2018

Domain-specific Knowledge Extraction from the Web of Data

Sarasi Lalithsena
Wright State University

Domain-specific Knowledge Extraction from the Web of Data

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

By

Sarasi Lalithsena
B.Sc., University of Colombo, 2008

2018
Wright State University

WRIGHT STATE UNIVERSITY
GRADUATE SCHOOL

March 26, 2018

I HEREBY RECOMMEND THAT THE DISSERTATION PREPARED UNDER MY SUPERVISION BY Sarasi Lalithsena ENTITLED Domain-specific Knowledge Extraction from the Web of Data BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Doctor of Philosophy.

Amit Sheth, Ph.D.
Dissertation Director

Michael Raymer, Ph.D.
Director, Computer Science and Engineering Ph.D. Program

Barry Milligan, Ph.D.
Interim Dean of the Graduate School

Committee on Final Examination

Amit Sheth, Ph.D.

Krishnaprasad Thirunarayan, Ph.D.

Derek Doran, Ph.D.

Cory Henson, Ph.D.

Saeedeh Shekarpour, Ph.D.

ABSTRACT

Lalithsena, Sarasi. PhD, Department of Computer Science and Engineering, Wright State University, 2018. Domain-specific Knowledge Extraction from the Web of Data.

Domain knowledge plays a significant role in powering a number of intelligent applications such as entity recommendation, question answering, data analytics, and knowledge discovery. Recent advances in Artificial Intelligence and Semantic Web communities have contributed to the representation and creation of this domain knowledge in a machine-readable form. This has resulted in a large collection of structured datasets on the Web which is commonly referred to as the Web of data. The Web of data continues to grow rapidly since its inception, which poses a number of challenges in developing intelligent applications that can benefit from its use. Majority of these applications are focused on a particular domain. Hence they can benefit from a relevant portion of the Web of Data. For example, a movie recommendation application predominantly requires knowledge of the movie domain and a biomedical knowledge discovery application predominantly requires relevant knowledge on the genes, proteins, chemicals, disorders and their interactions. Using the entire Web of data is both unnecessary and computationally intensive, and the irrelevant portion can add to the noise which may negatively impact the performance of the application. This motivates the need to identify and extract relevant data for domain-specific applications from the Web of data. Therefore, this dissertation studies the problem of domain-specific knowledge extraction from the Web of data.

The rapid growth of the Web of data takes place in three dimensions: 1) the number of knowledge graphs, 2) the size of the individual knowledge graph, and 3) the domain coverage. For example, the Linked Open Data (LOD), which is a collection of interlinked knowledge graphs on the Web, started with 12 datasets in 2007, and has evolved to more than 1100 datasets in 2017. DBpedia, which is a knowledge graph in the LOD, started with 3 million entities and 400 million relationships in

2012, and now has grown up to 38.3 million entities and 3 billion relationships. As we are interested in domain-specific applications and the domain of interest is already known, we propose to use the domain to restrict/reduce the other two dimensions from the Web of data. Reducing the first dimension requires to reduce the number of knowledge graphs by identifying relevant knowledge graphs to the domain. However, this still may results in large knowledge graphs such as DBpedia, Freebase, and YAGO that cover multiple domains including our domain of interest. Hence, it is required to reduce the size of the knowledge graphs by identifying the relevant portion of a large knowledge graph. This leads to two key research problems to address in this dissertation. (1) Can we identify the relevant knowledge graphs that represent a domain? and (2) Can we identify the relevant portion of a cross-domain knowledge graphs to represent the domain?

A solution to the first problem requires automatically identifying the domain represented by each knowledge graph. This can be challenging for several reasons: 1) Knowledge graphs represent domains at different levels of abstractions and specificity, 2) a single knowledge graph can represent multiple domains (i.e., cross-domain knowledge graphs), and 3) the represented domains by knowledge graphs keep evolving.

We propose to use existing crowd-sourced knowledge bases with their schema to automatically identify the domains and show its effectiveness in finding relevant knowledge graphs for specific domains. The challenge in addressing the second issue is the nature of the relationships connecting entities in these knowledge graphs. There are two types of relationships: 1) Hierarchical relationships, and 2) non-hierarchical relationships. While hierarchical relationships connect in-domain and out-of-domain entities using the same relationship type and hence represent uniform semantics, non-hierarchical relationships connect in-domain entities and out-of-domain entities using different relationships, i.e., they capture diverse semantics. We propose both data-driven and knowledge-driven approaches to capture the domain-relevancy of both hierarchical and non-hierarchical relationships. The solution encodes human knowledge on the domain specificity as probabilistic statements and infers the most probable explanation which captures the domain specificity of concepts and relation-

ships of the original knowledge graph. We present use cases related to entity recommendation for multiple domains to show the effectiveness in extracting the domain-specific subgraph. The domain-specific subgraphs extracted by our approach were 80% smaller in size in terms of the number of paths compared to the original knowledge graph and resulted in more than tenfold reduction of required computational time for entity recommendation task, yet produced better accuracy. We believe that this work will have major impact in utilizing knowledge graphs for domain-specific applications, especially with the extensive growth in the creation of knowledge graphs.

Contents

List of Figures

List of Tables

ACKNOWLEDGEMENTS

My graduate life at Kno.e.sis Center, Wright State University adds a memorable chapter to my life. This incredible journey would not have been possible without the people I met along the way.

I would like to start by thanking my advisor Dr. Amit P. Sheth for his continuous support throughout this journey. The opportunity he gave me to work with him changed my life in many ways. His research vision forced me to think outside the box and to identify impactful research problems. He has always emphasized the importance of social skills and makes sure his students develop these skills every day. I was lucky enough to benefit from the ecosystem he created where I was able to interact with my colleagues and to learn from them. During my graduate life, there were many times that things did not work as I expected. Dr. Sheth was patient with me during these times, and his understanding kept me moving forward. I am deeply grateful for his commitment towards the success of his students.

I would also like to express my gratitude to the rest of my dissertation committee Dr. T.K. Prasad, Dr. Cory Henson, Dr. Derek Doran, and Dr. Saeedeh Shekarpour. I used to discuss technical aspects of my research with Dr. Prasad from the beginning. His critical thinking and attention to detail have helped to improve my technical skills. I am grateful to the valuable time he selflessly spent with me whenever I needed his guidance. Even though I knew Cory since I joined Kno.e.sis, I got the opportunity to work with him closely during my Bosch internship. He was a great mentor, and I learned many things from him. Most importantly his unconventional way of looking at a problem greatly influenced me to become a better researcher. My interaction with Derek was short, yet his valuable comments during my dissertation proposal were helpful to shape up my dissertation. I am also grateful to Saeedeh for providing comments on my dissertation and papers under enormous pressure of time. Apart from my committee, I would also like to express my gratitude for Dr. Valerie Shalin and Dr. Tanvi Banerjee for their feedback.

Prateek Jain was my first mentor in Kno.e.sis. He introduced me to the Linked Open Data world, and we closely worked together during the first part of my dissertation. I learned the process of conducting research from Prateek. I am indebted to him always for that. On the second part of my dissertation, I worked closely with Pavan Kapanipathi. When I was struggling to find use cases which were critical to my research, he helped me with his expertise on recommendation systems to come up with a good use case. With this, I was able to publish two of my most important publications. He has helped enormously to improve my writing skills as well. His continuous support as a mentor, a colleague, and a friend helped me to overcome the struggles of my Ph.D. life. I would also like to appreciate the help I got from Dr. Pascal Hitzler in getting my first paper published. I am grateful for both Luis Tari and Steven Gustafson for their guidance during my internship at GE. I was able to identify the second problem of my dissertation while I was working with them. Michael Hausenblas and Richard Cyganiak gave me my first internship opportunity and taught me how to have fun while doing research.

I would not survive my graduate life without the strong support from my family. My husband Sujan's encouragement and constant belief in my abilities were the driving force for me to reach the end. His unconditional support both in good and bad times makes me stronger every day. He is not only an understanding husband but also the most supportive colleague. He reviewed and critiqued each paper I wrote and gave feedback for every presentation I did. I feel lucky to share this incredible journey with him and looking forward to continuing for many many years to come. My parents gave me their blessings and endless support throughout this journey. They always believed that I have the potential to succeed and silently cope the fact that their daughter is living thousands of miles away from them. After I left home for my graduate studies, my younger brother took all the responsibilities back in home upon himself and did not leave a chance for me to worry about anything. I hope I made my family proud.

My acknowledgment will be incomplete without mentioning the great friends and colleagues I met in Kno.e.sis. We have shared our best and worst times together by supporting each other. I would like to start with Ajith Ranabahu who helped us to settle down in the USA. I would like to thank Delroy, Wenbo, Lu, Pramod, Kalpa, Ashutosh, Vinh, Hemant, Sanjaya, and Shreyansh for reviewing my papers and/or insightful discussions. I love the company of Revathy, Lakshika, Siva, Adarsh, Gaurish, and Vaikunth. Their great company made my graduate life fun and entertaining. It was great working with Swati, Manas, Joy, Pavan, Nishita, and Roopteja. I learned how hard to become a good mentor by working with them. I got to spend a great deal of time with the younger generation in Kno.e.sis. I am going to miss that youthful energy. I also like to express my gratitude to Kno.e.sis alumni who helped me in many ways especially the great support I got from Pablo and Cartic during my job search. I would also like to thank Tonya Davis, Jennifer Limoli, and CS department staff for making my life easier by handling required paperwork. Outside of my academic world, Sri Lankan community in Dayton became the home away from home. I thank everyone for making me feel home.

Finally, I would like to acknowledge the funding sources. This dissertation is supported by the National Science Foundation under awards 1143717 III: EAGER Expressive Scalable Querying over Linked Open Data and EAR-1520870 Hazards SEES: Social and Physical Sensing Enabled Decision Support for Disaster Management". Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Dedicated to

my parents, Samantha and Lalith

my husband Sujan and my brother Kushan

1

Introduction

Thesis Statement: *Applications serving specific domains can be benefited by identifying the relevant knowledge from rapidly growing structured data on the Web. This can be accomplished by: (a) leveraging existing crowd-sourced knowledge bases as a reference schema to automatically determine the domains of knowledge graphs, and (b) exploiting the semantics and structure of entities and relationships with statistical techniques to extract the relevant portions of the knowledge graphs.*

Domain knowledge plays an essential role in human decision making [Devine and Kozlowski 1995] and problem solving [Nokes et al. 2010]. It can be helpful for various tasks ranging from deciding a movie to watch, planning a family vacation, to diagnosing disease. For example, I enjoy watching movies which discuss humanistic issues such as civil rights, war, terrorism, and slave trade. I recently enjoyed watching a couple of movies by *Steven Spielberg* of the same genre such as *Schindler's List*, *Saving Private Ryan*, *Lincoln* and *Bridge of Spies*. Now when I want to find a new movie to watch, I look for recent movies of the same genre and/or by the same director. Previous studies have shown that human decision making depends on their respective domain expertise [Devine and Kozlowski 1995].

Following the same analogy, many computer applications also use the domain knowledge in different ways to accomplish their tasks and achieve their goals. For example, IBM Watson [Ferrucci et al. 2010], which was built as a question answering system, used the knowledge about the type of candidate answers to rank the candidate answers for a given question. Given the question "*In 1610, Galileo named the moons of this planet for the Medic brothers.*" and the candidate answers "*Telescope*," "*Sidereus Nuncius*," "*Jupiter*," "*Giovanni Medici*," and "*Ganymede*", knowing that "*Jupiter*" is a *planet* helped Watson to come up with the correct answer. A movie recommendation system [Ostuni et al. 2013] uses knowledge about the movies such as genre, actors, and directors to recommend new movies to the user.

However, this requires the relevant knowledge to be represented in a machine-readable format. Over the years, Artificial Intelligent and Semantic Web communities have contributed to creating these knowledge representations which typically consist of entities and their relationships represented as a graph-based data model. These representations are referred to as *knowledge graphs*. For example, Figure 1.1 shows a snippet of the knowledge about *Cardiovascular disease* from two knowledge graphs: DBpedia[1] and Bio2RDF[2]. Each knowledge graph captures different aspects of *Cardiovascular disease*. While Bio2RDF has the knowledge on side effects and subcategories of the *Cardiovascular disease*, DBpedia has the knowledge of the people who have died from the disease, people who have studied the disease and the journals which publish research on the disease. The concepts in different KGs can be inter-connected via relationships. In this example, `owl:SameAs` relationship connects the *Cardiovascular disease* in two KGs. This has resulted in a large collection of knowledge graphs available on the Web which has also been referred to as the *Web of data*.

Web of data has continued to grow rapidly. This growth can be categorized into three dimensions:

1. The number of knowledge graphs: Linked Open Data (LOD) is a collection of interlinked knowledge graphs[3] on the Web. Starting with 12 knowledge graphs in 2007, it has evolved to more

[1] `http://wiki.dbpedia.org/`

[2] `http://bio2rdf.org/`

[3] Generally knowledge graphs in LOD are referred as just datasets. For the consistency, we use the term knowledge graph to refer to a LOD dataset throughout this dissertation.

