G}_2 = (\mathcal{G}_1 \backslash \mathcal{G}_2) \cup (\mathcal{G}_2 \backslash \mathcal{G}_1)$

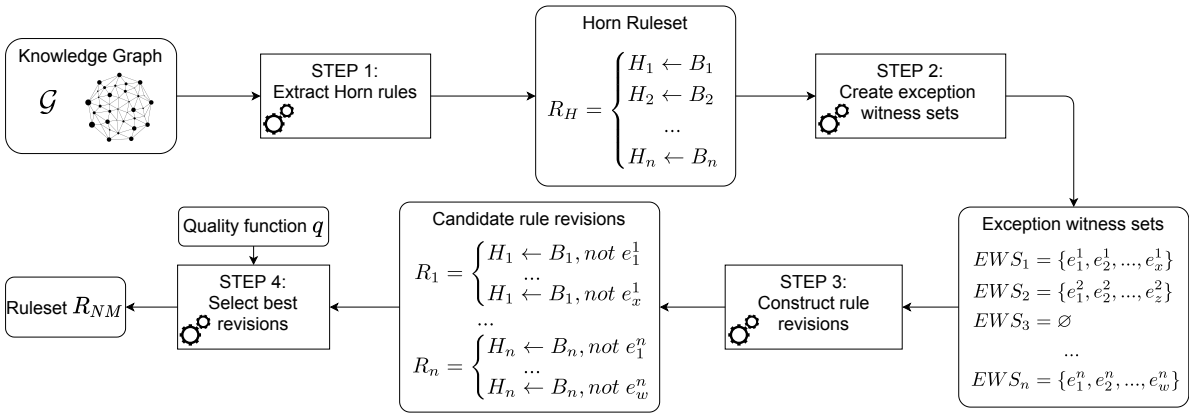


Figure 3.4: ExRuL general overview

Note that so far we did not specify the quality function q. In our approach, we estimate the quality of a ruleset by exploiting well-established measures proposed in the field of data mining [Azevedo and Jorge, 2007]. Even though none of these measures can offer any sort of guarantee, our hypothesis is that they still indicate to some extent the percentage of correctly predicted facts obtained as a result of completing a KG based on a given ruleset. We discuss in Section 3.5 the possible specific definition for q.

Approach overview. Figure 3.4 illustrates the main phases of our approach. In Step 1, we launch an off-the-shelf algorithm to mine Horn rules from the input KG. We use FPGrowth [Han et al., 2004], but any other, *e.g.*, [Chen et al., 2016], [Galárraga et al., 2015] can be likewise applied, *i.e.*, our overall revision approach is independent of the concrete technique used for Horn rule mining. Then, for each rule we compute *normal* and *abnormal* instance sets, defined as:

Definition 3.5 (r-(ab)normal Instance Set)**.** *Given the factual representation of a KG $\mathcal{G}$ and $r: a(X) \leftarrow b_1(X), \dots, b_k(X)$ a Horn rule mined from $\mathcal{G}$. Then,*

- $NS(r, \mathcal{G}) = \{c \mid b_1(c), \dots, b_k(c), a(c) \in \mathcal{G}\}$ *is an* r-normal instance set*;*
- $ABS(r, \mathcal{G}) = \{c \mid b_1(c), \dots, b_k(c) \in \mathcal{G}, a(c) \notin \mathcal{G}\}$ *is an* r-abnormal instance set*.*

Example 3.3. Given the propositional KG $\mathcal{G}$ from Figure 3.2 and the rule

$$r: \ livesInUS(X) \leftarrow bornInUS(X)$$

while the r-normal set is as $NS(r, \mathcal{G}) = \{p1, \dots, p5\}$, the r-abnormal instance sets is $ABS(r, \mathcal{G}) = \{p6, \dots, p11\}$. □

Intuitively, if the given data was complete, then the r-normal and r-abnormal instance sets would exactly correspond to instances for which the rule r holds (respectively does

Algorithm 1: *ComputeEWS*: compute $EWS(r, \mathcal{G})$

Input: KB $\mathcal{G}$, rule $r: \ a(X) \leftarrow b_1(X), \ldots, b_k(X)$
Output: $EWS(r, \mathcal{G})$

1 $N \leftarrow NS(r, \mathcal{G})$; $A \leftarrow ABS(r, \mathcal{G})$
2 $E^+ \leftarrow \{not\_a(c) \mid c \in A\}$; $E^- \leftarrow \{not\_a(c) \mid c \in N\}$
3 $\mathcal{R}_e \leftarrow Learn(E^+, E^-, \mathcal{G})$
4 $EWS \leftarrow \{\text{predicate } p \text{ in } Body^+(r') \mid r' \in \mathcal{R}_e, \text{ s.t. } , p \text{ is not in } Body^+(r)\}$
5 **return** EWS

not hold) in the real world. Since the KG is potentially incomplete, this is no longer the case and some r-abnormal instances might in fact be classified as such due to data incompleteness. In order to distinguish between the "wrongly" and "correctly" classified instances in the r-abnormal set, in Step 2 we construct *exception witness sets* (EWS), which are defined as follows:

Definition 3.6 (Exception Witness Set (EWS))**.** *Given a KG $\mathcal{G}$ and let r be a Horn rule mined from $\mathcal{G}$. An* r-exception witness set $EWS(r, \mathcal{G}) = \{e_1, \ldots, e_l\}$ *is a maximal set of predicates, such that*

- $e_i(c') \in \mathcal{G}$ *for some* $c' \in ABS(r, \mathcal{G})$, $1 \leq i \leq m$ *and*
- $e_1(c), \ldots, e_m(c) \notin \mathcal{G}$ *for all* $c \in NS(r, \mathcal{G})$.

Example 3.4. For $\mathcal{G}$ and r from Example 3.3 $EWS(r, \mathcal{G})$=$\{immigrant\}$ is an r-exception witness set. For $\mathcal{G}'$=$\mathcal{G}\backslash\{p5\}$ it holds that $EWS(r, \mathcal{G}')$=$\{immigrant, stateless\}$. □

After EWSs are computed for all rules in $\mathcal{R}_H$, we use them to create potential revisions in Step 3. Then, we rank the newly created revisions and select the best ones using different criteria (Step 4). These selected rules will constitute the new $\mathcal{R}_{NM}$.

3.4 Constructing Potential Rule Revisions

In this section we describe how we calculate the exception witness sets for Horn rules (Figure 3.4, Step 2) and how we create potential rule revisions (Figure 3.4, Step 3).

Computing exception witness sets. For constructing exception witness sets we use the algorithm *ComputeEWS* (Alg. 1), which given a propositional KG $\mathcal{G}$ and a rule $r \in \mathcal{R}_H$ as input, outputs the set $EWS(r, \mathcal{G})$.

The algorithm works as follows: First in line 1, r-normal $NS(r, \mathcal{G})$ and r-abnormal $ABS(r, \mathcal{G})$ instance sets are found and stored respectively in N and A. Then in line 2, the fresh predicate not_a the facts $not\_a(c)$ are added to E^+ for all $c \in ABS(r, \mathcal{G})$. In the same step the facts $not\_a(c)$ for $c \in N$ are stored in E^-. In line 3, a variant of a classical inductive learning procedure $Learn(E^+, E^-, \mathcal{G})$, *e.g.*, [Muggleton and Feng, 1990] is employed to induce a set of hypothesis $\mathcal{R}_e$ in the form of Horn rules with unary atoms, such that $\mathcal{G} \cup \mathcal{R}_e \models e$ for as many as possible $e \in E^+$, and $\mathcal{G} \cup \mathcal{R}_e \not\models e'$ for all $e' \in E^-$. Finally, in line 4 the bodies of rules in $\mathcal{R}_e$ not containing predicates from $Body^+(r)$ are put in EWS, which is output in line 5.

The correctness of *ComputeEWS* follows from the correctness of the procedure *Learn*. Indeed, by the procedure in line 4 for $p \in EWS$, a rule r' with p occurring in $Body(r')$ exists in $\mathcal{R}_e$. Since $r' \cup \mathcal{G} \not\models not\_a(c)$ for $not\_a(c) \in E^-$, we have that $p(c) \notin \mathcal{G}$ for r-normal c due to Lines 1 and 2. Moreover, $p(c') \in \mathcal{G}$ for some r-abnormal c', as otherwise $r' \notin \mathcal{R}_e$. Hence, (i) and (ii) of Definition 3.6 hold, *i.e.*, EWS is an exception witness set for r with respect to $\mathcal{G}$.

Constructing candidate rule revisions. After all EWSs are calculated for Horn rules in $\mathcal{R}_H$, we construct a search space of potential revisions by adding to rule bodies exceptions in the form of default negated atoms. More specifically, for every $r_i : a(X) \leftarrow b_1(X), \ldots, b_k(X)$ in $\mathcal{R}_H$ we create $m = |EWS(r_i, \mathcal{G})|$ revision candidates, *i.e.*, rules $r_i^{e_j}$, such that $Head(r_i^{e_j}) = Head(r_i)$, $Body^+(r_i^{e_j}) = Body(r_i)$, $Body^-(r_i) = e_j(X)$, where $e_j \in EWS(r_i, \mathcal{G})$. We denote with $\mathcal{R}_i$ the set of all $r_i^{e_j}$.

Example 3.5. For $EWS(r, \mathcal{G}') = \{immigrant, stateless\}$ from Example 3.4 in *Step 3*, we create the revision candidates

$$r^{im} : livesInUS(X) \leftarrow bornInUS(X), not \;\; immigrant(X) \text{ and}$$
$$r^{st} : livesInUS(X) \leftarrow bornInUS(X), not \;\; stateless(X)$$

□

3.5 Rules Quality Assessment

Given a potential R_{NM}, the function q should approximate the closeness between the completion $\mathcal{G}^a_{\mathcal{R}_{NM}}$ of the input KG $\mathcal{G}^a$ and the ideal KG $\mathcal{G}^i$. In this work, we follow usual practice in data mining and adapt standard association rule measures to our needs. Let

[2]We use the superscript *prop* to distinguish it from the *confidence* $conf(r, \mathcal{G})$ defined on relational data as in Chapter 2

Table 3.1: Evaluation measures for rules learned over **propositional KGs**. Note that $n(B)$ (and $n(H)$) denotes the **number of transactions** for which the body (and head) of the rule is satisfied.

Rule Measure	**Formula for** $r: H \leftarrow B$
Confidence (Prop.)[2]	$conf_{prop}(r, \mathcal{G}) = \frac{n(HB)}{n(B)}$
Lift	$lift(r, \mathcal{G}) = \frac{n(HB)}{n(H) * n(B)}$
Jaccard coef.	$jc(r, \mathcal{G}) = \frac{n(HB)}{n(H) + n(B) - n(HB)}$

rm be a generic rule measure, *e.g.*, one defined in Table 3.1. Then, naively generalizing rm for rulesets by taking the average of rm values for all rules in a given set we obtain

$$q_{rm}(\mathcal{R}_{NM}, \mathcal{G}) = \frac{\sum_{r \in \mathcal{R}_{NM}} rm(r, \mathcal{G})}{|\mathcal{R}_{NM}|} \tag{3.3}$$

In our case, q_{rm} alone is not sufficiently representative for being the target quality function q for two reasons: **(C1)** it does not penalize rules with noisy exceptions[3]; **(C2)** it does not measure how many contradicting beliefs our revisions reflect.

Example 3.6. Lets consider the following scenarios:

(1) For $r: livesInUS(X) \leftarrow hasUSPass(X), not\ \ poet(X)$ and $\mathcal{G}$, from Figure 3.2, we have $conf_{prop}(r, \mathcal{G})$=1, as all three non-poets with US passports live in the US, *i.e.*, r gets the highest individual score based on confidence. However, *poet* is a noisy exception due to p_3, who is a poet possessing a US passport and living in the US.

(2) Given the following nonmontonic ruleset:

$$\mathcal{R}_{NM} = \left\{ \begin{array}{l} r_1: livesInUS(X) \leftarrow hasUSPass(X), stateless(X) \\ r_2: livesInUS(X) \leftarrow bornInUS(X), not\ \ immigrant(X), \\ r_3: immigrant(X) \leftarrow stateless(X) \end{array} \right\}$$

While *immigrant* in r_2 may perfectly fit as exception with respect to some (unspecified here) original KG; once the KG is completed based on r_1 and r_3, *immigrant* might become noisy for r_2. Indeed, r_1 can easily bring new instances c in *livesInUS*, while r_3 can predict facts $immigrant(c)$. If this is the case, *i.e.*, $r_2 \in \mathcal{R}_{NM}$ becomes noisy after other rules in $\mathcal{R}_{NM}$ are applied, then intuitively rules in $\mathcal{R}_{NM}$ do not agree on the beliefs about $\mathcal{G}^i$ they express. □

[3] e is a *noisy* exception for r if $e(c) \in \mathcal{G}$ for some r-normal c.

To resolve the above issues, we introduce an additional quality function $q_{conflict}$, next to q_{rm}, whose purpose is to evaluate the ruleset with respect to (C1) and (C2). To measure $q_{conflict}$ for $\mathcal{R}_{NM}$, we create an extended set of rules $\mathcal{R}^{aux}$, containing every revised rule $r : a(X) \leftarrow b(X), not\ \ e(X)$ in $\mathcal{R}_{NM}$ and its auxiliary version $r_{aux} : not\_a(X) \leftarrow b(X), e(X)$, where not_a is a fresh predicate collecting instances that are not in a. Notice that r_{aux} is meaningless, and thus void in $\mathcal{R}^{aux}$, for rules r with positive bodies. Formally, we define $q_{conflict}$ as follows

$$q_{conflict}(\mathcal{R}_{NM}, \mathcal{G}) = \sum_{p \in pred(\mathcal{R}^{aux})} \frac{|\{c \mid p(c), not\_p(c) \in \mathcal{G}_{\mathcal{R}^{aux}}\}|}{|\{c \mid not\_p(c) \in \mathcal{G}_{\mathcal{R}^{aux}}\}|} \tag{3.4}$$

where $pred(\mathcal{R}^{aux})$ is the set of predicates appearing in $\mathcal{R}^{aux}$.

Intuitively, $\mathcal{G}_{\mathcal{R}^{aux}}$ contains both positive predictions of the form $p(c)$ and negative ones $not\_p(c)$ produced by the rules in $\mathcal{R}^{aux}$. The function $q_{conflict}$ computes the ratio of "contradicting" pairs $\{p(c), not\_p(c)\}$ over the number of $not\_p(c)$[4] in $\mathcal{G}_{\mathcal{R}^{aux}}$, which reflects how much the rules in $\mathcal{R}_{NM}$ disagree with each other on beliefs about the ideal KG $\mathcal{G}^i$ they express. The smaller $q_{conflict}$, the better is the ruleset $\mathcal{R}_{NM}$.

Revision based on partial materialization. Our goal in Step 4 is to find a set of revisions $\mathcal{R}_{NM}$, for which $q_{rm}(\mathcal{R}_{NM}, \mathcal{G})$ is maximal and $q_{conflict}(\mathcal{R}_{NM}, \mathcal{G})$ is minimal.

To determine such globally best set $\mathcal{R}_{NM}$ many candidate rule combinations have to be checked, which is unfortunately not feasible because of the large size of our $\mathcal{G}$ and EWS. Therefore, we propose an approach where we incrementally build $\mathcal{R}_{NM}$ by considering every $r_i \in \mathcal{R}_H$ and choose the best revision $r_i^j \in \mathcal{R}_i$ for it. In order to select the best r_i^j, we use a special ranking function, which estimates how well a rule r at hand describes the data and how noisy its exceptions are. In the remaining of this section, we will propose four different ranking functions, starting from the simplest to the most sophisticated one.

***Naive*-ranker.** The first implementation, which we call *rank_naive*, calculates the average value of the rm scores of r and r_{aux} and uses it to rank the rules. Formally, the average is computed by the following function:

$$est_{rm}(r, \mathcal{G}) = \frac{rm(\overbrace{H \leftarrow B, not\ E}^{r}, \mathcal{G}) + rm(\overbrace{not\_H \leftarrow B, E}^{r_{aux}}, \mathcal{G})}{2} \tag{3.5}$$

where rm is one of the measures in Table 3.1. For example, plugging in $conf_{prop}$

[4]Ratio over the number of $p(c)$ instead of $not\_p(c)$ is possible, but then $q_{conflict}$ is smaller and less representative.

instead of *rm*, gives

$$est_{conf\_prop}(r, \mathcal{G}) = \frac{1}{2}\Big(\frac{n(BH) - n(BHE)}{n(B) - n(BE)} + \frac{n(BE) - n(BHE)}{n(BE)}\Big) \tag{3.6}$$

where $n(X)$ is the number of transactions with items from X.

Example 3.7. For r and $\mathcal{G}$ from Example 3.6 (1), $est_{conf\_prop}(r, \mathcal{G}) = 0.75$, *i.e.*, due to noisiness of *poet* the value of est_{conf_prop} decreased. □

***PM*-ranker.** The main problem of *rank_naive* is that it does not exploit any knowledge about the properties that a final revision $\mathcal{R}_{NM}$ might have. In other words, ranking of revisions of a rule at hand is completely independent from ranking of revisions for other rules. To address this issue, we propose a second implementation called *revision based on partial materialization* (denoted as *rank_pm*). Here, the idea is to apply est_{rm} for a rule r not on $\mathcal{G}$ but on completion of $\mathcal{G}$ based on other rules, which according to our estimates constitute some approximation of $\mathcal{R}_{NM}$.

Example 3.8. Consider r_1:$livesInUS(X) \leftarrow bornInUS(X), not\ \ immigrant(X)$, and suppose there is only a single other rule r_2:$livesInUS(X) \leftarrow hasUSPass(X)$ given, for which $EWS(r_2, \mathcal{G}) = \emptyset$ for $\mathcal{G}$ from Figure 3.2. This knowledge can be exploited when ranking r_1. We have $est_{conf}(r_1, \mathcal{G}) = 0.8$, while $est_{conf\_prop}(r_1, \mathcal{G}_{r_2}) = 0.875$ due to the materialized fact $livesInUS(p11)$. This increase gives us an indication that r_1 agrees with r_2 on the predictions it makes.

On the contrary, for the rules

$$r_3 : livesInUS(X) \leftarrow hasUSPass(X), not\ \ poet(X) \text{ and}$$
$$r_4 : livesInUS(X) \leftarrow bornInUS(X)$$

we have $est_{conf\_prop}(r_3, \mathcal{G})$=0.75, but $est_{conf\_prop}(r_3, \mathcal{G}_{r_4})$=0.5, which witnesses that beliefs of r_3 and r_4 contradict. □

The function *rank_pm* first constructs the temporary rule set $\mathcal{R}^t$, which contains, for every rule $r_i \in \mathcal{R}_H$, a rule r_i^t with all exceptions from $EWS(r_i, \mathcal{G})$ incorporated, *i.e.*, $\mathcal{R}^t$ predicts the smallest number of facts, which are also predicted by any possible revision $\mathcal{R}_{NM}$. Then, for each $r_i \in \mathcal{R}_H$, we compute the est_{rm} value for all revision candidates r_i^j based on $\mathcal{G}_{\mathcal{R}^t \setminus r_i^t}$. Formally,

$$rank\_pm(r_i^j, \mathcal{G}) = est_{rm}(r_i^j, \mathcal{G}_{\mathcal{R}^t \setminus r_i^t}) \tag{3.7}$$

Once the scores for all revision candidates r_i^j for r_i are computed, we pick the revision with the highest score, add it to the current snapshot of $\mathcal{R}_{NM}$ and move to the next rule in the ruleset r_{i+1}.

***OPM*-ranker.** With *rank_pm*, facts inferred by rules of low quality might have a significant impact on more promising rules. To handle this issue, we propose a variation of *rank_pm* called *revision with ordered partial materialization* (abbr. *rank_opm*), which proceeds as follows. First we rank Horn rules based on some rm' (possibly same as rm) and obtain an ordered list $os_{\mathcal{R}_H}$. Then we go through $os_{\mathcal{R}_H}$ and for every rule r_i we compute a snapshot $\mathcal{G}_i$ of $\mathcal{G}$ by materializing only those rules $r_k^t \in \mathcal{R}^t$, for which r_k is ordered higher in the list $os_{\mathcal{R}_H}$ than r_i. More formally,

$$rank\_opm_{rm}(r_i, \mathcal{G}) = est_{rm}(r_i, \mathcal{G}_i) \tag{3.8}$$

where $\mathcal{G}_i = \mathcal{G}_{\mathcal{R}^t \setminus \{r_k^t \,|\, os_{\mathcal{R}_H}[k]=r_k; i \geq k\}}$.

***OWPM*-ranker.** With *rank_opm* as we have defined it, the facts inferred by rules count the same as the true facts in $\mathcal{G}$. Since the predicted facts are inferred based on statistically-supported assumptions, it is natural to distinguish them from the facts that are explicitly present in $\mathcal{G}$. To achieve this, we propose one last ranking function that exploits weights assigned to facts. Here, there is a clear distinction between facts from $\mathcal{G}$ (which get maximal weight) and the predicted facts (which inherit weights from rules that inferred them). We call this method *revision with ordered weighted partial materialization* (abbr. *rank_owpm*).

The method *rank_owpm* differs from *rank_opm* in that weights are used to estimate the revisions' scores. It is convenient (and a common practice) to assign weights of probabilistic nature between 0 and 1 (*e.g.*, *confidence* can be exploited). There are several ways to produce weighted partial materialization; for example, using probabilistic logic programming systems, such as Problog [Fierens et al., 2015] or PrASP [Nickles and Mileo, 2015].

However, normally, in such systems facts predicted by some rules in a ruleset at hand are used as input to other rules, *i.e.*, uncertainty is propagated through rule chains, which might be undesired in our setting. To avoid such propagation, when computing weighted partial materialization of $\mathcal{G}$ we keep predicted facts (*i.e.*, derived using rules) separately from the explicit facts (*i.e.*, those in $\mathcal{G}$), and infer new facts using only $\mathcal{G}$.

The method *rank_owpm* works as follows. Initially, we sort the rules in $\mathcal{R}_H$ and create the $\mathcal{G}_i$s with the same procedure as described for *rank_opm*. The only difference is that here every inferred fact in $\mathcal{G}_{\mathcal{R}^t}$ receives a specific weight that corresponds to $rm(r', \mathcal{G})$, where r' is the positive version of the rule that inferred the fact[5]. If the same fact is derived by multiple rules, we keep the highest weight.

The weights play a role when we evaluate a rule with respect to the partially materialized KG. To this end, we slightly change the *rm* function so that it considers weighted facts (we denote such function as rm^w). For example, $conf^w_{prop}(r, \mathcal{G})$ calculates a weighted sum of the instances for which the head (respectively body) of r is satisfied with respect to $\mathcal{G}$ (instead of a normal sum used in $conf_{prop}$). Formally, *rank_owpm* computes a score for a revision r_i^j as follows:

$$rank\_owpm_{rm^w}(r_i^j, \mathcal{G}) = est_{rm^w}(r_i^j, \mathcal{G}_i^w) \tag{3.9}$$

where $\mathcal{G}_i^w$ is the weighted version of $\mathcal{G}_i$ from Eq. 3.8. In the experiments, we analyze the performance of these four functions on some realistic KGs.

3.6 Exception-aware Rules for Relational Data

Up till this point, we studied mining exception-aware rules over propositinalized KGs *i.e.*, unary predicates only. In this section, we discuss the required modification to generalize the approach to relational datasets, *i.e.*, KGs including binary predicates.

3.6.1 Generalized Exception Witness Set

Predictive rules with binary predicates in the head have more than one variable, therefore, their grounding is not just a single constant, but rather a complex substitution of several variables. To this end, we begin with introducing the definition of *r-(Ab)Normal Substitutions* by extending the definition of *r-(Ab)Normal Instance Sets* as follows:

Definition 3.7 (r-(Ab)Normal Substitutions)**.** *Let $\mathcal{G}$ be a KG, r a Horn rule mined from $\mathcal{G}$, and let $\mathbf{V}$ be a set of variables occurring in r. Then*

- $NS(r, \mathcal{G}) = \{\theta \mid head(r)\theta, body(r)\theta \subseteq \mathcal{G}\}$ *is an* r-normal set of substitutions*;*
- $ABS(r, \mathcal{G}) = \{\theta' \mid body(r)\theta' \subseteq \mathcal{G}\ , head(r)\theta' \notin \mathcal{G}\}$ *is an* r-abnormal set of substitutions*,*

[5]We cannot consider the entire rule (*i.e.*, with all exceptions attached), since standard measures like *confidence* will return values very close to 1 for such rules.

where $\theta, \theta' : \mathbf{V} \rightarrow \mathcal{C}$.

Example 3.9. For $\mathcal{G}$ from Figure 3.1 and r_1, we have $NS(r_1, \mathcal{G}) = \{\theta_1, \theta_2, \theta_3\}$, where $\theta_1 = \{X/brad, Y/ann, Z/berlin\}$, $\theta_2 = \{X/john, Y/kate, Z/chicago\}$ and $\theta_3 = \{X/sui, Y/li, Z/beijing\}$ respectively. Besides, among substitutions in $ABS(r_1, \mathcal{G})$, we have $\theta_4 = \{X/mat, Y/lucy, Z/amsterdam\}$, yet there are others. □

Subsequently, we revise the definition of the *Exception Witness Set(EWS)* 3.6 to generalize to substitutions as follows:

Definition 3.8 (Exception Witness Set (EWS))**.** *Let* $\mathcal{G}$ *be a KG, let* r *be a rule mined from it, let* $\mathbf{V}$ *be a set of variables occurring in* r *and* $\vec{X} \subseteq \mathbf{V}$*. Exception witness set for* r *with respect to* $\mathcal{G}$ *and* $\vec{X}$ *is a maximal set of predicates* $EWS(r, \mathcal{G}, \tilde{X}) = \{e_1, \ldots, e_k\}$*, such that:*

- $e_i(\vec{X}\theta_j) \in \mathcal{G}$ *for some* $\theta_j \in ABS(r, \mathcal{G})$*,* $1 \leq i \leq k$ *and*
- $e_1(\vec{X}\theta'), \ldots, e_k(\vec{X}\theta') \notin \mathcal{G}$ *for all* $\theta' \in NS(r, \mathcal{G})$*.*

Example 3.10. For $\mathcal{G}$ in Figure 3.1 and rule r_1, we obtain $EWS(r, \mathcal{G}, Y) = \{researcher\}$ and $EWS(r, \mathcal{G}, X) = \{artist\}$. If *brad* with *ann* and *john* with *kate* did not live in metropolitan cities, then $EWS(r, \mathcal{G}, Z) = \{metropolitan\}$. □

In general, when binary atoms are allowed in the rules, there are potentially too many possible *EWS*s to construct. For a rule with n distinct variables, n^2 candidate *EWS*s might exist. Furthermore, many combinations of exception candidates could be an explanation for some missing links, so the search space is large. Therefore, we still restrict ourselves only to a single predicate as a final exception, and leave the extensions to arbitrary combinations for future research.

3.6.2 Updated Rules Quality Assessment

As discussed in chapter 2, there are several quality functions to assess the rules mined over relational data. In our extension, we utilized the *Conviction* measure [Brin et al., 1997] as our rule evaluation measure (rm). *Conviction* is accepted to be appropriate for estimating the actual implication of the rule at hand, and is thus particularly attractive for our KG completion task. For $r : \ H \leftarrow B, not \ E$, with $H = h(X, Y)$ and B, E involving variables from $\vec{Z} \supseteq X, Y$, the *conviction* is given by:

$$conv(r, \mathcal{G}) = \frac{1 - supp_{rel}(h(X, Y), \mathcal{G})}{1 - conf(r, \mathcal{G})} \tag{3.10}$$

where $supp_{rel}(h(X, Y), \mathcal{G})$ is the *relative support* of $h(X, Y)$ defined as follows:

$$supp_{rel}(h(X, Y), \mathcal{G}) = \frac{\#(X, Y) : h(X, Y) \in \mathcal{G}}{(\#X : \exists Y\ h(X, Y) \in \mathcal{G}) * (\#Y : \exists X\ h(X, Y) \in \mathcal{G})} \tag{3.11}$$

and *conf* is the standard confidence of r given as discussed in Section 2.3.

Example 3.11. The conviction of the rule r : $livesIn(Y, Z) \leftarrow isMarriedTo(X, Y)$, $livesIn(X, Z)$ based on KG in Figure 3.1 is $conv(r, \mathcal{G}) = \frac{1 - 0.3}{1 - 0.5} = 1.4$ □

3.6.3 Pipeline Realization

Due to the large number of exception candidates to consider, determining the globally best solution is not feasible in practice, especially given the substantial size of KGs. Therefore, we aim at finding an approximately good solution. Intuitively, our approach is to revise the rules one by one finding the locally best revision, while considering the predictive impact of other rules in a set. Our methodology for learning *exception-aware* rules over binary predicates proceeds as follows:

- **Step 1.** We start with a KG $\mathcal{G}$ and compute frequent conjunctive queries, which are then cast into Horn rules $\mathcal{R}_H$ based on some association rule measure *rm*. For that any state-of-the-art relational association rule learning algorithm can be used such as AIME [Galárraga et al., 2015]. We then compute for each rule $r \in \mathcal{R}_H$ the *r-normal* and *r-abnormal* substitutions.

- **Step 2 and 3.** Then, for every $r \in \mathcal{R}_H$ with $h(X, Y)$ in the head, we compose three sets $EWS(r, \mathcal{G}, X)$, $EWS(r, \mathcal{G}, Y)$ and $EWS(r, \mathcal{G}, \langle X, Y \rangle)$. The algorithm for computing *EWSs* is an extended version of the one reported in 3.4. Here, we first construct

  $$E^+ = \{not\_h(c, d),\ \text{s.t.}\ \theta = \{X/c, Y/d, \dots\}\ \text{is in}\ ABS(r, \mathcal{G})\}$$

  and

  $$E^- = \{not\_h(e, f),\ \text{s.t.}\ \theta' = \{X/e, Y/f, \dots\}\ \text{is in}\ NS(r, \mathcal{G})\}.$$

  A classical ILP procedure $learn(E^+, E^-, \mathcal{G})$ (*e.g.*, based on [Quinlan, 1990]) is then invoked, which searches for hypothesis with $not\_h(X, Y)$ in the head and a single body atom of the form $p(X)$, $p'(Y)$ or $p''(X, Y)$, where p, p', p'' are predicates in $\mathcal{G}$. The target hypothesis should not cover any examples in E^-, while covering at least

some examples in E^+. From the bodies of the obtained hypothesis, the predicates for EWS sets are extracted.

Then, for every $r \in \mathcal{R}_H$ we create potential revisions by adding to r a single negated atom from EWS set at a time. Overall, for each rule we obtain $|EWS(r, \mathcal{G}, X)| + |EWS(r, \mathcal{G}, Y)| + |EWS(r, \mathcal{G}, \langle X, Y \rangle)|$ candidate revisions.

- **Steps 4.** After all potential revisions are constructed, we rank them and determine the resulting set $\mathcal{R}_{NM}$ by selecting for every rule the revision that is ranked the highest. To find such globally best revised ruleset $\mathcal{R}_{NM}$, too many candidate combinations have to be checked, which is impractical due to the large size of both $\mathcal{G}$ and EWSs. Thus, instead we incrementally build $\mathcal{R}_{NM}$ by considering every $r_i \in \mathcal{R}_H$ and choosing the locally best revision r_i^j for it. For that, we exploit the **PM**-based ranking functions discussed in Section 3.5. Intuitively, the idea behind it is to rank candidate revisions not based on $\mathcal{G}$, but rather on its extension with predictions produced by other, selectively chosen, rules (grouped into a set $\mathcal{R}'$), thus ensuring a cross-talk between the rules.

3.7 Evaluation over Propositionalized KGs

This section reports the experiments assessing the effectiveness of the proposed *Exception-aware rule learning* algorithm, ExRuL over propositionalized KGs in Section 3.7. In Section 3.8, we discuss the quality of the rules learned over relational data, *i.e.*, normal KGs. Throughout the experiments, we evaluate different configurations of our method using the quality functions q_{rm} and $q_{conflict}$, defined in Sections 3.5 and 3.6. Furthermore, we evaluate the quality of the predictions produced by the generated rule sets. Ideally, the set difference between $\mathcal{G}_{\mathcal{R}_{NM}}$ and $\mathcal{G}^i$ (*i.e.*, the light red space in Figure 3.3) should be minimized, while the intersection between them should be maximized (*i.e.*, the light blue area in Figure 3.3). Finally, we report some example rules produced by ExRuL.

3.7.1 Experimental Setup

Datasets. We considered two knowledge graphs: a slice of more than 10M facts from YAGO3 [Mahdisoltani et al., 2015], a general purpose KG, and an RDF version of IMDB[6] data with 2M facts, a well known domain-specific KG of movies and artists. We chose these two KGs in order to evaluate our method's performance on both general-purpose and domain-specific KGs.

[6] http://imdb.com

Implementation and setup. We implemented ExRuL in Java[7] and realized the pipeline components as follows: (i) We first propositionalized the original KG, and then mined the Horn rules using the association rule mining implementation based on standard FPGrowth [Han et al., 2004] offered by SPMF Library [Fournier-Viger et al., 2016] for **Step 1**. In order to avoid over-fitting rules as well as to reduce the computation, we limited the extraction to rules with maximum four body atoms, a single head atom, a minimum support of $0.0001 \times \#$ *entities* and a minimum confidence of 0.25 for YAGO. Since IMDB is smaller and more connected, we set a higher minimum support of $0.005 \times \#$ *entities* and confidence of 0.6. On our machine, this process took approx. 10 seconds on YAGO and 2.5 second on IMDB, and it generated about 10K and 25K rules respectively.

(ii) In order to realize **Steps 2 and 3**, we implemented a simple inductive learning procedure, which performs manipulations on the set of facts instantiating the rule and its body to get the EWS. The generation of EWSs with minimum support of 0.05 took about 50 seconds for YAGO and 30 seconds for IMDB. The execution time is significantly affected by the size and distribution of the predicates in the KG. We could find EWSs for about 6K rules mined from YAGO, and 22K rules mined from IMDB. On average, the EWSs for the YAGO's rules contained 3 exceptions, and 28 exceptions on IMDB.

(iii) In **Step 4**, We implemented the four PM-based quality functions described in Section 3.5 and experimented with each one. Through the experiments, we use lift as rule quality measure and ordering criterion.

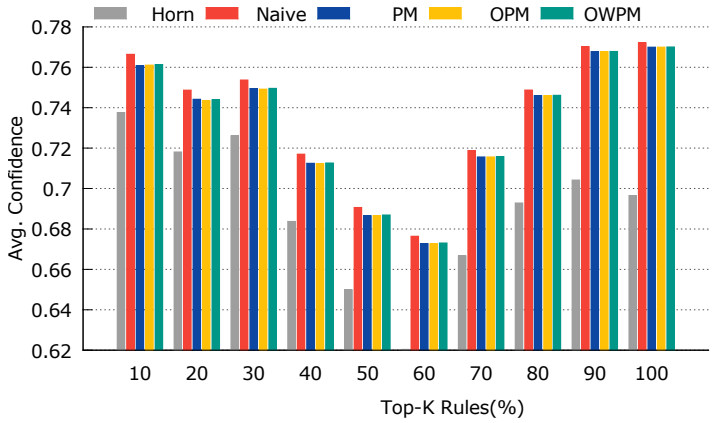
Metrics. We evaluated the quality of our rule selection procedure with respect to two dimensions, which reflect the two q proposed in Section 3.5: *average of the rules' confidence* (q_{conf}), and the *number of conflicts* ($q_{conflict}$). The average confidence shows how well the revised rules adhere to the input. The number of conflicts indicates how consistent the revised rules set is with respect to the final predictions it makes. We report the results using confidence as rule evaluation function (Eq. 3.6) and lift as rule ordering criterion, as we found this combination to be a good representative.

Machinery. Our experiments were performed on a machine with 40 cores and 400GB RAM.