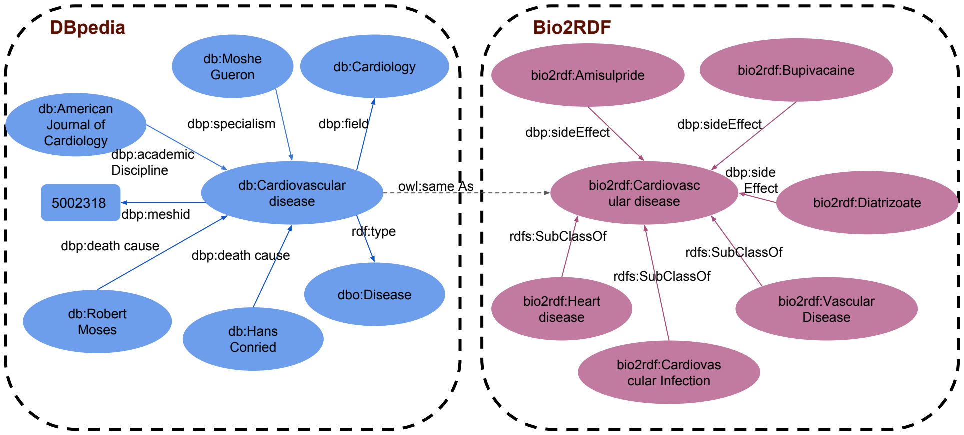


Figure 1.1: Snippet of Web of data

than 1100 knowledge graphs in 2017.[4]

2. Size of the knowledge graphs: One of the prominent knowledge graphs in LOD is DBpedia. Starting from 3 million entities and 400 million facts in 2012, it has now grown to 38.3 million entities and 3 billion facts. Table 1.1 shows the annual growth of the DBpedia English version.

3. The number of domains: Web of data cover a wide variety of domains such as government, geography, biology, publication, history, and entertainment. While some knowledge graphs capture knowledge about specific domains (e.g., Uniport on protein), others capture knowledge covering a number of domains (e.g., DBpedia contains knowledge about a number of domains).

The sheer volume of structured and diverse knowledge available on the Web challenges the consumers in identifying and extracting the relevant Web of data which best suits their application domain and/or tasks. **This dissertation studies the problem of extracting domain-specific knowledge from the Web of data.**

[4] http://lod-cloud.net/

Version	No of Entities	No of Facts
2012	3.77M	400M
2013	4.0M	470M
2014	4.5M	583M
2015	6.2M	1B
2016	6.6M	1.7B

Table 1.1: Growth of DBpedia English version

1.1 Domain-specific knowledge extraction from the Web of data

1.1.1 Knowledge graphs

As researchers identified the importance of embedding knowledge into computational algorithms, they were interested in the symbolic representation of the world knowledge in a machine interpretable way. In the mid 1970s, Artificial Intelligence (AI) research community argued that the captured knowledge can be used as computational models in AI systems. Since 1980, AI community started to use the term “Ontology” to refer to the symbolic representation of the captured knowledge [Gruber 1995]. During that time, two popular symbolic representations were WordNet [Miller 1995], which is a lexical dataset to group English words into sets of synonyms, and Cyc [Guha and Lenat 1993] which captures the common sense knowledge. In 2000, Taalee used a similar representation “WorldModel” (for multi-domain ontology or knowledge graph) to power its faceted and semantic search, semantic , semantic browsing, semantic personalization and semantic advertisement. [Sheth et al. 2001]. SCORE discussed an automatic approach for metadata extraction and annotation to define ontological components in 2002 [Sheth et al. 2002]. In 2012, Google introduced “Google Knowledge Graph”[5] which is their representation of the facts and is the backbone of Google’s entity search. After this, the term “Knowledge Graph” became prominent among the applied research community and commercial systems to refer to any graph-based data model to represent the facts.

1.1.2 Web of data

Generating and capturing knowledge took a drastic turn when Sir Tim Berners-Lee introduced the concept of Semantic Web, “a Web of data that can be processed directly and indirectly by machines” [Berners-Lee et al. 2000].[6] Semantic Web adopts the Resource Description Framework (RDF) [Lassila and Swick 1999] as its data model which represents knowledge as a labeled directed multi-graph.[7] For example, in Figure 1.1, `db:Cardiovascular disease` and `db:hans conried` are two nodes in the graph and the edge `dbp:death cause` represents the relationship between the two nodes. Nodes and edges in the graph are represented using URLs. For example, *Cardiovascular disease* in DBpedia is represented as `http://dbpedia.org/resource/Cardiovascular_disease`.

The recent adaptation of Semantic Web technologies has created trillions of facts on the Web. One of the prominent collection of a subset of Web of data is LOD. It follows four principles in publishing[8] [Bizer et al. 2009] and interlinking data on the Web. Another core contribution to the Web of data is Schema.org which is a vocabulary developed collaboratively by three search engine pioneers: Google, Bing, and Yahoo, to provide annotations on the web pages. Web Data Commons project [Meusel et al. 2015] extracts the structured data on the Web including schema.org annotation. Recent crawl by this project of 541 million HTML pages collected around 6 billion entities and 24 billion triples across multiple domains.[9]

[5] https://www.blog.google/products/search/introducing-knowledge-graph-things-not/

[6] `https://en.wikipedia.org/wiki/Semantic_Web`

[7] `https://www.w3.org/RDF/`

[8] `https://en.wikipedia.org/wiki/Linked_data`

[9] `http://webdatacommons.org/structureddata/2015-11/stats/stats.html#results-2015-1`

1.1.3 Utilization of Web of data for domain-specific application

In recent years, Web of data has been increasingly used as background knowledge for various applications in different ways. A named entity linking [Mendes et al. 2011] application uses the instances of well-known knowledge graph DBpedia [Auer et al. 2007] as a dictionary. Implicit entity recognition systems [Perera et al. 2015] [Perera et al. 2016] use existing knowledge graphs to represent the domain knowledge about entities. An early version of IBM Watson for playing Jeopardy used the type information from YAGO [Hoffart et al. 2013] to improve its accuracy in question answering. Recommendation [Ostuni et al. 2013] and document similarity/relevance [Schuhmacher and Ponzetto 2014] applications use relationships between entities of the Web of data to calculate the relatedness between entities. Knowledge discovery applications use knowledge graphs to find complex relationships among entities [Cameron et al. 2015].

While some of these applications require domain agnostic knowledge, most of the applications are domain-specific and require only the domain-specific knowledge. For example, a movie recommendation application requires knowledge of the movie domain and a biomedical knowledge discovery application would require knowledge of the genes, proteins, chemicals, disorders and their interactions. Twitris, which is a social media analytics tool that runs analysis on specific topics like natural disasters and elections, uses domain knowledge that is relevant to the specific topics [Jadhav et al. 2010].

To effectively leverage the existing Web of data for the domain-specific applications, it is important and adequate to identify the relevant portion of Web of data. The process of extracting relevant portion of the Web of data for domain-specific applications depends on the three dimensions of the growth of Web of data: (1) Number of knowledge graphs, (2) size of the knowledge graphs, and (3) number of domains. For domain-specific applications, we propose to use the domain information to restrict/reduce the number and size of knowledge graphs to be used, overcoming the challenges resulting from exclusive growth of Web of data. Hence, given the domain for the domain-specific application, identifying relevant Web of data consists of two steps.

1. Identify the relevant knowledge graphs for a given domain: Identifying the relevant knowledge graphs mainly has been done using manual inspection of the knowledge graphs and also based on the popularity of the knowledge graphs. For example, DBTune[10] captures the music related information while Geonames[11] captures geospatial information. Even though knowledge graphs such as DBTune and Geonames are well known and widely used in the community, there are other hidden gems like Climbdata[12] and Lingvoj[13] that have not been adequately exploited. Climbdata provides information about climbing routes, whereas Lingvoj provides various ways different languages relate to things such as country and organization.

2. Identifying the relevant portion of the knowledge graphs: Even though identifying relevant knowledge graphs narrow down the search space, it does not solve the problem entirely. Some of the identified knowledge graphs can be large and contain knowledge from diverse domains and hence, can contain the knowledge not specific to a domain. In fact, knowledge graphs in the LOD that are more popular belong to this category. DBpedia, YAGO, Freebase [Bollacker et al. 2008][14] and WikiData are knowledge graphs in the LOD cloud which are large and popular in the community for their coverage and availability. Schema.org data on the Web also comes under this category as there is no easier way to identify the relevant part for its consumers.

Hence, given the domain for the domain-specific application, this dissertation addresses these two key questions.

1. Can we automatically identify the relevant knowledge graphs to represent a domain adequately?

2. Can we automatically identify the relevant part of a knowledge graph to represent the domain?

[10] http://dbtune.org/

[11] https://datahub.io/dataset/geonames-semantic-web

[12] http://datahub.io/dataset/data-incubator-climb

[13] http://www.lingvoj.org/

[14] Google KG uses freebase and it is no longer being developed by the crowd

1.2 Challenges in extracting domain-specific knowledge from Web of data

Domain-specific knowledge extraction from Web of data is challenging for the following reasons:

- Different abstraction levels of domains: Domain of a knowledge graph can be defined at different abstraction levels. For example, both NCI Thesaurus[15] and Radiology Lexicon contain information regarding radiology domain. However, NCI Thesaurus covers general clinical care, translational and basic research. But, Radiology lexicon[16] contains fine-grained information regarding the radiology domain. While both cover the radiology information, the details of the abstraction levels are different.

- Cross-domain knowledge graphs: Some knowledge graphs can belong to more than one domain or topic. For example, DBpedia contains knowledge covering multiple domains such as music, movie, book, people, drug and protein. For some domains, these large cross-domain knowledge graphs may provide the up-to-date and more comprehensive information. DBpedia captures more up-to-date book information compared to Gutenberg knowledge graph[17] which also captures book information. In some cases, they can provide complementary information such as Disease information in DBpedia and Bio2RDF. However, domain-specific knowledge graphs may cover the fine-grained information than cross-domain knowledge graphs. For example, protein information is better represented in Uniport[18] compared to DBpedia.

- Evolving domains: Domains can evolve over time. It is not feasible to rely on a set of predefined tags or a predefined schema to describe domains.

- Relationships with diverse semantics connecting entities: Domain entities connect to in-domain

[15] `https://ncit.nci.nih.gov/ncitbrowser/`

[16] `http://www.radlex.org/`

[17] `https://github.com/mrcook/gutenberg_rdf`

[18] `http://www.uniprot.org/format/uniprot_rdf`

and out-of-domain entities via relationships. For example, `Brad Pitt` (an in-domain entity for the movie domain) is connected to `Rusty Ryan` (an in-domain entity for the movie domain) which is a fictional character in `Ocean's Eleven` via the relationship `portrayer` and also connected to city `Oklahoma` (an out-of-domain entity for the movie domain) via relationship `birthplace`. Also two in-domain entities `Brad Pitt` and `Angelina Jolie` have two relationships `spouse` and `colleague`. `colleague` relationship is more relevant in the movie domain, but `spouse` is not. Domain relationships (except hierarchical relationships) in a knowledge graph use different relationship labels for different relationships and contains different semantics. An algorithm needs to find the relationships that are relevant to the domain of these diverse relationships.

- Relationships with uniform semantics connecting entities: Hierarchical relationships in a knowledge graph use the same relationship label with uniform semantics. For example, Wikipedia category hierarchy uses `skos:broader` relationship to create a hierarchy of categories. But they connect entities from multiple domains. For example, Wikipedia category hierarchy assigns categories `Fast Food Mexican Restaurant` and `Building and Structures in Colorado` to the category `Chipotle`. The two categories represent two different domains even though they connect to the category `Chipotle` using the same relationship.

1.3 Dissertation Focus

In this dissertation, we study the problem of identifying relevant Web of data by addressing the challenges mentioned above. Specifically, we study the two questions described above in utilizing Web of data for domain-specific applications and present use cases for each of them.

1. Identifying the relevant knowledge graphs for a given application: We propose to create a registry of topics by automatically identifying the domains of each knowledge graph. We present and evaluate an application to identify relevant knowledge graphs by utilizing this registry of topics.

2. Identifying the relevant portion of a knowledge graph for a given application: We propose

techniques to extract relevant portion or subgraph from a large knowledge graph by capturing the semantics of both hierarchical and non-hierarchical relationships. We present a recommendation system as a use case of a domain-specific application for the domains of movie and book. We show that the quality of recommendations and efficiency of computation can be improved by using domain-specific knowledge. Furthermore, we analyze how the proposed techniques can be helpful in identifying the relevant portion of a heterogeneous domain such as US presidential election campaign.

The above two topics of investigations are divided into the following sub parts.

1.3.1 Automatic domain identification from Web of data

The key aspect in identifying the relevant knowledge graphs is identifying the domains represented by the knowledge graph. Existing techniques rely on the manually assigned predefined set of tags or keywords provided by users as descriptors in identifying the domain of the knowledge graphs. LOD data consumers mainly look at the LOD Cloud diagram[19] to identify the relevant knowledge graphs for their applications. LOD Cloud diagram is generated based on the knowledge graphs being added to the CKAN data hub.[20] CKAN allows data publishers to manually assign predefined sets of tags such as *media, geography, life sciences, publications, government, e-commerce, social web, user-generated content, schemata, and cross-domain* to classify the knowledge graphs to different domains. CKAN administrators manually review these assignments and use these tags to better organize the LOD diagram. This process has two problems: First, manual reviewing process has started to become unsustainable with the rapidly increasing number of knowledge graphs. Second, the diversity of the knowledge graphs makes it difficult to work with a fixed number of pre-defined tags. For example, it is hard to decide on proper tags for the Lingvoj knowledge graph using the predefined CKAN tags. Additionally, CKAN administrators are unlikely to have the required

[19] `http://lod-cloud.net/`

[20] https://datahub.io/

time and domain knowledge to make proper classification or labeling. To deal with the above mentioned issues in identifying relevant knowledge graphs, we need to have a mechanism which automatically tags the knowledge graphs using an evolving vocabulary to capture the diversity and different abstraction levels of the domains.

In this work, we provide an approach to automatically identify the domains of knowledge graphs by utilizing knowledge sources in other LOD knowledge graphs as the vocabulary, such as the hierarchy within Freebase. We believe community-driven knowledge sources and their hierarchy enable us to cover a wide variety of domains and also track their evolution. Our approach assigns domains to the knowledge graphs in LOD using Freebase types. While we have developed our approach with Freebase in mind, it is adaptable to any other hierarchy similar to Freebase. We present a search application built on the identified domains to search and identify relevant knowledge graphs within LOD. We also provide an evaluation to validate the domains identified and the effectiveness of the identified domains for searching knowledge graphs in comparison to the existing systems.

1.3.2 Identifying domain-specific subgraph

As discussed earlier, after identifying the knowledge graph, it may be necessary to identify the relevant portion of knowledge graph. Current applications that require domain-specific knowledge extract the relevant part of the knowledge graph as a subgraph by navigating a predefined number of hops from a set of given entities that represent a domain. Most of these applications set the predefined number of hops between 2 and 4 [Ostuni et al. 2013] [Schuhmacher and Ponzetto 2014] [Hulpuş et al. 2015] [Musto et al. 2014] [Passant 2010]. There are several limitations to this approach:

- Navigating a predefined number of hops can still cover a large part of the knowledge graph. For example, Figure 1.2 shows the percentage of unique entities reached by navigating up to 3-hops starting from $3,072$ movie entities in the MovieLens dataset [Harper and Konstan 2016] which is a popular dataset for the movie recommendation. The subgraph extracted by navigating 3-hops encompasses 66% of the DBpedia entities. This can be explained by the structure of DBpedia

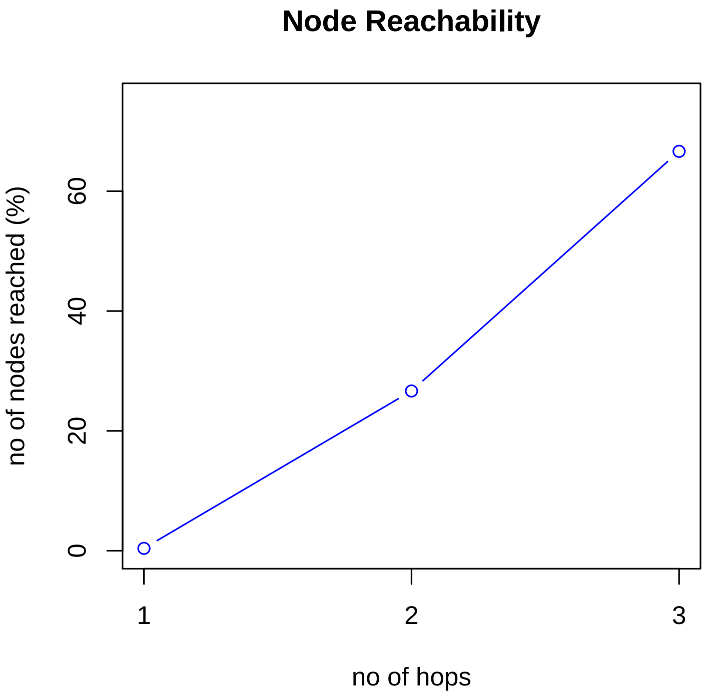


Figure 1.2: Percentage of nodes reached via n-hops for movie domain in DBpedia

knowledge graph. The mean shortest path between entities in DBpedia is around 5-hops,[21] and navigating 4-hops from a set of entities can cover a significant part of this large knowledge graph.

- Given a domain, not all entities connected via a predefined number of hops are necessarily relevant. For example, two movies connected because their directors are known for action movies is important for a movie recommendation system, whereas two movies connected because the directors of these movies died in the same city is irrelevant. Hence, extracting a subgraph with a predefined number of hops can encompass paths that do not contribute to the accuracy of the application.

When domain-specific applications leverage the large cross-domain knowledge graphs, the usage of complete knowledge graph or the subgraph created by navigating a predefined number of hops

[21] http://konect.uni-koblenz.de/networks/dbpedia-all

can be computationally intensive and result in suboptimal results. To deal with these issues in identifying the domain-specific part of the knowledge graph, we need a way to reduce the original graph to a domain-specific subgraph without compromising the accuracy of the application.

Our approach for domain-specific subgraph selection leverages three key elements in the knowledge graph.

1. The semantics of relationships: Relationships play a key role to extract the domain-specific subgraph. For example, if there is a way to identify that the relationship *genre* is specific to the movie domain but not the relationship *death city*, then we can use the relationship *genre* to navigate the graph but ignore the paths with *deathCity*.

2. The semantics of entities: Semantics of the entities such as types can be helpful to extract the domain-specific subgraph. For example, *James Cameron* is a type of *Director*, which is strongly associated with the type *Movie* is a good indicator to determine that the entity *James Cameron* is relevant to the *movie* domain.

3. Structure: The structure of a knowledge graph provides evidence in determining the domain-specific aspect of a graph. For example, how the *Director* typed entities are linked with the *Movie* typed entities versus *Country* typed entities are linked with the *Movie* typed entities can be helpful in extracting the domain-specific subgraph.

We use these three key elements with statistical measures to extract the domain-specific subgraph.

Domain-specific subgraph selection is challenged by the nature of the relationships connecting entities in these datasets. Corresponding to the two types of relationships - hierarchical relationships and non-hierarchical relationships, we divide this study into two parts based on the nature of the relationship.

1.3.2.1 Identifying domain-specific subgraph with non-hierarchical relationships

Our proposed methodology for non-hierarchical relationships treats relationships as first-class elements because relationships induce semantics between entities and, hence, can play a significant role in identifying domain-specific knowledge. We propose to identify the domain-specific relationships among all named non-hierarchical relationships in the knowledge graph. Then we use only these domain-specific relationships to restrict the original knowledge graph to a domain-specific subgraph. In order to identify the domain-specific relationships, we came up with domain-specific measures to capture the domain-specificity of these named relationships. These measures are based on the strength of association of domain entities to the relationships, and the strength of the association is calculated by the statistic and semantic-based metrics of the relationships.

1.3.2.2 Identifying domain-specific subgraph with hierarchical relationships

Capturing the domain-specificity of hierarchical relationships is challenging as it does not have different named relationships among entities. Even though it represents multiple aspects, it uses the same hierarchical relationship which makes it difficult to use the same techniques used for non-hierarchical relationships. Hence, we focused only on the entities and structural properties when it comes to hierarchical relationships. We propose an evidence-based approach for extracting a domain-specific subgraph from a hierarchical KG. Given a domain, the domain-specificity of entities are determined by combining different types of evidence using a probabilistic framework. To systematically combine the different sources of evidence and to manage the uncertainty associated with each of the sources, we use the probabilistic soft logic (PSL) framework [Bach et al. 2017].

For both the above two problems, we study a recommendation use cases on multiple domains to show the effectiveness of the domain-specific subgraph generated by our approach.

1.4 Dissertation Organization

The rest of the dissertation is organized as follows: Chapter 2 presents a background on important concepts in knowledge representation related to this dissertation and related work for identifying domains of the knowledge graphs and extracting domain-specific subgraph. The related work section for domain identification details the state-of-the-art for domain identification with a focus on identifying relevant knowledge graphs and the related work section for domain-specific subgraph extraction discusses how existing domain-specific applications have used these knowledge graphs.

Chapter 3 details the approach to automatically identify the domains of the knowledge graphs using an existing crowded source knowledge sources. It also presents the evaluation of the identified domain(s) via the proposed technique with an extensive user study and the impact of the identified domains to find relevant knowledge graphs in comparison to the state-of-the-art techniques.

Chapter 4 motivates the problem of extracting subgraph from large knowledge graphs and discusses two approaches to extract domain-specific subgraphs from large knowledge graphs with non-hierarchical relationships. The quality of the approaches is evaluated using a recommendation application which is the most common domain-specific application leveraging knowledge graphs.

Chapter 5 highlights the significance of hierarchical relationships in knowledge graphs and details an approach to extract domain-specific subgraph from large knowledge graphs with hierarchical relationships. It uses the same evaluation setup used in Chapter 4 to assess the effectiveness of the approach. Furthermore, it compares the proposed approach with an existing technique to identify domain-specific concepts in hierarchical knowledge graphs on domains movie, book, and US presidential election campaign.

Chapter 6 concludes the dissertation by summarizing the key insights of the contributions and also details the future research direction in this area and possible extensions to this work.

2

Background and Related Work

2.1 Background

This section discusses the evolution of knowledge representation and other related background to the dissertation.

2.1.1 Evolution of Knowledge Representation

The importance of knowledge for AI systems dates back to early stages of AI. One of the earliest works along this line was General Problem Solver (GPS) [Newell et al. 1959] that expresses the the domain knowledge and query as logical formulas (a directed graph) and then reason over it. However, this was only limited to solve simple problems and failed to scale to most real-world problems. Later on, expert systems [Feigenbaum et al. 1970] [Shortliffe 1974] became prominent in the AI for focused domains such as chemistry and medicine. Expert systems took the knowledge-based approach which typically consists of knowledge bases (facts and rules) and inference engines to solve a given problem. During this time, the term "Ontology" became prominent as a way to represent the knowledge and ontology engineering became the field to construct knowledge bases which can be reused in a number of projects. One of the leading projects in the ontology engineering area was the Cyc project [Lenat et al. 1985] which is used to capture the expert and common sense knowledge in a machine-readable way. One more notable work in the expert systems was a reasoning system [Russell et al. 1995] which uses knowledge bases for query assertions. [Mena et al. 2000] [Shah and Sheth 1999] were early search and browsing applications which used domain specific metadata and ontologies.

2.1.2 Semantic Web, Linked Data, and Knowledge Graphs

Semantic Web, coined by Sir Tim Berners-Lee, is a knowledge representation framework that introduced a standard set of representation formats, primitives, and conventions to represent knowledge. Sir Tim Berners-Lee envisioned the Semantic Web as the data on the Web (Web of Data). Semantic Web introduced three main representation formats; Resource Description Format (RDF), RDF Schema [McBride 2004], and Web Ontology Language [McGuinness et al. 2004]. As we mentioned above, Semantic Web uses a directed labelled graph data model which represents a fact as an edge in the graph using subject, predicate, and object. Subject and object are the nodes in the graph, and the predicate is the edge.

A Semantic Web dataset mainly consists of the schema-level knowledge and instance-level knowledge. For example, in Figure 1.1, `bio2rdf:Heart disease` is a `rdfs:subClassOf` `bio2rdf:Cardiovascular disease` is schema-level knowledge as it defines a relationship between two groups of objects. These groups of objects are known as classes. Classes can be associated with instances. `db:Disease` is a class, with an instance `db:Cardiovascular disease`. Aforementioned is typically referred as type information. Instances have facts about them such as `db:Cardiovascular disease dbp:field db:Cardiology`. These facts and type information form the instance-level knowledge.

Linked Data is a concept to create interconnected structured data. Sir Tim Berners-Lee came up with Linked Data to facilitate the integration of different Semantic Web datasets. It follows four basic principles as outlined in [Bizer et al. 2009],

1. Use URIs as names for things.

2. Use HTTP URIs so that people can look up those names.

3. When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL [Prud et al. 2006]).

4. Include links to other URIs, so that human and machine can discover more related things.

Linked Data embodies Sir Tim Berners-Lee's vision to create an interconnected data on the web which is commonly known as "Web of Data".

Semantic Web and Linked Data communities created several well known and widely used datasets. DBpedia [Auer et al. 2007], YAGO [Hoffart et al. 2013], and WikiData [Erxleben et al. 2014] are a few of the prominent datasets which capture information covering a number of domains. Linked Data also contains datasets capturing knowledge for specific domains such as life science (Bio2RDF [Belleau et al. 2008]), entertainment (BBC program [Kobilarov et al. 2009]), and education (DBLP[1]). Some of these knowledge graphs (KGs) are very large. For example, DBpedia English version contains 6.2 million entities and 1 billion facts, and Freebase contains 44 million entities and 2 billion facts. Not all Web of Data follows the Semantic Web and Linked Data standards, however they follow simple graph-based data model with their own representation format. Google Knowledge Graph is one such giant dataset which consists of 570 million entities and 18 billion facts as reported in 2014 [Dong et al. 2014]. With the emergence of Google Knowledge Graph, the term "Knowledge Graph" has become prominent for any graph-based data model including Semantic Web datasets.

Most of the early knowledge representation techniques captured [Russell et al. 1995] the knowledge manually. However, the manual creation of knowledge was time consuming and costly. Automatic knowledge graph creation techniques use structured sources such as Wikipedia. DBpedia [Auer et al. 2007] is a knowledge graph created using the structured portion of Wikipedia. NELL [Mitchell et al. 2015] and Knowledge Vault [Dong et al. 2014] extract facts from a textual corpus using NLP techniques and use a fixed ontology to guide the extraction process. Reverb [Kobilarov et al. 2009]

[1] http://dblp.l3s.de/d2r/

and OLLIE [Schmitz et al. 2012] perform open extraction without any schema even though it can lead to noisy facts.

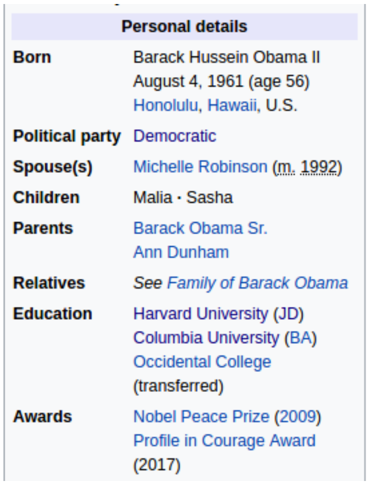
This dissertation uses DBpedia and Wikipedia Category Graph as the test beds to evaluate the effectiveness of the proposed approaches for domain-specific subgraph extraction for non-hierarchical and hierarchical relationships respectively.