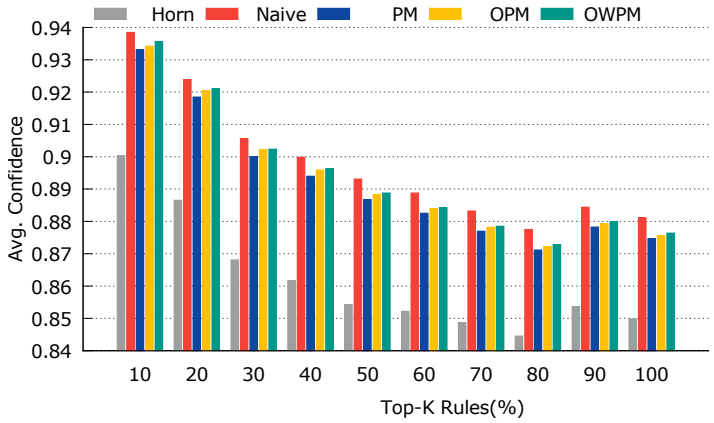
3.7.2 Results and Discussion

Ruleset quality. Figure 3.5 reports the obtained average rules' confidence using the four ranking functions to select the best revisions. *Horn* reports the average confidence

[7] https://github.molgen.mpg.de/gadelrab/Exception_Enriched_Rules



(a) Confidence for top-k YAGO revised rules

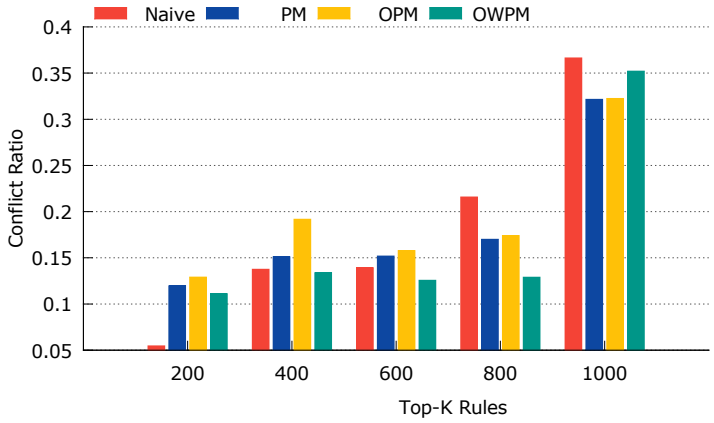


(b) Confidence for top-k IMDB revised rules

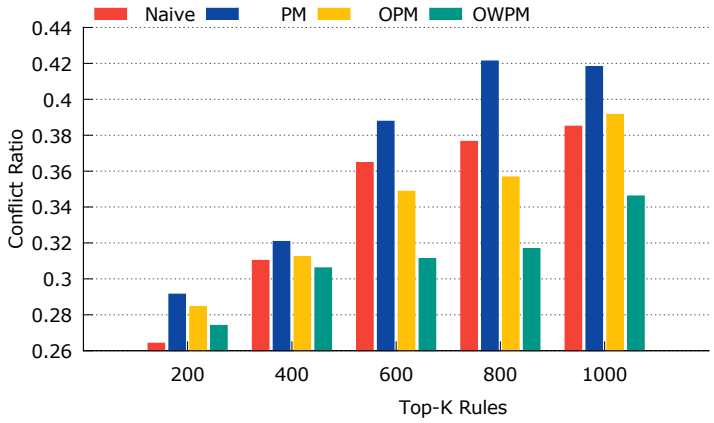
Figure 3.5: Average rules' confidence on YAGO and IMDB (higher is better)

of the original Horn rules; while *Naive*, *PM*, *OPM* and *OWPM* are our ranking methods described in Section 3.5. For both inputs, we show the results on the top $10, \ldots, 100\%$ rules ranked by lift.

We make the following three observations: (i) In general enriching Horn rules with exceptions increases the average confidence (approx. 11% for YAGO, 3.5% for IMDB). This indicates that our method is useful to mine rules that reflect the data more precisely. It is also worth mentioning that along with the increase in confidence, the average coverage of the revised rules dropped only by 13% for YAGO and 4% for IMDB (*i.e.*, the rules do not become too specific). (ii) The comparison between the four ranking methods shows that the highest confidence is achieved by the non-materialized (*Naive*) function followed by the weighted one (*OWPM*). (iii) Since we used lift for ordering the rules, and it is not neccessarily correlated with confidence, one can see that the confidence drops for around top 60% of the YAGO rules, and then slightly increases again. For



(a) Ratio of conflicts for YAGO rules



(b) Ratio of conflicts for IMDB rules

Figure 3.6: Ratio of conflicts on YAGO and IMDB (lower is better)

IMDB a smooth confidence decrease is observed with the addition of lower-ranked rules.

The higher value of *Naive-ranker* was expected, since this procedure is designed to maximize the confidence. However, confidence alone is not a sufficient indicator to determine the overall rule's quality, as we explained in Section 3.5. Figure 3.6 shows the number of conflicts (for YAGO and IMDB) that were obtained by executing the revised rules and their corresponding auxiliary versions (r_{aux}) using the DLV system [Leone et al., 2006].

Unfortunately, DLV was unable to scale to the entire ruleset; hence, we used up to 1000 rules. In our experiment, a conflict occurs when we derive both $p(c)$ and $not\_p(c)$. The graphs report the ratio between the number of conflicts and negated derived facts. From them, we observe that both *OPM* and *OWPM* produce less conflicts than the *Naive* function in most of the cases. By comparing the *OPM* and *OWPM* functions, we find that the weighted version is better, especially on the IMDB dataset when we can

$Y_1 : isMountain(X) \leftarrow isLocatedInAustria(X), isLocatedInItaly(X),$
$\quad \textbf{not}[\underline{isRiver(X)}|isLocatedInRussia(X)]$
$Y_2 : bornInUSA(X) \leftarrow actedInMovie(X), createdMovie(X), isPerson(X),$
$\quad \textbf{not}[\underline{wonFilmfareAwards(X)}|bornInNewYork(X)]$
$Y_3 : isPoliticianOfUSA(X) \leftarrow bornInUSA(X), isGovernor(X),$
$\quad \textbf{not}[\underline{isPoliticianOfPuertoRico(X)}|$
$\quad\quad isPoliticianOfHawaii(X)]$

$I_1 : hasLanguageEnglish(X) \leftarrow genreDrama(X), genreTriller(X), genreCrime(X),$
$\quad \textbf{not}[\underline{producedInIndia(X)}|createdByNovelist(X)]$
$I_2 : genreAnimation(X) \leftarrow directedByActor(X), inEnglish(X), producedInUSA(X),$
$\quad genreFamily(X),$
$\quad \textbf{not}[\underline{genreDrama(X)}|\ producedIn1984(X)]$

Figure 3.7: Example rules (Y=YAGO, I=IMDB) with good and bad exceptions

reduce the conflicts from 775 to 685 on a base of about 2000 negated facts.

Predictions quality. We executed the top-1000 revised rules using DLV and counted the number of derivations that our exceptions prevented. For YAGO with the original Horn rules, the reasoner inferred 924591 new triples. Our exception-aware ruleset decreased the number of inferred triples to 888215 (*Naive*), 892707 (*PM*), 892399 (*OPM*), and 891007 (*OWPM*). For IMDB we observed a smaller reduction. With the Horn rules the reasoner derived 38609 triples, while with the revised rules the inference set decreased to 36069 (*Naive*), 36355 (*PM*), 36021 (*OPM*), and 36028 (*OWPM*) triples.

Unfortunately, there is no automatic way available to assess whether the removed inference consists of genuine errors. Therefore, we selected the revised ruleset produced by the *OWPM* function and sampled 259 random facts from YAGO (we selected three facts for each binary predicate to avoid skewness). Then, we manually consulted online resources like Wikipedia to determine whether these triples were indeed incorrect. We found that 74.3% of these triples consisted of factual mistakes. This number provides a first empirical evidence that our method is indeed capable of detecting good exceptions and hence can improve the general quality of the Horn rules.

Example rules. We conclude by reporting some anecdotal examples of rules on YAGO and IMDB in Figure 3.7. Between the brackets we show examples of both good (underlined) and bad exceptions. In some cases, the rules have high quality exceptions such as rule $Y1$. In others, we found that the highest ranked exceptions mainly refer to disjoint classes of the head. The complete list of mined rules with the scores given to the determined exceptions is available in our repository.

3.8 Evaluation over Relational Data

3.8.1 Experimental Setup

Dataset. An automatic evaluation of the prediction quality requires an ideal graph $\mathcal{G}^i$ which is known to be complete as a ground truth. However, obtaining a real life complete KG is not possible. Therefore, we used the existing KG as an approximation of $\mathcal{G}^i$ ($\mathcal{G}^i_{appr}$), and constructed the available graph $\mathcal{G}^a$ by removing from $\mathcal{G}^i_{appr}$ 20% of the facts for each binary predicate. As an additional constraint, we ensure that every node in $\mathcal{G}^a$ is connected to at least one other node. We constructed two datasets for evaluating our approach: (i) YAGO3 [Mahdisoltani et al., 2015], as a general purpose KG, with more than 1.8M entities, 38 relations, and 20.7M facts, and (ii) a domain-specific KG extracted from the IMDB dataset with 112K entities, 38 relations, and 583K facts.

Implementation and setup. We implemented a prototype for the extension of ExRuL over relational data[8]. We start with mining Horn rules of the form $h(X, Z) \leftarrow p(X, Y), q(Y, Z)$ from $\mathcal{G}^a$ and ranking them with respect to their *absolute support*. Then, we revise the rules as described in Section 3.6.3, taking *conviction*, described in Section 3.6.2, as the *rm* measure. For every rule we rank the constructed revisions and pick the one with the highest score as the final result. This process is repeated for the proposed ranking methods, *i.e.*, *Naive*, *Partial Materialization*, and *Ordered Partial Materialization* resulting in the rulesets $\mathcal{R}_N$, $\mathcal{R}_{PM}$, and $\mathcal{R}_{OPM}$ respectively.

Metric. As an intrinsic measure, we report the average *Conviction* measure for the respective ruleset. We also report the number of the predicted facts for each revision approach. In order to analyze these predictions better, we report the number of facts which are not in the reference KG $\mathcal{G}^i_{appr}$. Finally, we sample the removed predictions and manually annotate them to compute the ratio of correctly removed predictions by the respective revisions.

3.8.2 Results and Discussion

Ruleset quality. In Table 3.2, we report the *average conviction* for the top-k (k=5,...100) Horn rules $\mathcal{R}_H$ and their revisions for YAGO and IMDB. The results show that the revision process consistently enhances the average ruleset conviction. Moreover, while the conviction per ruleset naturally decreases with addition of lower quality rules, improvement ratios are increasing with the best enhancement (7.6%) for the top-100 rules learned over IMDB.

[8] https://github.com/htran010589/nonmonotonic-rule-mining

Table 3.2: The average conviction for the *top-k* Horn rules and their revisions

top-k	**YAGO**				**IMDB**			
	$\mathcal{R}_H$	$\mathcal{R}_N$	$\mathcal{R}_{PM}$	$\mathcal{R}_{OPM}$	$\mathcal{R}_H$	$\mathcal{R}_N$	$\mathcal{R}_{PM}$	$\mathcal{R}_{OPM}$
5	1.3784	1.3821	1.3821	1.3821	2.2670	2.3014	2.3008	2.3014
30	1.1207	1.1253	1.1236	1.1237	1.5453	1.5644	1.5543	1.5640
50	1.0884	1.0923	1.0909	1.0913	1.3571	1.3749	1.3666	1.3746
60	1.0797	1.0837	1.0823	1.0829	1.3063	1.3221	1.3143	1.3219
70	1.0714	1.0755	1.0736	1.0744	1.2675	1.2817	1.2746	1.2814
80	1.0685	1.0731	1.0710	1.0720	1.2368	1.2499	1.2431	1.2497
100	1.0618	1.0668	1.0648	1.0659	1.3074	1.4100	1.3987	1.4098

Table 3.3: Predictions of sampled rules and their revisions (I=IMDB, Y=YAGO)

predicate	**predictions**				**outside $\mathcal{G}^i_{appr}$**				**corr. removed,%**		
	$\mathcal{R}_H$	$\mathcal{R}_N$	$\mathcal{R}_{PM}$	$\mathcal{R}_{OPM}$	$\mathcal{R}_H$	$\mathcal{R}_N$	$\mathcal{R}_{PM}$	$\mathcal{R}_{OPM}$	$\mathcal{R}_N$	$\mathcal{R}_{PM}$	$\mathcal{R}_{OPM}$
I:*actedIn*	1231	1214	1230	1214	1148	1131	1147	1131	90	100	90
I:*genre*	629	609	618	609	493	477	482	477	50	20	50
I:*hasLang*	173	102	125	102	163	92	115	92	60	100	60
I:*prodIn*	2489	2256	2327	2327	2488	2255	2326	2326	10	10	30
									52.50	45.16	**57.75**
Y:*direct*	41079	39174	39174	39174	41021	39116	39116	39116	100	100	100
Y:*grFrom*	3519	3456	3456	3456	3363	3300	3300	3300	100	100	70
Y:*citizOf*	3407	2883	2883	2883	3360	2836	2836	2836	50	50	70
Y:*bornIn*	110283	108317	109846	108317	109572	107607	109137	107607	90	90	100
									85	85	85

Prediction quality. To evaluate the quality of ruleset predictions, we sampled a set of 5 Horn rules $\mathcal{R}_H$ from the top-50 Horn rules both for IMDB and YAGO and compared them against their revisions with respect to the predictive power. For that, we run DLV [Leone et al., 2006] with these rulesets and the facts in $\mathcal{G}^a$ and obtained respectively $\mathcal{G}_{\mathcal{R}_H}$, $\mathcal{G}_{\mathcal{R}_N}$, $\mathcal{G}_{\mathcal{R}_{PM}}$ and $\mathcal{G}_{\mathcal{R}_{OPM}}$. Table 3.3 reports for each head predicate appearing in the sampled rules the number of newly predicted facts, *i.e.*, those not in $\mathcal{G}^a$ (second column) and the portion of predictions among them that are outside $\mathcal{G}^i_{appr}$ (third column).

First, observe that naturally relatively few predictions can be found in $\mathcal{G}^i_{appr}$ (≈9% for IMDB and ≈2% for YAGO). This is expected as the latter graph is highly incomplete. Second, it is important to note that $\mathcal{R}_H$ and the revised rulesets produced roughly the same number of correct predictions within $\mathcal{G}^i_{appr}$. For example, for YAGO we have $\mathcal{G}_{\mathcal{R}_H} \setminus \mathcal{G}_{\mathcal{R}_{PM}} \cap \mathcal{G}^i_{appr} = \emptyset$, *i.e.*, the green area within the approximation of the ideal graph in Figure 3.3 is empty, which shows that incorporated exceptions did not spoil the positive rules with respect to correct predictions in $\mathcal{G}^i_{appr}$.

$$
\begin{array}{lcl}
r_1 : \mathit{writtenBy}(X, Z) & \leftarrow & \mathit{hasPredecessor}(X, Y), \mathit{writtenBy}(Y, Z), \\
 & & \quad \textbf{not } \mathit{american\_film}(X) \\
r_2 : \mathit{actedIn}(X, Z) & \leftarrow & \mathit{isMarriedTo}(X, Y), \mathit{directed}(Y, Z), \\
 & & \quad \textbf{not } \mathit{silent\_film\_actor}(X) \\
r_3 : \mathit{isPoliticianOf}(X, Z) & \leftarrow & \mathit{hasChild}(X, Y), \mathit{isPoliticianOf}(Y, Z), \\
 & & \quad \textbf{not } \mathit{vicepresidentOfMexico}(X)
\end{array}
$$

Figure 3.8: Examples of the revised rules with binary relations

To make the comparison between $\mathcal{R}_H$ and the revised rulesets fair, we need to ensure that $\mathcal{R}_H$ on its own is not completely inaccurate. Indeed, if $\mathcal{R}_H$ makes only false predictions, then adding even irrelevant exceptions will reduce the number of incorrect instances, thus, improving the ruleset predictive quality. The number of $\mathcal{R}_H$ predictions outside $\mathcal{G}^i_{appr}$ is large, and we do not know the ground truth for these predictions. Therefore, we had to verify these facts manually using web resources.

Conducting such manual verification for all of the predictions is not feasible. Hence, we restricted ourselves to a uniform random sample of 20 predicted facts per head predicate in $\mathcal{R}_H$. Among the IMDB samples, the precision of 70% has been achieved, while for YAGO we have obtained precision of 30%. This shows that the rules in $\mathcal{R}_H$ are not completely erroneous.

To assess the impact of the revision methods, we also had to select a uniform sample due to the large size of the differences between $\mathcal{G}_{\mathcal{R}_H}$ and the graphs obtained by applying revised rulesets. More specifically, we have randomly sampled 10 predictions per head predicate from $\mathcal{G}_{\mathcal{R}_H} \backslash \mathcal{G}_{\mathcal{R}_N}$, $\mathcal{G}_{\mathcal{R}_H} \backslash \mathcal{G}_{\mathcal{R}_{PM}}$ and $\mathcal{G}_{\mathcal{R}_H} \backslash \mathcal{G}_{\mathcal{R}_{OPM}}$ respectively. The 4th column in Table 3.3 reports the percentage of erroneous predictions among the sampled facts in the difference for each revision method (referred to as correctly removed), *i.e.*, gray area in Figure 3.3. For IMDB $\mathcal{R}_{OPM}$ achieved the best improvement. For YAGO, all of the revision methods performed equally well. Moreover, the effect of YAGO revisions is more visible, since $\mathcal{R}_H$ for YAGO is of a lower quality than for IMDB as reported earlier.

Running times. Our main goal was to evaluate the predictive quality of computed rules rather then the running times of the implemented algorithms. Hence, the latter are only briefly reported to indicate the feasibility of our approach. For the *top-100* Horn YAGO and IMDB rules mined from $\mathcal{G}^a$, *EWS*s with an average of 1.6K and 10.9K exception candidates per rule were computed within 7 and 68 seconds respectively. As regards IMDB, the revisions $\mathcal{R}_N, \mathcal{R}_{PM}$, and $\mathcal{R}_{OPM}$ were determined in 9, 62, and 24 seconds respectively, while for YAGO, they required 45, 177, and 112 seconds. Besides, the predictions of each of the rulesets on $\mathcal{G}^a$ were found via DLV, on average, within 8 seconds for IMDB and 310 seconds for YAGO.

Example rules. Figure 3.8 shows examples of our revised rules, *e.g.*, r_1 extracted from IMDB states that movie plot writers stay the same throughout the sequel unless a movie is American, and r_3 learned from YAGO says that ancestors of politicians are also politicians in the same country with the exception of Mexican vice-presidents.

3.9 Related Work

In the association rule mining community, some works concentrated on finding (interesting) exception rules (*e.g.*, [Taniar et al., 2008]), which are defined as rules with low support (rare) and high confidence. Our work differs from this line of research because we do not necessarily look for rare interesting rules, but care about the quality of their predictions.

Another relevant stream of research is concerned with learning Description Logic TBoxes or schema (*e.g.*, [Lehmann et al., 2011]). However, these techniques focus on learning concept definitions rather than nonmonotonic rules.

In the context of inductive and abductive logic, learning nonmonotonic rules from complete datasets [Flach and Kakas, 2000] was studied in several works ([Sakama, 2005, Ray, 2009, Corapi et al., 2010, Katzouris et al., 2015, Law et al., 2020]. These methods rely on CWA and focus on describing a dataset at hand exploiting negative example, which are explicitly given unlike in our setting.

Learning Horn rules in presence of incompleteness was studied in hybrid settings in the work of [Józefowska et al., 2010] and [Lisi, 2010]. There a background theory or a hypothesis can be represented as a combination of a DL ontology and Horn rules. While the focus of this work is on the complex interaction between reasoning components and the learned rules are positive, we are concerned with techniques for deriving nonmonotonic rules with high predictive quality from huge KGs.

3.10 Conclusions and Future Work

We have presented, ExRuL, a method for revising Horn rules into nonmonotonic rules by adding negated atoms into their bodies with the goal of improving the quality of a rule set for data prediction. To select the best revision from potential candidates we devised rule-set ranking measures, based on data mining measures and the novel concept of partial materialization. We evaluated our method with various configurations on both general-purpose and domain-specific KGs and observed significant improvements over a baseline Horn rule mining.

There are various directions for future work. First, we investigate extracting evidence for (or against) exceptions from text and web corpora. Second, our framework can be enhanced by partial completeness assumptions for certain predicates (*e.g.*, all countries are available in KG) or constants (*e.g.*, knowledge about Barack Obama is complete). Finally, a promising future direction is to investigate more complex rules such as rules with aggregates or disjunctions in the head.

Chapter 4

RuLES: Rule Learning Over Knowledge Graph Embedding

Rules learned over knowledge graphs (KGs) capture interpretable patterns in the data and various methods for rule learning have been proposed, as shown in Chapter 2. More importantly, since KGs are inherently incomplete, rules can be used to deduce missing facts. Statistical measures for assessing the learned rules, such as confidence, reflect rule quality well when the KG is relatively complete; however, these measures might be misleading otherwise. Therefore, it is difficult to learn high-quality rules from the KG alone, and scalability dictates that only a small set of candidate rules could be generated. Therefore, the ranking and pruning of candidate rules are major problems.

To address this issue, we propose a rule learning method that utilizes probabilistic representations of missing facts. In particular, we iteratively extend rules induced from a KG by relying on feedback from a precomputed embedding model over the KG and external information sources, including text corpora. Experiments on real-world KGs demonstrate our approach's effectiveness, both with respect to the quality of the learned rules and the facts that they predict.

4.1 Introduction

Motivation. Rules are widely used to represent relationships and dependencies between data items in datasets and to capture the underlying patterns in data [Agrawal et al., 1993, Piatetsky-Shapiro, 1991]. Applications of rules include health-care [Wojtusiak, 2014], equipment diagnostics [Kharlamov et al., 2017, Mehdi et al., 2017], telecommunications [Mannila et al., 1995], and commerce [Ras and Wieczorkowska, 2000].To facilitate rule construction, a variety of rule learning methods have been developed, in particular over knowledge graphs, as discussed in Chapter 2. Moreover, various sta-

tistical measures such as confidence, actionability, and unexpectedness to evaluate the quality of the learned rules have been proposed.

When rules are automatically learned over KGs, statistical measures like standard support and confidence (see 2) are used to assess the quality of these rules. Most notably, the confidence of a rule is the fraction of facts predicted by the rule that are indeed true in the KG. However, this is a meaningful measure for rule quality only when the KG is reasonably complete. For rules learned from largely incomplete KGs, confidence and other measures may be misleading, as they do not reflect the patterns in the missing facts. For example, a KG that knows only (or mostly) male CEOs would yield a heavily biased rule $gender(X, male) \leftarrow isCEO(X, Y), isCompany(Y)$, which does not extend to the entirety of valid facts beyond the KG. Therefore, it is crucial that rules can be ranked by meaningful quality measures, which accounts for KG incompleteness.

Example 4.1. Consider a KG about people's jobs, residence and spouses as well as office locations and headquarters of companies. Suppose a rule learning method has computed the following two rules:

$$r_1 : livesIn(X, Y) \leftarrow worksFor(X, Z), hasOfficeIn(Z, Y)$$
$$r_2 : livesIn(Y, Z) \leftarrow marriedTo(X, Y), livesIn(X, Z)$$

□

Consider rules in Example 4.1, The rule r_1 is quite noisy, as companies have offices in many cities, but employees live and work in only one of them, while the rule r_2 clearly is of higher quality. However, depending on how the KG is populated with instances, the rule r_1 could nevertheless score higher than r_2 in terms of confidence measures. For example, the KG may contain only a specific subset of company offices and only people who work for specific companies. If we knew the complete KG, then the rule r_2 should presumably be ranked higher than r_1.

Suppose we had a perfect oracle for the true and complete KG. Then we could learn even more sophisticated rules such as:

$$r_3 : livesIn(X, Y) \leftarrow worksFor(X, Z), hasHeadquarterIn(Z, Y), not\ locatedIn(Y, USA).$$

This rule would capture that most people work in the same city as their employers' headquarters, with the USA being an exception (assuming that people there are used to long commutes). This is an example of a rule that contains a negated atom in the rule body (so it is no longer a Horn rule) and has a partially grounded atom with a variable and a constant as its arguments.

Problem. The problem of KG incompleteness has been tackled by methods that (learn to) predict missing facts for KGs. As discussed in Chapter 2, prominent class of approaches is statistics-based and includes tensor factorization, *e.g.*, [Nickel et al., 2011] and neural-embedding-based models, *e.g.*, [Bordes et al., 2013, Nickel et al., 2016b]. Intuitively, these approaches turn a KG, possibly augmented with external sources such as text [Xiao et al., 2017] or log files [Ringsquandl et al., 2018], into a probabilistic representation of its entities and relations, known as *embeddings*, and then predict the likelihood of missing facts by reasoning over the embeddings (see, *e.g.*, [Wang et al., 2017] for a survey).

Learned embeddings can complement the given KG and are a potential asset in overcoming the limitations that arise from incomplete KGs. Consider the following gedanken-experiment: we compute embeddings from the KG and external text sources, that can then be used to predict the complete KG that comprises all valid facts. This would seemingly be the perfect starting point for learning rules, without the bias and quality problems of the incomplete KG. However, this scenario is way oversimplified. The embedding-based fact predictions would themselves be very noisy, yielding also many spurious facts. Moreover, the computation of all fact predictions and the induction of all possible rules would come with a big scalability challenge; in practice, we need to restrict ourselves to computing merely small subsets of likely fact predictions and promising rule candidates.

Approach. In this work we propose a novel approach for rule learning guided by external sources that allows to learn high-quality rules from incomplete KGs. In particular, our method extends rule learning by exploiting probabilistic representations of missing facts computed by embedding models of KGs and possibly other external information sources. We iteratively construct rules over a KG and collect feedback from a precomputed embedding model, through specific queries issued to the model for assessing the quality of (partially constructed) rule candidates. This way, the rule induction loop is interleaved with the guidance from the embeddings, and we avoid scalability problems. Our machinery is also more expressive than many prior works on rule learning from KGs, by allowing non-monotonic rules with negated atoms as well as partially grounded atoms. Within this framework, we devise confidence measures that capture rule quality better than previous techniques and thus improve the ranking of rules.

While enhancing embeddings with precomputed rules or constraints has been studied in several works [Wang et al., 2015, Guo et al., 2018, Rastogi et al., 2017, Guo et al., 2016, Wang et al., 2015], accounting for embeddings in rule construction as we propose, has not been considered before to the best of our knowledge.

Contribution. The salient contributions of this chapter are as follows:

- We propose a rule learning approach guided by external sources, and show how to learn high-quality rules by utilizing feedback from embedding models.
- We implement our approach RuLES, and present extensive experiments on real-world KGs, demonstrating the effectiveness of our approach with respect to both the quality of the learned rules and the fact predictions that they produce.

4.2 Rule Learning Guided by External Sources

In this section, we introduce our framework for rule learning guided by external sources, discuss challenges associated with it, and finally propose a concrete instantiation of our framework with embedding models.

4.2.1 Problem Statement and General Solution

For a KG $\mathcal{G}$ with the signature $\Sigma_{\mathcal{G}} = (\mathcal{R}, \mathcal{C})$, a *probabilistic KG* $\mathcal{P}$ is a pair $\mathcal{P} = (\mathcal{G}, f)$ where $f : \mathcal{R} \times \mathcal{C} \times \mathcal{C} \to [0, 1]$ is a probability function over the facts over $\Sigma_{\mathcal{G}}$. We assume $f(a) = 1$ for each fact $a \in \mathcal{G}$, which is already known to be true.

The goal of our work is to learn rules that do not only describe the available graph $\mathcal{G}$ well, but also predict highly probable facts based on the function f. The key questions now are how to define the quality of a given rule r based on $\mathcal{P}$ and how to exploit this quality during rule learning for pruning out unpromising rules.

A quality measure μ for rules over probabilistic KGs is a function $\mu : (r, \mathcal{P}) \mapsto \alpha$, where $\alpha \in [0, 1]$. In order to measure the quality μ of r over $\mathcal{P}$ we propose:

- to measure the quality μ_1 of r over $\mathcal{G}$, where $\mu_1 : (r, \mathcal{G}) \mapsto \alpha \in [0, 1]$,
- to measure the quality μ_2 of $\mathcal{G}_r$ by relying on $\mathcal{P}_r = (\mathcal{G}_r, f)$, where $\mu_2 \colon (\mathcal{G}', (\mathcal{G}, f)) \mapsto \alpha \in [0, 1]$ for $\mathcal{G}' \supseteq \mathcal{G}$ is the quality of extension $\mathcal{G}'$ of $\mathcal{G}$ over $\Sigma_{\mathcal{G}}$ given f, and
- to combine the result as the weighted sum.

Formally, we define our hybrid rule quality function $\mu(r, \mathcal{P})$ as follows:

$$\mu(r, \mathcal{P}) = (1 - \lambda) \times \mu_1(r, \mathcal{G}) + \lambda \times \mu_2(\mathcal{G}_r, \mathcal{P}) \tag{4.1}$$

In this formula, μ_1 can be any classical quality measure of rules over the given KG $\mathcal{G}$. Intuitively, $\mu_2(\mathcal{G}_r, \mathcal{P})$ is the quality of $\mathcal{G}_r$ with respect to f that allows us to capture the information about facts missing in $\mathcal{G}$ that are relevant for r. The weighting factor λ,

we call it *embedding weight*, allows one to choose whether to rely more on the classical measure μ_1 or on the measure μ_2 of the quality of the extension $\mathcal{G}_r$ of r over $\mathcal{G}$.

Challenges. There are several challenges that one faces when realizing our approach. First, given an incomplete $\mathcal{G}$, one has to define f such that $(\mathcal{G}, f)$ satisfies the expectations, *i.e.*, reflects well the probabilities of missing facts. Second, one has to define both μ_1 and μ_2 to satisfy the expectations and admit efficient implementation. Finally, the adaptation of existing rule learning approaches to account for the probabilistic function f without the loss of scalability is not trivial. Indeed, materializing f by augmenting $\mathcal{G}$ with all possible probabilistic facts over $\Sigma_\mathcal{G}$ and subsequently applying standard rule learning methods on the obtained graph is not practical. Storing such potentially enormous augmented graph where many probabilistic facts are irrelevant for the extraction of meaningful rules might be simply infeasible.

4.2.2 Solution Realization

In this section, we present concrete realizations of f, μ_1 and μ_2 to address the above stated challenges. In Section 4.3, we discuss how we implemented and adapted them within an end-to-end rule learning system.

Realization of the probabilistic function f. We propose to define f by relying on embeddings of KGs. Embeddings are low-dimensional vector spaces that represent nodes and edges of KGs and can be used to estimate the likelihood (not necessary probability) of potentially missing binary atoms using a scoring function $\xi : \mathcal{R}_\mathcal{G} \times \mathcal{C}_\mathcal{G} \times \mathcal{C}_\mathcal{G} \to \mathbb{R}$. Examples of concrete scoring functions can be found, *e.g.*, in [Wang et al., 2017]. Note that our framework is not dependent on a concrete embedding model. What is important for us is that embeddings can be used to construct probabilistic representations [Nickel et al., 2016b] of the missing atoms in KGs, and we use this to define f.

Consider the following auxiliary definition: Given a KG $\mathcal{G}$, and an atom $a = p(s, o)$, the set $\mathcal{G}^s$ consists of a and all atoms a' that are obtained from a by replacing s with a constant from $\Sigma_\mathcal{G}$, except for those that are already in $\mathcal{G}$. Then, given a scoring function ξ, $[\mathcal{G}^s]$ is a list of atoms from $\mathcal{G}^s$ ordered in the descending order. Finally, the *subject rank* [Glorot et al., 2013] of a given ξ, $subject\_rank_\xi(a)$ is the position of a in $[\mathcal{G}^s]$. Analogously,one can define $[\mathcal{G}^o]$ and the corresponding *object rank* [Glorot et al., 2013] of a given ξ, that is, $object\_rank_\xi(a)$.

Now we are ready to define the function f for an atom $a \notin \mathcal{G}$ as the average of its

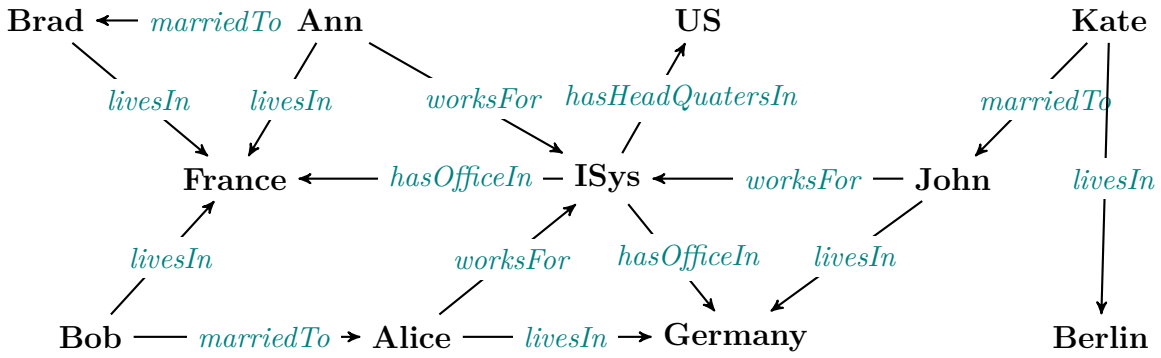


Figure 4.1: An example knowledge graph

subject and object inverted ranks given ξ [Glorot et al., 2013], such as:

$$f_\xi(a) = 0.5 \times (1/subject\_rank_\xi(a) + 1/object\_rank_\xi(a))$$

Note that we assume $f_\xi(a) = 1$ for $a \in \mathcal{G}$.

Realization of μ_1. This measure should reflect the descriptive quality of a given rule r with respect to $\mathcal{G}$. There are many classical data mining measures that can be used as μ_1, see, *e.g.*, [Tran et al., 2018, Galárraga et al., 2015, Tanon et al., 2017, Zupanc and Davis, 2018] for μ_1s proposed specifically for KGs.

In this chapter, we selected the following two measures for μ_1: *confidence* and *PCA confidence* [Galárraga et al., 2015], where PCA stands for the partial completeness assumption, that can be defined using rule support, *r-supp*, body support, *b-supp*, and partial body support, *pb-supp*.

Let $r:\ head \leftarrow body^+, not\ body^-$ be a rule, x be the subject variable of the *head*, and let h denote a *head*'s variable assignment that we with a slight abuse of notation use as a homomorphism on (sets of) atoms. Then,

$$\begin{aligned}
r\text{-}supp(r,\mathcal{G}) &= |\{h \mid h(head) \in \mathcal{G}, \exists h' \supseteq h \text{ s.t. } h'(body^+) \in \mathcal{G}, h'(body^-) \notin \mathcal{G}\}|,\\
b\text{-}supp(r,\mathcal{G}) &= |\{h \mid \exists h' \supseteq h \text{ s.t. } h'(body^+) \in \mathcal{G}, h'(body^-) \notin \mathcal{G}\}|,\\
pb\text{-}supp(r,\mathcal{G}) &= |\{h \mid \exists h' \supseteq h \text{ s.t. } h'(body^+) \in \mathcal{G}, h'(body^-) \notin \mathcal{G}, \text{and}\\
&\qquad \exists h'' \text{ s.t. } h(x) = h''(x), h''(head) \in \mathcal{G}\}|.
\end{aligned}$$

Finally, we are ready to define μ_1 as confidence or PCA confidence:

$$\begin{aligned}
\mu_1 &= conf(r,\mathcal{G}) = r\text{-}supp(r,\mathcal{G})/b\text{-}supp(r,\mathcal{G}),\\
\mu_{1,pca} &= conf_{pca}(r,\mathcal{G}) = r\text{-}supp(r,\mathcal{G})/pb\text{-}supp(r,\mathcal{G}).
\end{aligned}$$

Intuitively, confidence of a rule is the conditional probability of rule's head given its body, while PCA confidence is its generalization to the open world assumption (OWA), which does not penalize rules that predict facts $p(s, o)$, such that $p(s, o') \notin \mathcal{G}$ for any o'.

Example 4.2. Consider the KG $\mathcal{G}$ in Figure 4.1 and recall the rules r_1 and r_2 from Example 4.1. For r_1, we have $conf(r_1, \mathcal{G}) = conf_{pca}(r_1, \mathcal{G}) = \frac{3}{6}$, while for r_2 it holds that $conf(r_2, \mathcal{G}) = conf_{pca}(r_2, \mathcal{G}) = \frac{1}{3}$. In the case that *Alice* was not known to live in Germany, then the PCA confidence for r_2 is $conf_{pca}(r_2, \mathcal{G} \setminus \{livesIn(Alice, Germany)\}) = \frac{1}{2}$. Finally, for the following rule with negation:

$$r_4 : livesIn(Y, Z) \leftarrow marriedTo(X, Y), livesIn(X, Z), not\ researcher(X)$$

stating that married people live together unless one is a researcher, and the augmented graph $\mathcal{G}' = \mathcal{G} \cup \{researcher(bob)\}$, we have $conf(r_4, \mathcal{G}') = conf_{pca}(r_4, \mathcal{G}') = \frac{1}{2}$. □

Realization of μ_2. There are various ways to define the quality $\mu_2(\mathcal{G}_r, \mathcal{P})$ of $\mathcal{G}_r$. A natural candidate to define the quality of $\mathcal{G}_r$ is the probability of $\mathcal{G}_r$, that is, as $\mu_2(\mathcal{G}_r, \mathcal{P}) = \prod_{a \in \mathcal{G}_r} f(a) \times \prod_{a \in (\mathcal{R}_\mathcal{G} \times \mathcal{C}_\mathcal{G} \times \mathcal{C}_\mathcal{G}) \setminus \mathcal{G}_r} (1 - f(a))$. A disadvantage of such quality measure is that in practice it will be very low, as the product of many (potentially) small probabilities, and thus Equation 4.1 will be heavily dominated by $\mu_1(r, \mathcal{G})$. Therefore, we advocate to define $\mu_2(\mathcal{G}_r, \mathcal{P})$ as the average probability of predicted facts in $\mathcal{G}_r$:

$$\mu_2(\mathcal{G}_r, \mathcal{P}) = (\Sigma_{a \in \mathcal{G}_r \setminus \mathcal{G}} f(a))/|\mathcal{G}_r \setminus \mathcal{G}|.$$

Example 4.3. Consider the KG $\mathcal{G}$ in Figure 4.1, and the rules from Example (4.1) with their confidence values as presented in Example 4.2. Suppose that a text-enhanced embedding model produce a relatively accurate estimation of the probabilities of facts over *livesIn* relation. For instance, even though there is no direct connection between Germany and Berlin within the graph, relying on the living places of entities similar to John and hidden semantic relations between Germany and Berlin such as co-occurrences in text and other linguistic features, for the fact $a = livesIn(john, berlin)$ we obtained $f(a) = 0.9$, while for $a' = livesIn(john, france)$, a much lower probability $f(a') = 0.09$. These naturally support the predictions of r_2 but not those of r_1.