2.1.2.1 DBpedia and Wikipedia Category Graph (WCG)

Both DBpedia and WCG use Wikipedia as its source knowledge. Wikipedia[2] is a web-based encyclopedia curated by the crowd. As it has been crowdsourced data source, Wikipedia is known for its coverage and availability. Each Wikipedia page is intended to cover a single topic which can be a person, a location, an event, etc. DBpedia dataset has been created using the infobox feature in Wikipedia. Infobox captures the facts of a given Wikipedia page (topic). For example, Figure 2.1 shows some of the facts about Barack Obama as given in the Wikipedia page for Barack Obama. Each wikipedia page maps to an entity in DBpedia, and the facts about each of these entities are extracted using the facts expressed in these infoboxes. Figure 2.2 shows an example of some of the facts represented for the DBpedia entity `db:Barack Obama`. As shown in Figure 2.2, the fact about the Barack Obama's political party in the infobox maps to an edge (`db:Barack Obama` `dbo:party` `db:Democratic Party(United States)`) in the DBpedia graph. DBpedia maintains DBpedia ontology which captures all edge types (e.g. `dbo:party`), entity types (e.g. `dbo:President`)), and entity type (class) structure(e.g. `dbo:President` `rdfs:subClassOf` `dbo:Politician`). We consider all these edge types defined in DBpedia ontology as non-hierarchical relationships.

Each Wikipedia page is assigned to a set of categories as given at the end of the Wikipedia page. These categories are attached to super categories, which creates a category structure. WCH uses this category structure. For example, `db:Barack Obama` entity has categories `db:Democratic Party Presidents of the United States` and `db:American Nobel laureates` using the relationship

[2] https://www.wikipedia.org/



Personal details	
Born	Barack Hussein Obama II August 4, 1961 (age 56) Honolulu, Hawaii, U.S.
Political party	Democratic
Spouse(s)	Michelle Robinson (m. 1992)
Children	Malia · Sasha
Parents	Barack Obama Sr. Ann Dunham
Relatives	*See Family of Barack Obama*
Education	Harvard University (JD) Columbia University (BA) Occidental College (transferred)
Awards	Nobel Peace Prize (2009) Profile in Courage Award (2017)

Figure 2.1: Wikipedia infobox for Barack Obama

`dct:subject` as shown in Figure 2.2. These categories connect to its super categories using the relationship `skos:broader`.

2.1.3 Recommendation Systems

Recommendation systems automatically suggest items such as products, movies, and articles to users and have become increasingly popular in recent years in many application domains. The approaches used by these recommendation systems can be broadly classified into two groups:(1) content-based, and (2) collaborative filtering. While content-based approaches recommend the items based on the properties of the items to be recommended, collaborative filtering approaches recommend the items based on the preferences of similar users [Lu et al. 2015]. Hybrid approaches combine these two approaches to exploit the strengths of both these approaches [Burke 2002]. In addition to these, some advanced recommendation approaches leverage social networks [He and Chu 2010], fuzzy methods [Zhang et al. 2013], and context [Adomavicius and Tuzhilin 2015] to improve their recommendations.

Most of the existing content-based recommendation systems use the textual description of the items to analyze the properties of the items [Lops et al. 2011]. However, leveraging only the textual

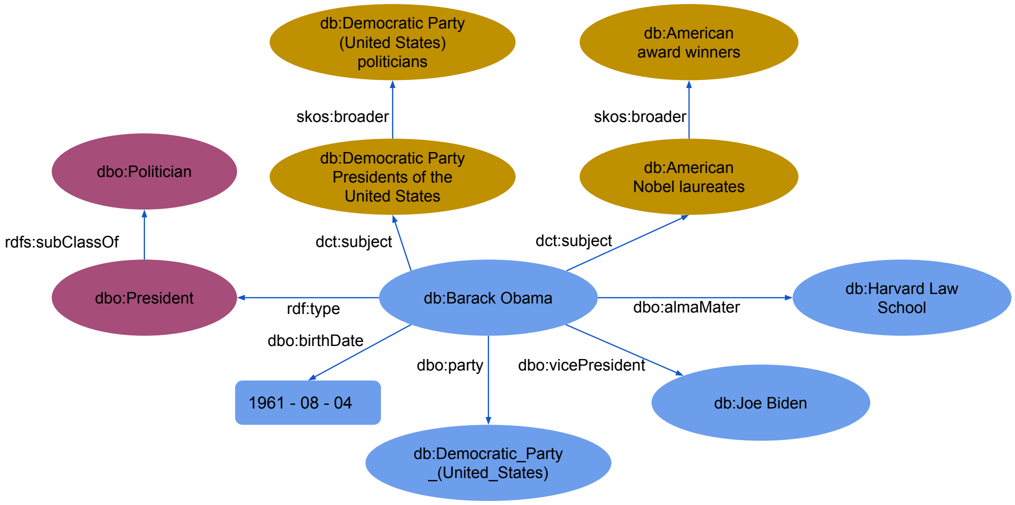


Figure 2.2: DBpedia example

description of items does not capture the semantic features among items. With the popularity of Linked data, content-based recommendation algorithms started to use knowledge graphs to capture the semantic features of items [Di Noia et al. 2012] [Ostuni et al. 2013] [Musto et al. 2014] [Passant 2010] [Piao and Breslin 2016]. Details of these recommendation systems use the Linked Data are provided in Section 2.2.2. A complete review of the Linked Data based recommendation systems can be found at [Figueroa et al. 2015]. In this dissertation, we use the content-based recommendation algorithm outlined in [Di Noia et al. 2012] as a use case to show the effectiveness of the domain-specific subgraph extraction. We pick this algorithm for its simplicity, popularity, and performance.

2.2 Related Work

In this chapter, we review the related work in the existing literature that covers the three parts of the proposed dissertation, namely, automatic domain identification from Web of data, identifying domain-specific subgraph with non-hierarchical relationships and identifying domain-specific subgraph with hierarchical relationships.

2.2.1 Automatic domain identification from Web of data

Domain identification of datasets will improve the identification of relevant datasets for domain-specific applications. CKAN and LODStats are the state of the art for finding relevant datasets on LOD. CKAN encourages data publishers to tag their datasets with a set of predefined labels, which are then manually reviewed by the CKAN administrators. CKAN is used to generate the LOD bubble diagram as shown in Figure 2.3 and it provides a search interface based on the metadata provided, assigned tags and keywords. To the best of our knowledge, our work is the first effort towards automatic domain identification for LOD datasets. However, LOD and information retrieval communities have worked on topics related to ours as described below.

LODStats [Auer et al. 2012] is a stream-based approach for gathering statistics about the datasets and it allows to search datasets based on keywords. Both CKAN and LODStats rely on the metadata provided by data publishers and hence rely on manually categorizing and describing the datasets. While this may lead to high-quality descriptions, this process is tedious and time-consuming and consequently different data providers may provide uneven descriptions or metadata. For enriching metadata about the datasets, in [Frosterus et al. 2011], authors have presented a system to create such metadata via annotation tools and a faceted search. This approach also relies on the annotations provided by the data providers. In addition to these systems, semantic search engines such as Sindice [Tummarello et al. 2007], Watson [d'Aquin and Motta 2011] and Swoogle [Finin et al. 2005] facilitate searching for entities but none of these systems are designed specifically for dataset search.

Dataset selection and identification has discussed in the context of federated querying and data interlinking. In SchemEX [Konrath et al. 2012], authors provide a scalable approach for indexing LOD datasets. It provides an index by leveraging type and property information of RDF instances. In [Harth et al. 2010], authors have proposed an index structure to store dataset summaries using QTree to identify relevant data sources. These data summaries are obtained by applying a hash

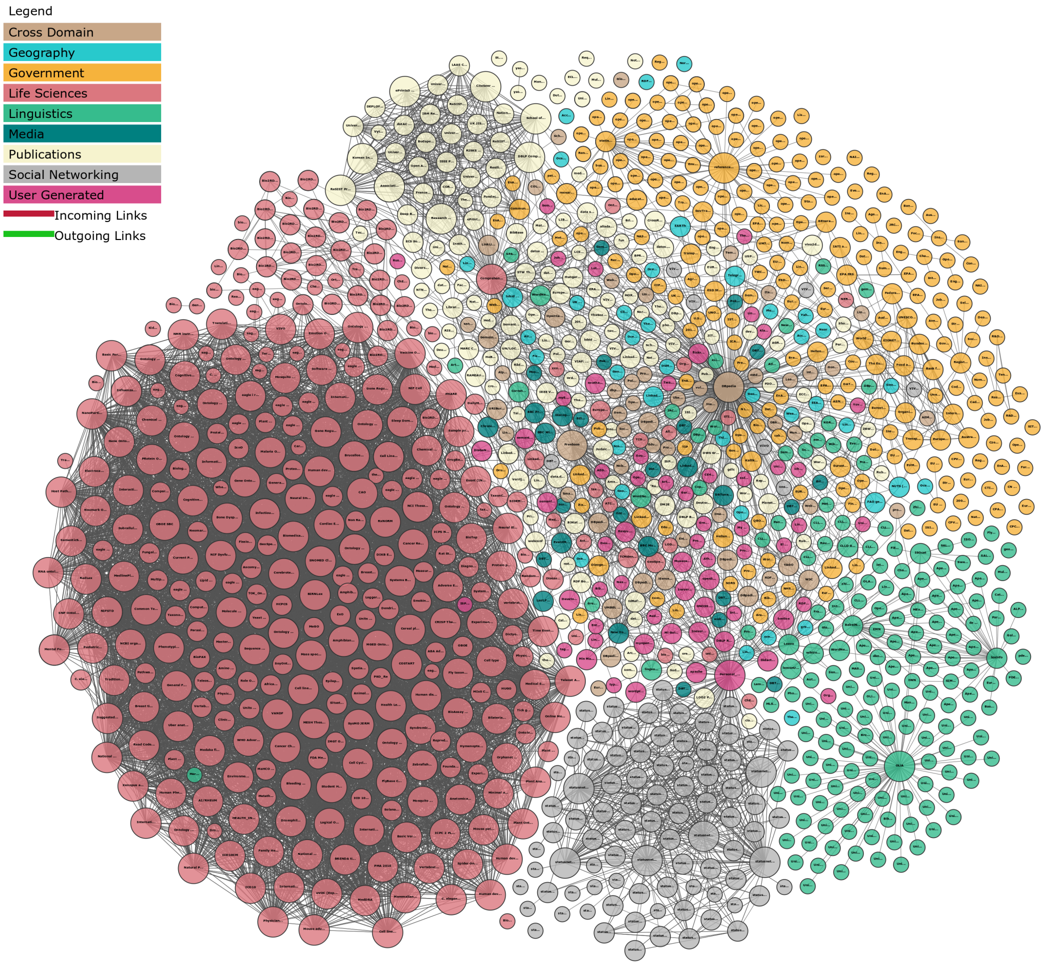


Figure 2.3: *Linked Open Data Cloud Diagram as of December-2017*

function to the triples of the dataset and mapped to the numerical space. In [Görlitz and Staab 2011], the authors have used voID descriptions [Alexander and Hausenblas 2009] containing metadata about datasets, to build an index which can be incorporated in query processing to determine the relevant dataset for querying. In [Schwarte et al. 2011] [Hartig et al. 2009] [de Oliveira et al. 2012] [Tran et al. 2010], authors have proposed different techniques for dataset identification for query answering. [Nikolov et al. 2011] presents an approach to identify relevant data sources for interlinking of a given particular dataset by using a semantic web index like sig.ma [Tummarello et al. 2010]. We believe that these approaches could also benefit from automatic domain identification for datasets, to reduce their search space and also to further identify more relevant results.

Another related body of work is topic modeling, which is about the identification of abstract topics (related clusters of words) that occur in a collection of documents. Latent Semantic Analysis [Deerwester et al. 1990] is a dimensionality reduction technique to identify the latent concepts which result in documents with similar topical content to be close to one another. Subsequently, probabilistic approaches such as the pLSI model [Hofmann 1999] and LDA [Blei et al. 2003] were used for topics models. But these latent concepts cannot be readily mapped into natural concepts in a way that a human would describe the concepts. Along these lines, Explicit Semantic Analysis (ESA) [Gabrilovich and Markovitch 2007] has been proposed to use machine learning techniques together with Wikipedia as a knowledge base, to augment keyword based representations with concepts from Wikipedia. Another body of related work is in the area of document/text classification into pre-defined topic hierarchies or taxonomies using machine learning techniques, such as [Cai and Hofmann 2004] and [Hao et al. 2007]. All these systems have the advantage of text being available in the documents for classification, but in our case, we only have one label for each typed instance in the dataset. A number of these systems utilize training data whereas our approach does not utilizes any training data at all.

In addition to the efforts described above, early work in Semantic Web investigated different ways to search for a relevant ontology among different ontologies. [Arumugam et al. 2002] proposes a Peer-to-Peer infrastructure to support sharing of independently created and maintained ontologies and facilitates search of shared ontologies. Searching for ontologies is based on keywords for ontological terms and then followed by discarding ontologies that do not have any common parent node with other ontologies from initial results. However, more recent KGs consist of shallow schemas where this technique would fail, but these KGs consist of a rich instance base which we utilize in the proposed techniques.

2.2.2 Identifying domain-specific subgraph with non-hierarchical relationships

The need to identify a domain-specific subgraph is a common problem when an application leverages large cross-domain datasets such as DBpedia, Freebase, and YAGO. Some of the prominent types of applications are: (1) recommendation, (2) named entity disambiguation, and (3) document similarity. In this section, we discuss usage of the KG in each of these applications and the applicability of domain-specific subgraph extraction in each of the applications.

Recommendation systems mainly use KGs to capture the item relatedness in recommending items. Content-based recommendation systems [Di Noia et al. 2012] [Ostuni et al. 2013] [Musto et al. 2014] [Passant 2010] [Piao and Breslin 2016] extract DBpedia subgraphs from a 2-hop expansion of in-domain entities with a set of manually selected relationships for the movie and book domains. Dbrec [Passant 2010] measures the relatedness of two items using all paths within 2-hop subgraph in recommending bands and solo artists. An improved version of this measure [Piao and Breslin 2016] penalizes the relationships based on its frequency in the whole graph. However, both these similarity measures are limited to measuring the similarity of two items directly connected or connected via only one intermediate node. A hybrid recommendation algorithm [Ostuni et al. 2013] extracts the DBpedia subgraph within 3-hops of movie entities to capture the relationships between the movies rated by the user and the movies to be recommended. They use a learning to rank function to rank the paths.

The n-hop expansion technique to extract the subgraph still contains a significant portion of KG and also includes irrelevant entities and relationships ignoring domain semantics. Manual selection of relationships for a given domain can help to capture the domain semantics, but this is a labor and time intensive process given the number of relationships in KGs. For instance, DBpedia has more than 1,000 relationships and only 65% relationships have domain and range defined that help to capture the domain semantics. Also, recommendation systems rank the paths to identify the relevant paths. However, this ranking depends on the two items being compared rather than the semantics of the domain of the two items. In this work, we capture the domain semantics for extracting the subgraph for a given domain. However, the proposed techniques can be complementary to ranking relevant paths.

Recommendation is a well-suited use case for domain-specific subgraph extraction as it considers the item relatedness among in-domain entities. Even though we focus on recommendation system as a use case in this work, it is not limited to recommendation. Named entity disambiguation links the entity mention in a text to an entity in a KG. Recent approaches [Usbeck et al. 2014] [Hulpuş et al. 2015] use KGs such as DBpedia as a way to generate the context for each ambiguous entity mention. AGDISTIS [Usbeck et al. 2014] extracts a DBpedia subgraph with up to 3-hop of all candidates and then use centrality-based measures to link the sense. A more recent approach [Hulpuş et al. 2015] jointly disambiguates entities by using path-based relatedness measures among all candidate senses in the DBpedia subgraph. Even though these existing techniques work well with generic entities, a domain-specific subgraph would benefit the named entity disambiguation in specialized domains like medical and financial due to the special domain characteristics [Zwicklbauer et al. 2015].

In addition to the approaches mentioned above for semantic relatedness, [Pirrò 2015] leverages the information content of the relationships and paths for semantic relatedness. Some approaches [Anyanwu et al. 2005] [Aleman-Meza et al. 2005] also employ the schema information to further improve the information content-based methods. While existing semantic relatedness techniques consider the features of the two items being compared, we restrict the features based on a given domain. In fact, [Hulpuş et al. 2015] emphasizes the importance of identifying relevant paths between two entities to improve the semantic relatedness measures.

2.2.3 Identifying domain-specific subgraph with hierarchical relationships

The n-hop extraction technique for domain-specific subgraph extraction treats both hierarchical and non-hierarchical relationships in the same way. But as we discussed, hierarchical and non-hierarchical relationships need to be handled differently. While some of the existing work manually pick the relationships [Di Noia et al. 2012] [Ostuni et al. 2013] [Musto et al. 2014] [Passant 2010] [Piao and Breslin 2016], hierarchical relationships carry uniform semantics to connect both in-domain and out-of-domain categories.

Due to the popularity and availability of the large-scale hierarchical knowledge graphs such as WCG and YAGO, there are number of techniques proposed to extract the relevant portion of the WCG. The Wikipedia taxonomy [Ponzetto and Strube 2011] extracts a taxonomy from WCH as it does not form a full-fledged subsumption hierarchy. Wikipedia Bitaxonomy [Flati et al. 2014] goes one step beyond [Ponzetto and Strube 2011] by creating an integrated taxonomy of Wiki pages and categories. While identifying the contextual differences of the categorization, [Jiang et al. 2013] proposed a domain-independent approach for ranking categories for a given concept with respect to a particular textual content.

However, none of these studies have focused on addressing the domain-specific aspect of WCH. Given a set of keywords, Doozer [Thomas et al. 2008] creates a domain model from the WCH. Doozer tries to capture the domain relevance by identifying semantically relevant entities (Wikipedia articles) and then traversing the WCH until finding a least common subsumer. In our work, we have shown that even starting with only domain entities can lead to subgraphs with an irrelevant portion for the domain. In bootstrapping domain ontologies, [Mirylenka et al. 2015] have used a classifier to classify whether a given category is relevant to a given domain. We have compared our approach against [Mirylenka et al. 2015] and showed that our approach performs better than theirs in extracting domain-specific hierarchical subgraphs from Wikipedia. However, [Mirylenka et al. 2015] also focuses on extracting a richer domain model which is not the focus of our work.

Another related body of work in extracting domain-specific subgraph from taxonomic relationships is the existing state of the art on automatic domain taxonomy creation [Liu et al. 2012]. Most of the existing approaches on automatic domain taxonomy creation rely on a domain corpus to extract the domain taxonomy. However, with the availability of large crowd-sourced structured resources such as DBpedia, YAGO, and Wikipedia there are few approaches which focus on extracting the domain-specific portion from these large cross-domain resources.

3

Automatic domain identification from the Web of data

3.1 Overview

Linked Open Data (LOD) has emerged as one of the largest collections of interlinked structured datasets on the Web. Despite the adoption, increase in the size and diversity of the datasets creates challenges in identifying the relevant datasets for the task at hand. As mentioned in the introduction this is mainly done using manual inspection of the datasets and also based on the popularity of the datasets. Existing approaches assign a pre-defined set of tags by manually inspecting the datasets, and these tags are being used to identify the relevant datasets. However, the growth of the number of datasets and diversity of the domains make this is an unsustainable process.

In this work, we propose [Lalithsena et al. 2013] a systematic and sophisticated approach to identify the domains of the datasets instead of a pre-defined set of tags. For this, we take a straightforward perspective on the domain identifiers needed: We use tags, which will usually be general terms such as "music," "geography," or "artist" as identifiers to describe domains. Automatic domain identification for LOD datasets is an interesting and challenging issue for several reasons. First, schema information plays a critical role in identifying the topic domains of a dataset, but most of the LOD datasets only contain very shallow schemas. Thus, schema information by itself will not be enough to identify domains. Second, people represent data that belongs to various domains in different granularities. For example, DBpedia contains information about diverse domains including music, while MusicBrainz focuses on the music domain. DBpedia and MusicBrainz use different schemas to represent data, so by looking at the schemas it is a nontrivial task to identify music as a common domain covered by both datasets.

In this work, we provide an approach to automatically identify the topic domains of datasets by utilizing knowledge sources in other LOD datasets, such as the hierarchy within Freebase. We believe community-driven knowledge sources and their hierarchy will enable us to cover a wide variety of domains, and in fact, this type of “bootstrapping” of LOD has been used before for other purposes [Jain et al. 2010]. Also, we present a search application built on the identified domains to search and identify relevant datasets within LOD. Furthermore, we provide an evaluation to validate the domains identified and also evaluate the effectiveness of the identified domains for searching datasets in comparison to existing systems.

3.2 Proposed approach

Our approach provides a technique to automatically identify the main topics of LOD datasets by utilizing Freebase as both background knowledge and to provide the vocabulary for the domain tags. We will in particular make use of the fact that each Freebase instance or article (we will call them *Freebase instances* in the following), is assigned one or more *Freebase types* within Freebase (such as *mountain*). Each of these types, in turn, is assigned to a *Freebase domain* (such as *geography*). Both *Freebase types* and *Freebase domains* will be used as domains in our work.

In a nutshell, our approach is based on assigning *Freebase types* and *Freebase domains* to the instances in an input LOD dataset, together with a weight computed from its frequency count.

While we have developed our approach with Freebase in mind, and we describe it as such below, it will be clear from the description that our approach is adaptable to other settings. We will discuss this further in the conclusions.

In more detail, our approach consists of the subsequent steps explained below in Sections 3.2.1 to 3.2.4. Figure 3.1 depicts the workflow of our approach with examples at each step.

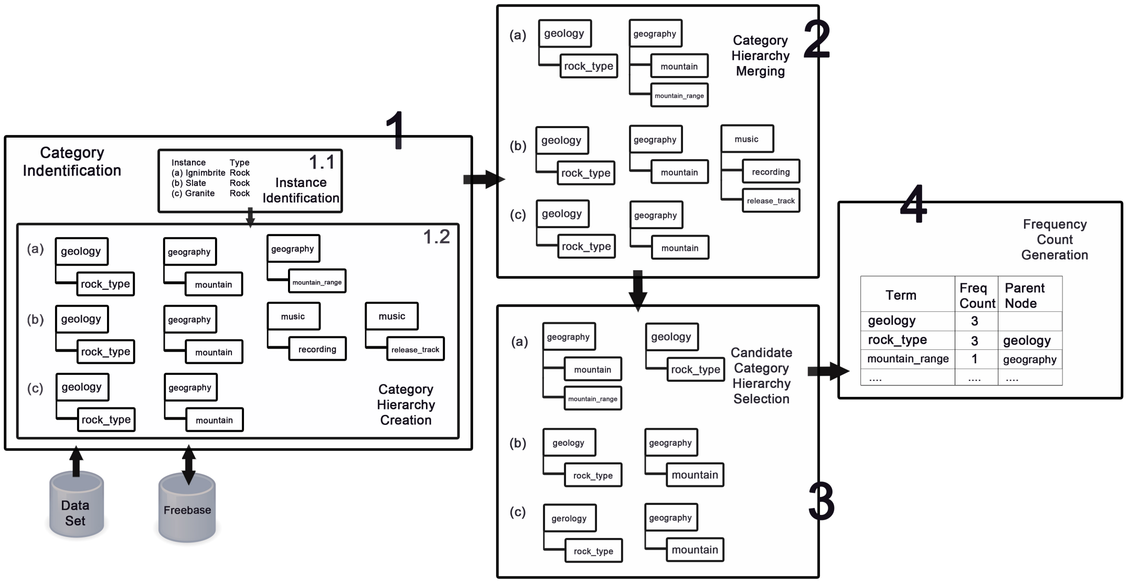

1
Category
Indentification
Instance Type
(a) Ignimbrite Rock
(b) Slate Rock
(c) Granite Rock
1.1
Instance
Identification
1.2
(a) geology rock_type geography mountain geography mountain_range
(b) geology rock_type geography mountain music recording music release_track
(c) geology rock_type geography mountain
Category
Hierarchy
Creation
Data
Set
Freebase
2
Category
Hierarchy
Merging
(a) geology rock_type geography mountain mountain_range
(b) geology rock_type geography mountain music recording release_track
(c) geology rock_type geography mountain
3
Candidate
Category
Hierarchy
Selection
(a) geography mountain mountain_range geology rock_type
(b) geology rock_type geography mountain
(c) gerology rock_type geography mountain
4
Frequency
Count
Generation
Term | Freq Count | Parent Node
geology | 3 |
rock_type | 3 | geology
mountain_range | 1 | geography
.... | .... | ....

Figure 3.1: *Workflow for identifying domains*

3.2.1 Category Identification

3.2.1.1 Instance Identification

The domains of a dataset are implicitly determined by the collection of entities it contains. As an example, the domain of 'GeoNames'[1] is predominantly geo-spatial because it contains a large number of geo-spatial entities such as countries, cities and villages. Therefore, our approach primarily utilizes the instances of the dataset in conjunction with type information of the instances to identify the domains of each dataset.

[1] http://www.geonames.org/ontology/documentation.html

As the first step, the input dataset is processed to retrieve (i) the instances, (ii) their corresponding labels or human readable values, (iii) classes of the instances, and (iv) class name labels or human readable values.

After this, the next step is the identification of corresponding or closely related Freebase instances for all instances of the input dataset.

This is achieved by utilizing the labels of the instances in the dataset combined with the concept names to which the instance belongs,[2] and executing a search on Freebase using its API[3]. The combination of instance labels with concept labels can improve the accuracy of the approach since in some cases instance labels by themselves return irrelevant results. For example, consider the instance 'Ignimbrite' from the Climb Dataincubator dataset[4]. Using 'Ignimbrite' as the search string on Freebase leads to multiple hits such as 'Ignimbrite' and the book 'Geology of a Miocene ignimbrite layer'. Appending the type information, i.e., 'Rock', to the query term enhances the precision by eliminating the book 'Geology of a Miocene ignimbrite layer'. The instance name alone is utilized as the search term where type information does not retrieve any results.

This step is illustrated in Step 1.1 of the Figure 3.1 for three different instances.

3.2.1.2 Category Hierarchy Creation

The search results, i.e., the identified Freebase instances for each query from the previous step, are used to obtain what we call *category hierarchies* for them: The Freebase search API is used to identify the *Freebase types* within which the Freebase instances have been categorized, and we also keep track of the corresponding *Freebase domains*. For the term *Ignimbrite*, for example, the Freebase API returns the type 'rock_type' in the domain 'geology', giving rise to the category hierarchy consisting of 'rock_type' and 'geology'.

At this point, the system has generated a set of category hierarchies for each given instance of

[2] i.e., to which it is explicitly assigned; we do not consider inferred types.

[3] https://developers.google.com/freebase/

[4] http://climb.dataincubator.org/

the dataset, as shown in Step 1.2 of Figure 3.1.

3.2.2 Category Hierarchy Merging

Once the category hierarchies have been created, this step merges all of those with the same *Freebase domain*, by creating a tree of depth 2 with the *Freebase domain* as the root and the *Freebase types* as leaves. Step 2 in Figure 3.1 shows the resulting category hierarchies for the instances (a) and (b) after merging. The two category hierarchies with *Freebase domain* 'geography' from instance (a) have been merged. This step is repeated for the two hierarchies with *Freebase domain* 'music' of instance (b). This step results in a forest-like data structure with a number of category hierarchy trees rooted at a common generic node.

3.2.3 Candidate Category Hierarchy Selection

At the end of the previous step the input LOD dataset now has multiple category hierarchy trees associated with it, due to the varied collection and classification of instances. However, not all of these hierarchies are relevant and/or significant for a given dataset. Therefore, this step in our approach filters out insignificant category hierarchies by using a simple heuristic. Given a concept C of the input dataset, each instance of C gives rise to several category hierarchy trees as a result of the previous steps. We now identify *Freebase domains* which occur most often as roots of these trees, and retain only the trees with these roots, discarding all others.