Generalizing this idea, assume that on the whole dataset we get $\mu_2(\mathcal{G}_{r_1}, \mathcal{P}) = 0.1$ and $\mu_2(\mathcal{G}_{r_2}, \mathcal{P}) = 0.8$, where $\mathcal{P} = (\mathcal{G}, f)$. Thus, for $\lambda = 0.5$ we have $\mu(r_1, \mathcal{P}) = (1 - 0.5) \times 0.5 + 0.5 \times 0.1 = 0.3$, while for $\mu(r_2, \mathcal{P}) = (1 - 0.5) \times \frac{1}{3} + 0.5 \times 0.8 \approx 0.57$, resulting in the desired ranking of r_2 over r_1 based on μ. □

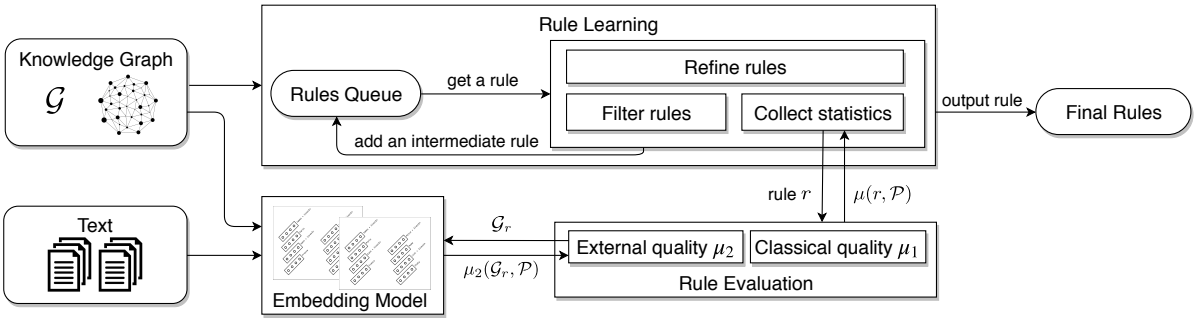


Figure 4.2: Overview of our system

4.3 Approach Description

In this section, we describe our rule learning system with embedding support. Conceptually, it extends the standard relational association rule learners [Galárraga et al., 2015, Goethals and den Bussche, 2002] to also take into account the feedback from embedding models through the probabilistic function f.

Following common practice [Galárraga et al., 2015], we restrict ourselves to rules that are *closed*, where every variable appears at least twice (moreover, we extract only rules whose Horn part is *closed*), and *safe*, where variables appearing in the negated part also appear in the positive part of the rule.

Overview. The input of the system are a KG, possibly a text corpus, and a set of user specified parameters that are used to terminate rule construction. These parameters include an embedding weight λ, a minimum threshold for μ_1, a minimum rule support *r-supp* and other *rule-related* parameters such as a maximum number of positive and negative atoms allowed in r. The KG and text corpus are used to train the embedding model that in turn is used to construct the probabilistic function f. The rules r are constructed in the iterative fashion, starting from the head, by adding atoms to its body one after another until at least one of the termination criteria (that depend on f) is met. In parallel with the construction of the rule r, the quality $\mu(r)$ is computed.

In Figure 4.2, we present a high level architecture of our system, where arrows depict information flow between blocks. The *Rule Learning* block constructs rules over the input KG, *Rule Evaluation* supplies it with quality scores μ for rules r, using $\mathcal{G}$ and f, where f is computed by the *Embedding Model* block from $\mathcal{G}$ and text.

We now discuss the algorithm behind the *Rule Learning* block in Figure 4.2. Following [Galárraga et al., 2015], we model rules as sequences of atoms, where the first atom is the head of the rule and other atoms are its body. The algorithm maintains a priority queue of intermediate rules (see the *Rules Queue* block in Figure 4.2). Initially all

possible binary atoms appearing in $\mathcal{G}$ are added to the queue with empty bodies. At each iteration, a single rule is selected from the queue. If the rule satisfies the *filtering criteria* (see the *Filter rules* block) which we define below, then the system returns it as an output. If the rule is not filtered, then it is processed with one of the *refinement operators* (see the *Refine rules* block) that we define below that expand the rule with one more atom and produce new rule candidates, which are then pushed into the queue (if not being pushed before). The iterative process is repeated until the queue is empty. All the reported rules will be finally ranked by the decreasing order of the hybrid measure μ, computed in *Collect statistics* block.

In the remainder of the section we discuss refinement operators and filtering criteria.

Refinement operators. We rely on the following three standard refinement operators [Galárraga et al., 2015] that extend rules:

(i) *add a positive dangling atom*: add a binary positive atom with one fresh variable and another one appearing in the rule, *i.e.*, *shared*.
(ii) *add a positive instantiated atom*: add a binary positive atom with one argument being a constant and the other one being a shared variable.
(iii) *add a positive closing atom*: add a binary positive atom with both of its arguments being shared variables.

Additionally, we introduce two more operators to allow negated atoms in rule bodies:

(iv) *add an exception instantiated atom*: add a binary negated atom with one of its arguments being a constant, and the other one being a shared variable.
(v) *add an exception closing atom*: add a binary negated atom to the rule with both of its arguments being shared variables.

These two operators are only applied to closed rules. Moreover, we ensure that the addition of exception atoms to the rule $r : head(r) \leftarrow body^+(r)$, should result in

$$r' : head(r) \leftarrow body^+(r), not\ body^-(r)$$

such that

$$r\text{-}supp(head(r) \leftarrow body^+(r), body^-(r), \mathcal{G}) = 0.$$

Intuitively, we aim at adding exceptions that explain the absence of predictions expected to be in the graph rather then their presence. Thus, the introduced exceptions should not affect the rule support, *i.e.*, $r\text{-}supp(r, \mathcal{G}) = r\text{-}supp(r', \mathcal{G})$.

Filtering criteria. After applying one of the refinement operators to a rule, a set of candidate rules is obtained. For each candidate rule we first verify that the hybrid measure μ has increased and discard the rule if it has not. Then, we compute its *h-cover* and our novel exception confidence measure *e-conf* that are defined as follows:

$$\textit{h-cover}(r, \mathcal{G}) = \textit{r-supp}(r, \mathcal{G}) / |\{h \mid h(\textit{head}(r, \mathcal{G})) \in \mathcal{G}\}|,$$
$$\textit{e-conf}(r, \mathcal{G}) = \textit{conf}(r'', \mathcal{G}),$$

where $r'' : \textit{body}^-(r) \leftarrow \textit{body}^+(r), \textit{not head}(r)$. If the *h-cover* and *e-conf* are below the user specified threshold, then the rule is discarded. Intuitively, *h-cover* quantifies the ratio of the known true facts that are implied by the rule. In contrast, *e-conf* is the conditional probability of the exception given predictions produced by the Horn part of r, which helps to disregard insignificant exceptions, *i.e.*, those that explain the absence in $\mathcal{G}$ of only a small fraction of predictions made by $\textit{head}(r) \leftarrow \textit{body}^+(r)$, as such exceptions likely correspond to noise. Observe that not all of the filtering criteria are relevant for all rule types. For example, exception confidence is relevant only for non-monotonic rules to ensure the quality of the added exceptions.

Finally, note that by exploiting the embedding feedback, we can now distinguish exceptions from noise. Consider the rule stating that married people live together. This rule can have several possible exceptions, *e.g.*, either one of the spouses is a researcher or he/she works at a company, which has headquarter in the US. Whenever the rule is enriched with an exception, naturally, the support of its body decreases, *i.e.*, the size of $\mathcal{G}_r$ goes down. Relying on our filtering criteria, we aim at adding such negated atoms, that the average quality of $\mathcal{G}_r$ increases, meaning that the introduced negated atoms prevent unlikely predictions.

4.4 Evaluation

We have implemented our hybrid rule learning approach in Java within a system prototype RuLES [1], and conducted experiments on a Linux machine with 80 cores and 500GB RAM. In this section we report the results of our experimental evaluation, which focuses on *(i)* the benefits of our hybrid embedding-based rule quality measure over traditional rule measures; *(ii)* the effectiveness of RuLES against the state-of-art Horn rule learning systems; and *(iii)* the quality of non-monotonic rules learned by RuLES compared to existing methods.

[1] RuLES is available at https://github.com/hovinhthinh/RuLES

4.4.1 Experimental Setup

Datasets. We performed experiments on the following two real world datasets:

- *FB15K* [Bordes et al., 2013]: a subset of Freebase with 592K binary facts over 15K entities and 1345 relations commonly used for evaluating KG embedding models [Wang et al., 2017].
- *Wiki44K*: a dataset with 250K binary facts over 44K entities and 100 relations, which is a subset of Wikidata dataset from December 2014 used in [Galárraga et al., 2015].

In the experiments for each incomplete KG $\mathcal{G}$ we need its *ideal* completion $\mathcal{G}^i$ that would give us a gold standard for evaluating our approach and comparing it to others. Since obtaining a real life $\mathcal{G}^i$ is hard, we used the KGs FB15K and Wiki44K as reference graphs $\mathcal{G}^i_{appr}$ that approximate $\mathcal{G}^i$. We then constructed $\mathcal{G}$ by randomly selecting 80% of its facts while preserving the distribution of facts over predicates.

Embedding models. We experimented with three embedding models; namely, TransE [Bordes et al., 2013], HolE [Nickel et al., 2016b], and the text-enhanced SSP [Xiao et al., 2017] model. We reuse the implementation of TransE, HolE[2], and SSP[3]. TransE and HolE were trained on $\mathcal{G}$ and SSP on $\mathcal{G}$ enriched with a textual description for each entity extracted from Wikidata. We compared the effectiveness of the models and selected for every KG the best one. Apart from SSP, which showed the best performance on both KGs, we also selected HolE for FB15K and TransE for Wiki44K. Note that in this work as a proof of concept we considered some of the most popular embedding models, but conceptually any model can be used in our system.

Evaluation metric. To evaluate the learned rules, we use the quality of the predictions produced when applying the rules on $\mathcal{G}$, *i.e.*, the more correct facts beyond $\mathcal{G}$ a ruleset produces, the better it is. We consider two evaluation settings: *closed world* (CW) and *open world* (OW). In the CW setting, we define the prediction precision of a rule r:

$$pred\_prec_{CW}(r) = \frac{|\mathcal{G}_r \cap \mathcal{G}^i_{appr} \setminus \mathcal{G}|}{|\mathcal{G}_r \setminus \mathcal{G}|}$$

, and a set of rules R as:

$$pred\_prec_{CW}(R) = \frac{\sum\limits_{r \in R} pred\_prec_{CW}(r)}{|R|}.$$

[2]https://github.com/mnick/scikit-kge (Last accessed December 2020)
[3]https://github.com/bookmanhan/Embedding (Last accessed December 2020)

In the OW setting, we also take into account the incompleteness of $\mathcal{G}^i_{appr}$ and consider the quality of predictions outside it by performing a random sampling and manually annotating the sampled facts relying on Web resources such as Wikipedia. Thus, we define the OW prediction precision $pred\_prec_{OW}$ for a set of rules R as follows:

$$pred\_prec_{OW}(R) = \frac{|\mathcal{G}' \cap \mathcal{G}^i_{appr}| + |\mathcal{G}' \backslash \mathcal{G}^i_{appr}| \times accuracy(\mathcal{G}' \backslash \mathcal{G}^i_{appr})}{|\mathcal{G}'|}.$$

where $\mathcal{G}' = \bigcup_{r \in R} \mathcal{G}_r \backslash \mathcal{G}$ is the union of predictions generated by rules in R, and $accuracy(S)$ is the approximated ratio of true facts inside S computed via manual checking of facts sampled from S. Finally, to evaluate the meaningfulness of exceptions in a rule (*i.e.*, negated atoms), we compute the *revision precision*, which according to [Tran et al., 2016] is defined as the ratio of incorrect facts in the difference between predictions produced by the Horn part of a rule and its non-monotonic version over the total number of predictions in this difference (the higher the revision precision, the better the rule exceptions) computed per ruleset. Formally,

$$rev\_prec_{OW}(R) = 1 - \frac{|\mathcal{G}'' \cap \mathcal{G}^i_{appr}| + |\mathcal{G}'' \backslash \mathcal{G}^i_{appr}| \times accuracy(\mathcal{G}'' \backslash \mathcal{G}^i_{appr})}{|\mathcal{G}''|}.$$

where $\mathcal{G}'' = \mathcal{G}_H \backslash \mathcal{G}_R$ and H is the set of Horn parts of rules in R. Intuitively, $\mathcal{G}''$ contains facts not predicted by the rules in R but predicted by their Horn versions.

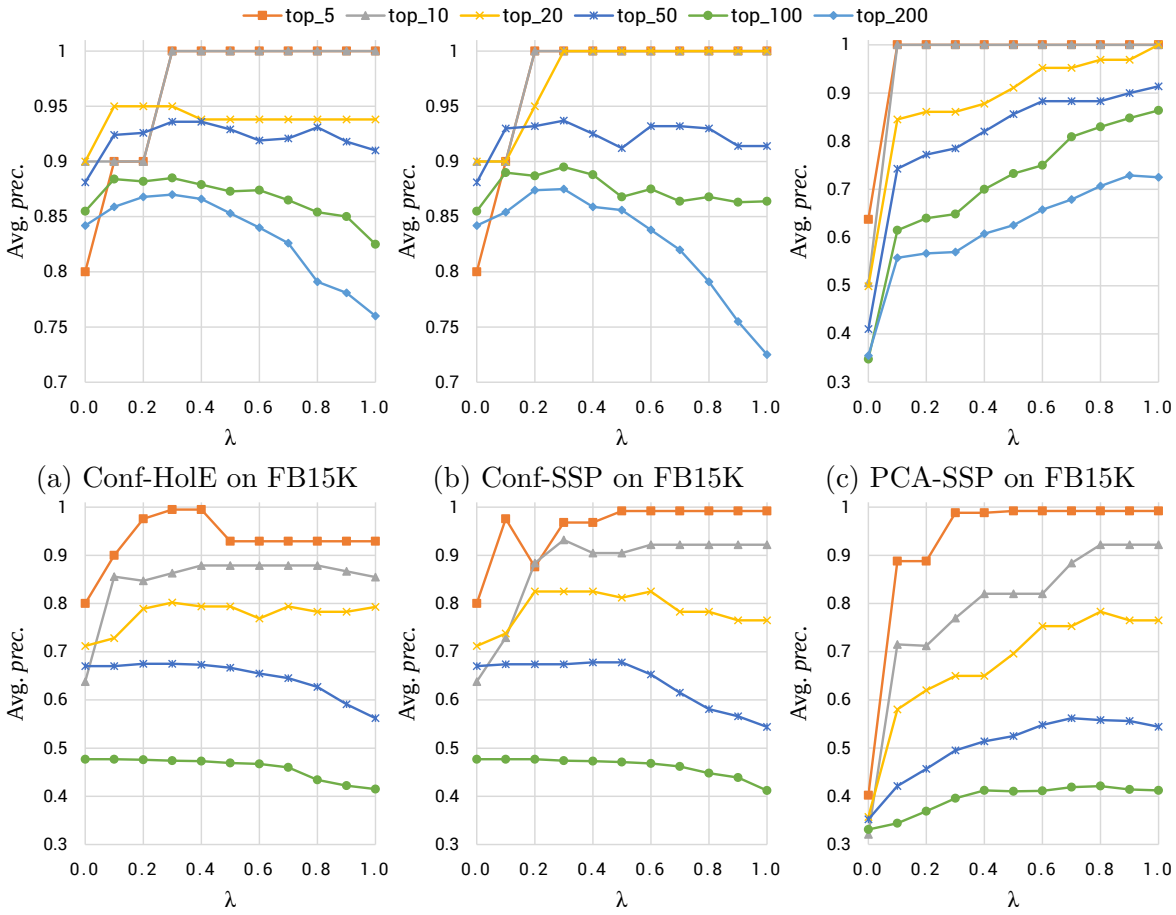
RuLES configuration. We run RuLES in several configurations where μ_1 is set to either *standard confidence (Conf)* or *PCA confidence (PCA)*, and μ_2 is computed based on either TransE, HolE, or SSP models. Through the experiments the configurations are named as μ_1-μ_2 (*e.g.*, Conf-HolE).

4.4.2 Embedding-Based Quality Function

In this experiment, we study the effect of using our hybrid embedding-based rule measure μ from Equation 4.1 on the rule ranking compared to traditional measures and embedding models independently. We start with learning rules of the form

$$r: \; h(X, Z) \leftarrow p(X, Y), q(Y, Z)$$

over $\mathcal{G}$ where $r\text{-}supp(r, \mathcal{G}) \geq 10$, $conf(r, \mathcal{G}) \in [0.1, 1)$ and $h\text{-}cover(r, \mathcal{G}) \geq 0.01$. Then, we rank these rules using Equation 4.1 with $\lambda \in \{0, 0.1, 0.2, \ldots, 1\}$, $\mu_1 \in \{conf, conf_{pca}\}$ and with μ_2 that is computed by relying on TransE, HolE and SSP. Note that $\lambda = 0$ simulates learning rules using the standard measure μ_1 similar to [Galárraga et al.,



(a) Conf-HolE on FB15K (b) Conf-SSP on FB15K (c) PCA-SSP on FB15K

(d) Conf-TransE on Wiki44K (e) Conf-SSP on Wiki44K (f) PCA-SSP on Wiki44K

Figure 4.3: $pred\_prec_{CW}$ of the *top-k* rules with various *embedding weights*

2015], while $\lambda = 1$ corresponds to ranking rules solely based on the predictions of the embedding models. Configuring λ indirectly allows us to compare our hybrid measure to both traditional measures and quality of embedding models.

Figure 4.3 shows the average prediction precision $pred\_prec_{CW}$ of the *top-k* rules ranked using our measure μ for different embedding weights λ (*x-axis*). In particular, in Figures 4.3a, 4.3b, 4.3d, and 4.3e, we observe that combining confidence with any embedding model increases the average prediction precision for $0 \leq \lambda \leq 0.3$. Moreover, we observe the decrease of prediction precision for $0.4 \leq \lambda \leq 1$ and *top-k* rules learned from FB15K when $k \geq 20$ and from Wiki44K when $k \geq 10$. This shows that the combination of μ_1 and μ_2 gives noticeable positive effect on the prediction results. Ranking using hybrid measure with λ around 0.3 achieves better results than both the traditional rule learning and embedding models. On the other hand, for $\mu_1 = conf_{pca}$ the precision increases significantly when combined with embedding models and only

Table 4.1: $pred\_prec_{CW}$ of the *top-k* rules learned using different measures

top-k	FB15K				Wiki44K			
	Conf ($\lambda = 0$)	**PCA** ($\lambda = 0$)	**Conf-HolE** ($\lambda = 0.3$)	**Conf-SSP** ($\lambda = 0.3$)	**Conf** ($\lambda = 0$)	**PCA** ($\lambda = 0$)	**Conf-TransE** ($\lambda = 0.3$)	**Conf-SSP** ($\lambda = 0.3$)
5	0.800	0.638	**1.000**	**1.000**	0.800	0.402	**0.995**	0.968
10	0.900	0.506	**1.000**	**1.000**	0.638	0.321	0.863	**0.932**
20	0.900	0.499	0.950	**1.000**	0.712	0.357	0.802	**0.825**
50	0.881	0.410	0.936	**0.937**	0.670	0.352	**0.675**	0.674
100	0.855	0.348	0.885	**0.895**	**0.477**	0.331	0.474	0.474
200	0.842	0.355	0.870	**0.875**	–	–	–	–

Table 4.2: $pred\_prec_{OW}$ of the *top-k* rules generated by RuLES and AMIE

top-k	FB15K						Wiki44K					
	AMIE-PCA		**AMIE-Conf**		**RuLES**		**AMIE-PCA**		**AMIE-Conf**		**RuLES**	
	Facts	*Prec.*	*Facts*	*Prec.*	*Facts*	*Prec.*	*Facts*	*Prec.*	*Facts*	*Prec.*	*Facts*	*Prec.*
20	1029	0.28	82	0.63	44	1.00	185	0.73	91	0.95	3291	0.98
50	1716	0.43	190	0.74	186	0.92	47099	0.10	3594	0.95	6154	0.88
100	3085	0.65	255	0.78	539	0.80	56831	0.20	13870	0.83	13253	0.82
200	10586	0.62	1210	0.83	1205	0.88	82288	0.39	19538	0.72	20408	0.73
500	40050	0.51	2702	0.75	7124	0.95	219264	0.35	124836	0.23	128256	0.48

decreases slightly for $\lambda = 1$ (Figures 4.3c,4.3f). Utilizing $conf_{pca}$ instead of *conf* as μ_1 in our hybrid measure is less effective, since our training data $\mathcal{G}$ is randomly sampled breaking the partial completeness assumption adopted by the PCA confidence.

Table 4.1 compactly summarizes the average prediction precision of *top-k* rules ranked by the standard rule measures and our μ for the best value of $\lambda = 0.3$ and highlights the effect of using the better embedding model (text-enhanced vs standard). We observe that the accuracy of a utilized embedding model is naturally propagated to the accuracy of the rules that we obtain using our hybrid ranking measure μ. This demonstrates that the use of a better embedding model positively effects the quality of learned rules.

4.4.3 Horn Rule Learning

In this experiment, we compare RuLES under Conf-SSP configuration (embedding weight $\lambda = 0.3$) with the state-of-art Horn rule learning system AMIE. We used the default AMIE-PCA configuration with $conf_{pca}$ and AMIE-Conf with *conf* measures, respectively. For a fair comparison, we set the two configurations of AMIE and our system to generate rules with at most three positive atoms and filtered them based on a minimum confidence of 0.1, head coverage of 0.01 and rule support of 10 in case of FB15K and 2

Table 4.3: $pred\_prec_{OW}$ of the *top-k* rules generated by NeuralLP and RuLES

top-k	**Family-NeuralLP**		**Family-Conf-TransE**	
	Facts	*Prec.*	*Facts*	*Prec.*
10	3709	0.72	4201	0.68
20	8821	0.53	6957	0.72
30	11337	0.49	9368	0.71
40	14662	0.46	11502	0.72
50	18768	0.40	14547	0.62

in case of Wiki44K. We then filtered out all rules with $conf(r, \mathcal{G}) = 1$, as they do not produce any predictions.

Table 4.2 shows the number of facts (see the *Facts* column) predicted by the set R of *top-k* rules in the described settings and their prediction precision $pred\_prec_{OW}(R)$ (see the *Prec.* column). The size of the random sample outside $\mathcal{G}^i_{appr}$ is 20. We can observe that on FB15K, RuLES consistently outperforms both AMIE configurations. The *top-20* rules have the highest precision difference (outperforming AMIE-PCA and AMIE-Conf by 72% and 37% respectively). This is explained by the fact that the hybrid embedding quality penalizes rules with higher number of false predictions. For Wiki44K, RuLES is capable of achieving better precision in most of the cases. Notably, for the *top-20* rules RuLES predicted significantly more facts then competitors yet with a high precision.

In table 4.3, we compare RuLES with the recently developed NeuralLP system [Yang et al., 2017]. For this we utilized the Family dataset used by NeuralLP with 28K facts over 3K entities and 12 relations. Starting from the *top-20* rules RuLES is capable of achieving significantly better precision. For the *top-10* rules the precision of NeuralLP is slightly better, but RuLES predicts significantly more facts.

4.4.4 Exception-Aware Rule Learning

In this experiment, we aim at evaluating the effectiveness of RuLES for learning exception-aware rules. First, consider in Figure 4.4 examples of such rules learned by RuLES over Wiki44K dataset. The first rule r_1 expresses the rule that "*a person is a citizen of the country where his alma mater is located, unless it is a research institution*", since most researchers in universities are foreigners. The second rule r_2 states that "*the scriptwriter of some artistic work is also the scriptwriter of its sequel unless it is a TV series*", which actually reflects the common practice of having several screenwriters for different seasons. Additionally, r_3 encodes that someone belonged to a noble family if his/her spouse is also from the same noble family, excluding the Chinese dynasties.

r_1: $nationality(X, Y) \leftarrow graduated\_from(X, Z), in\_country(Z, Y), \textbf{not}\ research\_uni(Z)$
r_2: $scriptwriter\_of(X, Y) \leftarrow preceded\_by(X, Z), scriptwriter\_of(Z, Y), \textbf{not}\ tv\_series(Z)$
r_3: $noble\_family(X, Y) \leftarrow spouse(X, Z), noble\_family(Z, Y), \textbf{not}\ chinese\_dynasties(Y)$

Figure 4.4: Example rules with exception generated by RuLES

Table 4.4: Comparing *top-k* rules learned by ExRuL and RuLES

(a) Predictive Quality $pred\_prec_{OW}$

top-k	FB15K				Wiki44K			
	ExRuL		**RuLES**		**ExRuL**		**RuLES**	
	Facts	*Prec.*	*Facts*	*Prec.*	*Facts*	*Prec.*	*Facts*	*Prec.*
20	672	0.95	34	0.97	5844	0.93	5640	0.93
50	1797	0.94	158	0.99	8585	0.83	13333	0.84
100	2672	0.94	434	0.99	21081	0.76	25265	0.81
200	4103	0.87	1155	0.96	50957	0.51	43677	0.67
500	13439	0.76	5466	0.90	–	–	–	–

(b) Revision Quality rev_prec_{OW}

top-k	FB15K				Wiki44K			
	ExRuL		**RuLES**		**ExRuL**		**RuLES**	
	Facts	*Prec.*	*Facts*	*Prec.*	*Facts*	*Prec.*	*Facts*	*Prec.*
20	76	0.70	111	0.68	63	0.47	81	0.94
50	126	0.51	435	0.74	191	0.28	611	0.69
100	183	0.43	680	0.76	543	0.49	1698	0.79
200	310	0.30	1112	0.87	4861	0.40	3175	0.80
500	1155	0.53	3760	0.59	–	–	–	–

To quantify the quality of RuLES in learning non-monotonic rules, we compare the Conf-SSP configuration of RuLES (with embedding weight $\lambda = 0.3$) with ExRuL for relational data introduced in the previous chapter in Section 3.6. ExRuL extracts rules of the form $r: h(X, Z) \leftarrow p(X, Y), q(Y, Z), not\ E$, where E is either $e(X, Z)$ or $e(X)$. For a fair comparison we restricted RuLES to learn rules of the same form. We configured both systems setting the minimum rule support threshold to 10 and exception confidence for RuLES to 0.05. To enable the systems to learn rules with exceptions of the form $e(X)$, we enriched our KGs with *types* from original Freebase and Wikidata KGs.

Table 4.5a reports the number of predictions produced by a rule set R of *top-k* non-monotonic rules learned by both systems as well as their precision $pred\_prec_{OW}(R)$ with a sample of 20 prediction outside $\mathcal{G}^i_{appr}$. The results show that RuLES consistently

outperforms ExRuL on both datasets. For Wiki44K, and $k \in \{50, 100\}$, the *top-k* rules produced by RuLES predicted more facts than those induced by the competitor achieving higher overall precision. Regarding the number of predictions, the converse holds for the FB15K KG; however, the rules learned by RuLES are still more accurate.

To evaluate the quality of the chosen exceptions, we compare the $rev\_prec_{OW}(R)$ with a sample of 20 predictions. Observe that in Table 4.5b, rules induced by RuLES prevented the generation of more facts than ExRuL. In all of the cases apart from *top-20* for FB15K, our system managed to remove a larger fraction of erroneous predictions. For Wiki44K, RuLES consistently performs twice as good as ExRuL. In conclusion, the guidance from the embedding model exploited in our system gives us hints on which among the possible exception candidates likely correspond to noise.

4.5 Related Work

Inductive Logic Programming (ILP) addresses the problem of rule learning from data. In its probabilistic setting, given a set of probabilistic examples for grounded atoms and a target predicate p, the task is to learn rules for predicting probabilities of atoms for p [Raedt and Thon, 2010, Corapi et al., 2011, Raedt et al., 2015]. which quickly grows to sizes that ILP methods cannot handle.

A recently proposed differentiable ILP framework [Evans and Grefenstette, 2018] has advantages over traditional ILP in its robustness to noise and errors in the underlying data. However, [Evans and Grefenstette, 2018] requires negative examples, which in our case are hard to get due to the large KG size. Moreover, [Evans and Grefenstette, 2018] is memory-expensive as authors admit, and cannot scale to the size of modern KGs.

In the context of KGs, [Galárraga et al., 2015, Chen et al., 2016] address the incompleteness of KGs by exploiting sophisticated measures over the original graph, possibly enhanced with a schema [d'Amato et al., 2016] or constraints on the number of missing edges [Tanon et al., 2017]. However, these methods do not tap any unstructured information like we do. Indeed, our hybrid embedding-based measure allows us to conveniently account for unstructured information implicitly via embeddings as well as making use of various graph-based rule metrics.

Exploiting embedding models for rule learning is a new research direction that has recently gained attention [Yang et al., 2017, Yang et al., 2015]. To the best of our knowledge, existing methods are purely statistics-based, *i.e.*, they reduce the rule learning problem to algebraic operations on neural-embedding-based representations of a given KG. The work [Yang et al., 2015] constructs rules by modeling relation composition as

multiplication or addition of two relation embeddings. The authors of [Yang et al., 2017] propose a differentiable system for learning models defined by sets of first-order rules that exploits a connection between inference and sparse matrix multiplication [Cohen, 2016]. However, existing approaches pose strong restrictions on target rule patterns, which often prohibit learning interesting rules, *e.g.*, non-chain-like or exception-aware ones, which we support.

Another line of work concerns enhancing embedding models with rules and constraints, *e.g.*, [Wang et al., 2015, Guo et al., 2016, Rastogi et al., 2017, Guo et al., 2018]. While our direction is related, we pursue a different goal of leveraging the feedback from embeddings to improve the quality of the learned rules. To the best of our knowledge, this idea has not been considered in any prior work.

4.6 Conclusion

We presented a method for learning rules that may contain negated atoms from KGs that dynamically exploits feedback from a precomputed embedding model. Our approach is general in that any embedding model can be utilized including text-enhanced ones, which indirectly allows us to harness unstructured web sources for rule learning. We evaluated our approach with various configurations on real-world datasets and observed significant improvements over state-of-the-art rule learning systems.

An interesting future direction is to extend our work to more complex non-monotonic rules with higher-arity predicates, aggregates and existential variables or disjunctions in rule heads, which is challenging due to inevitable scalability issues.

Chapter 5

ExFaKT: Explainable Fact Checking

Fact-checking is a crucial task for accurately populating and updating knowledge graphs. Manually validating candidate facts is time-consuming, which arises the need for automated fact-checking. Prior work on automating this task focuses on estimating truthfulness using numerical scores that are not human-interpretable. Others extract explicit mentions of the candidate fact in the text as evidence for the candidate fact, which can be hard to spot directly.

For that, we introduce ExFaKT, a framework for generating human-comprehensible explanations for candidate facts. ExFaKT uses background knowledge encoded as Horn clauses to rewrite the fact in question into a set of other easier-to-spot facts. The final output is a set of semantic traces for the candidate fact from both text and knowledge graphs. We also demonstrate by experiments that our rewritings significantly increase the recall of fact-spotting while preserving high precision. Moreover, we show that the generated explanations effectively help humans perform fact-checking and can be exploited to automate this task. Finally, we introduce *Tracy*, a user interface demonstrating the usability of our framework.

5.1 Introduction

Motivation and problem. Prominent knowledge graphs (KGs) projects exert much effort to ensure the quality of the KGs construction process. Nevertheless, KGs still contain doubtful if not incorrect triples. This raises the need for validating whether KG triple is correct or not, a task that is often referred to as *fact-checking* or *truth discovery* [Li et al., 2016].

Traditionally, fact-checking has been performed manually by human reviewers but this is time-consuming. Therefore, with the increase of false facts on the Web, the automation of fact-checking is gaining more attention. Methods for automatic fact-checking

(*e.g.*, [Pasternack and Roth, 2013, Nakashole and Mitchell, 2014, Gerber et al., 2015, Li et al., 2016]) proceed in two steps. First, they perform *fact-spotting* by searching for occurrences of a fact candidate, for example ⟨*Sadiq_Khan citizenOf UK*⟩, and possible common alternatives for the fact such as ⟨*Sadiq_Khan citizenOf Pakistan*⟩, in the Web sources. This is done by expanding the predicate into paraphrases (*e.g.*, "has nationality", "has passport") and searching for it jointly with the alias names of the S and O arguments. Then, the extracted evidence (or counter-evidence) is used to infer the truth value of the candidate fact.

Numerical scores produced by fully automated methods are not adequate whenever the final decision is made by KG curators. For humans, such scores are hard to understand or justify without explanations. Some approaches (*e.g.*, [Gerber et al., 2015, Dong and Srivastava, 2013]) attempt to show the sources used in computing the scores as an explanation. Yet, the collected syntactic clues using fact-spotting are often not sufficient since textual sources are *incomplete* and *biased* in what is stated explicitly. For instance, the citizenship of London's mayor *Sadiq Khan* would rarely be mentioned. In addition, some predicates (*e.g.*, *influencedBy*) are *ambiguous*, and have domain-specific interpretations.

Proposed approach. To better support KG curators in deciding the correctness of the candidate facts, we propose a novel framework for finding *semantically related evidence* in Web sources and the underlying KG, and for computing *human-comprehensible explanations* for facts. We refer to our framework, as ExFaKT (**Ex**plaining **Fa**cts over **K**Gs and **T**ext resources).

The key for detecting semantic evidence is intensional background knowledge in the form of *rules*, specifically, Horn rules of the form $H \leftarrow B_1, B_2, \ldots, B_n$. For example,

$$citizenOf(X, Y) \leftarrow mayorOf(X, Z), locatedIn(Z, Y)$$

intuitively states that mayors of cities are normally citizens of countries where these cities are located. Such rules can be specified by humans or automatically extracted from KGs using rule mining methods (*e.g.*, [Galárraga et al., 2015, Wang and Li, 2015]). As the latter may fall short of covering all interesting situations, hand-crafted rules are a valuable asset.

We utilize rules to decompose a fact-spotting query into more frequently stated and thus easier-to-spot related facts. This way, we counter the reporting sparseness and bias. Moreover, rules can encode domain-specific knowledge to better cope with ambiguous predicates. Finally, rules combine knowledge from both textual Web sources and the KG. For example, a rule could find the mayors of cities in news articles and look up the countries of cities in the KG.

Given a set of rules and a query for a fact candidate, ExFaKT rewrites the query into a set of subqueries. Whenever we find evidence in the KG or text that the body of the rule holds, the credibility of the head increases. This process creates semantic traces that *explain*, in a human-readable format, why a fact is likely to be true (or false).

A key difference between our setting and existing applications of query rewriting (*e.g.*, [Shi et al., 2014]) is that the latter assume that the data is contained exclusively in an indexed KG. This is in contrast to our scenario of interest, where some information partially exists in the KG, while the remaining pieces of knowledge are to be extracted from large and noisy text sources on-the-fly. Accounting for this, our framework is particularly tailored towards reducing the cost and uncertainty of the retrieval procedures.

Contributions. The salient contributions of this chapter are:

- We introduce ExFaKT, a framework for computing semantic traces for facts in question from both KG and an implicit external source in the form of a text corpora by utilizing Horn rules.
- We develop optimization strategies, whose target is an automatic search for an effective rewriting plan based on our cost model.
- We evaluate ExFaKT over real-world KGs and rules from various sources to illustrate the effectiveness of our rewriting strategy. We also show the benefits of the computed explanations in supporting human fact-checkers, and the viability of exploiting explanations in automated fact-checking.
- Finally, we introduce *Tracy*, a web user interface, allowing end-user to experiment with our framework.

5.2 Problem Statement

In this section, we start with providing the required background. Then, we describe the problem of computing human-comprehensible explanations for candidate facts.

Rewriting rules. To enable constructing explanations from KGs and other resources, we utilize sets of Horn rules. As discussed in Section 2.2, Horn rule is an expression of the form $H \leftarrow B_1, \ldots, B_n$, where $H, B_1, \ldots, B_n$ are the *atoms* of the rule, *i.e.*, expressions of the form $p(X)$ or $p(X, Y)$, where $p \in \mathcal{R}$ (*i.e.*, KG relations) and X, Y are either entities or variables. We refer to parts of the rule $head(r) = H$ and $body(r) = \{B_1, ..., B_n\}$ respectively as the *head* and *body* of r.

Given a rule r of the aforementioned form and a set I of facts, we define the set of facts inferred by r from I as

$$r(I) = \{H\theta \mid B_1\theta, \ldots, B_n\theta \in I\}, \tag{5.1}$$

where θ is a postfix operator which substitutes variables with constants. We denote by ϵ empty substitutions, *i.e.*, $q\epsilon = q$ for any q. Moreover, $\Pi(I) = \bigcup_{r\in\Pi} r(I)$ is the extension to inferences produced by all rules in Π. The output of multiple rule executions is recursively defined by setting $\Pi^0(I) = I$ and $\Pi^{i+1}(I) = \Pi(\bigcup_{j\in\{0,\ldots,i\}} \Pi^j(I))$. The set $\Pi^\infty(I)$ (called *closure*) contains all possible inferences derived using Π on I.