As an example, consider the 25 instances of type 'Rock' in the Climb Incubator dataset. Our system generates multiple category hierarchies for the 25 instances. 22 out of 25 instances have 'geology' as the root of the hierarchy, while 3 have 'music' as the root node. Using a simple majority as the deciding mechanism, all hierarchies with 'geology' as their root are retained and all hierarchies with 'music' as their root are discarded.

This is illustrated in Step 3 in Figure 3.1 for a small example consisting of only three instances. Category hierarchies rooted at 'geology' and 'geography' are retained while the one with 'music' as its root is removed. This process greatly reduces the impact made by false positives returned by the search API.

3.2.4 Frequency Count Generation

The next step involves assigning a frequency count to each of the terms in the resulting category hierarchies to describe their relative importance with regard to a given dataset. This count is generated by considering all category hierarchies from all instances of the dataset: Given an input LOD dataset D and a *Freebase type* or *Freebase domain* T, let $\mathcal{H}$ be the set of all category hierarchies generated from D using the steps described above, and let $\mathrm{Freq}_D(T)$, called the *frequency count of* T *for* D, be the number of occurrences of T in $\mathcal{H}$.

Note that the frequency count is generated both for the root node and for all child nodes occurring in the category hierarchies. The Table given in Step 4 in Figure 3.1 shows frequency counts calculated for 'geology', 'rock_type' and 'mountain_range' for our example. These terms which consist of *Freebase domains* and *Freebase types* can be considered as the domains for a given dataset. A higher frequency count for a term provides an evidence for the term being a good descriptor for the dataset, because it shows that a large number of instances can be described by the given term.

3.3 Implementation

Our system has been implemented in Java using Jena[5] and the Freebase API. In order to scale to large datasets, our system has been deployed on a Hadoop cluster[6] consisting of 15 nodes, using a Map-Reduce job. The list of instance and type labels collected from the dataset is given as input to the Mapper task as pairs <InstanceLabel, TypeLabel>. The Mapper task performs the category hierarchy building by querying the knowledge base, and merges the category hierarchies as described in Step 2 in Figure 3.1. The Mapper writes its output as <TypeName, CategoryHierarchies> for

[5] http://jena.apache.org/

[6] http://hadoop.apache.org/

each instance. Here CategoryHierarchies refer to the hierarchies generated in Step 2 in Figure3.1. Once the Mapper tasks are done, the Reducer initializes its task by taking the TypeName as the key, i.e., all the instances with the same TypeName will be performed by a single reducer. The Reducer processes candidate category hierarchy selection, performs frequency count generation, and keeps track of the root nodes associated with non-root nodes.

3.4 Evaluation

In order to evaluate our approach, we ran our system on 30 LOD datasets which cover a variety of domains. These datasets include some prominent ones such as BBCMusic, DailyMed, VIVO Indiana, LinkedMovieDB, and SemanticWebDogFood.

In order to evaluate the quality of our approach, we use two different settings. The first setting (see Section 3.4.1) aims to validate the domains we identified involving users as subjects, and the second setting (see Section 3.4.2) evaluates the identified domains in terms of their effectiveness for finding LOD datasets on given topics.

3.4.1 Appropriateness of identified domains

Since there is no existing benchmark for this purpose we validate the identified domains using human subjects. To do so, we extracted the two highest ranked domains from the set of roots (*Freebase domains*) and the two other highest ranked terms from the leaves (*Freebase type*) for each dataset. Then we mixed these with four other random *Freebase types/domains* from the Freebase hierarchy. The reason behind selecting terms from both roots and leaves instead of taking just the four highly ranked terms is to ensure that the terms cover more than one *Freebase domain*. This allows us to assess the validity of assigning more than one *Freebase domain* as a domain. We then presented these eight terms to each of twenty users and asked them to select the terms that best represent the domains of the dataset. Of the twenty people, ten people responded to our request for participation

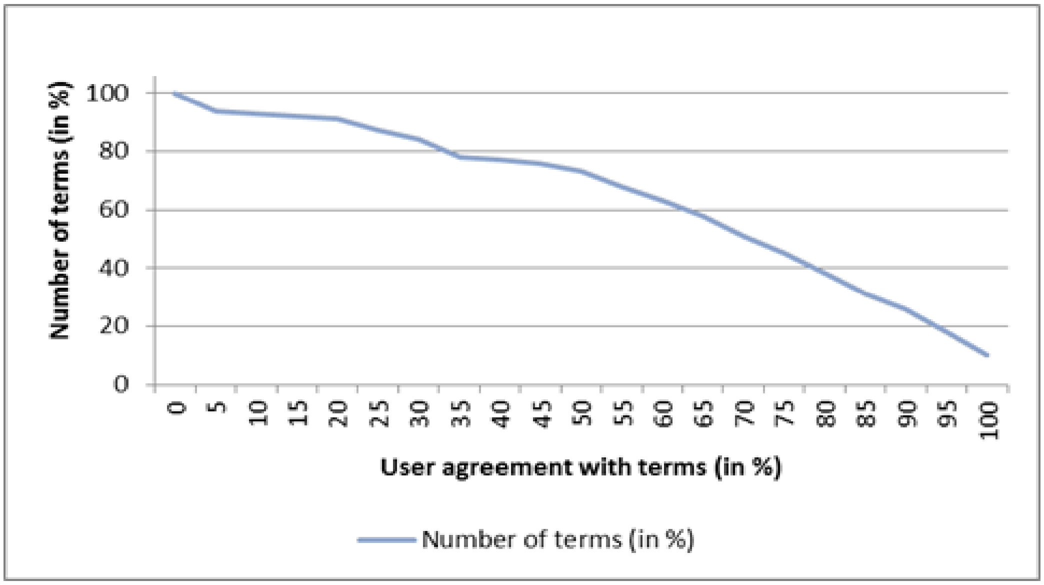


Figure 3.2: *User agreement on appropriateness of terms.*

from W3C Semantic Web and LOD mailing lists[7] and an other ten were members of two different organizations, DERI and Kno.e.sis, who work with LOD datasets and are familiar with them.

To summarize the results, we calculated the percentage of users which agree for each term generated by our approach. The graph in Figure 3.2 shows how many users agreed on how many terms being appropriate descriptors, from a total of 20 users (=100%, horizontal axis) and 120 terms (=100%, vertical axis). The data shows that 50% of the users agreed on 73% (88 out of 120) of the terms being appropriate descriptors. Table 3.1 shows in more detail the results for the terms which had the highest user agreement for each dataset.

Even though the system performs well with most datasets, with some datasets, such as LinkedEnergyData and UKPatentInfo, our approach fails to identify the prominent topic domains. The likely reason for this is the lack of matching Freebase instances for the entities in these datasets.

[7] http://lists.w3.org/Archives/Public/public-lod/2013May/0110.html

Dataset	**Term**	%	**Dataset**	Term	%
BBCMusic	music*	100	BBCProgram	TV	100
BBCWildLife	animal*	100	Climbdata	location*	85
DailyMed	medicine*	90	DBTuneClassic	music*	100
Diseasome	disease*	100	DrugBank	medicine*	100
Eumida	university	80	EuroStat	location*	95
Foodalista	food*	100	GeneBank	gene	100
GeoSpecies	biology*	95	Lingvoj	language*	95
EnergyData	organization	50	LMDB	film*	100
NASA	spaceflight*	100	ordinanceSurvey	citytown	95
semwebdogFood	people	65	UKPatentInfo	organization*	65
VIVOIndiana	organization	80	WorldFactBook	country	90
Airport	aviation	100	ECSRKB	people*	65
EUInstitutions	organization*	95	SIDER	drug	85
FarmersMarket	location*	75	Medicare	Medicine*	100
Gutenburg*	book*	65	Telegraphic	location*	85

Table 3.1: Terms with highest user agreement for each dataset. We indicate by a star (*) that a term was also the highest ranked by our system.

3.4.2 Usefulness of identified domains for dataset search

In this section we present comparative evaluations of our approach by demonstrating its effectiveness for finding LOD datasets compared with (1) a baseline obtained by a user study employing existing LOD lookup services such as semantic search engines (Section 3.4.2.1), and (2) searching on the CKAN data hub repository (Sections 3.4.2.2 and 3.4.2.3).

For the evaluation, we created a LOD dataset search application based on the identified domains. The application makes use of an index, more specifically it leverages the terms and statistical information collected during our process of topic domain identification. Each term is indexed with a list of datasets ranked by the normalized frequency count $\mathrm{NFreq}_D(T)$ of the term. The normalized frequency count is calculated as

$$\mathrm{NFreq}_D(T) = \frac{\mathrm{Freq}_D(T)}{\text{Total No of Instances in } D}$$

3.4.2.1 User study

We conducted a user study to evaluate how useful the results generated by our approach are for a dataset search, compared to using CKAN, LODStats[8] or the Sindice semantic search engine. CKAN and LODStats are two systems which allow people to identify relevant datasets based on keywords. Furthermore, CKAN uses metadata provided by users. More details on these systems are given in Section 2.2.1. There are a number of semantic web search engines such as Watson and Swoogle. We choose Sindice mainly because (1) it allows us to group the search results by datasets which is directly relevant to our approach, and (2) it is a very recent system and regularly updated.

For the evaluation, we performed the following steps.

1. We asked four users to come up with twenty terms each that reflect some topic domains of datasets present in the LOD. Table 3.2 presents the list of 20 terms for each of the 4 users.

[8] http://stats.lod2.eu/rdfdocs

From these eighty terms selected by the users, 20 terms were selected which were most often mentioned.

2. These twenty terms were used in order to evaluate our approach compared with CKAN, LODStats and Sindice. We retrieved the top ten results for each term for all systems. The results for all the terms can be found at the web page we used for the evaluation.[9]

3. The results for each term and each system were presented to 27 different users and they were asked to identify which set of results they preferred the most. The familiarity of the users with the LOD datasets varied from medium to expert. The results were provided to the users in a blind fashion, i.e., the users were not provided with the names of the systems which generated each set of results. The users were asked to rank the four result sets from 1 (best) to 4 (worst) based on their familiarity with the datasets and expectations based on the terms.

4. We calculated a user preference score using the user rankings to assess the performance of each system for each term. The score $R(S,T)$ for term T in system S is calculated using the weighted average[10]

   $$R(S,T) = \frac{\sum_{i=1}^{4}((5-i) * (N_{iTS}))}{\text{Total Number of Users}},$$

   where N_{iTS} is the number of users which rated rank i for the term T in the system S.

   This is essentially the most common method to summarize user ratings in product ranking systems. Note that a higher score indicates stronger performance.

Table 3.3 summarizes the results: CKAN ranked best for 12 terms while our approach ranked best for 9 terms. LODStats ranked best for 1 term. While our approach generates only second best results in some cases, it needs to be noted that our system indexes only 30 datasets, while other systems index over 290 datasets. Note, also, that CKAN uses keywords, user's metadata and manual tagging, while our system creates topic domain tags automatically, and thus scales better.

[9] http://knoesis-hpco.cs.wright.edu/LODYellowPagesEvaluation/

[10] http://en.wikipedia.org/wiki/Weighted_mean

User	**Terms**
User1	music, animal, drug, gene, food, conference, spacecraft, energy, language, university, tv program, film, mountain, geology, biology, spacecraft, instrument, recipe, disease, artist
User2	music, animal, drug, gene, food, conference, spacecraft, energy, language, university, tv program, rock, geology , astronaut, phenotypes, composer, recipe, country, artist, organism
User3	music, animal, drug, food, conference, spacecraft, energy, language, university, tv program, invention, book, geology, biology, phenotypes, composer, student, location, researcher, region
User4	music, animal, drug, gene, food, conference, energy, language, university, patent, film, book, geography, biology, instrument, student, astronaut, disease, artist, nasa

Table 3.2: Terms selected by users to describe the domain

Term	Our Approach	CKAN	LOD Stat	Sindice
music	2.037	**3.74**	3.11	1.333
artist	2.815	**3.926**	1	2.259
biology	**3.481**	3.333	1	2.185
animal	2.926	1.63	**3.481**	1.926
geology	2.852	**3.666**	1	2.481
drug	2.926	**3.148**	2	2.555
gene	2.148	**3.333**	3.074	1.222
university	**3.185**	3.148	2.37	1.222
food	**3.259**	2.296	3	1.259
language	3.148	**3.74**	1	2.11
spacecraft	**4**	**4**	1	2
conference	2.814	**3.555**	1	2.666
astronaut	**4**	**4**	1	2
composer	**3.815**	3.037	1	2.11
tv program	**3.666**	2.923	1	2.370
instrument	**3.852**	2	2	3.148
recipe	**3.926**	2	2	3.074
student	2	**3.889**	2	3.111
phenotypes	2	**3.923**	2	3.037
energy	1	**3.74**	3.26	3.03

Table 3.3: Comparative dataset search evaluation results

This evaluation demonstrates that our approach is nearly as effective as the manual tagging of datasets by CKAN for dataset search.

3.4.2.2 Evaluation with CKAN as baseline

In order to better understand our automated system in comparison with the CKAN, we performed a more detailed and focused evaluation against CKAN. For this, we again utilized the twenty terms from the previous evaluation, and retrieved the search results for those twenty terms from both CKAN and our search application. By considering the CKAN results as the baseline, we calculated the Precision (P), Recall, and F-measure for our search application. Here we calculated two recall values R1 and R2, where R1 considers only the 30 datasets we used for our approach and R2 considers all the datasets.

Table 3.4 summarizes the results. Some of the extremal values can be explained as follows (marked by a * in the table): (1) Our search application did not return any results for the terms 'student', 'phenotypes' and 'energy'.[11] (2) CKAN did not return any results for 'instrument' and 'recipe'.[12] (3) There is no overlap between results returned by our system and CKAN for the terms 'animal', 'geology', 'food' and 'tv program'.

The results demonstrate that our approach is able to provide high recall for some terms like 'music', 'language' and 'spacecraft'. The poor precision and recall values for some terms can be due to (i) inaccuracies within CKAN which we consider as baseline here, or (ii) shortcomings in our system. We further investigate this issue in Section 3.4.2.3 below. We also believe that our results could be improved further by increasing the number of datasets utilized by our system for generating the results.

[11] We have listed a precision of 1 in this case, indicating that we did not have any false-positives. The precision could also be considered *undefined* in this case.

[12] We have listed a recall of 1 in this case, indicating that we did not have any true-negatives. The recall could also be considered *undefined* in this case.

Term	**P**	**R1**	**F1**	**R2**	**F2**
music	0.286	1	0.445	0.1	0.148
artist	0.4	1	0.571	0.2	0.267
biology	0.125	1	0.222	0.333	0.182
animal	0*	0*	n/a*	0*	n/a*
geology	0*	0*	n/a*	0*	n/a*
drug	0.6	0.667	0.632	0.75	0.667
gene	0.333	1	0.5	0.125	0.182
university	0.5	1	0.667	0.0512	0.093
food	0*	0*	n/a*	0*	n/a*
language	1	1	1	0.045	0.0861
spacecraft	1	1	1	1	1
conference	1	1	1	0.125	0.2222
astronaut	1	1	1	1	1
composer	0.25	1	0.4	0.5	0.3333
tv program	0*	0*	n/a*	0*	n/a*
instrument	0*	1*	0*	1*	0*
recipe	0*	1*	0*	1*	0*
student	1*	0*	0*	0*	0*
phenotypes	1*	0*	0*	0*	0*
energy	1*	0*	0*	0*	0*

Table 3.4: Evaluation with CKAN as baseline

3.4.2.3 Comparison of CKAN and our approach against a manually curated gold standard

Although CKAN is manually populated, it does have omissions and contains erroneous values. For example, when we search for the term ’food’, CKAN gives ’Semantic Web Dog Food’ as a relevant result even though it is obvious that this dataset has nothing to do with ’food’ as such. Hence, in order to perform a fair evaluation and to establish a baseline for our system and any future applications in the same spirit, we asked 3 different human graders to manually assign the datasets to the given terms. These human graders are researchers in semantic web technologies and they have a high expertise level on LOD datasets as they often deal with these datasets for various applications such as querying and mapping. The output is presented in Table 3.5. We did this, on the one hand, to achieve higher quality results which have been manually verified. On the other hand, since CKAN utilizes keyword based indexing, it affects the results obtained by using its search interface, as explained earlier for the Semantic Web Dog Food example.

We then compared our system and CKAN with this manually curated gold standard, the results are presented in Table 3.6. We use P for Precision, R for Recall, and F for F-Measure. Table 3.6 shows that our search application provides nearly 90% better recall with respect to the manually verified standard, while being at par with CKAN in terms of precision.

To summarize, our approach can be helpful for systematically categorizing and finding relevant datasets from LOD. Our evaluations demonstrate that our approach provides significantly better precision and recall in retrieving LOD datasets compared to other approaches. It also demonstrates that the state of the art of LOD searching systems fails to provide the support required for searching and retrieving relevant datasets from the LOD cloud. The reasons for the superior performance of our system lies in the utilization of a diverse classification hierarchy such as Freebase in comparison to approaches which utilize traditional indexing and manual tagging based approaches. In addition, our system is automated, and thus scales well compared to manual approaches such as the tagging used in CKAN.

Term	**Dataset**
music	BBCMusic, DBTuneClassic, BBCProgram, LinkedMovieDB
artist	DBTuneClassic, LinkedMovieDB, BBCMusic, BBCProgram
biology	GeoSpecies, BBCWildLife, GeneBank, Diseasome, DrugBank
animal	BBCWildLife
geology	OrdanaceSurvey, Climb Data
drug	DrugBank, DailyMed, Diseasome, SIDER, MediCare
gene	GenBank, Diseasome, DrugBank
university	VIVOIndiana, eumida, ECSRKBExplorer
food	Foodalista
language	lingvoj
spacecraft	NASA Space Flight and Astronaut data
conference	Semantic Web Dog Food
tv program	BBC Program, BBC Music
instrument	BBCProgram, BBCMusic, DBTuneClassic
astronaut	NASA Space Flight and Astronaut data
composer	BBCMusic, BBCProgram, DBtuneClassic, LinkedMovieDB
recipe	Foodalista
phenotypes	Diseasome
student	VIVO Indiana, eumida, ECSRKBExplorer
energy	Linked Clean Energy Data

Table 3.5: Manually Classified Datasets

3.5 Conclusion

We have presented a solution for systematically identifying domains of LOD datasets. We have evaluated our approach against existing systems and reported on a user study. Our evaluation shows the effectiveness of our approach and that it can be very useful for the LOD community in number of ways. This work has the potential to be a basis for creating a search catalogue for LOD datasets as we have shown in our evaluation. Furthermore, our work is also potentially useful for identifying datasets for the purpose of interlinking.

Our approach currently draws some of its strength from the richness of Freebase. However, only the first *Category Identification* step described in Section 3.2 actually depends on Freebase, the remainder of the approach is completely generic. Returning to the description of the Category Identification step in Section 3.2, note that the only thing we need is a way to assign both a general *domain* and a more specific *type* to each instance in a dataset. Several alternatives suggest how to approach this, some of which will again reuse LOD datasets. Complementing the use of Freebase with other appropriate knowledge sources (including dictionaries or thesauri to help with synonyms) is not only interesting in order to improve performance or topic coverage of our system, it is also needed for full-scale domain identification for all LOD datasets, as the full use of Freebase is restricted on a daily basis. We intend to work on such alternatives. Furthermore, we can leverage the interlinks between datasets as these interlinked datasets are likely to share some topics.

We are confident that the LOD community can benefit from our approach. Our work can in principle easily be integrated with LOD meta data repositories such as CKAN, LOD Stats and Sindice, to allow people to gain a better understanding of the datasets. CKAN can use this for topic identification as an alternative or replacement for the manual assignment of topics. Furthermore, the LOD Bubble Diagram could be organized in a better way with improved topic domain identifications.

	CKAN			Our Approach		
Term	**P**	**R**	**F**	**P**	**R**	**F**
music	1	0.5	0.667	0.571	1	0.727
artist	1	0.25	0.4	0.8	1	0.9
biology	1	0.2	0.333	0.625	1	0.769
animal	0	0	n/a	0.333	1	0.5
geology	0	0	n/a	1	0.5	0.667
drug	1	0.6	0.75	1	1	1
gene	1	0.333	0.5	1	1	1
university	0.5	0.667	0.572	0.6	1	0.75
food	0	0	n/a	0.25	1	0.4
language	1	1	1	1	1	1
spacecraft	1	1	1	1	1	1
conference	1	1	1	1	1	1
tv program	0	0	n/a	1	1	1
instrument	1	0	0	0.75	1	0.857
astronaut	1	1	1	1	1	1
composer	1	0.25	0.4	1	1	1
recipe	1	0	0	1	1	1
phenotypes	1	1	1	1	0	0
student	1	0.5	0.667	1	0	0
energy	1	0.333	0.5	1	0	0
Mean	0.775	0.432	0.489	0.846	0.825	0.728

Table 3.6: Comparison of our approach and CKAN with a manual curated gold standard

4

Identifying domain-specific subgraph with non-hierarchical relationships

4.1 Overview

The large cross-domain knowledge graphs available on the Web gets wider adoption due to their availability and broader coverage. However, these KGs are large and utilizing the complete graph is computationally intensive and also, the irrelevant portion may negatively impact the performance in particular for domain-specific applications.

In this work, we address the task of extracting a domain-specific subgraph from a large KG. Our approach [Lalithsena et al. 2016] identifies entities and relationships that are strongly associated with the domain of interest. To determine the associations, our methodology treats relationships as first-class elements because relationships induce semantics between entities and, hence, can play a significant role in identifying domain-specific knowledge [Sheth et al. 2004]. As we discussed in the introduction, there are two main types of relationships between entities: (1) non-hierarchical relationships, and (2) hierarchical relationships which capture different semantics. Due to their different semantics discussed in Section 1.2, these two types of relationships have to handled differently. This chapter discusses the problem of identifying domain-specific subgraph with non-hierarchical relationships.

The practical usage of our approach is demonstrated with a recommendation system use case. We harness the domain-specific subgraphs extracted by our approach using DBpedia for recommending entities of two domains, i.e., movie and book. In the evaluation, we show that the use of a domain-specific subgraph can, (1) reduce the graph by over 80% without compromising the accuracy by identifying domain-specific entities and relationships; (2) decrease the computation time by more than tenfold in comparison to the existing methods that extract subgraphs.

4.2 Proposed approach

In order to extract a domain-specific subgraph from a large KG, the primary input is *the domain*. For example, the domain for a movie or a book recommendation system is *movie* and *book* respectively. The entities of the given domain are identified as *in-domain entities* i.e. movies and books. These in-domain entities can be leveraged to initiate and expand the subgraph. Our approach uses types of the entities in the KG as the domain and entities of the given type as in-domain entities. For instance, a movie recommendation system that utilizes DBpedia as a KG, the input to our approach would be type `dbo:Film`, and entities of type `dbo:Film` are the in-domain entities.

Starting with the in-domain entities, the extraction of the domain-specific subgraph now translates to expanding the in-domain entities with relationships and other entities that are specific to the domain of interest. Existing applications perform this expansion with a simple n-hop navigation with a pre-defined value for n [Ostuni et al. 2013; Musto et al. 2014; Passant 2010]. This would end up creating a subgraph with knowledge that is not specific to the domain. In order to extract

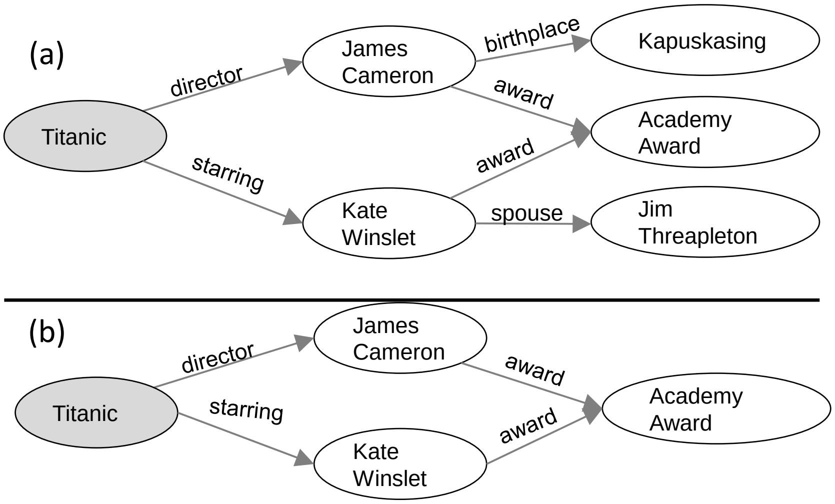


Figure 4.1: (a) 2-hop expansion subgraph (b) Domain-specific subgraph

a domain-specific subgraph, this n-hop navigation should only pick the facts that are specific to the domain. For an illustrative example, consider expanding the entity `dbpedia:Titanic` to a movie domain-specific subgraph on DBpedia. Fig. 4.1(a) shows the subgraph extracted by navigating 2-hops from entity `dbpedia:Titanic`. If we want to extract a movie domain-specific subgraph from Fig. 4.1(a) which consists of set of entities and relationships specific to the movie domain, facts such as `dbpedia:Kapuskasing` is the `dbprop:birthplace` of `dbpedia:James Cameron` and `dbpedia:Jim Threapleton` is the `dbprop:spouse` of `dbpedia:Kate Winslet` have less significance compared to facts such as `dbpedia:James Cameron` and `dbpedia:Kate Winslet` have won the `dbpedia:Academy Awards`.

Inspired by this observation, our approach focuses on assessing the domain-specificity of a fact. The specificity of a fact with respect to a domain can intuitively be assessed using the relationships. For example, identifying that relationships `dbprop:starring`, `dbprop:director`, and `dbprop:award` are more specific to the movie domain whereas `dbprop:spouse` and `dbprop:deathdate` are less specific would help to generate domain-specific subgraph in Fig. 4.1(b).[1] Therefore, in our

[1] Jim Threapleton would be in the movie domain-specific subgraph as a film director but not as an ex-spouse of Kate Winslet.

approach we consider relationships as the first-class elements in identifying the domain-specific subgraph and propose measures to capture the domain specificity of relationships.

To get to the crux of the approach, we formally define a KG and a domain-specific subgraph as follows:

Definition 1. *A knowledge graph $KG = (V, E, P)$ is a graph-based data model where V is a set of entities,*[2] *E is a set of labeled edges where $E \subseteq V \times P \times V$ and P is a set of relationships.*

Even though edges are directed in a KG, we assume that all semantic relations can be considered to have semantically inverse relations. Hence, our KG is undirected.

Definition 2. *A domain-specific subgraph for domain d with a set of in-domain entities D is $DSG = (V_d, E_d, P_d)$ where P_d is a set of domain-specific relationships with each relationship having a domain specificity score w_d, $V_d \subseteq V$ and $E_d \subseteq E$ that can be reached by navigating KG starting with D and using only P_d.*

The primary focus of this work is to determine the domain specificity scores that can be utilized to restrict the graph to a domain-specific subgraph. Section 4.2.1 describes our novel measures to determine domain specificity scores w_d.

4.2.1 Domain Specificity Measures

Association between a relationship and a domain determines the domain specificity of the relationship. Since the in-domain entities represent the domain in a KG, we consider the association between a relationship and the in-domain entities as the domain specificity of the relationship. For example, relationship `dbprop:starring` is strongly associated with the movie domain because it appears specifically with the in-domain entities, whereas, relationship `dbprop:country` is generic and can be associated with many non-domain entities of types such as `dbo:Person`, `dbo:Organization` and

[2] We consider only the edges connecting the entities and ignore the literals.