Fact spotting. Let *textspot* be a textual fact-spotting procedure, which gets as input an atom q and a set $\mathcal{T}$ of textual documents such as Wikipedia ($textspot(q, \mathcal{T})$), and outputs a set of tuples of the form $\langle\theta, s\rangle$, where $q\theta$ is a fact spotted in the text and s is a textual string containing this fact.

Example 5.1. For the query $q = directed(lucas, star\_wars)$ and the text corpus $\mathcal{T} = \{$*"G. Lucas, the director of Star Wars, signs.."*, *"Nolan got inspired by Star Wars"*, *"Star Wars 1977 directed by Lucas.."*$\}$, $textspot(q, \mathcal{T})$ returns $\{\langle\theta, s =$ *"G. Lucas, the director of Star Wars.."*$\rangle$, $\langle\theta, s =$*"Star Wars 1977 directed by Lucas.."*$\rangle\}$, where $\theta = \epsilon$. □

We define the set of all facts involving KG entities $\mathcal{C}$ and relations $\mathcal{R}$ which can be extracted from the text $\mathcal{T}$ using the syntactic fact-spotting procedure *textspot* by $I_\mathcal{T} = \{p(s, o) \mid s, o \in \mathcal{C}, p \in \mathcal{R} \wedge textspot(p(s, o), \mathcal{T}) \neq \emptyset\}$.

Fact explanations. Given a fact in question, a KG, text, and rules, our goal is to compute a set of semantic traces or *explanations* as we call them for a given fact, which are formally defined as follows:

Definition 5.1 (Explanation)**.** *Given a fact q, a KG $\mathcal{G}$, text corpus $\mathcal{T}$ and a ruleset Π, a set $E \subseteq \mathcal{G} \cup I_\mathcal{T}$ of facts is an* explanation *for q with respect to $\Pi, \mathcal{G}, \mathcal{T}$ if $q \in \Pi^\infty(E)$.*

The presence of unstructured textual resources in the input makes the problem of computing explanations particularly challenging. Naturally, a given fact q might have multiple explanations or a single *trivial* one, *i.e.*, q itself. Obviously, explanations as defined above may be subsumed by others. Among all explanations, ideally, we aim at computing non-trivial ones that are

(D1) **concise**, *i.e.*, contain a small number of atoms;
(D2) **close to the query**, *i.e.*, obtained by using few rules;
(D3) **reliable**, *i.e.*, contain as many facts from the KG as possible, since KGs are usually more reliable than text.

5.3 ExFaKT Framework

ExFaKT utilizes rules to compute explanations over the content of KGs and textual resources. We first start with an overview of how these explanations are computed. Then, we illustrate the technical details underlying our framework.

5.3.1 Computing Explanations

Computing the entire KG closure is not feasible as it requires the extraction of *all* possible facts from $\mathcal{T}$. Thus, we proceed *backwards*, *i.e.*, we start from the input fact, and check whether any of the rules can potentially produce such derivation. If so, then we move to the body of the rule, and search for possible instantiations for each body atom. This triggers a recursive process, where the new input is constituted by the body atoms. The recursion stops in case no rules can be found or all atoms are instantiated either by facts in the KG or by text. In this last case, the rules produce new derivations which are returned to earlier recursive calls.

We illustrate the overview of computing explanations using the following example:

Example 5.2. Consider the following input

- a query $q = \mathit{influencedBy}(\mathit{nolan}, \mathit{lucas})$
- a text corpus $\mathcal{T}$ composed of Wikipedia articles
- a KG $\mathcal{G} = \{\mathit{directed}(\mathit{lucas}, \mathit{star\_wars}), \mathit{isDirector}(\mathit{nolan}),$
  $\mathit{directed}(\mathit{lucas}, \mathit{amer\_graffiti})\}$
- a ruleset $\Pi = \{r_1, r_2\}$, where
  $r_1 : \mathit{influencedBy}(X, Y) \leftarrow \mathit{isDirector}(X), \mathit{directed}(Y, Z), \mathit{inspiredBy}(X, Z);$
  $r_2 : \mathit{inspiredBy}(X, Y) \leftarrow \mathit{liked}(X, Y), \mathit{isArtist}(X).$

As $q \notin \mathcal{G}$, and assuming that $\mathit{textspot}(q, \mathcal{T}) = \emptyset$, ExFaKT utilizes those rules in Π, such that $\mathit{Head}(r) = \mathit{influencedBy}$ to explore the potential explanations resulting in the following set of partially grounded explanations :

$$P = \{\{\mathit{isDirector}(\mathit{nolan}),\ \mathit{directed}(\mathit{lucas}, Z),\ \mathit{inspiredBy}(\mathit{nolan}, Z)\}\}.$$

For this candidate, it holds that the atom $\mathit{isDirector}(\mathit{nolan}) \in \mathcal{G}$, hence, we move to the second atom $\mathit{directed}(\mathit{lucas}, Z)$. Grounding this atom from the KG results in the updated set of candidate explanations:

$$P = \{\{\mathit{isDirector}(\mathit{nolan}), \mathit{directed}(\mathit{lucas}, \mathit{star\_wars}), \mathit{inspiredBy}(\mathit{nolan}, \mathit{star\_wars})\},$$
$$\{\mathit{isDirector}(\mathit{nolan}), \mathit{directed}(\mathit{lucas}, \mathit{amer\_graffiti}), \mathit{inspiredBy}(\mathit{nolan}, \mathit{amer\_graffiti})\}\}$$

We start processing the first explanation, and since the first two atoms are in $\mathcal{G}$, they are marked as *found*, and we move to the third atom $inspiredBy(nolan, star\_wars)$. As this third atom is spotted in the text, we get the following evidence set:

$$E_1=\{isDirector(nolan), directed(lucas, star\_wars), inspiredBy(nolan, star\_wars)\}$$

Since all atoms in E_1 were *found*, it is added to the output set.

However, since the last atom in E_1 was found in text which is a noisy resource, we would still rewrite it, seeking for more reliable evidences. In this case, r_2 is used for rewriting leading to the second evidence:

$$E_2 = \{isDirector(nolan), directed(lucas, star\_wars), isArtist(nolan), liked(nolan, star\_wars)\}.$$

Assuming that the last two atoms of E_2 are spotted in $\mathcal{T}$, we get E_2 to be in the output set. Similarly, we process the second explanation candidate in P, *i.e.*, $\{isDirector(nolan)$, $directed(lucas, amer\_graffiti), \dots\}$. □

ExFaKT explanations are human-interpretable; allowing KG curators to judge the validity of the fact. They can also be used as input features to automatically assess the candidate fact truthfulness.

5.3.2 Prerequisites

Besides the KG, ExFaKT requires two key resources, rules and fact-spotting engine over textual sources, which are obtained as follows:

Ruleset acquisition. In general, rules can be either automatically extracted as previously discussed, or manually specified by domain experts as in the context of ontology engineering [Gómez-Pérez et al., 2004]. We performed experiments to evaluate the effectiveness of using rulesets from both sources. We have also conducted a pilot experiment to ensure the feasibility of manual rule construction where we asked non-experienced participants to create useful rules. From these experiments, we observed that adequate rules can be produced not only by KG curators (which are our target group), but also by non-experts (more details in Section 5.4).

Fact-spotting realization. ExFaKT utilizes the syntactic fact-spotting subroutine *textspot*, defined above. In practice, similar to [Nakashole and Mitchell, 2014, Popat et al., 2017], it is implemented relying on a textual-search-based method, in which

the SPO query q is converted to textual representation (*i.e.*, verbalization) using paraphrasing dictionaries for relations such as PATTY [Nakashole et al., 2012b] and entity-mentions dictionaries for entity name aliases. Then, q with its paraphrasing is issued to get the documents containing it. Finally, a named entity recognizer, *e.g.*, [Finkel et al., 2005], is utilized to collect entities from the documents and compute the substitution θ. This versatile approach is easily scalable without extensive training, but any other fact-spotting method can easily be plugged in ExFaKT.

5.3.3 ExFaKT Algorithm

We identify two main operations in ExFaKT's recursive process, namely, *bind* and *rewrite*. The first retrieves answers for a given query from the underlying data sources, while the second rewrites a query into subqueries. We formally specify them below.

Bind. Prior to defining *bind*, we introduce some formal notations. Let g be an atom, $\mathcal{G}$ a KG, and $\mathcal{T}$ a text corpus. We define $\Sigma_{\mathcal{G}}(g) = \{\sigma \mid g\sigma \in \mathcal{G}\}$ and $\Sigma_{\mathcal{T}}(g) = \{\sigma \mid \mathit{textspot}(g\sigma, \mathcal{T}) \neq \emptyset\}$ as the sets of substitutions that result in answers to the query g from the KG and text respectively. Every substitution $\sigma \in \Sigma_{\mathcal{G}}(g)$ is annotated with the metadata $\mathit{source}[\sigma] =$`KG`, while every $\sigma' \in \Sigma_{\mathcal{T}}(g)$ is annotated with $\mathit{source}[\sigma'] =$`TEXT`; moreover, each σ' is annotated with another metadata $\mathit{text}[\sigma_{\mathcal{T}}]$ which contains the string that mentions g in $\mathcal{T}$ (as returned by *textspot*).

Metadata of substitutions is passed to atoms they substitute, for example, if $\mathit{text}[\sigma] =$`X`, then $\mathit{text}[g\sigma] =$`X`. In our procedures, we use another two types of metadata, $\mathit{status}[.]$, to mark atoms which are already processed, and $\mathit{depth}[.]$, to record the number of rewritings that led to the atom at hand. Finally, metadata is not considered for the equality of substitutions and atoms, *i.e.*, two atoms can be equivalent even if they are annotated with different metadata.

Definition 5.2. *The function* $\mathrm{bind}(g, \mathcal{G}, \mathcal{T})$ *receives as input an atom* g*, a KG* $\mathcal{G}$*, and a text corpus* $\mathcal{T}$*. It returns in output the set* $\Sigma = \{\Sigma_{\mathcal{G}}(g) \cup \{\Sigma_{\mathcal{T}}(g) \setminus \Sigma_{\mathcal{G}}(g)\}\}$.

Example 5.3. Let us assume that the target atom $g = \mathit{directed}(\mathit{lucas}, Z)$, a minimal KG $\mathcal{G} = \{\mathit{directed}(\mathit{lucas}, \mathit{star\_wars})\}$ and text corpus $\mathcal{T} =$"Along with *Star Wars,* Lucas has directed short documentaries *Herbie*, ...". Then, $\Sigma_{\mathcal{G}} = \{\sigma_1\}$, $\Sigma_{\mathcal{T}} = \{\sigma_2\}$, where $\sigma_1 = \{Z \rightarrow \mathit{star\_wars}\}$, $\sigma_2 = \{Z \rightarrow \mathit{herbie}\}$, Also, $\mathit{source}[\sigma_1] =$`KG`, $\mathit{source}[\sigma_2] =$ `TEXT`, and $\mathit{text}[\sigma_2]$, store string coordinates of films in $\mathcal{T}$. □

Rewrite. The function *rewrite* moves the search space of the answers for a given query from the rule's head to its body by rewriting an input query into a respective set of subqueries.

Definition 5.3. *The function* rewrite(g,E,Π) *receives as input an atom g, a set of atoms E and a program Π. It computes the set $B_g = \{body(r)\sigma_g\sigma_E | r \in \Pi \wedge head(r)\sigma_g = g\}$ where σ_g is a substitution that unifies the rule head with g and propagates the substituted terms to the rule body, while σ_E renames among the rest of the body variables those that appear in E into fresh ones. Moreover, for each atom a in B_g the function sets $status[a] =$* `TODO`*, $depth[a] = depth[g] + 1$ and returns the set $\{E \cup a \mid a \in B_g\}$.*

Example 5.4. For the input predicate $g = inspiredBy(nolan, Z)$, the intermediate explanation $E = \{isDirector(nolan), directed(Y, Z)\}$ and the ruleset Π=$\{r_2$: $inspiredBy(X, Y) \leftarrow liked(X, Y), isArtist(X)\}$, the function $rewrite(g, E, \Pi)$ first computes $B_g = body(r_2)\sigma_g\sigma_E$, where $\sigma_g = \{X \rightarrow nolan, Y \rightarrow Z\}$ and $\sigma_E = \epsilon$, since σ_g already handled all variables appearing in $body(r_2)$. Finally, in the output we obtain $E = \{isDirector(nolan), directed(Y, Z), liked(nolan, Z),\ 0isArtist(nolan)\}$, where for every atom $a \in B_g$, the status is assigned as $status[a]$=`TODO`, and the depth as $depth[a] = depth[g] + 1$. □

Main procedure. Algorithm 2 shows the main procedure underlying ExFaKT. Our procedure takes as input a query q, a KG $\mathcal{G}$, a text corpus $\mathcal{T}$, a ruleset Π and a global parameter *max_depth* specifying a maximum number of allowed rewritings to ensure termination, and as output provides a set O of explanations for q.

Initially, the *status* of the input query is set to `TODO`, and the set P of potential explanations is initialized with $\{q\}$. In (3) the algorithm iterates over the set P of potential explanations. Explanations $E \in P$, all of whose atoms have status `FOUND`, are moved to the output set O in (7). For other candidate explanations, atoms with the status `TODO` are processed by the procedure *process_goal*.

Procedure *process_goal* takes as input an atom g and first retrieves all substitutions of variables to constants that result in answers to g in the KG and text using *bind* function and copies g into the set TR (to be rewritten). Then, it iterates over the retrieved substitutions and sets the depth of every obtained answer (a) to the one of g and the status to `FOUND` in line (19). After that, the answer is added to the existing atom in the current explanation (E), which is included in O (line 21).

Lastly, *process_goal* rewrites the input query using the rules in the program by invoking the function *rewrite*. Notice that TR might be empty: This occurs if g is a fact, which was verified in the KG. In this case, we do not need to rewrite it, since the fact is already explicitly stated; thus we remove it from TR in line (20).

Note that since no restriction is put on the form of allowed rules, Algorithm 2 might not terminate. To avoid this, we bound the number of rewritings using the *max_depth*

Algorithm 2: Algorithm for computing explanations.

Input: fact q, KG $\mathcal{G}$, text corpus $\mathcal{T}$, ruleset Π, nonegative parameter *max_depth* for ensuring termination

Output: set of explanations O

```
1  function explain(q, G, T, Π)
2      depth[q] ← 0; status[q] ← TODO; P ← {{q}}; O ← {}
3      while P ≠ ∅ do
4          Pick explanation E from P (i.e., E ∈ P)
5          P ← P \ {E}
6          if status[g] =FOUND for all g ∈ E then
7              O ← O ∪ {E} ▷We found a valid explanation.
8          else
9              Pick an atom g from E s.t. status[g] =TODO
10             NT ← process_goal(g, E \ {g}, G, T, Π)
11             P ← P ∪ NT
12     return O
13 function process_goal(g, E, G, T, Π)
14     O ← ∅
15     Σ ← bind(g, G, T)
16     TR ← {g}
17     for σ ∈ Σ do
18         a ← gσ
19         depth[a] ← depth[g]; status[a] ← FOUND
20         if source[σ] =KG then TR ← TR \ {a}
21         O ← O ∪ {Eσ ∪ {a}}
22     for gr ∈ TR s.t. depth[gr] < max_depth do
23         O ← O ∪ rewrite(gr, E, Π)
24     return O
```

parameter, ensuring that the algorithm always terminates. As output, Algorithm 2 returns explanations for the input candidate, which is a property formally stated in the following theorem:

Theorem 5.1. *Let $\mathcal{G}$ be a KG, $\mathcal{T}$ a text corpus, Π a program, q an input fact, and the output $O = explain(q, \mathcal{G}, \mathcal{T}, \Pi)$. If $E \in O$ then E is an explanation of q with respect to $\Pi, \mathcal{G}, \mathcal{T}$.*

Optimizations. Often, we do not need to calculate all explanations; few relevant ones are sufficient for establishing the truth value of a fact in question. To improve the performance, we introduce anytime behavior and incrementally collect new explanations as they are added to the output set (line 7 of Algorithm 2). In this new setting, we can stop the algorithm whenever it has returned satisfactory explanations, but it is crucial

that the most relevant explanations based on the criteria **(D1)**-**(D3)** from Section 5.2 are computed first.

Example 5.5. If *influencedBy*(*nolan*, *lucas*) from Example 5.2 was found in text, then $E_0 = \{influencedBy(nolan, lucas)\}$ would be the most *concise* explanation, since it contains only a single atom. Moreover, for E_1 and E_2 from the same example we have that E_1 is *closer to the query* than E_2, since less rules were used to produce E_1. If there is another explanation E_3 which equals to E_2 but with *star_wars* substituted by *Herbie* and *directed*(*lucas*, *herbie*) is found in text, it holds that E_2 is more *reliable* than E_3, since E_2 contains fewer atoms from text sources. □

Our optimizations along these lines affect two operations: the *explanation selection* and the *atom selection criteria*.

Explanation selection criterion. In line 4, Algorithm 2 selects one explanation to be processed from those in P. To prioritize the processing of promising explanations we change the order in which they are picked based on the following criteria: (i) we favor *shorter explanations*, *i.e.*, explanations with the lowest number of atoms **(D1)**; (ii) we favor *explanations produced with fewer rewritings* by picking the query with the smallest *depth* value **(D2)**.

Atom selection criterion. In line 9 of Algorithm 2, a naive strategy would always pick the first atom in E; resulting in a huge search space. To counter this, we first process atoms without variables. Then, we pick atoms with some constants and keep those with only variables till the end. Moreover, to favor explanations that can be proven from the KG **(D3)**, we prefer atoms with KG substitutions.

5.4 Evaluation

We start in Section 5.4.2 with evaluating the effectiveness of ExFaKT with respect to the increase in the coverage of the collected pieces of evidence while preserving their precision (as shown later in Section 5.4.3). Additionally, we present a user-study which targets assessing the quality of the explanations and their readability as well as usefulness for human-reviewers in Section 5.4.3. Later in Section 5.4.4, we demonstrate the effectiveness of ExFaKT using automatically mined rules as input. Moreover, a showcase for integrating ExFaKT into the pipeline of a fully automated fact-checking system is given in Section 5.4.5. Finally, Section 5.4.6 reports the results of our study on the feasibility of manual rule construction.

5.4.1 Experimental Setup

Datasets. We conducted our experiments on two datasets:

- **YAGO-based** benchmark, which consists of 300 true candidate facts, uniformly distributed over six relations (listed in Table 5.1). The instances of the first three relations were randomly selected from YAGO [Suchanek et al., 2007]. The other three relations do not appear in the KG; hence, their instances were semi-automatically curated using simple logical rules. Then, 50% of the facts used during their creation were removed at random; thus, intentionally introducing incompleteness in the KG, and hence, the need for textual sources.
- **DBpedia-based** benchmark is a subset of the dataset proposed by [Shiralkar et al., 2017] containing facts over the predicates, for which AMIE, a state-of-the-art rule mining system [Galárraga et al., 2015], managed to learn at least 5 rules having them in the head. This benchmark contains four predicates with a total of 1763 correct facts.

Rules. We constructed a ruleset for each benchmark as follows: For the *YAGO-based* benchmark, we selected rules from the top-ranked ones mined by AMIE from YAGO, and added further rules with new predicates that do not exist in the KG. We refer to these new predicates as *text-based*, as they need to be verified from the text. This way, we obtained 20 rules on average for each head predicate. For the *DBpedia-based* benchmark, we used the rules mined by AMIE from DBpedia without altering them. Each head predicate in the dataset obtained a set of 100 related rules on average. This setup is designed to study the case of fully relying on automatically learned rules.

Knowledge graph. We used YAGO3 [Suchanek et al., 2007] as *KG* in all experiments. YAGO3 contains around 5.5M facts and 35 relations. This KG is geared towards precision rather than recall, allowing us to treat it as a trusted resource.

Text corpora. As for text, we experimented with two different sources: (i) *Wiki* which contains 5.5M Wikipedia articles, whose textual parts were split into sentences and indexed as separate documents using Elasticsearch [Gormley and Tong, 2015] and (ii) *Web* constructed relying on the Bing API for searching in Web pages.

Baselines. We compared against three baselines:

- **B-Wiki**: a method that syntactically spots candidate facts and their paraphrases in the *Wiki* corpus.
- **B-Web**: a method that retrieves fact occurrences in the Web using the Bing search API and post-filters the results to obtain the relevant text snippets (this baseline was extracted from [Popat et al., 2017]).

- **B-Search**: a simulation for a user issuing verbalized versions of candidate facts to commercial search engine and retrieving the top-5 results.

Configurations. To analyze the influence of the various sources, we ran ExFaKT[1] with the following configurations:

- ***KG***: Rules are used over the KG facts only.
- ***Wiki***: Rules are used only with syntactic fact-spotting over the *Wiki* corpus, *i.e.*, no KG facts are exploited.
- ***Web***: Rules are used only with fact-spotting over the *Web*, *i.e.*, no KG facts are used.
- ***KG+Wiki***: Rules used together with both *Wiki* corpus and KG.
- ***KG+Web***: Both *Web* corpus and KG are used with the rules.

In all of the experiments, we set the parameter *max_depth* to 5 unless otherwise stated.

5.4.2 Explanations Coverage

Experimental details. In this experiment, we show the effectiveness of ExFaKT in retrieving more explanations for candidate facts. We used the *YAGO-based* dataset and compared all five configurations of ExFaKT against the baselines. We computed the **recall** of each configuration as the ratio of the queries for which at least one explanation was retrieved.

Results. Table 5.1 reports the recall for each predicate in the dataset and the recall of the whole dataset as the total. These results show that ExFaKT configured with *KG+Wiki* or *KG+Web* almost doubled the recall of the baselines *B-Wiki* and *B-Web* respectively, whereas rules on the KG alone (*KG*) have the worst recall. Similarly, configurations over text corpora alone; namely, *Wiki* or *Web* could not compensate the absence of the KG.

Observe that since *mayorOf* is prominent enough to be easily spotted in the text, there is no increase in the recall for this predicate. In contrast, our method is particularly successful for *countryWon*, since the baselines fail to spot facts over this newly created predicate. Overall, the results demonstrate that while neither the KG nor textual sources alone are sufficient to collect strong evidence, their combination doubles the recall, and is doubtlessly the best configuration. Moreover, comparing *KG+Wiki* with *KG+Web* shows that increasing the size of the textual corpus enhances the results.

[1]ExFaKT is available at https://www.mpi-inf.mpg.de/impact/exfakt.

Table 5.1: Recall of the baselines vs. ExFaKT configurations

	B-Wiki	**B-Web**	**KG**	**Wiki**	**Web**	**KG+Wiki**	**KG+Web**
influences	0.30	0.24	0.00	0.38	0.88	0.42	0.92
isPolitOf	0.02	0.16	0.26	0.18	0.88	0.42	0.92
wroteMusic	0.08	0.28	0.00	0.10	0.72	0.24	0.78
mayorOf	0.66	0.9	0.00	0.66	0.90	0.66	0.90
actedWith	0.26	0.52	0.18	0.26	0.60	0.54	0.94
countryWon	0.18	0.38	0.00	0.18	0.38	0.70	0.92
Total	**0.25**	**0.41**	**0.07**	**0.29**	**0.73**	**0.50**	**0.90**

Table 5.2: Examples of explanations produced by ExFaKT

Fact candidates	**Explanations**
$countryWon(guatemala, nobel)$	$isCitizenOf(miguel\_asturias, guatemala)$, $source =$ `TEXT`, "Asturias is a Nobel Prize-winning Guatemalan poet..." $hasWonPrize(miguel\_asturias, nobel)$, $source =$ `KG`
$influences(s\_fitzgerald, r\_yates)$	$wrote(s\_fitzgerald, the\_great\_gatsby)$, $source =$ `KG` $read(r\_yates, the\_great\_gatsby)$, $source =$ `TEXT`, "Yates, called "The Great Gatsby" the most nourishing..."

Examples. As anecdotal evidence, Table 5.2 shows two examples of candidate facts for which our approach managed to compute supporting explanations even though fact-spotting failed to find direct mentions for them. The first fact to be spotted is about the country *Guatemala* and the predicate *country has won prize*. Our method was able to find positive evidence for Guatemala winning the Nobel prize by spotting the Nobel laureate Miguel Asturias, and combining this information with the fact that he is a citizen of Guatemala.

In the second example, we were able to extract an evidence about the influence of a writer on another one by spotting the fact that the latter read books written by the former. Note that here the relation *read* is not present in the KG but mentioned in the Wikipedia text.

5.4.3 Explanations Quality and Usefulness

In this experiment, we evaluate the quality of the retrieved explanations by estimating the precision of the results and their readability based on human judgment.

Experimental details. We designed a Mechanical Turk [Buhrmester et al., 2011] task

Table 5.3: Mechanical Turk task statistics

Config	Candidates	Explanations	Question 1			Question 2	
			Yes	*No*	*Cannot*	Yes	No
B-Wiki	75	75	0.87	0.04	0.10	0.90	0.10
B-Web	122	122	0.85	0.0	0.15	0.80	0.20
B-Search	228	228	0.58	0.01	0.42	0.55	0.45
KG+Wiki	159	311	0.64	0.35	0.02	0.63	0.37
KG+Web	267	1021	0.82	0.01	0.17	0.74	0.26

to collect human judgments on the quality of the computed explanations. To facilitate readability, we translated our task into a natural language question. The participants were shown a candidate fact and some extracted explanation (*i.e.*, in a human readable format), and their task was to judge the correctness of the fact relying only on the provided information.

Since it may be hard for non-experts to give a solid judgment, the participants were asked to choose one out of five answers: (i) *"For sure, yes"*, (ii) *"Probably, yes"*, (iii) *"Probably, no"*, (iv) *"For sure, no"*, or (v) *"Can not judge"*. As a confirmation, they were also asked for their explicit feedback on the usefulness of the provided information. Participants were provided with a set of instructions and clarification examples covering all possible cases. Each record was assigned to 5 different participants.

We evaluated the top-5 explanations produced by ExFaKT's configurations *KG+Wiki* and *KG+Web* on the *YAGO-based* dataset as reported in Section 5.4.2. We compared them to the results of the baselines *B-Wiki* and *B-Web* respectively. Additionally, we included the results of the third baseline *B-Search* for annotations.

Evaluation metrics. The target of this experiment is to assess the relevance of the extracted traces for the query by relying on whether participants were able to make a correct judgment. To this end, we used the annotations given by the participants to perform majority voting on three categories: *correct judgment* (*i.e.*, yes case) *incorrect judgment* (*i.e.*, no case), and *unable to judge*. Tie cases are randomly broken. Explanations with a majority of the *correct judgment* are counted as relevant.

We calculated **Precision@K**, which is defined as the ratio of candidate facts for which our method returned one or more *relevant* explanations to those that have at least one (relevant or irrelevant) explanation. Finally, we computed the f_1 **score** and **the mean average precision** at *top-5* ($MAP@5$).

Results. Table 5.3 reports the sizes of each dataset and the distribution of the majority voting over the classes. The second column shows the number of candidate

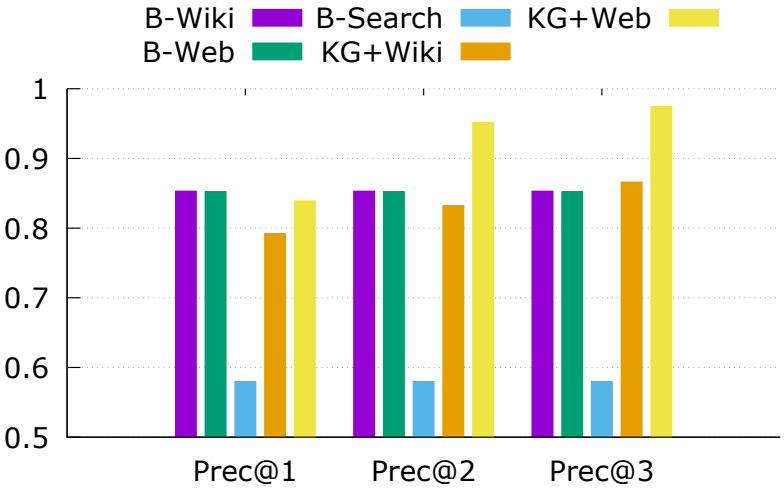


Figure 5.1: *Precision@k* computed based on human annotations

facts involved in the evaluation, while the third column indicates the total number of explanations produced for all of the candidates. The annotations demonstrate a fair Fleiss Kappa [Fleiss et al., 1971] agreement of 0.35 for the Wiki-based configurations and a slight agreement of 0.03 for Web-based configurations, reflecting the difficulty and subjectivity of the task in the presence of noisy results. Deeper analysis showed that the lower the *ratio of atoms binded from text*, the easier it is to judge the explanation correctly. While reducing the number of rules has a similar effect, increasing the the number of atoms in the explanation increases human-comprehensibility of explanations.

Observe that the average time required for the annotator to judge the truthfulness of a fact candidate based on explanations retrieved by our method (27 seconds) was almost half the time required to judge it based on standard Web search results (51 seconds), yet the quality of the judgments in the former case is higher. This illustrates the benefits of using our method for increasing the productivity and accuracy of human fact-checkers.

Figure 5.1 shows the results for *Prec@k* for $k = 1, 2, 3$. First, we observe that the Prec@1 of ExFaKT's configurations *KG+Wiki* and *KG+Web* are 6% and 2% lower than for the respective baselines *B-Wiki* and *B-Web* which both have precision of 0.85, yet all configurations have significantly higher precision compared to the case when a traditional commercial search engine was exploited as for *B-Search*. From the figure we can observe that *KG+Web* exceeds the baselines starting from top-2 with $Prec@2 = 0.95$ and *KG+Wiki* overcomes the baselines for the top-3 explanations. The results also indicate that the precision for the both configurations *KG+Wiki* and *KG+Web* consistently increase till they achieve $Prec@5$ of 0.87 and 0.97 respectively. Furthermore, $MAP@5$ for *KG+Web* is 0.84, which exceeds $MAP@5 = 0.77$ of *KG+Wiki*.

Table 5.4: Recall with automatically mined rules

	B-Wiki	**KG**	**Wiki**	**KG+Wiki**
vicePresident	0.96	1.00	0.96	1.00
diedIn	0.46	0.11	0.51	0.56
nationality	0.64	0.08	0.68	0.68
graduatedFrom	0.14	0.11	0.14	0.30
Total	**0.20**	**0.14**	**0.21**	**0.35**

Since our method doubles the recall (Table 5.1), it also significantly enhances the F_1 score achieving 0.64 and 0.93 with *KG+Wiki* and *KG+Web* compared to 0.4 and 0.55 for the corresponding baselines.

5.4.4 Extracting Explanations using Mined Rules

So far we have considered manually refined rules. We now evaluate ExFaKT using automatically learned, hence noisier rules.

Experimental details. We used the *DBpedia-based* dataset with the ruleset automatically mined by AMIE from DBpedia. These rules contain many predicates that are not in YAGO; thus, textual sources are vital. We compare the recall of *KG+Wiki* against the configurations *B-Wiki*, *KG*, and *Wiki*. For this experiment, we set the rewriting depth *max_depth* to 2 to reduce propagation of noisy rule rewritings. We computed $Prec@5$, $MAP@5$ and $f_1@5$ scores in the same way as in Section 5.4.3.

Results. Table 5.4 reports the recall for the fact candidates over four different predicates. We observe that our method improves the recall for all predicates, with the best result obtained for *graduatedFrom*, where the recall has even doubled. Based on human annotations, we obtained $Prec@5 = 0.84$, $MAP@5 = 0.84$ for *B-Wiki* and $Prec@5 = 0.64$, $MAP@5 = 0.56$ for *KG+Wiki*. Despite that, *KG+Wiki* configuration achieved a better f_1 with $f_1@5 = 0.45$ compared to a lower value $f_1@5 = 0.32$ obtained by *B-Wiki*. These results demonstrate the effectiveness of ExFaKT also when automatically mined rules are used as input.

5.4.5 Rule-based Automatic Fact-checking

In this experiment, we focus on the viability of exploiting extracted explanations in automated fact-checking. To this end, we implemented a simple automated fact-checker that uses the explanations to determine whether a given fact should be supported or

rejected.

For the sake of computing numerical scores over explanations, we define the notion of explanation confidence as

$$confidence(E) = \frac{1}{|E|} \sum_{a \in E} \frac{trust(source[a])}{depth[a]}$$

where E is an explanation, $trust(\cdot)$ is 1 if $source[a]$ is `KG`, or 0.5 otherwise, representing the trust in the source, and $depth[a]$ is number of rewritings performed to obtain the atom as in Section 5.3. Then, for each fact candidate $f = p(a, b)$, we exploit two rules sets Π_+ for supporting it and Π_- for refuting it. Then, we compute the supporting explanations set $O_+ = explain(p(a, b), \mathcal{G}, \mathcal{T}, \Pi_+)$, and another set of refutation evidence $O_- = explain(not\_p(a, b), \mathcal{G}, \mathcal{T}, \Pi_-)$ where not_p is newly introduced predicate representing the negation of p. Then, we compute the *truthfulness score* for f as:

$$truth\_score(f) = quality(O_+) - quality(O_-)$$

where $quality(\cdot)$ is the average confidence of explanations belonging to this set. The truth score acts as a threshold value which determines whether the fact should be accepted or rejected.

Dataset. We applied the automated fact-checking on two datasets containing both correct and erroneous candidate facts.

- *Politicians benchmark* [Nakashole and Mitchell, 2014] with 275 candidates, which contains for each true fact a set of its alternatives. For fairness, we removed the existing candidates from the KG which would trivially support the input facts. For this dataset, we used a ruleset of both manually specified and automatically mined rules by AMIE.
- The previously used *DBpedia-based* dataset enriched by adding a set of erroneous alternative instances from [Shiralkar et al., 2017] for each true fact. In this case, we used the set of automatically learned rules from Section 5.4.4 extended with rules for not_p for each predicate p in the given dataset.

Experimental details. The explanations were computed using the *KG+Wiki* configuration. We compare the results of our simple ExFaKT-based fact checker to two state-of-the-art approaches:

- *TruthFinder* [Yin et al., 2008] which uses a voting approach over the retrieved documents weighted based on their sources.
- *Language-Stance-Credibility (LSC)* [Popat et al., 2017] which decides the truthfulness

Table 5.5: Rule-based fact checking vs prior methods

Method	DBpedia-Based		Politicians	
	Recall	Accu.	Recall	Accu.
ExFaKT (*KG+Wiki*)	0.68	0.93	0.83	0.81
TruthFinder [Yin et al., 2008]	0.66	0.97	0.73	0.79
LSC [Popat et al., 2017]	0.99	0.59	1.00	0.55

of a claim based on language, stance, and reliability of the evidence sources.

We considered the first 5 explanations retrieved by the *KG+Wiki* configuration per supporting O_+ and refuting O_- sets respectively. Note that *LSC* collects clues from the whole Web, while *TruthFinder* and ExFaKT rely only on the Wikipedia text corpus.

Evaluation metric. In order to compare the performance of the various approaches, we grouped the true facts with their alternatives and ranked the elements of the group using the scoring function of each method. For instance, let f_t be a true fact and $f_{f_1}, \ldots, f_{f_n}$ be alternative false facts. In this setting, each method assigns a score value to every fact thus producing a ranking. After the ranked list is computed, we compute the **recall** as the number of groups where the method at hand was capable of returning some truthfulness scores for the true fact. Then, for each given group $G_i = \{f_t, f_{f_1}, \ldots, f_{f_n}\}$, we compute the **accuracy** as $Accuracy(G_i) = \frac{1}{n} \sum_{f_{f_j} \in G} [score(f_t) \geqslant score(f_{f_j})]$ where $[\cdot]$ is the Iverson bracket and $score(f)$ is the truthfulness score computed by each method. This accuracy estimates the probability that the true fact f_t has the highest rank among its alternative facts as in [Nakashole and Mitchell, 2014].

Results. Table 5.5 shows the results of the average accuracy (*Accu.*) for all groups of facts and recall of our method compared to the competitors. According to the first two rows, we can observe that our method is on par with the accuracy of TruthFinder, despite the simplicity of our ranking function. Moreover, our method is advantageous, as it offers clear explanations of the results, and has a better recall compared to TruthFinder, especially for the politicians dataset. Moreover, importantly while LSC is utilizing the benefits of scrapping the whole Web, our system still achieves significantly higher accuracy. We analyzed the cases where our method failed, and observed that some of the relations in the test set require more complex rules to be properly supported or refuted. For instance, this dataset contains facts over such predicates as *isMarriedTo* or *holdsPosition*, which are better explained using temporal rules, as they change over time (*e.g.*, one person can be married to multiple persons, but not at the same time). Extending our approach to support also rules of this kind is a promising future direction.

5.4.6 Rule Specification

In addition to automatically mined rules, our approach benefits from hand-crafted rules. In this experiment, we examine the feasibility and cost of this kind of rule specification.

Experimental details. We asked 10 students from different fields to create rules. Most of these had no prior exposure to logical rules or KGs; so we gave them 20 minutes of explanation on KGs and Horn rules. Each participant picked 5 predicates from a list of KG predicates, and was asked to write at least one supporting rule and at least one refuting rule. The participants were given 30 minutes to create the rules.

Evaluation metric. We classified the resulting rules into three categories: (i) *strong rules* that represent causality and generalize (*e.g.*, $politicianOf(X, Y) \leftarrow electedIn(X, Z)$, $in(Z, Y)$.); (ii) *valid rules* that capture typical correlations but may be tied to specific cases that cannot be generated (*e.g.*, $citizenOf(X, Y) \leftarrow grewUpIn(X, Y)$); (iii) *invalid rules* that are logically flawed or do not properly reflect typical situations (*e.g.*, $not\_memberOf(X, Y) \leftarrow leader(X, Y)$). We consider *strong* and *valid* rules as suitable.

Results. Within 30 minutes, the 10 participants created 96 rules for 23 different head predicates (*i.e.*, with average of 3 minutes per rule). Table 5.6 shows their distribution over the three categories. More than half of the rules were strong, and more than 80% were strong or valid. The remaining incorrect rules could be filtered out by having the same participants judge the validity of each others' rules, using a voting scheme. This study clearly demonstrates that manual construction of rules is not a bottleneck, and can be accomplished by informed crowdsourcing at fairly low cost.

5.5 ExFaKT Demonstration

5.5.1 Implementation

We implemented a demonstration tool, Tracy, to illustrate the benefits of ExFaKT in supporting KG curators deciding the correctness of the candidate facts. In Tracy, we realize the two main components: (i) *Query rewriting engine*, which is responsible for rewriting facts and generating explanations, and (ii) *Fact-spotting*, which is the component that finds evidences in Web sources.

Our framework receives as input a candidate fact (*i.e.*, the query), a set of rules, and two knowledge sources: (i) A KG, which we view as a *reliable* source; and (ii) an unstructured collection of text corpora, which we consider as an *unreliable* source, since its extractions by the fact spotting might be noisy.

In general, rules can be automatically extracted from KGs as shown in Chapters 2, 3, and 4. However, extracted rules are restricted to the predicates in the KG. Thus, Tracy also accepts manually specified rules with user-defined predicates, which later can be spotted in the text. This allows obtaining rich and domain-aware rules beyond KG.

The workflow of Tracy is as follows: First, a user submits an input fact Q (*i.e.*, the query) to the query rewriting engine. Then, this component possibly rewrites it into easier facts according to the logic specified by the rules contacting the *fact-spotting* component every time it cannot find evidences in the KG. If a sufficient number of evidences is found, then the query rewriting engine uses them to return one or more explanations for Q.

Query rewriting engine. The process starts by considering rules, whose head predicates are the same as in the query. Then, it visits the rules' bodies and effectively rewrites the query into additional subqueries. The subqueries might trigger new rules and this results in a recursive process until all subqueries are answered. Such rule-based evaluation strategy is well-known in logic programming and commonly used for query answering. Tracy implements an adaptation of a set-based version of standard SLD resolution [Kowalski and Kuehner, 1971] as discussed in Section 5.3.3.

At the end, the answers (either from the KG or Web sources) are used to compute the explanations. This is done by revisiting the rules bottom-up. Figure 5.2 shows an extract of an example explanation provided by Tracy. In this case, the explanation illustrates that the fact that *Sadiq Khan is a citizen of UK* is likely to be true according to the rule $isCitizenOf(X, Y) \leftarrow mayorOf(X, Z),\ hasCapital(Y, Z)$ and additional evidence found both in the KG and textual corpora.

As discussed in Section5.2, ideally, we aim at computing non-trivial explanations that are **(D1) concise** with a small number of atoms; **(D2) close to the query**, *i.e.*, obtained by using few rules; **(D3) reliable**, *i.e.*, contain as many facts from the KG as possible, since KGs are usually more reliable than text.

In order to recognize these explanations, we compute a confidence score for each explanation E, defined as:

$$confidence(E) = \frac{1}{|E|} \sum_{a \in E} \frac{trust(sources[a])}{depth[a]}$$

where a is the atom of the explanation E. This confidence score is directly correlated with the quality of the sources containing the atom a ($trust(sources[a])$) and inversely correlated with the depth of the rewriting performed to reach the atom a (*i.e.*, the number of used rewriting rules) and the number of distinct atoms in the explanation.

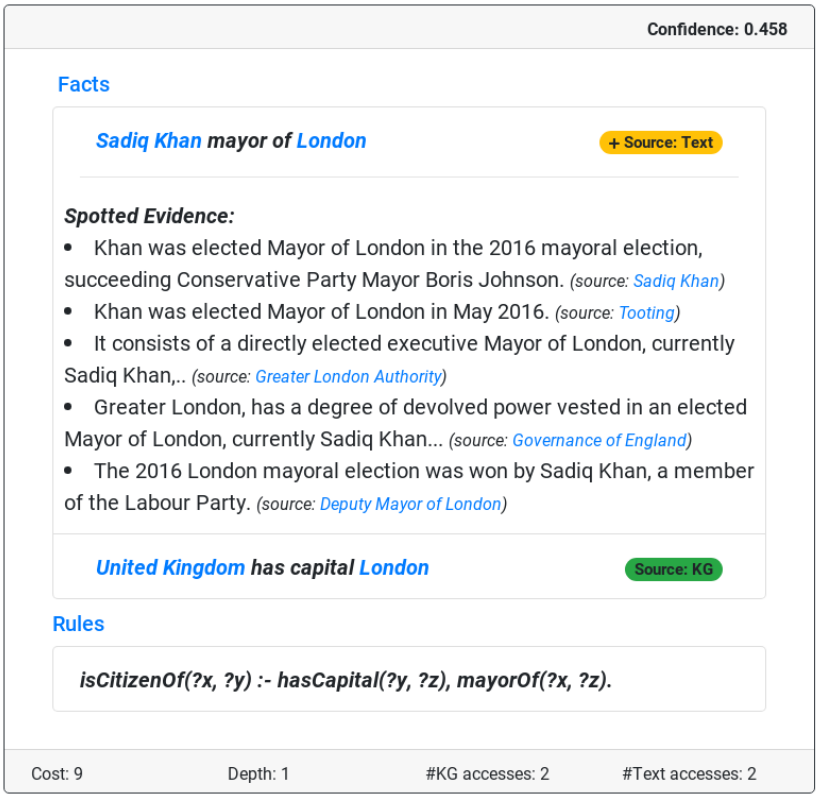


Figure 5.2: Example explanation returned by Tracy

Fact spotting. Tracy follows a modular design allowing integrating any fact-spotting method. As default, we implemented a dictionary based fact-spotting procedure similar to [Nakashole and Mitchell, 2014, Popat et al., 2017]. It is a simple and highly scalable method that does not require training. The main idea consists of first converting the triple query into a textual representation (*i.e.*, verbalization) using a paraphrasing dictionary. Then, queries with the paraphrases are issued to a spotting engine to retrieve documents that mention them.

We use two types of dictionaries, depending on whether the used KG is YAGO or Wikidata (these are the only two which are currently supported). With YAGO, we use the paraphrases learned by PATTY [Nakashole et al., 2012b] after some manual filtering. With Wikidata, we exploit the name aliases included in the KG.

Tracy also provides two types of spotting engines, one that uses a *local* text corpus and another one that consults a *remote* one:

- **Local corpus:** We utilize Elasticsearch [Gormley and Tong, 2015] to index all Wikipedia articles. As a prepossessing step, we filtered the textual parts in the articles by removing semi-structured parts such as tables and info-boxes. Then,

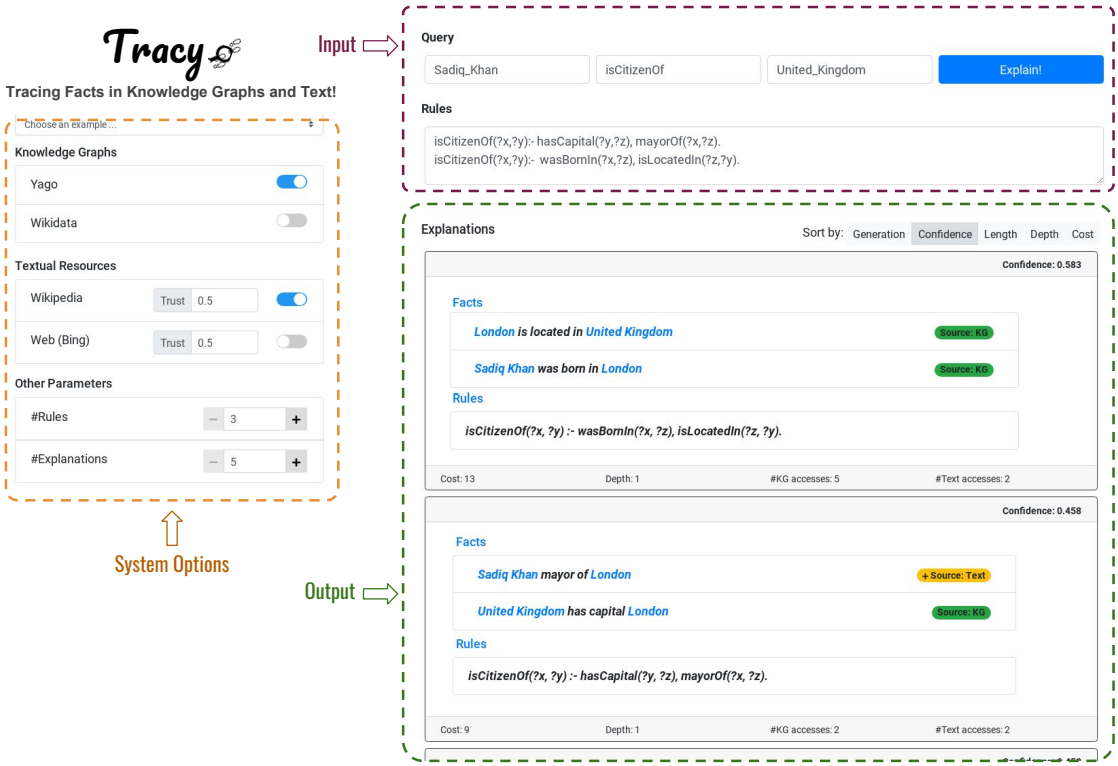


Figure 5.3: Tracy interface

we index the article sentences separately along with the title. For spotting a triple query, we issue a boolean query with the paraphrases of each part of the triple separately. Then, we collect the *top-5* matching sentences as evidence for the provided query.

- **Remote corpus:** We also provide the option for searching the Web using Bing Web search API [2]. To query the API, we compose a string query containing all possible paraphrases of the triple and use the *top-5* search results as evidence.

5.5.2 User Interface

We built a Web interface to allow an easy interaction with the system. Figure 5.3 shows a screenshot illustrating its main components.

System options. The left-hand side of Figure 5.3 shows the *system options menu*, which is the part where the user can select from a predefined list of KGs and textual sources to work with. Moreover, the interface allows the user to tune the *query rewriting* procedure by specifying a *trust* value for each textual source and the limit on the number

[2] http://azure.microsoft.com/en-us/services/cognitive-services/bing-web-search-api

Table 5.6: Statistics for manually specifying rules experiment

	Strong	**Valid**	**Invalid**	**Total**
Supporting Rules	37	22	10	69
Refutation Rules	10	12	5	27
Total	**47**	**34**	**15**	**96**

of used rules and output explanations.

Input. The top part in Figure 5.3 contains the *input form* with three fields for the triple *query* and a text area for the *rules*. In the *query* fields, the user can select the subject and the object of the fact in question from a set of KG entities and specify the predicate, which might be out of KG. In the *rules* text area, the user is expected to input the rules using the standard logic programming syntax. For example, a rule expressing that "*a person is a citizen of a country if he was born in one of its cities*" is written as:

$$isCitizenOf(?x, ?y) \text{ :- } bornIn(?x, ?z),\ isA(?z, `City'),\ in(?z, ?y)$$

where $?x, ?y, ?z$ are variables, '*City*' is a constant, *isCitizenOf* is a head predicate and the rest are body predicates. These rules can also involve out-of-KG predicates, which Tracy spots in the text.

Output. Below the rules text area, Tracy returns the list of explanations for the input fact. Each *explanation* card contains:

- A set of facts that support the correctness of the query;
- In case the explanation includes facts with textual evidence, the user can view top-5 matching sentences with the links to the Web-pages where they have been found;
- The set of rules used by Tracy to find the evidence;
- Explanation confidence score computed as described in Equation 5.5.1 and some execution insights (*e.g.*, resources accesses).

Explanations can be sorted based on their generation order, quality, length, rewriting depth, or cost.

5.6 Related work

Fact-checking. Starting with TruthFinder [Yin et al., 2008] and T-Verifier [Li et al., 2011], text-based fact-checking has become an established research field [Li et al., 2016].

The majority of the state-of-the-art methods, *e.g.*, [Li et al., 2014, Mukherjee et al., 2014, Nakashole and Mitchell, 2014, Dong et al., 2015, Gerber et al., 2015, Popat et al., 2017], perform joint estimation of the fact's truth value based on the credibility of the fact candidates syntactically spotted in text and the trustworthiness of the underlying sources. These approaches, unlike ours are purely syntactic, *i.e.*, they ignore domain background knowledge often vital for inferring the fact in question.

A query language to assess the validity of claims in multiple contexts over structured information sources has been proposed in [Leblay, 2017]. Other works, such as [Ciampaglia et al., 2015, Shi and Weninger, 2016, Aggarwal et al., 2016, Shiralkar et al., 2017], estimate the support for factual statements by mining and ranking connectivity patterns in KGs. In contrast to ours, all of these approaches ignore textual sources. The inclusion of textual sources make the execution of bottom-up procedures (like the well-known chase in [Leblay, 2017]) problematic, as they require the execution of fact-spotting for any possible fact, which is unfeasible due to the high cost of this procedure. This problem has been circumvented by our top-down approach. The same holds for less related works that explain semantic connections among given KG entities (*e.g.*, [Arenas et al., 2016, Fionda and Pirrò, 2017]).

Checking facts jointly over KGs and textual resources has been explored in the context of natural language question answering in, *e.g.*, [Das et al., 2017], where given a relation and an entity (*i.e.*, subject or object), the task of retrieving a set of other entities that form a true triple is solved using neural networks. Our work differs from [Das et al., 2017] in several aspects. First, instead of retrieving entities we are given a fact to be checked. Second, we do not merely estimate the truthfulness of the fact, but also output human-readable semantic traces that support or refute it, while the output of [Das et al., 2017] cannot be explained to humans.

Inductive and deductive Reasoning over KGs. Mining rules for KG completion is orthogonal to our work. Indeed, instead of inducing rules from KGs, we are rather interested in exploiting them effectively for fact-checking over both structured and unstructured (textual) resources.

Logical query rewriting is well-studied in databases, *e.g.*, for the Datalog language. This line of research (see [Ceri et al., 1989] for an overview) focused on structured databases, though. In contrast, we consider also external sources in text form, which introduces additional challenges and requires optimizations. Deductive reasoning over multiple external sources has been explored, *e.g.*, in [Eiter et al., 2017]. However, joint query rewriting over structured and textual sources has not been studied in this context.

5.7 Conclusion

Prior methods for automatic fact-checking focused on producing final truthfulness scores, which are hard to interpret for humans. Moreover, existing approaches spot solely explicit occurrences of facts in text. In this chapter, we moved forward towards deriving more human understandable evidence based on background knowledge in the form of rules. ExFaKT utilizes these rules to collect pieces of evidence from both KGs and textual sources. The conducted experiments demonstrate the usefulness of our method for supporting human curators in making accurate decisions about the truthfulness of facts as well as the potential of our explanations for improving automated fact-checking systems. We also, provided Tracy, a demonstration tool that allows the user to observe the results of changing the ruleset and other parameters to experience the performance of ExFaKT in different scenarios.

Regarding the future work, it is interesting to study how more complex rules, *e.g.*, with negations, can be included. While such rules are useful to represent additional background knowledge, spotting negative mentions of facts in the text is challenging, and requires additional advances in the syntactic *textspot* procedure. Another important research stream concerns with further optimizations of our rewriting algorithm. This includes adaptations of intelligent query rewriting plans [Ceri et al., 1989] to account for unstructured textual resources and application of other advanced optimization strategies to our scenario of interest. Finally, methods for summarizing our explanations into a homogeneous piece of text can further facilitate the assessment by humans.

Chapter 6

ExCut: Explainable Clustering

Clustering entities over knowledge graphs (KGs) is an asset for explorative search and knowledge discovery. KG embedding methods have been intensively investigated, mostly for KG completion, and have the potential for facilitating entity clustering. However, KG embedding models are latent and do not convey user-interpretable labels for clusters.

In this chapter, we present ExCut, a novel approach that combines KG embedding with rule mining methods to compute informative clusters of entities and comprehensible explanations. The explanations are in the form of concise combinations of entity relations. ExCut jointly enhances the quality of entity clusters and their explanations in an iterative manner that interleaves the learning of embeddings and rules. Experiments on real-world KGs demonstrate the effectiveness of our approach for discovering high-quality clusters and their explanations.

6.1 Introduction

Motivation. Knowledge graphs (KGs) became essential resources for important tasks such as entity search, question answering and text analytics, by providing rich repositories of typed entities and associated properties. For example, *Tedros Adhanom* is known as a health expert, director of the World Health Organization (*WHO*), alumni of the University of London, and many more.

KGs can support analysts in exploring sets of interrelated entities and discovering interesting structures. This can be facilitated by **entity clustering**, using unsupervised methods for grouping entities into informative subsets. Consider, for example, an analyst or journalist who works on a large corpus of topically relevant documents, say on the Coronavirus crisis. Assume that key entities in this collection have been spotted and linked to the KG already. Then the KG can guide the user in understanding what kinds of entities are most relevant. With thousands of input entities, from health experts, geo-

locations, political decision-makers all the way to diseases, drugs, and vaccinations, the user is likely overwhelmed and would appreciate a group-wise organization. This task of computing entity clusters [Lisi, 2006, Fanizzi et al., 2008, Dumancic and Blockeel, 2017] is the problem we address.

Merely clustering the entity set is insufficient, though. The user also needs to understand the nature of each cluster. In other words, clusters must be explainable, in the form of **user-comprehensible labels**. As entities have types in the KG, an obvious solution is to label each cluster with its prevalent entity type. However, some KGs have only coarse-grained types and labels like "people" or "diseases" cannot distinguish health experts from politicians or virus diseases from bacterial infections. Switching to fine-grained types, such as Wikipedia categories, on the other hand, causes the opposite problem: each entity is associated with tens or hundreds of types, and it is unclear which of these would be a good cluster label. The same holds for an approach where common triple properties (*e.g.*, *educatedIn UK*) are considered as labels. Moreover, once we switch from a single KG to a set of linked open data (LOD) sources as a joint entity repository, the situation becomes even more difficult.

Problem. Given a large set of entities, each with a substantial set of KG properties in the form of categorical values or relations to other entities, our problem is to jointly tackle:

- **Clustering:** group the entities into k clusters of semantically similar entities;
- **Explanation:** generate a user-comprehensible concise labels for the clusters, based on the entity relations to other entities.

State-of-the-art and its limitations. The problem of clustering relational data is traditionally known as conceptual clustering (see, *e.g.*, [Suárez et al., 2019] for overview). Recently, it has been adapted to KGs in the Semantic Web community [Lisi, 2006, Fanizzi et al., 2008]. Existing approaches aim at clustering graph-structured data itself by, *e.g.*, introducing novel notions of distance and similarity directly on the KG [Dumancic and Blockeel, 2017, Dumancic et al., 2018]. Due to the complexity of the data, finding such universally good similarity notions is challenging [Dumancic et al., 2018].

Moreover, existing relational learning approaches are not sufficiently scalable to handle large KGs with millions of facts, *e.g.*, YAGO [Suchanek et al., 2007] and Wikidata [Vrandecic and Krötzsch, 2014]. Clustering entities represented in latent space, *e.g.*, [Idahl et al., 2019, Wang et al., 2019a], helps to overcome this challenge, yet, the resulting clusters are lacking explanations, clustering process is prone to the embedding quality, and hyperparameters are hard to tune [Dumancic et al., 2018]. Explaining clusters over

KGs, such as [Tiddi et al., 2014, Tiddi et al., 2015] focus on the discovery of explanations for given perfect clusters. However, obtaining such high-quality clusters in practice is not straightforward.

Approach. To address the above shortcomings, we present **ExCut**, a new method for computing explainable clusters of large sets of entities. The method uses KG embedding as a signal for finding plausible entity clusters, and combines it with logical rule mining, over the available set of properties, to learn interpretable labels. The labels take the form of concise conjunctions of relations that characterize the majority of entities in a cluster. For example, for the above Coronavirus scenario, we aim at mining such labels as $worksFor(X, Y) \wedge type(Y, health\_org) \wedge hasDegreeIn(X, life\_sciences)$ for a cluster of health experts, $type(X, disease) \wedge causedBy(X, Y) \wedge type(Y, virus)$ for a cluster of virus diseases, and more. A key point in our approach is that these labels can in turn inform the entity embeddings, as they add salient information. Therefore, we interleave and iterate the computation of embeddings and rule mining adapting the embeddings using as feedback the information inferred by the learned rules.

Contributions. The main contributions in this chapter are:

- We introduce ExCut, a novel approach for computing explainable clusters, which combines embedding-based clustering with symbolic rule learning to generate explanations for the resulting clusters in human-understandable format. These explanations can also serve as new types for entities.
- We propose several strategies to iteratively fine-tune the embedding model to maximize the explainability and accuracy of the discovered clusters based on the feedback from the learned explanations.
- We evaluate ExCut on real-world KGs. In many cases, it out-performs state-of-the-art methods with respect to both clustering and explanations quality.
- We provide preliminary results for using explainable clustering to compare KG embedding models with respect to their ability to capture complex concepts.

6.2 Computing Explainable Clusters

Given a KG, a subset of its entities and an integer k, our goal is to find a "good" split of entities into k clusters and compute explanations for the constructed groups that would serve as informative cluster labels. For example, consider the KG in Figure 6.1, the set of target entities $\{e_1, \ldots, e_6\}$ and the integer $k = 2$. One of the possible solutions

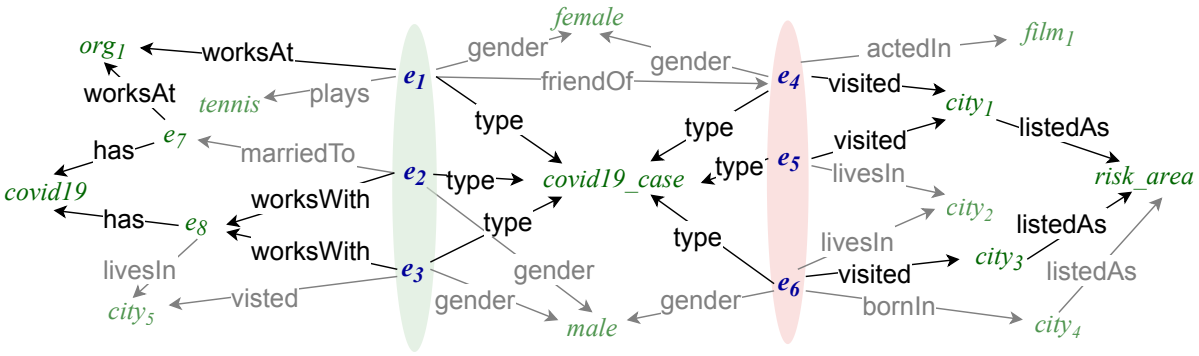


Figure 6.1: An example KG with potential *covid19* cases split into two entity clusters (in green and red). Black edges are relevant for the potential explanations of these clusters.

is to put e_{1-3} into the first cluster C_1 and the other three entities into the second one C_2. Explanations for this split would be that C_1 includes those who got infected via interacting with their coworkers, while the others were infected after visiting a risk area. Obviously, in general there are many other splits and identifying the criteria for the best ones is challenging.

Formally, we define the problem of *computing explainable entity clusters* as follows:

Definition 6.1 (Computing Explainable Entity Clusters Problem)**.**
Given:

- *a knowledge graph* $\mathcal{G}$ *over* $\Sigma_{\mathcal{G}} = \langle \mathcal{R}, \mathcal{C} \rangle$*;*
- *a set* $T \subseteq \mathcal{C}$ *of target entities;*
- *a number of desired clusters* $k > 1$*;*
- *an explanation language* L*; and*
- *an explanation evaluation function* $d : 2^L \times 2^T \times \mathcal{G} \to [0..1]$

Find: *a split* $Cls = \{C_1, \ldots C_k\}$ *of entities in* T *into* k *clusters and a set of explanations* $\mathbf{E} = \{r_1, \ldots, r_k\}$ *for them, where* $r_i \in L$*, such that* $d(\mathbf{E}, Cls, \mathcal{G})$ *is maximal.*

6.2.1 Explanation Language

Explanations (*i.e.*, informative labels) for clusters can be characterized as conjunctions of common entity properties in a given cluster; thus, Horn rules are sufficient for representing the explanations. Specifically, our explanation language relies on (cluster) explanation rules defined as follows:

Definition 6.2 (Cluster Explanation Rules)**.** *Let $\mathcal{G}$ be a KG with the signature $\Sigma_{\mathcal{G}} = \langle \mathcal{R}, \mathcal{C} \rangle$, let $C \subseteq \mathcal{C}$ be a subset of entities in $\mathcal{G}$, i.e., a cluster, and $\mathbf{X}$ a set of variables. A* (cluster) explanation rule *r for C over $\mathcal{G}$ is of the form*

$$r : \ belongsTo(X, e_C) \leftarrow p_1(\vec{X_1}), \ldots, p_m(\vec{X_m}), \tag{6.1}$$

where $e_C \notin \mathcal{C}$ is a fresh unique entity representing the cluster C, $belongsTo \notin \mathcal{R}$ is a fresh predicate, and $body(r)$ is a finite set of atoms over $\mathcal{R}$ and $\mathbf{X} \cup \mathcal{C}$.

Example 6.1. A possible explanation rule for $C_1 = \{e_1, e_2, e_3\}$ in $\mathcal{G}$ from Figure 6.1 is

$$r : belongsTo(X, e_{C_1}) \leftarrow worksWith(X, Y), has(Y, covid19)$$

which describes C_1 as a set of people working with infected colleagues.

Out of all possible cluster explanation rules we naturally prefer **succinct** ones. Therefore, we put further restrictions on the explanation language L by limiting the number of rule body atoms (an adjustable parameter in our method).

6.2.2 Evaluation Function

The function d from Definition 6.1 compares solutions to the problem of explainable entity clustering with respect to their quality, and ideally d should satisfy the following two criteria:

(i) **Coverage:** Given two explanation rules for a cluster, the one covering more entities should be preferred and
(ii) **Exclusiveness:** Explanation rules for different clusters should be (approximately) mutually exclusive.

The coverage measure from data mining is a natural choice for satisfying (i).

Definition 6.3 (Explanation Rule Coverage)**.** *Let $\mathcal{G}$ be a KG, C a cluster of entities, and r a cluster explanation rule. The* coverage *of r on C with respect to $\mathcal{G}$ is*

$$cover(r, C, \mathcal{G}) = \frac{|\{c \in C | r \models_{\mathcal{G}} belongsTo(c, e_C)\}|}{|C|} \tag{6.2}$$

Example 6.2. Consider clusters $C_1 = \{e_1, e_2, e_3\}$, $C_2 = \{e_4, e_5, e_6\}$ shown in Figure 6.1. The set of potential cluster explanation rules along with their coverage scores for C_1 and C_2 respectively, is given as follows:

		$i=1$	$i=2$
$r_1 : belongsTo(X, e_{C_i})$	$\leftarrow type(X, covid19\_case)$	1	1
$r_2 : belongsTo(X, e_{C_i})$	$\leftarrow gender(X, male)$	0.67	0.33
$r_3 : belongsTo(X, e_{C_i})$	$\leftarrow worksWith(X, Y), has(Y, covid19)$	0.67	0
$r_4 : belongsTo(X, e_{C_i})$	$\leftarrow visited(X, Y), listedAs(Y, risk\_area)$	0	1

While addressing (i), the coverage measure does not account for the criteria (ii). Indeed, high coverage of a rule for a given cluster does not imply a low value of this measure for other clusters. For instance, r_1 in Example 6.2 is too general, as it perfectly covers entities from both clusters. This motivates us to favour (approximately) mutually exclusive explanation rules, *i.e.*, explanation rules with high coverage for a given cluster but low coverage for others (similar to [Knobbe and Ho, 2006]). To capture this intuition, we define the *exclusive explanation coverage* of a rule for a cluster given other clusters as follows.

Definition 6.4 (Exclusive Explanation Rule Coverage)**.** *Let $\mathcal{G}$ be a KG, let Cls be a set of all clusters of interest, $C \in Cls$ a cluster, and r an explanation rule. The* exclusive explanation rule coverage *of r for C with respect to Cls and $\mathcal{G}$ is defined as*

$$exc(r, C, Cls, \mathcal{G}) = \begin{cases} 0, \ \textit{if} \min\limits_{C' \in Cls \setminus C} \{cover(r, C, \mathcal{G}) - cover(r, C', \mathcal{G})\} \leq 0 \\ cover(r, C, \mathcal{G}) - \frac{\sum\limits_{C' \in Cls \setminus C} cover(r, C', \mathcal{G})}{|Cls \setminus C|}, \ \textit{otherwise.} \end{cases} \tag{6.3}$$

Example 6.3. Consider $Cls = \{C_1, C_2\}, \mathbf{E} = \{r_1, r_2, r_3, r_4\}$ from Example 6.2 and the KG $\mathcal{G}$ from Figure 6.1. We have $exc(r_1, C_1, Cls, \mathcal{G}) = exc(r_1, C_2, Cls, \mathcal{G}) = 0$, which disqualifies r_1 as an explanation for either of the clusters. For r_2, we have $exc(r_2, C_1, Cls, \mathcal{G}) = 0.34$ making it less suitable for the cluster C_1 than r_3 which has $exc(r_3, C_1, Cls, \mathcal{G}) = 0.67$. Finally, r_4 perfectly explains C_2, since $exc(r_4, C_2, Cls, \mathcal{G}) = 1$.

Similarly, we can measure the *quality* of a collection of clusters with their explanations by averaging their per-cluster exclusive explanation rule coverage.

Definition 6.5 (Quality of Explainable Clusters)**.** *Let $\mathcal{G}$ be a KG, $Cls = \{C_1, \ldots, C_k\}$ a set of entity clusters, and $\mathbf{E} = \{r_1, \ldots, r_k\}$ a set of cluster explanation rules, where each r_i is an explanation for C_i, $1 \leq i \leq k$. The* explainable clustering quality *q of $\mathbf{E}$ for Cls with respect to $\mathcal{G}$ is defined as follows:*

$$q(\mathbf{E}, Cls, \mathcal{G}) = \frac{1}{|Cls|} \sum_{i=1}^{|Cls|} exc(r_i, C_i, Cls, \mathcal{G}) \tag{6.4}$$

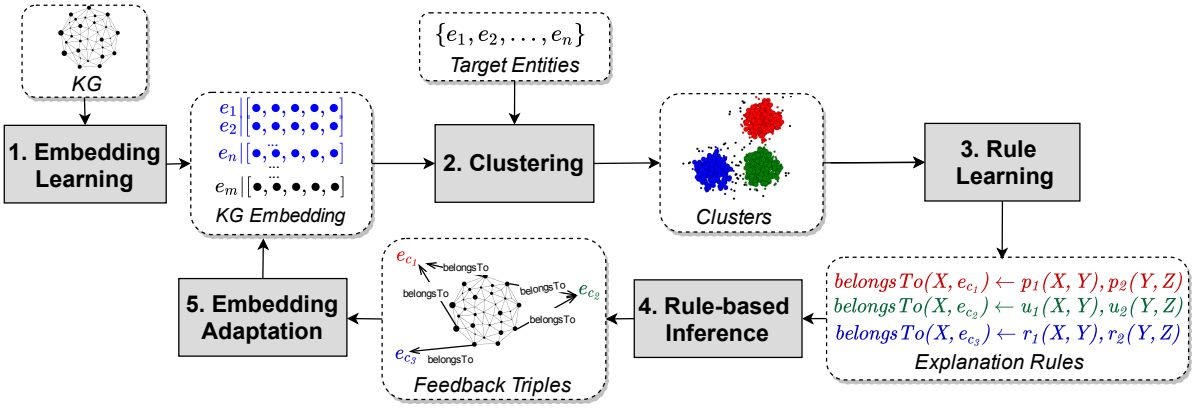


Figure 6.2: ExCut pipeline overview

Realizing the function d in Definition 6.1 by the above measure allows us to conveniently compare the solutions of the explainable clusters discovery problem.

Example 6.4. Consider $\mathcal{G}$ from Figure 6.1, the set of target entities $T = \{e_1, \ldots, e_6\}$, $k = 2$, language L of cluster explanation rules with at most 2 body atoms, and the evaluation function d given as q from Definition 6.5. The best solution to the respective problem of computing explainable entity clusters is $Cls = \{C_1, C_2\}$, $\mathbf{E} = \{r_3, r_4\}$, where C_1, C_2, r_3, r_4 are from Example 6.2. We have that $q(\mathbf{E}, Cls, \mathcal{G})$=0.83.

6.3 Method

We now present our method ExCut, which iteratively utilizes KG *Embedding-based Clustering* and *Rule Learning* to compute explainable clusters. More specifically, as shown in Figure 6.2, ExCut starts with (1) *Embedding Learning* for a given KG. Then, it performs (2) *Clustering* of the entities in the target set over the learned embeddings. Afterwards, (3) *Rule Learning* is utilized to induce explanation rules for the constructed clusters, which are ranked based on the exclusive coverage measure. Using the learned explanation rules, we perform (4) *Rule-based Inference* to deduce new entity-cluster assignment triples reflecting the learned structural similarities among the target entities. Then, ExCut uses the rules and the inferred assignment triples in constructing feedback to guide the clustering in the subsequent iterations. We achieve that by fine-tuning the embeddings of the target entities in Step (5) *Embedding Adaptation*.

In what follows we present the detailed description of ExCut's components.

6.3.1 Embedding Learning and Clustering

Embedding learning. ExCut starts with learning vector representations of entities and relations. We adopt KG embeddings in this first step, as they are well-known for their ability to capture semantic similarities among entities, and thus could be suited for defining a robust similarity function for *clustering relational data* [Dumancic et al., 2018]. Embeddings are also effective for dealing with data incompleteness, *e.g.*, predicting the potentially missing fact $worksWith(e_1, e_7)$ in Figure 6.1. Moreover, embeddings facilitate the inclusion of unstructured external sources during training, *e.g.*, textual entity descriptions [Xie et al., 2016b].

Conceptually, any embedding method can be used in our approach. We experimented with TransE [Bordes et al., 2013] and ComplEx [Trouillon et al., 2016] as prominent representatives of translation-based and linear map embeddings. To account for the context surrounding the target entities, we train embeddings using the whole KG.

Clustering. The *Clustering* step takes as input the trained embedding vectors of the target entities and the number k of clusters to be constructed. We perform clustering relying on the embeddings as features to compute pairwise distances among the target entities using standard distance functions, *e.g.*, *cosine distance*. Various classical clustering approaches or more complex embedding-driven clustering techniques [Wang et al., 2019a] could be exploited here too. In this chapter, we mainly rely on the traditional *Kmeans* method [MacQueen, 1967] as a proof of concept.

For KGs with types, the majority of embedding models [Bordes et al., 2013, Trouillon et al., 2016] would map entities of a certain type to similar vectors [Wang et al., 2019a]. For example, e_1 and e_2 in Figure 6.3.A are likely to be close to each other in the embedding space, and thus have a high chance of being clustered together. An ideal embedding model for explainable clustering should follow the same intuition even if types in the KG are missing. In other words, it should be capable of assigning similar vectors to entities that belong to structurally similar subgraphs of certain pre-specified complexity. For instance, in Figure 6.3.B, both e_1 and e_2 belong to subgraphs reflecting that these entities are married to politicians with some *covid19* symptom, and hence should be mapped to similar vectors.

Despite certain attempts to consider specific graph patterns (*e.g.*, [Lin et al., 2015]), to the best of our knowledge none of the existing embedding models is general enough to capture patterns of arbitrary complexity. We propose to tackle this limitation (see Section 6.3.3) by passing to the embedding model feedback created using cluster explanation rules learned in the Step 3 of ExCut.

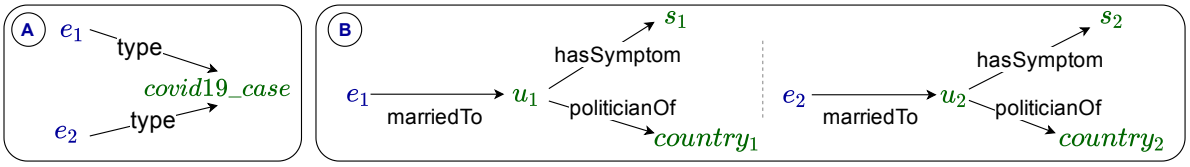


Figure 6.3: KG fragments

6.3.2 Explanation Mining

KG-based explanations. KG embeddings and the respective clusters constructed in Steps 1 and 2 of our method are not interpretable. However, since KG embeddings are expected to preserve semantic similarities among entities, the clusters in the embedding space should intuitively have some meaning. Motivated by this, in ExCut, we aim at decoding these similarities by learning rules over the KG extended by the facts that reflect the cluster assignments computed in the *Clustering* step.

Rule learning procedure. After augmenting $\mathcal{G}$ with the facts $belongsTo(e, e_{C_i})$ for all entities e clustered in C_i, we learn Horn rules of the form (6.1) from Definition 6.2. Following [Galárraga et al., 2015], we model rules as sequences of atoms, where the first atom is the head of the rule (*i.e.*, $belongsTo(X, e_{C_i})$ with C_i being the cluster to be explained), and other atoms form the rule's body.

For each cluster C_i, we maintain an independent queue of intermediate rules, initialized with a single rule atom $belongsTo(X, e_{C_i})$, and then exploit an iterative breadth-first search strategy. At every iteration, we expand the existing rules in the queue using the following *refinement operators*:

(i) *add a positive dangling atom*: add a binary positive atom with one fresh variable and another variable appearing in the rule, *i.e.*, *shared variable* , *e.g.*, adding $worksAt(X, Y)$, where Y is a fresh variable not appearing in the current rule;

(ii) *add a positive instantiated atom*: add a positive atom with one argument being a constant and the other one a shared variable , *e.g.*, adding $locatedIn(X, usa)$, where usa is a constant, and X appears elsewhere in the rule constructed so far.

These operators produce a set of new rule candidates, which are then filtered relying on the given explanation language L. Suitable rules with a minimum coverage of 0.5, *i.e.*, rules covering the majority of the entities within the respective cluster, are added to the output set. We refine the rules until the maximum length specified in the language bias is reached. Finally, we rank the constructed rules based on the *exclusive explanation coverage* (Definition 6.4), and select the top m rules for each cluster.

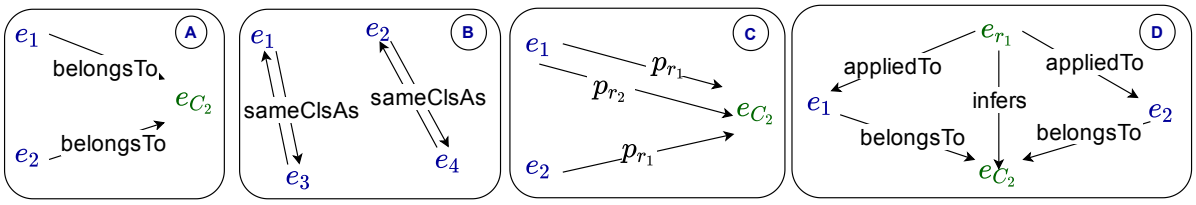


Figure 6.4: Inferred clusters assignment triples modeling options

Example 6.5. Assume that for $\mathcal{G}$ in Figure 6.1, and $T = \{e_1, \ldots, e_6\}$, the embedding-based clustering resulted in the following clusters $C_1 = \{e_1, e_2, \underline{e_4}\}$ and $C_2 = \{e_5, e_6, \underline{e_3}\}$, where e_4 and e_3 are incorrectly placed in wrong clusters. The top cluster explanation rules for C_2 ranked based on *exc* measure from Definition. 6.4 are:

	$exc(r_i, C_2)$
$r_1 : belongsTo(X, e_{C_2}) \leftarrow visited(X, Y)$	0.67
$r_2 : belongsTo(X, e_{C_2}) \leftarrow gender(X, male)$	0.33
$r_3 : belongsTo(X, e_{C_2}) \leftarrow visited(X, Y), listedAs(Y, risk\_area).$	0.33

Inferring entity-clusters assignments. In the *Rule-based Inference* (Step 4 in Figure 6.2), we apply the top-m rules obtained in the *Rule Learning* step on the KG to predict the assignments between the target entities and the discovered clusters over *belongsTo* relation using standard deductive reasoning techniques. The computed assignment triples are ranked and filtered based on the *exc* score of the respective rules that inferred them.

Example 6.6. Application of the rules from Example 6.5 on $\mathcal{G}$ with respect to the target entities e_{1-6} results in the cluster assignment triples:

$$\{belongsTo(e_3, e_{C_2}), belongsTo(e_4, e_{C_2}), belongsTo(e_2, e_{C_2})\}.$$

Note that based on r_1, entity e_4 is now assigned to C_2 instead of C_1.

6.3.3 Embedding Adaptation

Learned explanation rules capture explicit structural similarities among the target entities. We propose to utilize them to create feedback to guide the embedding-based clustering towards better explainable clusters. This feedback is passed to the embedding model in the form of additional training triples reflecting the assignments inferred by the learned rules. Our intuition is that such added triples should potentially help

other similarities of analogous nature to be discovered by the embeddings, compensating for the embedding-based clustering limitation discussed in Section 6.3.1.

Specifically, the embedding adaptation (Step 5 in Figure 6.2) is summarized as follows:

- From the *Rule Learning* and *Rule-based Inference* steps, described above, we obtain a set of *cluster assignment triples* of the form $belongsTo(e, e_C)$ together with rules inferring them, where e is an entity in the input KG $\mathcal{G}$ and e_C is a new entity uniquely representing the cluster C.
- We then model the cluster assignments from (a) and rules that produce them using one of our four strategies described below and store the results in $\mathcal{G}^{inf}$.
- A subset $\mathcal{G}^{context}$ of $\mathcal{G}$ consisting of triples that surround the target entities is then constructed.
- Finally, we fine-tune the embedding model by training it further on the data compiled from $\mathcal{G}^{inf}$ and $\mathcal{G}^{context}$.

Modeling rule-based feedback. Determining the adequate structure and amount of training triples required for fine-tuning the embedding model is challenging. On the one hand, the training data should be rich enough to reflect the learned structure, but on the other hand, it should not corrupt the current embedding. We now present our proposed four strategies for representing the inferred cluster-assignments along with the corresponding rules as a set of triples $\mathcal{G}^{inf}$ suitable for adapting the embedding. The strategies are listed in the ascending order of their complexity.

- **Direct**: As a straightforward strategy, we directly use the inferred entity-cluster assignment triples in $\mathcal{G}^{inf}$ as shown in Figure 6.4.A, *e.g.*, $belongsTo(e_1, e_{C_2})$.

- **Same-cluster-as**: In the second strategy, we model the inferred assignments as edges only. As shown in Figure 6.4.B, we compile $\mathcal{G}^{inf}$ using triples of *sameClsAs* relations between every pair of entities belonging to the same cluster as the learned rules suggest, *e.g.*, $sameClsAs(e_1, e_2)$. Modeling the cluster assignments using fresh relations allows us to stress the updates related to the target entities, as no extra entities are added to the KG in this strategy.

- **Rules as edges**: Third, we propose to model the inferred assignments together with the rules which led to their prediction. More precisely, for every rule r which deduced the fact $belongsTo(e, e_{C_i})$, we introduce a fresh predicate p_r and add a triple $p_r(e, e_{C_i})$ to the training set $\mathcal{G}^{inf}$, as illustrated in Figure 6.4.C. This allows us to encode all conflicting entity-cluster assignments (*i.e.*, assignments, in which an entity belongs to

two different clusters) and supply the embedding model with richer evidence about the rules that predicted these assignments.

- **Rules as entities**: Rules used in the deduction process can also be modeled as entities. In the fourth strategy, we exploit this possibility by introducing additional predicates *infers* and *appliedTo*, and for every rule r a fresh entity e_r. Here, each $belongsTo(e, e_{C_i})$ fact deduced by the rule r is modeled in $\mathcal{G}^{inf}$ with two triples $infers(e_r, e_{C_i})$ and $appliedTo(e_r, e)$ as shown in Figure 6.4.D.

Embedding fine-tuning. At every iteration i of ExCut, we start with the embedding vectors obtained in the previous iteration $i-1$ and train the embedding further with a set of adaptation triples $\mathcal{G}^{adapt}$. The set $\mathcal{G}^{adapt}$ is composed of the union of all $\mathcal{G}^{inf}_j$ for $j = 1 \dots i$ and a set of context triples $\mathcal{G}^{context}$. For $\mathcal{G}^{context}$, we only consider the triples that directly involve the target entities as a subject or object. For example, among the facts in the surrounding context of e_1, we have $worksAt(e_1, org_1)$ and $plays(e_1, tennis)$.

Our empirical studies showed that including assignment triples from previous iterations $j < i$ leads to better results; thus, we include them in $\mathcal{G}^{adapt}$, but distinguish entity and relation names from different iterations. Additionally, considering the context subgraph helps in regulating the change caused by the cluster assignment triples by preserving some of the characteristics of the original embeddings.

6.4 Evaluation

We evaluate the effectiveness of ExCut for computing explainable clusters. More specifically, we report the experimental results covering the following aspects: (i) the quality of the clusters produced by ExCut compared to existing clustering approaches; (ii) the quality of the computed cluster explanations; (iii) the usefulness and understandability of the explanations for humans based on a user study; (iv) the benefits of interleaving embedding and rule learning for enhancing the quality of the clusters and their explanations; and (v) the impact of using different embedding paradigms and our strategies for modeling the feedback from the rules.

Results overview. In seven out of eight datasets, ExCut outperforms the baselines with regard to the overall clustering and explanation quality metrics. Additionally, the quality of the computed explanations increases after just few iterations (≤ 10). Significant enhancements were observed on large-scale datasets. Above all, the conducted user study demonstrated that the generated explanations are elaborative descriptions for the discovered clusters. In the following, we discuss the conducted experiments in detail.

Table 6.1: Datasets statistics

	UWCSE	WebKB	Terror.	IMDB	Mutag.	Hepatitis	LUBM	YAGO
Target Entities	209	106	1293	268	230	500	2850	3900
Target Clusters	2	4	6	2	2	2	2	3
KG Entities	991	5906	1392	578	6196	6511	242558	4295825
Relations	12	7	4	4	14	19	22	38
Facts	2216	72464	17117	1231	30805	77585	2169451	12430700

Table 6.2: The predictive quality of the trained KG embedding models

	TransE				**ComplEx**			
	MRR	*Hit@1*	*Hit@3*	*Hit@10*	*MRR*	*Hit@1*	*Hit@3*	*Hit@10*
UWCSE	0.903	0.838	0.962	0.999	0.998	0.996	1.000	1.000
IMDB	0.981	0.965	0.999	0.999	1.000	1.000	1.000	1.000
Hepatitis	0.929	0.903	0.947	0.974	0.946	0.919	0.970	0.988
Mutagenesis	0.896	0.840	0.959	0.983	0.953	0.919	0.988	0.998
WebKB	0.210	0.158	0.223	0.293	0.415	0.329	0.441	0.584
Terrorist	0.320	0.140	0.429	0.623	0.930	0.876	0.984	0.998
YAGO-Art	0.105	0.085	0.111	0.133	-	-	-	-

6.4.1 Experiment Setup

ExCut Configurations. We implemented ExCut [1] and configured it as follows:

- *Embedding-based Clustering:* We extended the implementation of *TransE* and *ComplEx* provided by Ampligraph [Costabello et al., 2019] to allow embedding fine-tuning. We set the size of the embeddings to 100, and trained a base model with the whole KG for 100 epochs, using stochastic gradient descent with a learning rate of 0.0005. For fine-tuning, we trained the model for 25 epochs with a learning rate of 0.005. Table 6.2 reports the standard predictive quality of the trained embedding models: *Mean Reciprocal Rank (MRR)* and *Hit@k* measures [Rossi et al., 2020]. In these experiments, we use *Kmeans* for clustering as a proof of concept.
- *Rule Learning:* We implemented the algorithm described in Section 6.3.2. For experiments, we fix the language bias of the explanations to paths of length two, *e.g.*, $belongsTo(x, e_{C_i}) \leftarrow p(x, y), q(y, z)$, where z is either a free variable or bind to a constant.
- *Modeling Rule-based Feedback:* We experiment with the four strategies from Section 6.3.3: direct (*belongToCl*), same cluster as edges (*sameClAs*), rules as edges (*entExplCl*), and rules as entities (*followExpl*).

[1] ExCut is available at github.com/mhmgad/ExCut.

Table 6.3: Clustering results of ExCut compared to the baselines

	Methods	UWCSE			IMDB			Hepatitis		
		ACC	ARI	NMI	ACC	ARI	NMI	ACC	ARI	NMI
Baselines	ReCeNT	0.90	0.60	0.54	0.61	0.02	0.01	0.51	-0.01	0.01
	DEC	0.67	0.17	0.12	0.54	0.00	0.01	0.55	0.01	0.01
	Kmeans-T	0.91	0.66	0.51	0.58	0.03	0.08	0.51	0.00	0.00
	Kmeans-C	0.54	0.00	0.01	0.53	0.00	0.00	0.52	0.00	0.00
ExCut-T	belongToCl	0.99	0.96	0.92	**1.00**	1.00	1.00	**0.83**	0.43	0.35
	sameClAs	**1.00**	1.00	1.00	**1.00**	1.00	1.00	0.56	0.01	0.01
	entExplCl	**1.00**	1.00	1.00	**1.00**	1.00	1.00	0.82	0.41	0.33
	followExpl	**1.00**	1.00	1.00	**1.00**	1.00	1.00	0.82	0.41	0.33
Excut-C	belongToCl	0.96	0.85	0.77	**1.00**	1.00	1.00	0.63	0.07	0.05
	sameClAs	0.98	0.91	0.86	**1.00**	1.00	1.00	0.58	0.02	0.02
	entExplCl	0.97	0.88	0.81	0.65	0.08	0.19	0.69	0.15	0.11
	followExpl	0.99	0.97	0.94	**1.00**	1.00	1.00	0.66	0.10	0.08

	Methods	Mutagenesis			WebKB			Terrorist		
		ACC	ARI	NMI	ACC	ARI	NMI	ACC	ARI	NMI
Baselines	ReCeNT	**0.77**	0.30	0.24	**0.52**	0.00	-0.25	0.37	0.10	0.13
	DEC	0.51	0.00	0.00	0.31	0.03	0.05	0.37	0.16	0.26
	Kmeans-T	0.52	0.00	0.00	0.33	0.01	0.06	0.53	0.33	0.44
	Kmeans-C	0.73	0.21	0.18	0.49	0.21	0.34	0.51	0.23	0.28
ExCut-T	belongToCl	0.68	0.12	0.13	0.43	0.13	0.17	0.52	0.27	0.31
	sameClAs	0.65	0.08	0.08	0.36	0.06	0.08	0.35	0.03	0.06
	entExplCl	0.64	0.07	0.08	0.43	0.13	0.20	0.45	0.17	0.23
	followExpl	0.64	0.08	0.08	0.44	0.15	0.22	0.45	0.16	0.22
Excut-C	belongToCl	0.73	0.21	0.18	0.51	0.23	0.37	**0.54**	0.26	0.29
	sameClAs	0.73	0.21	0.18	0.38	0.08	0.17	0.34	0.03	0.08
	entExplCl	0.73	0.21	0.19	**0.52**	0.24	0.36	0.53	0.25	0.29
	followExpl	0.73	0.20	0.18	0.51	0.22	0.34	0.52	0.24	0.29

Datasets. We performed experiments on six datasets (Table 6.1) with a pre-specified set of target entities, which are widely used for relational clustering [Dumancic and Blockeel, 2017]. Additionally, we considered the following large-scale KGs:

- *LUBM-Courses*: a subset of entities from LUBM syntactic KG [Guo et al., 2005] describing the university domain, where target entities are distributed over *graduate* and *undergraduate courses*; and
- *YAGO-Artwork*: a set of target entities randomly selected from *YAGO* [Suchanek et al., 2007]. The entities are uniformly distributed over three types, *book*, *song*, and *movie*. To avoid trivial explanations, type triples for target entities were removed from the KG.

Table 6.4: Quality of Clusters Explanations by ExCut compared to the baselines

	Methods	UWCSE			IMDB			Hepatitis		
		Cov	Exc	WRA	Cov	Exc	WRA	Cov	Exc	WRA
Baselines	ReCeNT	0.91	0.88	0.14	1.00	0.04	0.01	1.00	0.00	0.00
	DEC	0.73	0.31	0.07	1.00	0.03	0.01	1.00	0.01	0.00
	Kmeans-T	0.83	0.76	0.16	0.74	0.11	0.01	0.81	0.09	0.02
	Kmeans-C	0.59	0.06	0.01	0.73	0.04	0.01	0.61	0.09	0.02
ExCut-T	belongToCl	0.89	0.89	**0.19**	1.00	1.00	**0.11**	0.76	0.64	0.13
	sameClAs	0.90	0.90	**0.19**	1.00	1.00	**0.11**	0.94	0.45	0.09
	entExplCl	0.90	0.90	**0.19**	1.00	1.00	**0.11**	0.75	0.64	0.13
	followExpl	0.90	0.90	**0.19**	1.00	1.00	**0.11**	0.75	0.63	0.13
ExCut-C	belongToCl	0.88	0.86	0.18	1.00	1.00	**0.11**	0.73	0.50	0.12
	sameClAs	0.91	0.89	**0.19**	1.00	1.00	**0.11**	0.80	0.45	0.11
	entExplCl	0.88	0.88	**0.19**	0.73	0.18	0.01	0.85	0.73	**0.18**
	followExpl	0.90	0.89	**0.19**	1.00	1.00	**0.11**	0.81	0.66	0.12
Ground truth		0.92	0.90	0.19	1.00	1.00	0.11	0.92	0.57	0.14

	Methods	Mutagenesis			WebKB			Terrorist		
		Cov	Exc	WRA	Cov	Exc	WRA	Cov	Exc	WRA
Baselines	ReCeNT	1.00	0.00	0.00	1.00	1.00	0.00	0.93	0.42	0.06
	DEC	1.00	0.00	0.00	1.00	0.06	0.01	0.60	0.13	0.02
	Kmeans-T	0.75	0.11	0.03	0.75	0.11	**0.03**	0.49	0.17	0.02
	Kmeans-C	0.87	0.30	0.08	0.98	0.04	0.01	0.64	0.28	0.02
ExCut-T	belongToCl	0.94	0.39	0.09	0.98	0.12	0.01	0.68	0.26	0.03
	sameClAs	0.96	0.50	**0.12**	0.99	0.04	0.01	0.87	0.49	0.06
	entExplCl	0.99	0.48	0.12	0.99	0.10	0.01	0.94	0.80	**0.11**
	followExpl	0.98	0.46	0.11	0.99	0.09	0.01	0.95	0.79	**0.11**
ExCut-C	belongToCl	0.87	0.31	0.08	0.98	0.08	0.01	0.68	0.32	0.02
	sameClAs	0.87	0.30	0.08	0.98	0.10	0.01	0.85	0.61	0.07
	entExplCl	0.87	0.31	0.08	0.97	0.08	0.01	0.68	0.33	0.03
	followExpl	0.87	0.31	0.08	0.97	0.07	0.01	0.67	0.30	0.03
Ground truth		1.00	0.16	0.04	1.00	0.04	0.01	0.64	0.33	0.03

Baselines. We compare ExCut to the following clustering methods:

- *ReCeNT* [Dumancic and Blockeel, 2017], a state-of-the-art relational clustering approach, that clusters entities based on a similarity score computed from entity neighborhood trees;
- *Deep Embedding Clustering (DEC)* [Xie et al., 2016a], an embedding-based clustering method that performs dimensionality reduction jointly with clustering and
- Standard *Kmeans* applied directly over embeddings: *TransE (Kmeans-T)* and *ComplEx (Kmeans-C)*. This baseline is equivalent to a single iteration of ExCut.

Machinery. All experiments were performed on a Linux machine with 80 cores and 500GB RAM. The average results over 5 runs are reported.

6.4.2 Clustering Quality

Metrics. We measure the clustering quality with respect to the ground truth with three standard metrics: *Accuracy (ACC)*, *Adjusted Rand Index (ARI)* [Steinley, 2004], and *Normalized Mutual Information (NMI)* [Manning et al., 2008] (the higher, the better). To assess the quality of the solution to the explainable clustering problem from Definition 6.1 found by ExCut, we compare the computed quality value to the quality of the explanations computed over the ground truth. To map the ground truth classes to the predicted clusters, we construct a confusion matrix, and hence, we match each cluster to one class in a greedy fashion, *i.e.*, class with highest intersections.

Results. Table 6.3 presents the quality of the clusters computed by the baselines, in the first 4 rows, followed by ExCut with the four feedback strategies, where *ExCut-T* and *ExCut-C* stand for ExCut with TransE and ComplEx respectively.

For all datasets except for *Mutagensis*, ExCut achieved, in general, better results with respect to the *ACC* value than the state-of-the-art methods. Furthermore, ExCut-T results in significantly better clusters on all datasets apart from *Terrorists* compared to Kmeans-T, *i.e.*, the direct application of *Kmeans* on the TransE embedding model. Since the *Terrorists* dataset contains several attributed predicates (*e.g.*, facts over numerical values), a different language bias for the explanation rules is required.

Our system managed to fully re-discover the ground truth clusters for the two datasets: *UWCSE* and *IMDB*. The accuracy enhancement by ExCut-T compared to the respective baseline (Kmeans-T) exceeds 30% for *IMDB* and *Hepatitis*. Other quality measurements indicate similar increments.

6.4.3 Explanation Quality

Metrics. The quality of the generated explanations is measured using the coverage metrics defined in Section 6.2.2, namely, *per cluster coverage (Cov)* and *exclusive coverage (Exc)*. In addition, we adapted the "novelty" metric *Weighted Relative Accuracy (WRA)* [Lavrac et al., 1999], representing a trade-off between the coverage and the accuracy of the discovered explanations. We compute the average of respective qualities of the top explanations for all clusters.

Results. Table 6.4 shows the average quality of the top explanations for the discovered clusters, where the average per cluster coverage (*Cov*) and exclusive coverage (*Exc*) are

Table 6.5: Quality of the clusters and the explanations found in Large-scale KGs

	Methods	LUBM Courses						YAGO Artwork					
		ACC	ARI	NMI	Cov	Exc	WRA	ACC	ARI	NMI	Cov	Exc	WRA
Bas.	DEC	0.92	0.70	0.66	0.96	0.95	0.19	0.56	0.44	0.57	0.92	0.49	0.11
	Kmeans-T	0.50	0.00	0.00	0.46	0.03	0.01	0.52	0.42	0.58	0.92	0.42	0.11
ExCut-T	belongToCl	1.00	1.00	1.00	1.00	1.00	0.25	0.82	0.63	0.59	0.85	0.70	0.16
	sameClAs	0.88	0.57	0.53	0.91	0.79	0.19	0.97	0.91	0.90	0.95	0.93	0.21
	entExplCl	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**0.25**	**0.97**	**0.92**	**0.91**	**0.95**	**0.93**	**0.21**
	followExpl	1.00	1.00	1.00	1.00	1.00	0.25	0.88	0.73	0.70	0.86	0.78	0.17
Ground truth		-	-	-	1.00	1.00	0.25	-	-	-	0.95	0.93	0.21

intrinsic evaluation metrics used as our optimization functions, while the *WRA* is the extrinsic measure.

The last row presents the quality of the learned explanations for the ground truth clusters; these values are not necessarily 1.0, as perfect explanations under the specified language bias may not exist. We report them as reference points.

ExCut enhances the average *Exc* and *WRA* scores of the clusters' explanations compared to the ones obtained by the baselines. These two measures highlight the exclusiveness of the explanations; making them more representative than *Cov*. Thus, the decrease in the *Cov*, as in *Terrorist*, is acceptable, given that it is in favor of increasing both *Exc* and *WRA*.

Similar to the clustering results, for *UWCSE* and *IMDB* our method achieved the explanations quality of the ground truth. For other datasets, our method obtained higher explanations quality than the respective baselines. This demonstrates the effectiveness of the proposed feedback mechanism in adapting the embedding model to better capture the graph structures in the input KGs.

6.4.4 Effectiveness on Large-scale KGs

Table 6.5 presents quality measures for clustering and explainability of ExCut running with TransE on *LUBM* and *YAGO*. ExCut succeeds to compute the ground truth clusters on *LUBM*. Despite the noise in *YAGO*, it achieves approximately 40% enhancement of the clustering accuracy. The explanation quality is also improved. ReCent did not scale on *LUBM* and *YAGO* due to memory requirements.

For illustration, Tables 6.6 present the *top-3* explanations for each cluster computed by ExCut along with their quality on the *YAGO* KG. In the ground truth, C_1, C_2, C_3 are clusters for entities of the type *Songs*, *Books*, and *Movies* respectively. One can observe that the explanations generated by ExCut-T (Table 6.7b) are more intuitive and of higher

Table 6.6: Explanations of clusters *song*, *book*, and *movie* from YAGO KG. ($\forall X \in C_i$)

(a) **Kmeans-T**

	Explanations	Cov	Exc	WRA
C_1	$created(Y, X)$, $bornIn(Y, Z)$	0.94	0.55	0.13
	$created(Y, X)$, $type(Y, artist)$	0.49	0.45	0.10
	$created(Y, X)$, $type(Y, writer)$	0.52	0.44	0.10
C_2	$directed(Y, X)$	0.92	0.56	0.11
	$directed(Y, X)$, $gender(Y, male)$	0.89	0.54	**0.10**
	$created(Y, X)$, $type(Y, person)$	0.71	0.52	**0.06**
C_3	$actedIn(Y, X)$, $type(Y, person)$	0.58	0.30	0.07
	$locatedIn(X, Y)$, $hasLang(Y, Z)$	0.60	0.29	0.07
	$locatedIn(X, Y)$, $currency(Y, Z)$	0.60	0.29	0.07

(b) **ExCut-T**

	Explanations	Cov	Exc	WRA
C_1	$created(Y, X)$, $type(Y, artist)$	0.99	0.96	**0.21**
	$created(Y, X)$, $won(Y, grammy)$	0.57	0.57	**0.12**
	$created(Y, X)$, $type(Y, person)$	0.84	0.48	**0.11**
C_2	$created(Y, X)$, $type(Y, writer)$	0.99	0.91	**0.19**
	$created(Y, X)$, $diedIn(Y, Z)$	0.46	0.20	0.04
	$created(Y, X)$	1.00	0.00	0.05
C_3	$actedIn(Y, X)$	0.81	0.81	**0.19**
	$actedIn(Y, X)$, $bornIn(Y, Z)$	0.79	0.79	**0.18**
	$actedIn(Y, X)$, $type(Y, person)$	0.78	0.78	**0.18**

quality than those obtained using Kmeans-T (Table 6.7a) . The correlation between the explanation relevance and the used quality metrics can also be observed.

6.4.5 Understanbility of the Clusters

To assess the human-understandability and usefulness of the explanation rules, we analyze whether ExCut explanations are the best fitting labels for the computed clusters based on the user opinion. The study was conducted on Amazon MTurk.

User study details. Based on the YAGO KG, we provided the participants with:

- Three clusters of entities, each represented with three entities pseudo-randomly selected from these clusters along with a brief summary for each entity, and a link to its Wikipedia page;

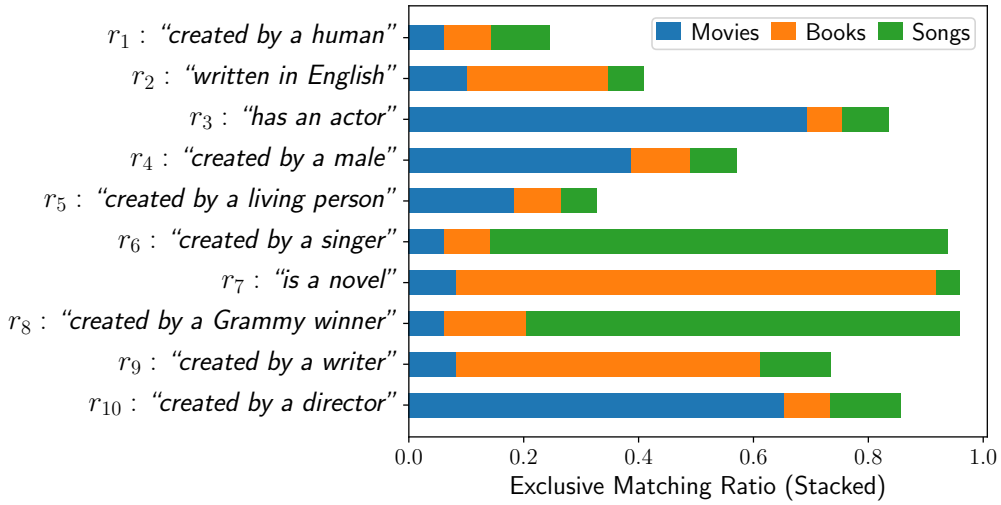


Figure 6.5: Ratio of explanation-to-cluster pairs exclusively matched

- A set of 10 potential explanations composed of the top explanations generated by ExCut and other explanations with high *Cov* but low *Exc*. Explanations were displayed in natural language for the ease of readability.

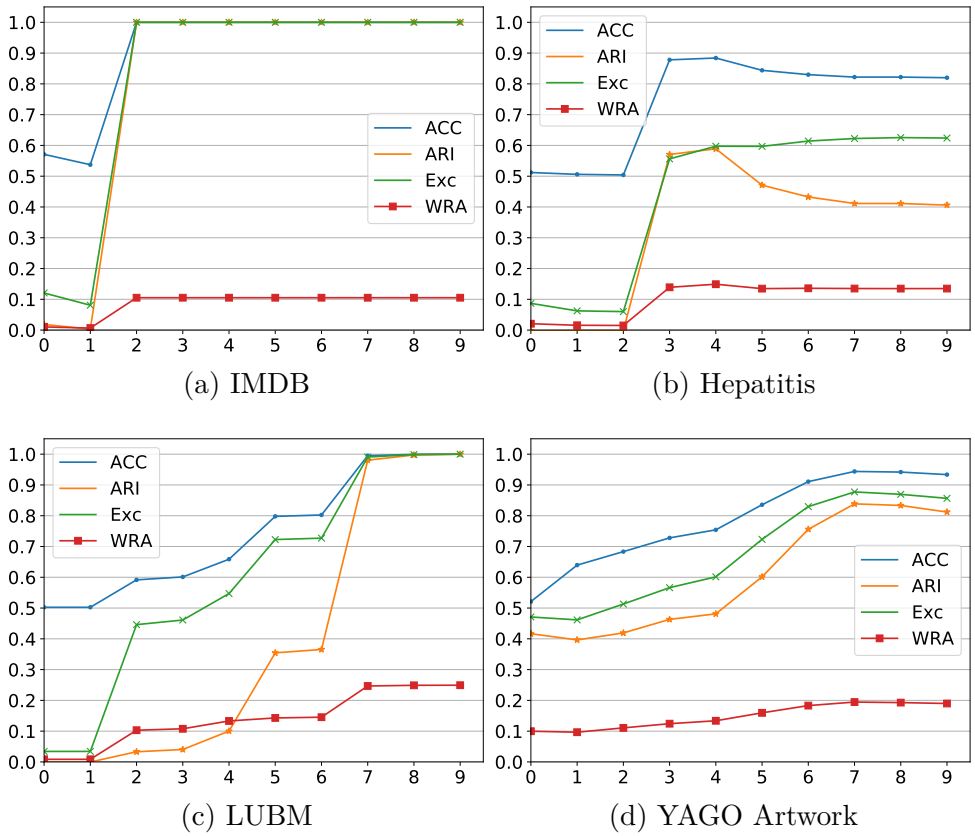
We asked the participants to match each explanation to all relevant clusters.

Metric. A *useful* explanation is the one that is *exclusively matched* to the correct cluster by the participants. To detect useful explanations, for every *explanation-cluster* pair, we compute the ratio of responses where the pair is *exclusively matched*. Let $match(r_i, c_m) = 1$ if the user *matched* explanation r_i to the cluster c_m (otherwise 0). Then, r_i is *exclusively matched* to c_m if additionally, $match(r_i, c_j) = 0$ for all $j \neq m$.

Results. Figure 6.5 summarizes the results of the 50 responses collected via the user-study. Each bar shows the ratio of responses exclusively matching explanation r_i to each of the provided clusters. The results show that the majority of the participants exclusively matched explanations r_3 and r_{10} to *movies*; r_7 and r_9 to *books*; and r_6 and r_8 to *songs*. The explanations r_3, r_6, and r_9 have been learned by ExCut. The high relative exclusive matching ratio to the corresponding correct cluster for the ExCut explanations demonstrates their usefulness in differentiating between the given clusters.

6.4.6 Effectiveness of ExCut Configurations

In Figure 6.6, we present a sample for the quality of the clusters and the aggregated quality of their top explanations over 10 iterations of ExCut-T using the *followExpl* configuration. In general, clustering and explanations qualities consistently improved



(a) IMDB (b) Hepatitis

(c) LUBM (d) YAGO Artwork

Figure 6.6: ExCut-T clustering and explanations quality over the iterations(x-axis)

over iterations, which demonstrates the advantage of the introduced embedding fine-tuning procedure. For *IMDB*, the qualities drop at the beginning, but increase and reach the highest values at the third iteration. This highlights the benefit of accumulating the auxiliary triples for enhancing the feedback signal, thus preventing the embedding tuning from diverging. The charts also show a correlation between the clustering and explanation quality, which proves our hypothesis that the introduced exclusive coverage measure (*Exc*) is useful for computing good clusters.

With respect to the effects of different embeddings and feedback modeling, as shown in Tables 6.3 and 6.4, we observe that ExCut with *TransE* is more robust than with *ComplEx* regardless of the feedback modeling method. Furthermore, modeling the feedback using *followExpl* strategy leads to better results on the majority of the datasets, especially for large-scale KGs. This reflects the benefit of passing richer feedback to the embedding, as it allows for better entity positioning in the latent space.

6.5 Use-case: Understanding KG embeddings

In this section, we report our preliminary attempt for utilizing explainable clustering in investigating the ability of well-known KG embedding models to capture complex concepts. We particularly examine whether prominent regions in the embedding space, constructed by existing KG embeddings, correspond to any conjunctive queries. Comparing embeddings with respect to their ability to capture such queries opens new perspectives for their applicability for various data mining tasks beyond link prediction, *e.g.*, *conceptual clustering* [Suárez et al., 2019] (see Section 2.1.3).

6.5.1 Pipeline

We use our proposed explainable clustering pipeline to discover prominent regions in the embedding space and learn explanations (*i.e.*, queries) for them. The higher the quality of these queries, the better is the model in capturing complex concepts. After obtaining the clusters, we apply the rule learning procedure described in Section 6.3.2 to discover the underlying queries. In this section, we focus on a single iteration of our method, *i.e.*, the Steps 4 and 5 in Figure 6.2 are omitted. Hence, the KG embedding is not modified.

Additionally, instead of employing the standard clustering method KMeans, we adapt the *Multicut* graph clustering approach [Chopra and Rao, 1993] as an effective clustering method, which does not require the number of clusters as input. Multicut method has previously not been applied in the context of KGs to the best of our knowledge, despite its apparent advantages compared to other algorithms, *e.g.*, the small number of parameters to be tuned.

In the following, we briefly describe applying *Multicut clustering* algorithm over KGs.

Multicut clustering over KG embeddings. First step to apply Multicut is to construct a weighted undirected graph providing the cutting costs on the edges. For that, we construct a complete graph $G = (V, E)$ over the target entities and compute pair-wise costs $\Phi\colon E \mapsto \mathbb{R}$ between entity pairs using the *normalized* cosine similarity of their embedding vectors.

A multicut of a graph is a subset of its edges such that no cycle in the graph intersects this subset exactly once. If we label edges in the multicut as 1, and all other edges as 0, the set of all valid multicuts can be formalized by the following set of linear inequalities:

$$Y_G = \left\{ y\colon E \mapsto \{0,1\} \mid \forall T \in \text{cycles}(G), \forall e \in T\colon y_e \leq \sum_{f \in T \setminus \{e\}} y_f \right\} \tag{6.5}$$

where y_e and y_f are labels of the edges e and f respectively obtained using the labeling

function y. Considering only chordless cycles is sufficient [Chopra and Rao, 1993], and any valid multicut $y \in Y_G$ uniquely defines a graph decomposition. Given the above definitions, we can formulate the **minimum cost multicut problem** as follows:

$$\min_{y \in Y_G} \sum_{e \in E} (\Phi_e + \beta)\, y_e \tag{6.6}$$

By solving (6.6) using efficient local search methods [Keuper et al., 2015], we find an optimal multicut and the respective optimal graph decomposition, which allows us to detect prominent regions in the embedding space without knowing their number even for large KGs. The cutting prior value $\beta \in \mathbb{R}$ can be tuned to discover more ($\beta < 0$) or less ($\beta > 0$) clusters, than given by the pair-wise costs Φ only.

6.5.2 Preliminary Experiments

We aim at: ***(Q1)*** comparing KG embeddings with respect to their suitability for conceptual clustering, ***(Q2)*** evaluating the correlation between the predictive quality of embeddings and their performance in conceptual clustering, and ***(Q3)*** verifying the effectiveness of the Multicut clustering algorithm in our context compared to other common clustering algorithms.

Embedding models. We experiment with *TransE (T)* and *ComplEx (C)*, which are respectively representatives of translation-based and bilinear map embedding models. Embeddings are trained on the whole KG as described in Section 6.4.1.

Clustering methods. We try several cutting prior values β for the Multicut algorithm and report the best results. Multicut results are compared against commonly used clustering algorithms; namely, DBSCAN [Ester et al., 1996], Kmeans [MacQueen, 1967], and Spectral clustering [Shi and Malik, 2000], whose parameters are tuned and the best results are reported. We pass the number of clusters produced by Multicut to Kmeans and Spectral clustering for a fair comparison, as both require this parameter as input.

Datasets. We uses the datasets introduced in Section 6.4; namely, Hepatitis, Mutagensis, WebKB, and Terrorist Attacks. In addition, we experiment with *YAGO-Artwork* as a large scale KG.

Results. Recalling the predictive quality of the KG embeddings reported in Table 6.2, ComplEx consistently achieves better results than TransE on all datasets, except YAGO. Training embeddings for this KG is particularly challenging for both models, and ComplEx could not converge within the training epochs due to its complexity.

In Table 6.8, we report the estimated quality of the discovered queries. For each

Table 6.8: Quality of the learned queries; "–" refers to the failure of finding clusters

Methods	Hepatitis				Mutagenesis				WebKB				Terrorist				YAGO-Art			
	cov	*exc*	*wra*	*cls*	*cov*	*exc*	*wra*	*cls*	*cov*	*exc*	*wra*	*cls*	*cov*	*exc*	*wra*	*cls*	*cov*	*exc*	*wra*	*cls*
Multicut-T	0.57	0.57	0.04	6	**0.92**	0.67	0.00	4	**0.95**	0.15	**0.01**	5	0.79	0.38	**0.05**	4	0.76	**0.7**	0.05	3
Multicut-C	**0.83**	**0.76**	**0.01**	4	0.77	0.63	**0.10**	2	0.85	**0.27**	0.00	2	**0.80**	**0.70**	**0.05**	3	0.82	0.21	0.00	6
DBSC.-T	0.70	0.58	0.03	3	–	–	–	–	–	–	–	–	0.76	0.42	0.01	2	**0.91**	0.58	**0.07**	4
DBSC.-C	0.73	0.62	0.04	2	**0.92**	**0.75**	0.02	5	–	–	–	–	0.75	0.56	0.01	3	–	–	–	–
Kmeans-T	0.61	0.49	0.07	6	0.79	0.04	0.01	4	0.97	0.02	0.01	5	0.62	0.23	0.06	4	0.67	0.40	0.10	3
Kmeans-C	0.66	0.56	0.11	4	0.94	0.88	0.22	2	0.95	0.11	0.03	2	0.77	0.52	0.10	3	0.75	0.01	0.00	6
Spectral-T	0.59	0.29	0.08	6	0.93	0.04	0.01	4	0.97	0.04	0.02	5	0.67	0.27	0.06	4	0.68	0.40	0.10	3
Spectral-C	0.76	0.62	0.06	4	1.00	0.48	0.01	2	0.92	0.43	0.00	2	0.84	0.75	0.05	3	0.81	0.01	0.00	6

Table 6.9: Example rules from YAGO-Art dataset learned over TransE

Query	**cov**	**exc**	**wra**
$r_1 : belongsTo(X, c_1) \leftarrow directed(Y, X), acted(Z, X)$	0.768	0.749	0.153
$r_2 : belongsTo(X, c_1) \leftarrow actedIn(Y, X), type(Y, film\_actor)$	0.724	0.708	0.144
$r_3 : belongsTo(X, c_2) \leftarrow created(Y, X), hasWonPrize(Y, Z)$	0.564	0.424	0.093
$r_4 : belongsTo(X, c_2) \leftarrow created(Y, X), type(Y, writer)$	0.504	0.420	0.091

dataset, we present the average *cov*, *exc*, *wra* measures described in Chapter 6, and the *number of discovered clusters (cls)*. Conceptual clustering over ComplEx results in better average *exc* and *wra*, which answers our first research question ***(Q1)***. In addition, the higher predictive quality of ComplEx in Table 6.2, supports the hypothesis ***(Q2)***, suggesting correlation between the predictive quality and the quality of the discovered queries. This also holds for YAGO, where TransE performs better than ComplEx in both prediction and clustering.

Regarding ***(Q3)***, Multicut achieved better results compared to DBSCAN for the majority of datasets. Moreover, DBSCAN failed in several cases regardless of the used parameters. Interestingly, even compared to other clustering algorithms that require the number of clusters, Multicut performed better on several datasets. This demonstrates the suitability of this algorithm for the KG domain.

Table 6.9 shows example rules mined from YAGO over TransE. For example, r_1 describes entities in c_1 as *"artifacts that have a director and an actor"*, while r_3 describes entities in c_2 as: *"artifacts created by an award winner"*.

In conclusion, we believe that preliminary experimental results contribute to a better understanding of the strengths and weaknesses of the existing KG embeddings beyond fact prediction. As a future future work, we plan to compare more embedding models, especially those trained on complex patterns [Ren et al., 2020].

6.6 Related Work

Clustering relational data has been actively studied (*e.g.*, [Dumancic and Blockeel, 2017, Fanizzi et al., 2008, Fonseca et al., 2011, Lisi, 2006, Suárez et al., 2019]). The majority of the existing approaches are based on finding interesting features in KGs and defining distance measures between their vectors. Our work is conceptually similar, but we let embedding model identify the features implicitly instead of computing them on the KG directly, in spirit of linked data propositionalization [Ristoski and Paulheim, 2014].

A framework for explaining given high-quality clusters using linked data and inductive logic programming has been proposed in [Tiddi et al., 2014, Tiddi et al., 2015]. While [Tiddi et al., 2015] aims at explaining existing clusters, we focus on performing clustering and explanation learning iteratively to discover high-quality clusters with explanations. The work [Idahl et al., 2019] targets interpreting embedding models by finding concept spaces in node embeddings and linking them to a simple external type hierarchy. This is different from our method of explaining clusters computed over embeddings by learning rules from a given KG. The authors of [Bouraoui and Schockaert, 2018] proposes a method for learning conceptual space representations of known concepts by associating a Gaussian distribution over a learned vector space with each concept. In [Hamilton et al., 2018, Ren et al., 2020] the authors introduce methods for answering logical queries over the embedding space. In contrast, in our work, the concepts are not given but rather need to be discovered.

While the step of explanation learning in our method is an adaptation of [Galárraga et al., 2015], the extension of other exact symbolic rule learning methods [Meilicke et al., 2019, Ortona et al., 2018] is likewise possible. In principle, one can also employ neural-based rule learners for our needs, such as [Yang et al., 2017, Omran et al., 2018, Ghiasnezhad Omran et al., 2019]; however the integration of our exclusive rule coverage scoring function into such approaches is challenging, and requires further careful investigation.

Several methods recently focused on combining [Zhang et al., 2019a] and comparing [Dumancic et al., 2018, Meilicke et al., 2018] rule learning and embedding methods. In Chapter 4, we propose to rank rules learned from KGs by relying both on their embedding-based predictive quality and traditional rule measures. Along similar lines, [Zhang et al., 2019a] introduces an iterative method for joint learning of linear-map embeddings and OWL axioms (without nominals). The triples inferred by the learned rules are injected into the KG, before the embedding is re-trained from scratch in the subsequent iteration. In contrast, the rule-based feedback generated by ExCut is not limited to only fact predictions, but encodes further structural similarities across enti-

ties. Furthermore, we do not re-train the whole model from scratch, but rather adapt the embedding of target entities accounting for the feedback. Finally, unlike [Zhang et al., 2019a], the rules that we learn support constants, which allow to capture a larger variety of explanations.

6.7 Conclusion

We have proposed ExCut, an approach for explainable KG entity clustering, which iteratively utilizes embeddings and rule learning methods to compute accurate clusters and human-readable explanations for them. Our approach is flexible, as any embedding model can be used. Experiments show the effectiveness of ExCut on real-world KGs. We have also shown a primary use-case for utilizing explainable clustering to assess KG embedding's ability to preserve KG structures.

There are several directions for future work. Considering more general rules (*e.g.*, with negations) in the *rule learning* component of our method or exploiting several embedding models instead of a single one in the *embedding-based clustering* step should lead to cleaner clusters. Further questions to study include analyzing the suitability of our method for large number of clusters and how the feedback from the rules can automatically determine the number of clusters. We also plan to exploit explainable clustering to understand the capabilities of advanced embedding models better.

Chapter 7

Conclusion and Outlook

7.1 Summary

In this dissertation, we presented four methods to address the challenges of *KG completion*, *validation*, and *exploration*. The introduced methods utilized symbolic reasoning combined with other techniques, namely, KG embedding and text mining. This combination enabled the integration of large-scale and noisy sources during reasoning and producing human-comprehensible output.

To improve the precision of rule-based *KG completion*, we introduced *ExRuL*, a method for revising Horn rules into exception-aware rules. We also presented *RuLES*, a rule learning method that utilizes KG embedding to provide probabilistic representations of the missing facts. Experiments on real-world KGs show that both methods enhance the overall quality of the learned set of rules. Besides, the learned exception-aware rules vastly reduce the errors of fact prediction compared with their positive counterparts. Moreover, the incorporation KG embeddings into the rule learning process shows better results compared to using KGs solely.

To support the explainability of *KG validation*, we presented *ExFaKT*, a framework for collecting semantic traces (*i.e.*, evidence) to support a candidate fact relying on both text and KGs. *ExFaKT* uses Horn rules to rewrite the candidate fact into a set of other facts that are easier to spot and confirm. Experiments show that rule-based rewriting significantly increases the recall of the discovered evidence for candidate facts while preserving high precision. Furthermore, the discovered traces support both manual and automatic fact-checking.

Finally, to facilitate *KG exploration*, we proposed *ExCut*, a method that combines rule learning with KG embedding models to compute entity clusters with comprehensible explanations. Experiments demonstrate that interleaving the learning of KG embeddings and rules improves the quality of the produced clusters and their explanations. More

importantly, the user study show that the explanations help to grasp the semantic similarities among entities in these clusters, which is vital while exploring the KG.

Throughout the dissertation, we have empirically demonstrated the benefits of combining symbolic reasoning with KG embedding and text mining. For instance, incorporating feedback from KG embedding models improved the quality of the learned rules in *RuLES*. Similarly, in *ExCut*, retraining the embedding based on the feedback from the learned rules enhances the overall quality of embedding-based clustering. These observations confirm our hypothesis that bridging both symbolic and sub-symbolic approaches is beneficial for creating scalable methods with interpretable results.

7.2 Outlook

We conclude by discussing future research directions related to this thesis.

Knowledge graph bias. Bias is one of the significant challenges faced by modern KGs (see Chapter 2), affecting KGs downstream tasks. Rule learning is also prone to such bias, which appears in the learned rules' diversity and quality. For example, many of the rules learned over the general propose KGs reflect the behavior of American celebrities. *ExRul* reduces such bias by learning the exceptions for the rules. Additionally, in *RuLES* and *ExCuT*, we utilize KG embedding to compensate for the missing facts; hence, reducing the effect of the KG bias.

Nevertheless, the bias in KGs remains an unresolved challenge, even for KG embedding models [Arduini et al., 2020]. Developing techniques to address KG bias is necessary to improve the subsequent tasks. One possibility for reducing the effect of the KG bias in the obtained rules, is to investigate mining rules locally over semantically related subgraphs or specific domains. This direction raises several research questions, such as finding the appropriate divisions and learning rules over domains with sparse knowledge.

Learning other rule forms. Inference rules are key assets for explainable KG refinement and exploration. However, the expressiveness of the rules is restricted to a predefined structure that is typically specific to the rule learning approach. More notably, available rule learning methods focus on extracting specific forms of Horn or nonmonotonic rules.

On the other hand, inducing rules of other, more complex, forms would be beneficial for express hidden data correlations. For instance, disjunctive rules (*e.g.*, *"Having a sibling implies having a sister or a brother", "Korean speakers are normally either from South or North Korea"*) can be used in combination with other resources to complete the KG. Also, existential rules, *i.e.*, rules with existential quantifiers in the head (*e.g.*,

"Being a musician in a band implies playing some musical instrument") are useful for discovering missing entities in KGs. Nevertheless, learning rules of such forms is not directly addressed in the existing works; only few attempts for key mining in KGs, *e.g.*, [Symeonidou et al., 2017, Lajus and Suchanek, 2018], might be relevant in this regard.

Similarly, temporal rules or constraints (*e.g.*, *"A person cannot graduate from a university before being born"*) can be a great asset for KG validation and completion. Only recently, temporal deductive [Chekol et al., 2017] and inductive [Omran et al., 2019] reasoning have been applied to KGs, yet there are still many challenges to tackle.

Extracting rules jointly from KGs and text. As shown in Chapter 5 (*ExFaKT*), rules mined over the KG solely are not always sufficient to represent real-world correlations. The diversity in the relations included in the rules is essential to collect more evidence. While modern KGs are rich in facts and typically relatively clean, they contain a limited set of encyclopedic relations (*e.g.*, *bornIn*, *marriedTo*). On the other hand, textual resources certainly cover a richer set of predicates (*e.g.*, *gotAquintedWith*, *celebratedWedding*), but suffer from noise.

A natural way to address the above issues is to combine text-based rule extraction relying on natural language processing and textual entailment techniques [Schoenmackers et al., 2010, Dragoni et al., 2017] with inductive rule learning from KGs. Recent large-scale word embedding and language models, *e.g.*, BERT [Devlin et al., 2019], can also be useful in this context. This interesting research direction comes with many challenges due to the heterogeneity of the input sources.

Hybrid symbolic and sub-symbolic approaches. As discussed earlier, interleaving rule learning with other sub-symbolic methods, particularly KG embeddings, benefits both methods. However, existing approaches, including those introduced in this dissertation, study their interaction as independent components, *i.e.*, black-boxes.Such decoupled interaction is usually limited to specific predefined queries with simple responses, which lack the global context, as in *RuLES*. Alternatively, the interaction between rule learning and KG embeddings can also be through augmenting each component's input with the feedback from the other, as in *ExCut*. However, it is not easy to control the impact of the added data, notably in large-scale datasets.

To the best of our knowledge, deeply integrating both statistical and rule-based methods is an unexplored area. Such coupling will allow for interchanging more complex signals and better control towards the desired target. For instance, integrating feedback from rules straight into the clustering algorithm may produce better results in *ExCut*. Also, adapting the embedding model's loss functions to directly incorporate the rules can improve the resulting embeddings. It is also worth investigating the possibility of

learning both entity and rule embeddings simultaneously, which is now feasible given neural rule learning approaches.

Finally, with the rise of *sub-symbolic machine learning*, we would like to emphasize the importance of producing human-comprehensible output for the developed approaches. Explainability can help humans to understand the advantages and limitations of these methods. Combining knowledge graphs with symbolic reasoning and natural language generation techniques could lead to a leap in understanding how machines think.

List of Figures

List of Tables

Bibliography

[Abujabal, 2019] Abujabal, A. (2019). *Question answering over knowledge bases with continuous learning*. PhD thesis, Saarland University, Saarbrücken, Germany.

[Aggarwal et al., 2016] Aggarwal, N., Bhatia, S., and Misra, V. (2016). Connecting the dots: Explaining relationships between unconnected entities in a knowledge graph. In Sack, H., Rizzo, G., Steinmetz, N., Mladenic, D., Auer, S., and Lange, C., editors, *The Semantic Web - ESWC 2016 - Satellite Events, Heraklion, Crete, Greece, May 29 - June 2, 2016, Revised Selected Papers*, volume 9989 of *Lecture Notes in Computer Science*, pages 35–39. Springer.

[Agrawal et al., 1993] Agrawal, R., Imielinski, T., and Swami, A. N. (1993). Mining Association Rules between Sets of Items in Large Databases. *SIGMOD Record*, 22(2):207–216.

[Agrawal et al., 1996] Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., and Verkamo, A. I. (1996). Fast Discovery of Association Rules. In *Advances in Knowledge Discovery and Data Mining*, page 307–328. AAAI Press, USA.

[Albrecht et al., 2019] Albrecht, J., Belger, A., Blum, R., and Zimmermann, R. (2019). Business analytics on knowledge graphs for market trend analysis. In *Proceedings of the Conference on "Lernen, Wissen, Daten, Analysen", Berlin, Germany, September 30 - October 2, 2019*, volume 2454 of *CEUR Workshop Proceedings*, pages 371–376. CEUR-WS.org.

[Arduini et al., 2020] Arduini, M., Noci, L., Pirovano, F., Zhang, C., Shrestha, Y. R., and Paudel, B. (2020). Adversarial learning for debiasing knowledge graph embeddings. *CoRR*, abs/2006.16309.

[Arenas et al., 2016] Arenas, M., Diaz, G. I., and Kostylev, E. V. (2016). Reverse Engineering SPARQL Queries. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*, pages 239–249. ACM.

[Auer et al., 2007] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. G. (2007). DBpedia: A Nucleus for a Web of Open Data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735. Springer.

[Azevedo and Jorge, 2007] Azevedo, P. J. and Jorge, A. M. (2007). Comparing Rule Measures for Predictive Association Rules. In *European Conference on Machine Learning - ECML 2007*, pages 510–517, Berlin, Heidelberg. Springer.

[Bader et al., 2020] Bader, S. R., Grangel-González, I., Nanjappa, P., Vidal, M., and Maleshkova, M. (2020). A knowledge graph for industry 4.0. In *The Semantic Web - ESWC 2020 - 17th International Conference, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings*, volume 12123 of *Lecture Notes in Computer Science*, pages 465–480. Springer.

[Bamler et al., 2019] Bamler, R., Salehi, F., and Mandt, S. (2019). Augmenting and tuning knowledge graph embeddings. In *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2019*, page 172. AUAI Press.

[Bianchi et al., 2020] Bianchi, F., Rossiello, G., Costabello, L., Palmonari, M., and Minervini, P. (2020). Knowledge Graph Embeddings and Explainable AI. In *Knowledge Graphs for eXplainable Artificial Intelligence: Foundations, Applications and Challenges*, volume 47 of *Studies on the Semantic Web*, pages 49–72. IOS Press.

[Bienvenu et al., 2016] Bienvenu, M., Bourgaux, C., and Goasdoué, F. (2016). Query-driven repairing of inconsistent dl-lite knowledge bases. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 957–964. IJCAI.

[Bizer et al., 2011] Bizer, C., Heath, T., and Berners-Lee, T. (2011). Linked data. In *Semantic Services, Interoperability and Web Applications - Emerging Concepts*, pages 205–227. CRC Press.

[Bollacker et al., 2008] Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250. ACM.

[Bordes et al., 2013] Bordes, A., Usunier, N., García-Durán, A., Weston, J., and Yakhnenko, O. (2013). Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.

[Bouraoui and Schockaert, 2018] Bouraoui, Z. and Schockaert, S. (2018). Learning Conceptual Space Representations of Interrelated Concepts. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 1760–1766. IJCAI.

[Brin et al., 1997] Brin, S., Motwani, R., Ullman, J. D., and Tsur, S. (1997). Dynamic Itemset Counting and Implication Rules for Market Basket Data. In *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data*, SIGMOD '97, page 255–264, New York, NY, USA. ACM.

[Buhrmester et al., 2011] Buhrmester, M., Kwang, T., and Gosling, S. D. (2011). Amazon's Mechanical Turk: A New Source of Inexpensive, Yet High-Quality, Data? *Perspectives on Psychological Science*, 6(1):3–5.

[Cebiric et al., 2019] Cebiric, S., Goasdoué, F., Kondylakis, H., Kotzinos, D., Manolescu, I., Troullinou, G., and Zneika, M. (2019). Summarizing semantic graphs: a survey. *The VLDB Journal*, 28(3):295–327.

[Ceri et al., 1989] Ceri, S., Gottlob, G., and Tanca, L. (1989). What you Always Wanted to Know About Datalog (And Never Dared to Ask). *IEEE Transactions on Knowledge and Data Engineering*, 1(1):146–166.

[Chekol et al., 2017] Chekol, M. W., Pirrò, G., Schoenfisch, J., and Stuckenschmidt, H. (2017). Marrying uncertainty and time in knowledge graphs. In Singh, S. P. and Markovitch, S., editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 88–94. AAAI Press.

[Chen et al., 2016] Chen, Y., Goldberg, S., Wang, D. Z., and Johri, S. S. (2016). Ontological Pathfinding. In *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 835–846. ACM.

[Chopra and Rao, 1993] Chopra, S. and Rao, M. R. (1993). The partition problem. *Mathematical Programming*, 59:87–115.

[Chu et al., 2015] Chu, X., Ouzzani, M., Morcos, J., Ilyas, I. F., Papotti, P., Tang, N., and Ye, Y. (2015). KATARA: reliable data cleaning with knowledge bases and crowdsourcing. *Proceedings of the VLDB Endowment*, 8(12):1952–1955.

[Ciampaglia et al., 2015] Ciampaglia, G. L., Shiralkar, P., Rocha, L. M., Bollen, J., Menczer, F., and Flammini, A. (2015). Computational Fact Checking from Knowledge Networks. *PLOS ONE*, 10(6):1–13.

[Cohen, 2016] Cohen, W. W. (2016). TensorLog: A Differentiable Deductive Database. *CoRR*, abs/1605.06523.

[Corapi et al., 2010] Corapi, D., Russo, A., and Lupu, E. (2010). Inductive Logic Programming as Abductive Search. In *Technical Communications of the 26th International Conference on Logic Programming*, volume 7 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 54–63, Dagstuhl, Germany. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[Corapi et al., 2012] Corapi, D., Russo, A., and Lupu, E. (2012). Inductive Logic Programming in Answer Set Programming. In *Inductive Logic Programming*, pages 91–97, Berlin, Heidelberg. Springer.

[Corapi et al., 2011] Corapi, D., Sykes, D., Inoue, K., and Russo, A. (2011). Probabilistic Rule Learning in Nonmonotonic Domains. In *Computational Logic in Multi-Agent Systems*, pages 243–258, Berlin, Heidelberg. Springer.

[Costabello et al., 2019] Costabello, L., Pai, S., Van, C. L., McGrath, R., McCarthy, N., and Tabacof, P. (2019). AmpliGraph: a Library for Representation Learning on Knowledge Graphs. https://docs.ampligraph.org/en/1.3.2/ (Last visited: December 2020).

[Dai et al., 2020] Dai, Y., Wang, S., Xiong, N. N., and Guo, W. (2020). A Survey on Knowledge Graph Embedding: Approaches, Applications and Benchmarks. *Electronics*, 9(5).

[d'Amato et al., 2016] d'Amato, C., Staab, S., Tettamanzi, A. G. B., Minh, T. D., and Gandon, F. (2016). Ontology Enrichment by Discovering Multi-Relational Association Rules from Ontological Knowledge Bases. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, SAC '16, page 333–338, New York, NY, USA. ACM.

[Darari et al., 2013] Darari, F., Nutt, W., Pirrò, G., and Razniewski, S. (2013). Completeness Statements about RDF Data Sources and Their Use for Query Answering. In *The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part I*, volume 8218 of *Lecture Notes in Computer Science*, pages 66–83. Springer.

[Das et al., 2017] Das, R., Zaheer, M., Reddy, S., and McCallum, A. (2017). Question Answering on Knowledge Bases and Text using Universal Schema and Memory Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*, pages 358–365. ACL.

[De Raedt and Bruynooghe, 1991] De Raedt, L. and Bruynooghe, M. (1991). CLINT : a multi-strategy interactive concept-learner and theory revision system. *Proceedings of the Multi-Strategy Learning Workshop, Virginia*, pages 175–191.

[Dehaspe and Raedt, 1997] Dehaspe, L. and Raedt, L. D. (1997). Mining Association Rules in Multiple Relations. In *Inductive Logic Programming, 7th International Workshop, ILP-97, Prague, Czech Republic, September 17-20, 1997, Proceedings*, volume 1297 of *Lecture Notes in Computer Science*, pages 125–132. Springer.

[Demartini, 2019] Demartini, G. (2019). Implicit bias in crowdsourced knowledge graphs. In *Companion of The 2019 World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 624–630. ACM.

[Dettmers et al., 2018] Dettmers, T., Minervini, P., Stenetorp, P., and Riedel, S. (2018). Convolutional 2D Knowledge Graph Embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pages 1811–1818. AAAI Press.

[Devlin et al., 2019] Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. ACL.

[Diefenbach et al., 2018] Diefenbach, D., López, V., Singh, K. D., and Maret, P. (2018). Core techniques of question answering systems over knowledge bases: a survey. *Knowledge and Information Systems*, 55(3):529–569.

[Dietz et al., 2018] Dietz, L., Kotov, A., and Meij, E. (2018). Utilizing knowledge graphs for text-centric information retrieval. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 1387–1390. ACM.

[Dong et al., 2014] Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., and Zhang, W. (2014). Knowledge Vault: A Web-scale Approach to Probabilistic Knowledge Fusion. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 601–610. ACM.

[Dong et al., 2015] Dong, X. L., Gabrilovich, E., Murphy, K., Dang, V., Horn, W., Lugaresi, C., Sun, S., and Zhang, W. (2015). Knowledge-Based Trust: Estimating the Trustworthiness of Web Sources. *Proceedings of the VLDB Endowment*, 8(9):938–949.

[Dong and Srivastava, 2013] Dong, X. L. and Srivastava, D. (2013). Compact explanation of data fusion decisions. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, pages 379–390. ACM.

[Drabent et al., 2009] Drabent, W., Eiter, T., Ianni, G., Krennwallner, T., Lukasiewicz, T., and Maluszynski, J. (2009). Hybrid reasoning with rules and ontologies. In *Semantic Techniques for the Web, The REWERSE Perspective*, volume 5500 of *Lecture Notes in Computer Science*, pages 1–49. Springer.

[Dragoni et al., 2017] Dragoni, M., Villata, S., Rizzi, W., and Governatori, G. (2017). Combining natural language processing approaches for rule extraction from legal documents. In *AI Approaches to the Complexity of Legal Systems - AICOL International Workshops 2017, Revised Selected Papers*, volume 10791 of *Lecture Notes in Computer Science*, pages 287–300. Springer.

[Dumancic and Blockeel, 2017] Dumancic, S. and Blockeel, H. (2017). An Expressive Dissimilarity Measure for Relational Clustering Using Neighbourhood Trees. *Machine Learning*, 106(9–10):1523–1545.

[Dumancic et al., 2018] Dumancic, S., García-Durán, A., and Niepert, M. (2018). On embeddings as an alternative paradigm for relational learning. *CoRR*, abs/1806.11391[v2].

[Dzeroski and Lavrac, 1991] Dzeroski, S. and Lavrac, N. (1991). Learning Relations from Noisy Examples: An Empirical Comparison of LINUS and FOIL. In *Machine Learning*, pages 399–402. Morgan Kaufmann.

[Dzyuba and van Leeuwen, 2017] Dzyuba, V. and van Leeuwen, M. (2017). Learning What Matters - Sampling Interesting Patterns. In *Advances in Knowledge Discovery and Data Mining - 21st Pacific-Asia Conference, PAKDD 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part I*, volume 10234 of *Lecture Notes in Computer Science*, pages 534–546. ACM.

[Eiter et al., 2016] Eiter, T., Fink, M., and Stepanova, D. (2016). Computing repairs of inconsistent dl-programs over EL ontologies. *Journal of Artificial Intelligence Research (JAIR)*, 56:463–515.

[Eiter et al., 2009] Eiter, T., Ianni, G., and Krennwallner, T. (2009). Answer Set Programming: A Primer. In *Reasoning Web. Semantic Technologies for Information Systems, 5th International Summer School 2009, Brixen-Bressanone, Italy, August 30 - September 4, 2009, Tutorial Lectures*, volume 5689 of *Lecture Notes in Computer Science*, pages 40–110. Springer.

[Eiter et al., 2017] Eiter, T., Kaminski, T., Redl, C., Schüller, P., and Weinzierl, A. (2017). Answer Set Programming with External Source Access. In *Reasoning Web. Semantic Interoperability on the Web - 13th International Summer School 2017, London, UK, July 7-11, 2017, Tutorial Lectures*, volume 10370 of *Lecture Notes in Computer Science*, pages 204–275. Springer.

[Ernst et al., 2015] Ernst, P., Siu, A., and Weikum, G. (2015). Knowlife: a versatile approach for constructing a large knowledge graph for biomedical sciences. *BMC Bioinformatics*, 16:157:1–157:13.

[Ester et al., 1996] Ester, M., Kriegel, H., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*, pages 226–231. AAAI Press.

[Evans and Grefenstette, 2018] Evans, R. and Grefenstette, E. (2018). Learning Explanatory Rules from Noisy Data. *Journal of Artificial Intelligence Research (JAIR)*, 61:1–64.

[Faber et al., 2011] Faber, W., Pfeifer, G., and Leone, N. (2011). Semantics and complexity of recursive aggregates in answer set programming. *Artificial Intelligence*, 175(1):278–298.

[Fanizzi et al., 2008] Fanizzi, N., d'Amato, C., and Esposito, F. (2008). Conceptual Clustering and Its Application to Concept Drift and Novelty Detection. In *The Semantic Web: Research and Applications*, pages 318–332, Berlin, Heidelberg. Springer.

[Fierens et al., 2015] Fierens, D., den Broeck, G. V., Renkens, J., Shterionov, D. S., Gutmann, B., Thon, I., Janssens, G., and Raedt, L. D. (2015). Inference and learning in probabilistic logic programs using weighted Boolean formulas. *Theory and Practice of Logic Programming*, 15(3):358–401.

[Finkel et al., 2005] Finkel, J. R., Grenager, T., and Manning, C. D. (2005). Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA*, pages 363–370. ACL.

[Fionda and Pirrò, 2017] Fionda, V. and Pirrò, G. (2017). Explaining and Querying Knowledge Graphs by Relatedness. *Proceedings of the VLDB Endowment*, 10(12):1913–1916.

[Fisher et al., 2020] Fisher, J., Mittal, A., Palfrey, D., and Christodoulopoulos, C. (2020). Debiasing knowledge graph embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7332–7345. ACL.

[Flach and Kakas, 2000] Flach, P. A. and Kakas, A. C. (2000). *Abduction and Induction: essays on their relation and integration*, volume 18 of *Applied Logic Series*. Springer.

[Fleiss et al., 1971] Fleiss, J. et al. (1971). Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

[Fonseca et al., 2011] Fonseca, N. A., Costa, V. S., and Camacho, R. (2011). Conceptual Clustering of Multi-Relational Data. In *Inductive Logic Programming - 21st International Conference, ILP 2011, Windsor Great Park, UK, July 31 - August 3, 2011, Revised Selected Papers*, volume 7207 of *Lecture Notes in Computer Science*, pages 145–159. Springer.

[Fournier-Viger et al., 2016] Fournier-Viger, P., Lin, J. C., Gomariz, A., Gueniche, T., Soltani, A., Deng, Z., and Lam, H. T. (2016). The SPMF Open-Source Data Mining Library Version 2. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part III*, volume 9853 of *Lecture Notes in Computer Science*, pages 36–40. Springer.

[Fürnkranz et al., 2012] Fürnkranz, J., Gamberger, D., and Lavrac, N. (2012). *Foundations of Rule Learning*. Cognitive Technologies. Springer.

[Gad-Elrab et al., 2020a] Gad-Elrab, M. H., Ho, V. T., Levinkov, E., Tran, T., and Stepanova, D. (2020a). Towards utilizing knowledge graph embedding models for conceptual clustering. In *Proceedings of the ISWC 2020 Demos and Industry Tracks: From Novel Ideas to Industrial Practice co-located with 19th International Semantic Web Conference (ISWC 2020), Globally online, November 1-6, 2020 (UTC)*, volume 2721 of *CEUR Workshop Proceedings*, pages 281–286. CEUR-WS.org.

[Gad-Elrab et al., 2020b] Gad-Elrab, M. H., Stepanova, D., Tran, T., Adel, H., and Weikum, G. (2020b). Excut: Explainable embedding-based clustering over knowledge graphs. In *The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, Athens, Greece, November 2-6, 2020, Proceedings, Part I*, volume 12506 of *Lecture Notes in Computer Science*, pages 218–237. Springer.

[Gad-Elrab et al., 2016] Gad-Elrab, M. H., Stepanova, D., Urbani, J., and Weikum, G. (2016). Exception-enriched Rule Learning from Knowledge Graphs. In *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I*, pages 234–251.

[Gad-Elrab et al., 2019] Gad-Elrab, M. H., Stepanova, D., Urbani, J., and Weikum, G. (2019). Exfakt: A framework for explaining facts over knowledge graphs and text. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, pages 87–95. ACM.

[Gad-Elrab et al., 2019] Gad-Elrab, M. H., Stepanova, D., Urbani, J., and Weikum, G. (2019). Tracy: Tracing facts over knowledge graphs and text. In *The World Wide Web Conference*, WWW '19, page 3516–3520, New York, NY, USA. Association for Computing Machinery.

[Galárraga et al., 2015] Galárraga, L., Teflioudi, C., Hose, K., and Suchanek, F. M. (2015). Fast Rule Mining in Ontological Knowledge Bases with AMIE+. In *The VLDB Journal*, volume 24, pages 707–730. VLDB Endowment.

[Gelfond and Lifschitz, 1988] Gelfond, M. and Lifschitz, V. (1988). The Stable Model Semantics for Logic Programming. In *Logic Programming, Proceedings of the Fifth International Conference and Symposium, Seattle, Washington, USA, August 15-19, 1988 (2 Volumes)*, pages 1070–1080. MIT Press.

[Gerber et al., 2015] Gerber, D., Esteves, D., Lehmann, J., Bühmann, L., Usbeck, R., Ngomo, A. N., and Speck, R. (2015). DeFacto - Temporal and multilingual Deep Fact Validation. *Semantic Web*, 35:85–101.

[Ghiasnezhad Omran et al., 2019] Ghiasnezhad Omran, P., Wang, K., and Wang, Z. (2019). An Embedding-based Approach to Rule Learning in Knowledge Graphs. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1.

[Glorot et al., 2013] Glorot, X., Bordes, A., Weston, J., and Bengio, Y. (2013). A Semantic Matching Energy Function for Learning with Multi-relational Data. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

[Goasdoué et al., 2013] Goasdoué, F., Karanasos, K., Katsis, Y., Leblay, J., Manolescu, I., and Zampetakis, S. (2013). Fact checking and analyzing the web. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pages 997–1000. ACM.

[Goethals and den Bussche, 2002] Goethals, B. and den Bussche, J. V. (2002). Relational Association Rules: Getting WARMeR. In *Pattern Detection and Discovery, ESF Exploratory Workshop, London, UK, September 16-19, 2002, Proceedings*, volume 2447 of *Lecture Notes in Computer Science*, pages 125–139. Springer.

[Gómez-Pérez et al., 2004] Gómez-Pérez, A., Fernández-López, M., and Corcho, Ó. (2004). *Ontological Engineering: With Examples from the Areas of Knowledge Management, e-Commerce and the Semantic Web*. Advanced Information and Knowledge Processing. Springer.

[Gómez-Romero et al., 2018] Gómez-Romero, J., Molina-Solana, M., Oehmichen, A., and Guo, Y. (2018). Visualizing large knowledge graphs: A performance analysis. *Future Generation Computer Systems*, 89:224–238.

[Gonzalez et al., 2014] Gonzalez, J. E., Xin, R. S., Dave, A., Crankshaw, D., Franklin, M. J., and Stoica, I. (2014). GraphX: Graph Processing in a Distributed Dataflow Framework. In *11th USENIX Symposium on Operating Systems Design and Implementation, OSDI '14, Broomfield, CO, USA, October 6-8, 2014*, pages 599–613. USENIX Association.

[González-Conejero et al., 2018] González-Conejero, J., Casanovas, P., and Teodoro, E. (2018). Business requirements for legal knowledge graph: the LYNX platform. In *Proceedings of the 2nd Workshop on Technologies for Regulatory Compliance co-located with the 31st International Conference on Legal Knowledge and Information Systems (JURIX 2018), Groningen, The Netherlands, December 12, 2018*, volume 2309 of *CEUR Workshop Proceedings*, pages 31–38. CEUR-WS.org.

[Gormley and Tong, 2015] Gormley, C. and Tong, Z. (2015). *Elasticsearch: The Definitive Guide*. O'Reilly Media, Inc., 1st edition.

[Guo et al., 2019] Guo, L., Sun, Z., and Hu, W. (2019). Learning to Exploit Long-term Relational Dependencies in Knowledge Graphs. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2505–2514. PMLR.

[Guo et al., 2020] Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., and He, Q. (2020). A survey on knowledge graph-based recommender systems. *CoRR*, abs/2003.00911.

[Guo et al., 2016] Guo, S., Wang, Q., Wang, L., Wang, B., and Guo, L. (2016). Jointly Embedding Knowledge Graphs and Logical Rules. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 192–202, Austin, Texas. ACL.

[Guo et al., 2018] Guo, S., Wang, Q., Wang, L., Wang, B., and Guo, L. (2018). Knowledge Graph Embedding With Iterative Guidance From Soft Rules. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 4816–4823. AAAI Press.

[Guo et al., 2005] Guo, Y., Pan, Z., and Heflin, J. (2005). LUBM: A benchmark for OWL knowledge base systems. *Journal of Web Semantics*, 3(2-3):158–182.

[Hamilton et al., 2018] Hamilton, W. L., Bajaj, P., Zitnik, M., Jurafsky, D., and Leskovec, J. (2018). Embedding Logical Queries on Knowledge Graphs. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pages 2030–2041.

[Han et al., 2004] Han, J., Pei, J., Yin, Y., and Mao, R. (2004). Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. *Data Mining and Knowledge Discovery*, 8(1):53–87.

[Hassanzadeh et al., 2009] Hassanzadeh, O., Chiang, F., Lee, H. C., and Miller, R. J. (2009). Framework for Evaluating Clustering Algorithms in Duplicate Detection. *Proceedings of the VLDB Endowment*, 2(1):1282–1293.

[Haussmann, 2017] Haussmann, E. (2017). *Towards precise and convenient semantic search on text and knowledge bases*. PhD thesis, University of Freiburg, Freiburg im Breisgau, Germany.

[Ho et al., 2018] Ho, V. T., Stepanova, D., Gad-Elrab, M. H., Kharlamov, E., and Weikum, G. (2018). Rule Learning from Knowledge Graphs Guided by Embedding Models. In *The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part I*, volume 11136 of *Lecture Notes in Computer Science*, pages 72–90. Springer.

[Hogan et al., 2020] Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., de Melo, G., Gutierrez, C., Gayo, J. E. L., Kirrane, S., Neumaier, S., Polleres, A., Navigli, R., Ngomo, A. N., Rashid, S. M., Rula, A., Schmelzeisen, L., Sequeda, J. F., Staab, S., and Zimmermann, A. (2020). Knowledge graphs. *CoRR*, abs/2003.02320.

[Huang et al., 2017] Huang, X., Kwiatkowska, M., Wang, S., and Wu, M. (2017). Safety verification of deep neural networks. In *Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I*, volume 10426 of *Lecture Notes in Computer Science*, pages 3–29. Springer.

[Idahl et al., 2019] Idahl, M., Khosla, M., and Anand, A. (2019). Finding Interpretable Concept Spaces in Node Embeddings Using Knowledge Bases. In *Machine Learning and Knowledge Discovery in Databases - International Workshops of ECML PKDD 2019, Würzburg, Germany, September 16-20, 2019, Proceedings, Part I*, volume 1167 of *Communications in Computer and Information Science*, pages 229–240. Springer.

[Janowicz et al., 2018] Janowicz, K., Yan, B., Regalia, B., Zhu, R., and Mai, G. (2018). Debiasing Knowledge Graphs: Why Female Presidents are not like Female Popes. In *Proceedings of the ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, USA, October 8th - to - 12th, 2018*, volume 2180 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Ji et al., 2011] Ji, Q., Gao, Z., and Huang, Z. (2011). Reasoning with noisy semantic data. In *The Semanic Web - ESWC 2011 - 8th Extended Semantic Web Conference, Heraklion, Crete, Greece, May 29 - June 2, 2011, Proceedings, Part II*, volume 6644 of *Lecture Notes in Computer Science*, pages 497–502. Springer.

[Józefowska et al., 2010] Józefowska, J., Lawrynowicz, A., and Lukaszewski, T. (2010). The role of semantics in mining frequent patterns from knowledge bases in description logics with rules. *Theory and Practice of Logic Programming*, 10(3):251–289.

[Junior et al., 2020] Junior, A. C., Orlandi, F., Graux, D., Hossari, M., O'Sullivan, D., Hartz, C., and Dirschl, C. (2020). Knowledge graph-based legal search over german court cases. In *The Semantic Web - ESWC 2020 - Satellite Events, Heraklion, Crete, Greece, May 31 - June 4, 2020, Revised Selected Papers*, volume 12124 of *Lecture Notes in Computer Science*, pages 293–297. Springer.

[Kalayci et al., 2020] Kalayci, E. G., Grangel-González, I., Lösch, F., Xiao, G., ul Mehdi, A., Kharlamov, E., and Calvanese, D. (2020). Semantic integration of bosch manufacturing data using virtual knowledge graphs. In *The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, Athens, Greece, November 2-6, 2020, Proceedings, Part II*, volume 12507 of *Lecture Notes in Computer Science*, pages 464–481. Springer.

[Katzouris et al., 2015] Katzouris, N., Artikis, A., and Paliouras, G. (2015). Incremental learning of event definitions with Inductive Logic Programming. *Machine Learning*, 100(2-3):555–585.

[Keuper et al., 2015] Keuper, M., Levinkov, E., Bonneel, N., Lavoué, G., Brox, T., and Andres, B. (2015). Efficient decomposition of image and mesh graphs by lifted multicuts. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1751–1759. IEEE.

[Kharlamov et al., 2017] Kharlamov, E., Savkoviý, O., Xiao, G., Penaloza, R., Mehdi, G., Roshchin, M., and Horrocks, I. (2017). Semantic Rules for Machine Diagnostics:

Execution and Management. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 2131–2134, New York, NY, USA. ACM.

[Klyne and Carroll, 2004] Klyne, G. and Carroll, J. J. (2004). Resource Description Framework (RDF): Concepts and Abstract Syntax. *W3C Recommendation*.

[Knobbe and Ho, 2006] Knobbe, A. J. and Ho, E. K. Y. (2006). Pattern Teams. In *Knowledge Discovery in Databases: PKDD 2006, 10th European Conference on Principles and Practice of Knowledge Discovery in Databases, Berlin, Germany, September 18-22, 2006, Proceedings*, volume 4213 of *Lecture Notes in Computer Science*, pages 577–584. Springer.

[Kolitsas et al., 2018] Kolitsas, N., Ganea, O., and Hofmann, T. (2018). End-to-End Neural Entity Linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning, CoNLL 2018, Brussels, Belgium, October 31 - November 1, 2018*, pages 519–529. ACL.

[Kowalski and Kuehner, 1971] Kowalski, R. A. and Kuehner, D. (1971). Linear Resolution with Selection Function. *Artificial Intelligence*, 2(3):227–260.

[Kramer et al., 2001] Kramer, S., Lavrač, N., and Flach, P. (2001). *Propositionalization Approaches to Relational Data Mining*, pages 262–291. Springer.

[Lajus and Suchanek, 2018] Lajus, J. and Suchanek, F. M. (2018). Are All People Married?: Determining Obligatory Attributes in Knowledge Bases. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1115–1124. ACM.

[Lavrac et al., 1999] Lavrac, N., Flach, P. A., and Zupan, B. (1999). Rule Evaluation Measures: A Unifying View. In *Inductive Logic Programming, 9th International Workshop, ILP-99, Bled, Slovenia, June 24-27, 1999, Proceedings*, volume 1634 of *Lecture Notes in Computer Science*, pages 174–185. Springer.

[Law et al., 2020] Law, M., Russo, A., and Broda, K. (2020). The ILASP system for Inductive Learning of Answer Set Programs. *CoRR*, abs/2005.00904.

[Leblay, 2017] Leblay, J. (2017). A Declarative Approach to Data-Driven Fact Checking. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 147–153. AAAI Press.

[Lehmann et al., 2011] Lehmann, J., Auer, S., Bühmann, L., and Tramp, S. (2011). Class expression learning for ontology engineering. *Journal of Web Semantics*, 9(1):71–81.

[Lehmann et al., 2015] Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., and Bizer, C. (2015). DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*, 6(2):167–195.

[Lenat, 1995] Lenat, D. B. (1995). CYC: A Large-Scale Investment in Knowledge Infrastructure. *Communications of the ACM*, 38(11):32–38.

[Lenat and Feigenbaum, 1991] Lenat, D. B. and Feigenbaum, E. A. (1991). On the thresholds of knowledge. *Artificial Intelligence*, 47(1):185–250.

[Leone et al., 2006] Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G., Perri, S., and Scarcello, F. (2006). The DLV system for knowledge representation and reasoning. *ACM Transactions on Computational Logic*, 7(3):499–562.

[Li et al., 2020] Li, L., Wang, P., Yan, J., Wang, Y., Li, S., Jiang, J., Sun, Z., Tang, B., Chang, T., Wang, S., and Liu, Y. (2020). Real-world data medical knowledge graph: construction and applications. *Artificial Intelligence Medicine*, 103:101817.

[Li et al., 2014] Li, Q., Li, Y., Gao, J., Zhao, B., Fan, W., and Han, J. (2014). Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In *International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014*, pages 1187–1198. ACM.

[Li et al., 2011] Li, X., Meng, W., and Yu, C. T. (2011). T-verifier: Verifying Truthfulness of Fact Statements. In *Proceedings of the 27th International Conference on Data Engineering, ICDE 2011, April 11-16, 2011, Hannover, Germany*, pages 63–74. IEEE.

[Li et al., 2016] Li, Y., Gao, J., Meng, C., Li, Q., Su, L., Zhao, B., Fan, W., and Han, J. (2016). A Survey on Truth Discovery. *SIGKDD Explorations*, 17(2):1–16.

[Lin et al., 2015] Lin, Y., Liu, Z., Luan, H., Sun, M., Rao, S., and Liu, S. (2015). Modeling Relation Paths for Representation Learning of Knowledge Bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 705–714. ACL.

[Lisi, 2006] Lisi, F. A. (2006). A Pattern-Based Approach to Conceptual Clustering in FOL. In *Conceptual Structures: Inspiration and Application, 14th International Conference on Conceptual Structures, ICCS 2006, Aalborg, Denmark, July 16-21, 2006, Proceedings*, volume 4068 of *Lecture Notes in Computer Science*, pages 346–359. Springer.

[Lisi, 2010] Lisi, F. A. (2010). Inductive Logic Programming in Databases: From Datalog to DL+log. *Theory and Practice of Logic Programming*, 10(3):331–359.

[Lissandrini et al., 2020] Lissandrini, M., Mottin, D., Palpanas, T., and Velegrakis, Y. (2020). Graph-query suggestions for knowledge graph exploration. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 2549–2555. ACM.

[Liu et al., 2018] Liu, Y., Safavi, T., Dighe, A., and Koutra, D. (2018). Graph summarization methods and applications: A survey. *ACM Computing Surveys*, 51(3):62:1–62:34.

[Liu et al., 2020] Liu, Y., Wan, Y., He, L., Peng, H., and Yu, P. S. (2020). KG-BART: knowledge graph-augmented BART for generative commonsense reasoning. *CoRR*, abs/2009.12677.

[Lv et al., 2018] Lv, X., Hou, L., Li, J., and Liu, Z. (2018). Differentiating Concepts and Instances for Knowledge Graph Embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1971–1979. ACL.

[MacQueen, 1967] MacQueen, J. B. (1967). Some Methods for Classification and Analysis of MultiVariate Observations. In *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press.

[Mahdisoltani et al., 2015] Mahdisoltani, F., Biega, J., and Suchanek, F. M. (2015). YAGO3: A Knowledge Base from Multilingual Wikipedias. In *CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings*. www.cidrdb.org.

[Mami et al., 2019] Mami, M. N., Graux, D., Scerri, S., Jabeen, H., Auer, S., and Lehmann, J. (2019). Uniform access to multiform data lakes using semantic technologies. In *Proceedings of the 21st International Conference on Information Integration and Web-based Applications & Services, iiWAS 2019, Munich, Germany, December 2-4, 2019*, pages 313–322. ACM.

[Mannila et al., 1995] Mannila, H., Toivonen, H., and Verkamo, A. I. (1995). Discovering Frequent Episodes in Sequences. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95), Montreal, Canada, August 20-21, 1995*, pages 210–215. AAAI Press.

[Manning et al., 2008] Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.

[Martires et al., 2020] Martires, P. Z. D., Nitesh, K., Persson, A., Loutfi, A., and Raedt, L. D. (2020). Symbolic learning and reasoning with noisy data for probabilistic anchoring. *Frontiers in Robotics and AI*, 7:100.

[Mcmillan et al., 1992] Mcmillan, C., Mozer, M. C., and Smolensky, P. (1992). Rule Induction through Integrated Symbolic and Subsymbolic Processing. In *Advances in Neural Information Processing Systems 4*, pages 969–976. Morgan Kaufmann.

[Mehdi et al., 2019] Mehdi, A., Kharlamov, E., Stepanova, D., Loesch, F., and Grangel-González, I. (2019). Towards semantic integration of bosch manufacturing data. In *Proceedings of the ISWC 2019 Satellite Tracks (Posters & Demonstrations, Industry, and Outrageous Ideas) co-located with 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 26-30, 2019*, volume 2456 of *CEUR Workshop Proceedings*, pages 303–304. CEUR-WS.org.

[Mehdi et al., 2017] Mehdi, G., Kharlamov, E., Savkovic, O., Xiao, G., Kalayci, E. G., Brandt, S., Horrocks, I., Roshchin, M., and Runkler, T. A. (2017). Semantic Rule-Based Equipment Diagnostics. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II*, volume 10588 of *Lecture Notes in Computer Science*, pages 314–333. Springer.

[Meij, 2019] Meij, E. (2019). Understanding news using the bloomberg knowledge graph. In *TheWebConf - WWW2019 - Invited talk at the Big Data Innovators Gathering*.

[Meilicke et al., 2019] Meilicke, C., Chekol, M. W., Ruffinelli, D., and Stuckenschmidt, H. (2019). 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. IJCAI.

[Meilicke et al., 2018] Meilicke, C., Fink, M., Wang, Y., Ruffinelli, D., Gemulla, R., and Stuckenschmidt, H. (2018). Fine-Grained Evaluation of Rule- and Embedding-Based Systems for Knowledge Graph Completion. In *The Semantic Web - ISWC 2018 -*

17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part I, volume 11136 of *Lecture Notes in Computer Science*, pages 3–20. Springer.

[Miller, 1995] Miller, G. A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41.

[Mitchell et al., 2015] Mitchell, T. M., Cohen, W. W., Jr., E. R. H., Talukdar, P. P., Betteridge, J., Carlson, A., Mishra, B. D., Gardner, M., Kisiel, B., Krishnamurthy, J., Lao, N., Mazaitis, K., Mohamed, T., Nakashole, N., Platanios, E. A., Ritter, A., Samadi, M., Settles, B., Wang, R. C., Wijaya, D., Gupta, A., Chen, X., Saparov, A., Greaves, M., and Welling, J. (2015). Never-Ending Learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 2302–2310. AAAI Press.

[Mohanty and Ramanath, 2019] Mohanty, M. and Ramanath, M. (2019). Insta-search: Towards effective exploration of knowledge graphs. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 2909–2912, New York, NY, USA. ACM.

[Morik, 1993] Morik, K. (1993). Balanced Cooperative Modeling. *Machine Learning*, 11(2):217–235.

[Mottin et al., 2016] Mottin, D., Lissandrini, M., Velegrakis, Y., and Palpanas, T. (2016). Exemplar queries: a new way of searching. *The VLDB Journal*, 25(6):741–765.

[Muggleton, 1991] Muggleton, S. (1991). Inductive Logic Programming. *New Generation Computing*, 8(4):295–318.

[MUGGLETON and BUNTINE, 1988] MUGGLETON, S. and BUNTINE, W. (1988). Machine Invention of First-order Predicates by Inverting Resolution. In *Machine Learning Proceedings 1988*, pages 339–352. Morgan Kaufmann, San Francisco (CA).

[Muggleton and Feng, 1990] Muggleton, S. and Feng, C. (1990). Efficient Induction of Logic Programs. In *Algorithmic Learning Theory, First International Workshop, ALT '90, Tokyo, Japan, October 8-10, 1990, Proceedings*, pages 368–381. Springer.

[Mukherjee et al., 2014] Mukherjee, S., Weikum, G., and Danescu-Niculescu-Mizil, C. (2014). People on drugs: credibility of user statements in health communities. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 65–74. ACM.

[Nakashole and Mitchell, 2014] Nakashole, N. and Mitchell, T. M. (2014). Language-Aware Truth Assessment of Fact Candidates. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, volume 1, pages 1009–1019. ACL.

[Nakashole et al., 2012a] Nakashole, N., Sozio, M., Suchanek, F. M., and Theobald, M. (2012a). Query-time reasoning in uncertain RDF knowledge bases with soft and hard rules. In *Proceedings of the Second International Workshop on Searching and Integrating New Web Data Sources, Istanbul, Turkey, August 31, 2012*, volume 884 of *CEUR Workshop Proceedings*, pages 15–20. CEUR-WS.org.

[Nakashole et al., 2012b] Nakashole, N., Weikum, G., and Suchanek, F. M. (2012b). PATTY: A Taxonomy of Relational Patterns with Semantic Types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, pages 1135–1145. ACL.

[Navigli and Ponzetto, 2012] Navigli, R. and Ponzetto, S. P. (2012). BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

[Nguyen et al., 2018] Nguyen, D. Q., Nguyen, T. D., Nguyen, D. Q., and Phung, D. (2018). 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)*, pages 327–333, New Orleans, Louisiana. ACL.

[Nguyen et al., 2019] Nguyen, D. Q., Vu, T., Nguyen, T. D., Nguyen, D. Q., and Phung, D. Q. (2019). A Capsule Network-based Embedding Model for Knowledge Graph Completion and Search Personalization. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2180–2189. ACL.

[Nickel et al., 2016a] Nickel, M., Murphy, K., Tresp, V., and Gabrilovich, E. (2016a). A Review of Relational Machine Learning for Knowledge Graphs. *Proceedings of the IEEE*, 104(1):11–33.

[Nickel et al., 2016b] Nickel, M., Rosasco, L., and Poggio, T. A. (2016b). Holographic Embeddings of Knowledge Graphs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 1955–1961. AAAI Press.

[Nickel et al., 2011] Nickel, M., Tresp, V., and Kriegel, H.-P. (2011). 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. Omnipress.

[Nickles and Mileo, 2015] Nickles, M. and Mileo, A. (2015). A Hybrid Approach to Inference in Probabilistic Non-Monotonic Logic Programming. In *Proceedings of the 2nd International Workshop on Probabilistic Logic Programming co-located with 31st International Conference on Logic Programming (ICLP 2015)*, volume 1413 of *CEUR Workshop Proceedings*, pages 57–68. CEUR-WS.org.

[Niklaus et al., 2018] Niklaus, C., Cetto, M., Freitas, A., and Handschuh, S. (2018). A Survey on Open Information Extraction. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3866–3878, Santa Fe, New Mexico, USA. ACL.

[Noy et al., 2019] Noy, N. F., Gao, Y., Jain, A., Narayanan, A., Patterson, A., and Taylor, J. (2019). Industry-scale knowledge graphs: lessons and challenges. *Communications of the ACM*, 62(8):36–43.

[Omran et al., 2018] Omran, P. G., Wang, K., and Wang, Z. (2018). Scalable Rule Learning via Learning Representation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 2149–2155. IJCAI.

[Omran et al., 2019] Omran, P. G., Wang, K., and Wang, Z. (2019). Learning temporal rules from knowledge graph streams. In *Proceedings of the AAAI 2019 Spring Symposium on Combining Machine Learning with Knowledge Engineering (AAAI-MAKE 2019) Stanford University, Palo Alto, California, USA, March 25-27, 2019.*, volume 2350 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Ortona et al., 2018] Ortona, S., Meduri, V. V., and Papotti, P. (2018). Robust Discovery of Positive and Negative Rules in Knowledge Bases. In *34th IEEE International Conference on Data Engineering, ICDE 2018, Paris, France, April 16-19, 2018*, pages 1168–1179. IEEE.

[Pandey et al., 2018] Pandey, S., Banerjee, S., and Chatterjee, A. (2018). ReiNN: Efficient error resilience in artificial neural networks using encoded consistency checks. In *23rd IEEE European Test Symposium, ETS 2018, Bremen, Germany, May 28 - June 1, 2018*, pages 1–2. IEEE.

[Pasternack and Roth, 2013] Pasternack, J. and Roth, D. (2013). Latent credibility analysis. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, pages 1009–1020. ACM.

[Paulheim, 2017] Paulheim, H. (2017). Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, 8(3):489–508.

[Paulheim and Bizer, 2014] Paulheim, H. and Bizer, C. (2014). Improving the quality of linked data using statistical distributions. *International Journal on Semantic Web and Information Systems*, 10(2):63–86.

[Piatetsky-Shapiro, 1991] Piatetsky-Shapiro, G. (1991). Discovery, Analysis, and Presentation of Strong Rules. In *Knowledge Discovery in Databases*, pages 229–248. AAAI Press.

[Poole and Mackworth, 2017] Poole, D. L. and Mackworth, A. K. (2017). *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press, USA, 2nd edition.

[Popat et al., 2017] Popat, K., Mukherjee, S., Strötgen, J., and Weikum, G. (2017). Where the Truth Lies: Explaining the Credibility of Emerging Claims on the Web and Social Media. In *Proceedings of the 26th International Conference on World Wide Web Companion, Perth, Australia, April 3-7, 2017*, pages 1003–1012. ACM.

[Quinlan, 1990] Quinlan, J. R. (1990). Learning Logical Definitions from Relations. *Machine Learning*, 5:239–266.

[Raad et al., 2020] Raad, J., Beek, W., van Harmelen, F., Wielemaker, J., Pernelle, N., and Saïs, F. (2020). Constructing and cleaning identity graphs in the LOD cloud. *Data Intelligence Nsl*, 2(3):323–352.

[Raad et al., 2019] Raad, J., Pernelle, N., Saïs, F., Beek, W., and van Harmelen, F. (2019). The sameas problem: A survey on identity management in the web of data. *CoRR*, abs/1907.10528.

[Raedt, 2008] Raedt, L. D. (2008). *Logical and relational learning*. Cognitive Technologies. Springer.

[Raedt et al., 2015] Raedt, L. D., Dries, A., Thon, I., den Broeck, G. V., and Verbeke, M. (2015). Inducing Probabilistic Relational Rules from Probabilistic Examples. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 1835–1843. AAAI Press.

[Raedt et al., 1993] Raedt, L. D., Lavrac, N., and Dzeroski, S. (1993). Multiple Predicate Learning. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence. Chambéry, France, August 28 - September 3, 1993*, pages 1037–1043. Morgan Kaufmann.

[Raedt and Thon, 2010] Raedt, L. D. and Thon, I. (2010). Probabilistic Rule Learning. In *Inductive Logic Programming - 20th International Conference, ILP 2010, Florence, Italy, June 27-30, 2010. Revised Papers*, volume 6489 of *Lecture Notes in Computer Science*, pages 47–58. Springer.

[Ras and Wieczorkowska, 2000] Ras, Z. W. and Wieczorkowska, A. (2000). Action-Rules: How to Increase Profit of a Company. In *Principles of Data Mining and Knowledge Discovery, 4th European Conference, PKDD 2000, Lyon, France, September 13-16, 2000, Proceedings*, volume 1910 of *Lecture Notes in Computer Science*, pages 587–592. Springer.

[Rastogi et al., 2017] Rastogi, P., Poliak, A., and Durme, B. V. (2017). Training Relation Embeddings under Logical Constraints. In *Proceedings of the First Workshop on Knowledge Graphs and Semantics for Text Retrieval and Analysis (KG4IR 2017) co-located with the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2017), Shinjuku, Tokyo, Japan, August 11, 2017*, volume 1883 of *CEUR Workshop Proceedings*, pages 25–31. CEUR-WS.org.

[Ray, 2009] Ray, O. (2009). Nonmonotonic abductive inductive learning. *Journal of Applied Logic*, 7(3):329–340. Special Issue: Abduction and Induction in Artificial Intelligence.

[Rebele et al., 2016] Rebele, T., Suchanek, F. M., Hoffart, J., Biega, J., Kuzey, E., and Weikum, G. (2016). YAGO: A Multilingual Knowledge Base from Wikipedia, Wordnet, and Geonames. In *The Semantic Web - ISWC 2016 - 15th International*

Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part II, volume 9982 of *Lecture Notes in Computer Science*, pages 177–185. Springer.

[Ren et al., 2020] Ren, H., Hu, W., and Leskovec, J. (2020). Query2box: Reasoning over Knowledge Graphs in Vector Space using Box Embeddings. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

[Reuters, 2017] Reuters, T. (2017). Thomson reuters launches first of its kind knowledge graph feed allowing financial services customers to accelerate their ai and digital strategies. *Press Release*. (Last visited: December 2020).

[Ringsquandl et al., 2018] Ringsquandl, M., Kharlamov, E., Stepanova, D., Hildebrandt, M., Lamparter, S., Lepratti, R., Horrocks, I., and Kröger, P. (2018). Event-enhanced learning for KG completion. In *European Semantic Web Conference*, pages 541–559. Springer.

[Ristoski and Paulheim, 2014] Ristoski, P. and Paulheim, H. (2014). A Comparison of Propositionalization Strategies for Creating Features from Linked Open Data. In *Proceedings of the 1st Workshop on Linked Data for Knowledge Discovery co-located with European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2014), Nancy, France, September 19th, 2014*, volume 1232 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Ristoski et al., 2019] Ristoski, P., Rosati, J., Noia, T. D., Leone, R. D., and Paulheim, H. (2019). RDF2Vec: RDF graph embeddings and their applications. *Semantic Web*, 10(4):721–752.

[Rossi et al., 2020] Rossi, A., Firmani, D., Matinata, A., Merialdo, P., and Barbosa, D. (2020). Knowledge Graph Embedding for Link Prediction: A Comparative Analysis. *CoRR*, abs/2002.00819.

[Sadeghian et al., 2019] Sadeghian, A., Armandpour, M., Ding, P., and Wang, D. Z. (2019). DRUM: End-To-End Differentiable Rule Mining On Knowledge Graphs. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 15321–15331.

[Saeedi et al., 2018] Saeedi, A., Peukert, E., and Rahm, E. (2018). Using Link Features for Entity Clustering in Knowledge Graphs. In *The Semantic Web - ESWC 2018 -*

15th International Conference, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings, volume 10843 of *Lecture Notes in Computer Science*, pages 576–592. Springer.

[Sakama, 2005] Sakama, C. (2005). Induction from answer sets in nonmonotonic logic programs. *ACM Transactions on Computational Logic*, 6(2):203–231.

[Sazonau and Sattler, 2017] Sazonau, V. and Sattler, U. (2017). Mining Hypotheses from Data in OWL: Advanced Evaluation and Complete Construction. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I*, volume 10587 of *Lecture Notes in Computer Science*, pages 577–593. Springer.

[Schoenmackers et al., 2010] Schoenmackers, S., Davis, J., Etzioni, O., and Weld, D. S. (2010). Learning first-order horn clauses from web text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP 2010, 9-11 October 2010, MIT Stata Center, Massachusetts, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1088–1098. ACL.

[Shi and Weninger, 2016] Shi, B. and Weninger, T. (2016). Discriminative predicate path mining for fact checking in knowledge graphs. *Knowledge Based Systems*, 104:123–133.

[Shi et al., 2014] Shi, H., Maly, K., and Zeil, S. J. (2014). Optimized Backward Chaining Reasoning System for a Semantic Web. In *4th International Conference on Web Intelligence, Mining and Semantics (WIMS 14), WIMS '14, Thessaloniki, Greece, June 2-4, 2014*, pages 34:1–34:6. ACM.

[Shi and Malik, 2000] Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905.

[Shiralkar et al., 2017] Shiralkar, P., Flammini, A., Menczer, F., and Ciampaglia, G. L. (2017). Finding Streams in Knowledge Graphs to Support Fact Checking. In *2017 IEEE International Conference on Data Mining, ICDM 2017, New Orleans, LA, USA, November 18-21, 2017*, pages 859–864. IEEE.

[Singhal, 2012] Singhal, A. (2012). Introducing the knowledge graph: things, not strings. *Google Blog*. (Last visited: December 2020).

[Socher et al., 2013] Socher, R., Chen, D., Manning, C. D., and Ng, A. Y. (2013). Reasoning With Neural Tensor Networks for Knowledge Base Completion. In *Advances in*

Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States, pages 926–934.

[Song et al., 2018] Song, Q., Wu, Y., Lin, P., Dong, X., and Sun, H. (2018). Mining summaries for knowledge graph search. *IEEE Transactions on Knowledge and Data Engineering*, 30(10):1887–1900.

[Srinivasan, 2001] Srinivasan, A. (2001). *The Aleph manual*. Oxford University, http://www.cs.ox.ac.uk/activities/machlearn/Aleph/aleph.html (Last visited: December 2020).

[Steinley, 2004] Steinley, D. (2004). Properties of the hubert-arabie adjusted rand index. *Psychological methods*, 9-3:386–96.

[Stepanova et al., 2018] Stepanova, D., Gad-Elrab, M. H., and Ho, V. T. (2018). Rule Induction and Reasoning over Knowledge Graphs. In *Reasoning Web. Learning, Uncertainty, Streaming, and Scalability - 14th International Summer School 2018, Esch-sur-Alzette, Luxembourg, September 22-26, 2018, Tutorial Lectures*, volume 11078 of *Lecture Notes in Computer Science*, pages 142–172. Springer.

[Suárez et al., 2019] Suárez, A. P., Trinidad, J. F. M., and Carrasco-Ochoa, J. A. (2019). A review of conceptual clustering algorithms. *Artificial Intelligence Review*, 52(2):1267–1296.

[Suchanek et al., 2007] Suchanek, F. M., Kasneci, G., and Weikum, G. (2007). Yago: A Core of Semantic Knowledge. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, pages 697–706. ACM.

[Symeonidou et al., 2017] Symeonidou, D., Galárraga, L., Pernelle, N., Saïs, F., and Suchanek, F. M. (2017). VICKEY: mining conditional keys on knowledge bases. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I*, volume 10587 of *Lecture Notes in Computer Science*, pages 661–677. Springer.

[Taniar et al., 2008] Taniar, D., Rahayu, W., Lee, V., and Daly, O. (2008). Exception rules in association rule mining. *Applied Mathematics and Computation*, 205(2):735–750.

[Tanon et al., 2017] Tanon, T. P., Stepanova, D., Razniewski, S., Mirza, P., and Weikum, G. (2017). Completeness-aware rule learning from knowledge graphs. In d'Amato, C., Fernández, M., Tamma, V. A. M., Lécué, F., Cudré-Mauroux, P., Sequeda, J. F., Lange, C., and Heflin, J., editors, *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I*, volume 10587 of *Lecture Notes in Computer Science*, pages 507–525. Springer.

[Tanon et al., 2020] Tanon, T. P., Weikum, G., and Suchanek, F. M. (2020). YAGO 4: A Reason-able Knowledge Base. In *The Semantic Web - ESWC 2020 - 17th International Conference, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings*, volume 12123 of *Lecture Notes in Computer Science*, pages 583–596. Springer.

[Terolli et al., 2020] Terolli, E., Ernst, P., and Weikum, G. (2020). Focused query expansion with entity cores for patient-centric health search. In *The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, Athens, Greece, November 2-6, 2020, Proceedings, Part I*, volume 12506 of *Lecture Notes in Computer Science*, pages 547–564. Springer.

[Tiddi et al., 2014] Tiddi, I., d'Aquin, M., and Motta, E. (2014). Dedalo: Looking for Clusters Explanations in a Labyrinth of Linked Data. In *The Semantic Web - ESWC 2014 - Trends and Challenges - 11th International Conference*, pages 333–348. Springer.

[Tiddi et al., 2015] Tiddi, I., d'Aquin, M., and Motta, E. (2015). Data Patterns Explained with Linked Data. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2015, Porto, Portugal, September 7-11, 2015, Proceedings, Part III*, volume 9286 of *Lecture Notes in Computer Science*, pages 271–275. Springer.

[Tran et al., 2018] Tran, D. M., d'Amato, C., Nguyen, B. T., and Tettamanzi, A. G. B. (2018). Comparing Rule Evaluation Metrics for the Evolutionary Discovery of Multi-relational Association Rules in the Semantic Web. In *Genetic Programming - 21st European Conference, EuroGP 2018, Parma, Italy, April 4-6, 2018, Proceedings*, volume 10781 of *Lecture Notes in Computer Science*, pages 289–305. Springer.

[Tran et al., 2016] Tran, H. D., Stepanova, D., Gad-Elrab, M. H., Lisi, F. A., and Weikum, G. (2016). Towards Nonmonotonic Relational Learning from Knowledge Graphs. In *Inductive Logic Programming - 26th International Conference, ILP 2016,*

London, UK, September 4-6, 2016, Revised Selected Papers, volume 10326 of *Lecture Notes in Computer Science*, pages 94–107. Springer.

[Tran et al., 2020] Tran, T., Gad-Elrab, M. H., Stepanova, D., Kharlamov, E., and Strötgen, J. (2020). Fast computation of explanations for inconsistency in large-scale knowledge graphs. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 2613–2619. ACM / IW3C2.

[Trouillon et al., 2016] Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., and Bouchard, G. (2016). Complex Embeddings for Simple Link Prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org.

[Tylenda et al., 2014a] Tylenda, T., Kondreddi, S. K., and Weikum, G. (2014a). Spotting Knowledge Base Facts in Web Texts. In *Proceedings of the 4th Workshop on Automated Knowledge Base Construction*, pages 1–6.

[Tylenda et al., 2014b] Tylenda, T., Wang, Y., and Weikum, G. (2014b). Spotting facts in the wild. In *Workshop on Automatic Creation and Curation of Knowledge Bases*.

[Urbani et al., 2016] Urbani, J., Jacobs, C. J. H., and Krötzsch, M. (2016). Column-Oriented Datalog Materialization for Large Knowledge Graphs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, pages 258–264. AAAI Press.

[Vahdati et al., 2018] Vahdati, S., Palma, G., Nath, R. J., Lange, C., Auer, S., and Vidal, M. (2018). Unveiling Scholarly Communities over Knowledge Graphs. In *Digital Libraries for Open Knowledge, 22nd International Conference on Theory and Practice of Digital Libraries, TPDL 2018, Porto, Portugal, September 10-13, 2018, Proceedings*, volume 11057 of *Lecture Notes in Computer Science*, pages 103–115. Springer.

[Vrandecic and Krötzsch, 2014] Vrandecic, D. and Krötzsch, M. (2014). Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

[Wan et al., 2019] Wan, H., Zhang, Y., Zhang, J., and Tang, J. (2019). Aminer: Search and mining of academic social networks. *Data Intelligence Nsl*, 1(1):58–76.

[Wang et al., 2019a] Wang, C., Pan, S., Hu, R., Long, G., Jiang, J., and Zhang, C. (2019a). Attributed Graph Clustering: A Deep Attentional Embedding Approach. In

Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019, pages 3670–3676. IJCAI.

[Wang et al., 2020] Wang, P., Stepanova, D., Domokos, C., and Kolter, J. Z. (2020). Differentiable learning of numerical rules in knowledge graphs. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

[Wang et al., 2017] Wang, Q., Mao, Z., Wang, B., and Guo, L. (2017). Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743.

[Wang et al., 2015] Wang, Q., Wang, B., and Guo, L. (2015). Knowledge Base Completion Using Embeddings and Rules. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 1859–1866. AAAI Press.

[Wang et al., 2019b] Wang, X., Wang, D., Xu, C., He, X., Cao, Y., and Chua, T. (2019b). Explainable reasoning over knowledge graphs for recommendation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 5329–5336. AAAI Press.

[Wang and Li, 2015] Wang, Z. and Li, J. (2015). RDF2Rules: Learning Rules from RDF Knowledge Bases by Mining Frequent Predicate Cycles. *CoRR*, abs/1512.07734.

[Wang et al., 2014a] Wang, Z., Zhang, J., Feng, J., and Chen, Z. (2014a). Knowledge Graph and Text Jointly Embedding. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1591–1601. ACL.

[Wang et al., 2014b] Wang, Z., Zhang, J., Feng, J., and Chen, Z. (2014b). Knowledge Graph Embedding by Translating on Hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, pages 1112–1119. AAAI Press.

[Wei et al., 2015] Wei, Z., Zhao, J., Liu, K., Qi, Z., Sun, Z., and Tian, G. (2015). Large-scale Knowledge Base Completion: Inferring via Grounding Network Sampling over Selected Instances. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 1331–1340. ACM.

[Weikum et al., 2020] Weikum, G., Dong, L., Razniewski, S., and Suchanek, F. M. (2020). Machine Knowledge: Creation and Curation of Comprehensive Knowledge Bases. *CoRR*, abs/2009.11564.

[Wojtusiak, 2014] Wojtusiak, J. (2014). Rule Learning in Healthcare and Health Services Research. In *Machine Learning in Healthcare Informatics*, volume 56 of *Intelligent Systems Reference Library*, pages 131–145. Springer.

[Wrobel, 1996] Wrobel, S. (1996). First Order Theory Refinement. In *Advances in Inductive Logic Programming*, pages 14 — 33. IOS Press.

[Xiao et al., 2017] Xiao, H., Huang, M., Meng, L., and Zhu, X. (2017). SSP: Semantic Space Projection for Knowledge Graph Embedding with Text Descriptions. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3104–3110. AAAI Press.

[Xie et al., 2016a] Xie, J., Girshick, R. B., and Farhadi, A. (2016a). Unsupervised Deep Embedding for Clustering Analysis. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 478–487. JMLR.org.

[Xie et al., 2016b] Xie, R., Liu, Z., Jia, J., Luan, H., and Sun, M. (2016b). Representation Learning of Knowledge Graphs with Entity Descriptions. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 2659–2665. AAAI Press.

[Xie et al., 2017] Xie, R., Liu, Z., Luan, H., and Sun, M. (2017). Image-embodied Knowledge Representation Learning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 3140–3146. IJCAI.

[Yang et al., 2015] Yang, B., Yih, W., He, X., Gao, J., and Deng, L. (2015). Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[Yang et al., 2017] Yang, F., Yang, Z., and Cohen, W. W. (2017). Differentiable Learning of Logical Rules for Knowledge Base Reasoning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 2319–2328.

[Yin et al., 2008] Yin, X., Han, J., and Philip, S. Y. (2008). Truth Discovery with Multiple Conflicting Information Providers on the Web. *IEEE Transactions on Knowledge and Data Engineering*, 20(6):796–808.

[Zeng et al., 2014] Zeng, Q., Patel, J. M., and Page, D. (2014). QuickFOIL: Scalable Inductive Logic Programming. *Proceedings of the VLDB Endowment*, 8(3):197–208.

[Zhang et al., 2019a] Zhang, W., Paudel, B., Wang, L., Chen, J., Zhu, H., Zhang, W., Bernstein, A., and Chen, H. (2019a). Iteratively Learning Embeddings and Rules for Knowledge Graph Reasoning. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 2366–2377. ACM.

[Zhang et al., 2019b] Zhang, W., Paudel, B., Zhang, W., Bernstein, A., and Chen, H. (2019b). Interaction Embeddings for Prediction and Explanation in Knowledge Graphs. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 96–104. ACM, ACM.

[Zhang et al., 2020] Zhang, Z., Cai, J., Zhang, Y., and Wang, J. (2020). Learning Hierarchy-Aware Knowledge Graph Embeddings for Link Prediction. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 3065–3072. AAAI Press.

[Zupanc and Davis, 2018] Zupanc, K. and Davis, J. (2018). Estimating Rule Quality for Knowledge Base Completion with the Relationship between Coverage Assumption. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1073–1081. ACM.