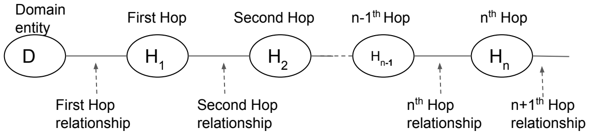


Figure 4.2: Hops and relationships

dbo:Place. Therefore, we consider that the domain specificity of dbprop:starring is higher than the domain specificity of dbprop:country with respect to the movie domain.

Our approach focuses on expanding the graph starting from in-domain entities by identifying domain-specific relationships. Determining the domain specificity of the first hop relationships (relationships that connect in-domain entities to the first hop entities as shown in Fig. 4.2), from the in-domain entities is trivial because we can find the direct association of the relationship with the in-domain entities. For example, in Fig. 4.3(a), dbprop:director and dbprop:country are directly connected to in-domain entity (m_1) and its association can be determined based on its frequency of occurrence with in-domain entities versus all entities. On the other hand, measuring the domain specificity of relationships that are not directly connected to in-domain entities is non-trivial. For example, in Fig. 4.3(a) it is not straightforward to find the association of relationships such as dbprop:award, dbprop:spouse, and dbprop:president with in-domain entities because they may require more information about their connectedness to the in-domain entities. For simplicity, in the rest of the paper we refer to relationships that connect in-domain entities to its first-hop entities as *direct relationships* and the relationships that are more than one-hop away from in-domain entities as *indirect relationships.*

We identify two characteristics in the KG that can be harnessed to measure the association of relationships to in-domain entities: (1) type and (2) path. We use these as ways to obtain the connectedness of relationships to in-domain entities. In Sections 4.2.1.1 and 4.2.1.2, we detail the intuition and formalizations of measures with these characteristics.

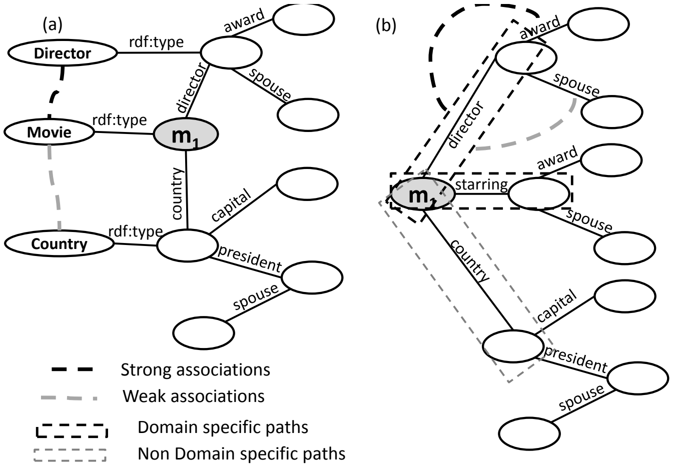


Figure 4.3: (a) Type-based connectedness; (b) Path-based connectedness m_1 is a movie (in-domain) entity

4.2.1.1 Type-based scoring

For this measure, we consider the KG as entities of certain types connected via relationships. For instance, `dbpedia:Titanic` is an entity of type `dbo:Film` connected to `dbpedia:James Cameron`, an entity of type `dbo:Director` via the relationship `dbprop:director`. We utilize the association between types to determine the domain specificity of n^{th} hop relationship. For example, as shown in Fig. 4.3(a), in order to calculate the domain specificity of relationships `dbprop:award` and `dbprop:spouse` that are 2^{nd} hop relationships, we utilize the strength of association between the type `dbo:Film` and `dbo:Director`. If entities of the type `dbo:Director` are strongly associated with the entities of type `dbo:Film` and the relationship `dbprop:award` is strongly associated with the type `dbo:Director`, then there is a higher likelihood that `dbprop:award` relationship has high association to the movie domain. To capture this intuition in calculating the domain specificity of an indirect n^{th} hop relationship we utilize two factors:

(1) The strength of association between the domain entity type t_d and the entity type at the $n-1^{th}$ hop t_{n-1} is determined by calculating the association between each pair of directly connected intermediate types leading to $n-1^{th}$ hop from in-domain entity type. The strength of association

between directly connected entity types t_i and t_j is determined using Eq. 4.1.

$$d\_typerel(t_i, t_j) = \frac{edgecount_{t_i,t_j}}{edgecount_{t_i} \times edgecount_{t_j}} \tag{4.1}$$

where $edgecount_{t_i,t_j}$ is the number of edges that connect entities of type t_i and t_j, and $edgecount_{t_j}$ is the number of edges with entities of type t_j.

The strength of the association between t_d and t_{n-1} connected via an ordered set of intermediate entity types $T = t_1, \ldots, t_{n-2}$ can be determined using Eq. 4.2

$$ind\_typerel(t_d, t_{n-1}, n) = \prod_{k=1}^{n-1} d\_typerel(t_{k-1}, t_k) \tag{4.2}$$

where $t_0 = t_d$. When there are multiple ordered sets of intermediate entity types that connect t_d and t_{n-1}, we calculate $ind\_typerel(t_d, t_{n-1}, n)$ for each ordered set and take the maximum value.

(2) The strength of association between entity types and their direct relationships is determined using Eq. 4.3,

$$proprel(p, t) = \frac{edgecount_{p,t}}{edgecount_p} \tag{4.3}$$

where t can be any type, p is a direct relationship of t, $edgecount_{p,t}$ is the number of edges with relationship p directly connected to entities of type t and $edgecount_p$ is the total number of edges with relationship p.

For indirect relationships, type-based scoring combines both $proprel(p, t)$ and $ind\_typerel(t_d, t_{n-1}, n)$ to compute a score for each n hop relationship p, $propscore(p, n)$, for a given domain entity type t_d, as given in Eq 4.4.

$$\begin{aligned} propscore(p, n) = \sum_{t_{n-1_j} \in C} & ind\_typerel(t_d, t_{n-1_j}, n) \\ & \times proprel(p, t_{n-1_j}) \end{aligned} \tag{4.4}$$

where C is the set of all entity types at $n - 1^{th}$ hop that have a direct relationship with relationship p. When the same relationship appears at hops i and j where $i < j$, we take the $propscore$ at i^{th}

hop. Type-based scoring uses Eq. 4.3 to find the domain specificity of direct relationships, where p is a direct relationship of t_d.

4.2.1.2 Path-based scoring

While type-based scoring utilizes the types of *intermediate entities* to determine the domain specificity, path-based scoring utilizes the domain specificity of *intermediate relationships*. The intuition is, if `dbprop:director` is already detected as a domain-specific relationship in Fig. 4.3(b), then `dbprop:award`, which is an indirect relationship connected via the intermediate domain-specific relationship `dbprop:director`, has a higher possibility to be a domain-specific relationship.

In order to determine the domain specificity of a relationship, we introduce an iterative approach that determines the domain specificity of intermediate relationships along the path from the in-domain entities. In the first iteration, it ranks direct relationships connected to the in-domain entities as given in Fig. 4.4(a) and selects the `top-K` relationships as domain-specific relationships. The `top-2` relationships in the example are `dbprop:director` and `dbprop:starring`. In the next iteration, as shown in Fig. 4.4(b), it uses the domain-specific relationships identified at the first-hop to traverse to the next hop and score the second-hop relationships such as `dbprop:knownFor`, `dbprop:award`, and `dbprop:writer`. In summary, this methodology determines the domain specificity of a n^{th} hop relationship based on its association with the ordered set of domain specific relationships from the in-domain entities up to $n-1^{th}$ hops. The ordered set of domain-specific relationships are termed as *domain-specific paths*.

According to Fig. 4.4(b) `m1 starring x award y` is a domain-specific path formed using the ordered set of relationships `starring` and `award`. The `x` and `y` are variables and can be replaced with any entity and `m1` is any in-domain entity. In other words, the intermediate entities are of no concern in representing a domain-specific path. These are utilized to determine the domain specificity of relationships that are connected to a subgraph at the next hop.

In this approach the domain specificity of a n^{th} hop relationship depends on its association to

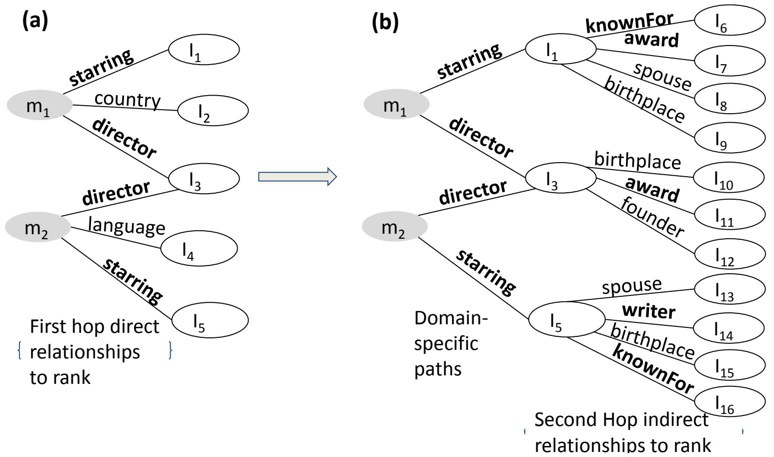


Figure 4.4: Iterative scoring of relationships (a) Iteration 1: scoring of direct (1-hop) relationship; (b) Iteration 2: scoring of indirect (2-hop) relationships; m_1 *and* m_2 *are movie (in-domain) entities.*

the domain-specific path obtained from domain entities to $n-1^{th}$ hop. To measure the association, we adopt Pointwise Mutual Information (PMI) [Church and Hanks 1990] which is a well-known association measure. PMI quantifies the association of two items appearing together using the co-occurrence of the two items. Here, the two items are the n^{th} hop relationship and the domain-specific paths until the $n-1^{th}$ hop.

Hence, we define the path-based measure $propscore(p,n)$ of a n^{th} hop relationship p as,

$$propscore(p,n) = PMI(p, dsp_{n-1}) \tag{4.5}$$

$$PMI(p, dsp_{n-1}) = log\frac{Prob(p, dsp_{n-1})}{Prob(p) \times Prob(dsp_{n-1})} \tag{4.6}$$

where $dsp_{n-1} \in DSP_{n-1}$, DSP_{n-1} is the set of domain-specific paths up to $n-1^{th}$ hop.

$$Prob(p, dsp_{n-1}) = \frac{Path(dsp_{n-1}, p)}{\sum_{p \in P} Path(dsp_{n-1}, p)} \tag{4.7}$$

where $Path(dsp_{n-1}, p)$ is the number of paths of length n with a domain-specific path dsp of length $n-1$ connected to p. The probability $Prob(p)$ is the fraction of edges that contain p. $Prob(dsp_{n-1})$ is the fraction of paths of length $n-1$ that has the domain-specific path dsp. If there are multiple domain-specific paths dsp_{n-1} that can reach p, we calculate the $propscore(p, n)$ for each dsp_{n-1} and take the maximum value.

PMI is known for its sensitivity to low frequency values. Therefore, we normalize the $PMI(p, dsp_{n-1})$ to $NPMI(p, dsp_{n-1})$ as proposed in [Bouma 2009].

$$propscore(p, n) = NPMI(p, dsp_{n-1}) \tag{4.8}$$

$$NPMI(p, dsp_{n-1}) = \frac{log\frac{Prob(p, dsp_{n-1})}{Prob(p) \times Prob(dsp_{n-1})}}{-logProb(p, dsp_{n-1})} \tag{4.9}$$

Similar to the type-based ranking, if the same relationship appears in multiple hops i and j where $i < j$, we take the *propscore* at i^{th} hop.

4.2.2 Creating a Domain Specific Subgraph

Definition 2 defines a domain-specific subgraph where the relationships in the graph are scored based on their domain specificity. These scores as detailed in Section 4.2.1, are utilized to restrict the KG to a domain-specific subgraph. The restriction is performed by rank-based thresholding of the scores. The graph extracted using Top-k relationships according to the domain specificity scores is identified as domain-specific subgraph.

4.3 Evaluation

To show the effectiveness of our approach, we use the domain-specific subgraph for a recommendation use case. Recent developments in recommendation algorithms [Di Noia et al. 2012; Ostuni et al. 2013; Musto et al. 2014; Piao and Breslin 2016] recognize the importance of incorporating KGs as background knowledge to improve the accuracy. Our evaluation methodology adapts an existing recommendation algorithm as outlined in Section 4.3.1.1 and compares its performance with subgraphs generated from: (1) n-hop expansion, and (2) n-hop domain-specific subgraph. We perform the evaluation for two domains: movie and book by using different evaluation metrics as detailed in Section 4.3.2.

4.3.1 Evaluation Setup

4.3.1.1 Recommendation algorithm

We adapt an existing recommendation algorithm [Di Noia et al. 2012] for our evaluation. This uses KGs from Linked Open Data to calculate the similarity between the items. The similarity function used by [Di Noia et al. 2012] considers only the entities reached via one-hop, which restricts the recommendations of movies connected via one-hop to the user selected movies. For a fair comparison, we replace the similarity function with a measure proposed by [Leal 2013] which measures the similarity of two entities x and y connected via n number of hops. This similarity function is given below.

$$sim(x,y) = \sum_{n=1}^{r} \frac{1}{2n^2} \times \sum_{path \in Paths(x,y)} \sum_{e \in edges(path)} w(e)$$

where r is the maximum number of hops between two entities, $Paths(x,y)$ returns all the paths between two entities within n-hops, $edges(path)$ returns all the edges in the $path$ and $w(e)$ is the weight of the edge. We consider all the edges to have an equal weight.

4.3.1.2 Baseline - n-hop expansion subgraph

The baseline is the recommendation algorithm presented in Section 4.3.1.1 with the input KG being a simple n-hop expansion of DBpedia.[3] We experiment with n = 2, 3.

4.3.1.3 Our approach - Domain-specific subgraph

While the recommendation algorithm remains the same as baseline, the input KG is created by scoring and ranking domain-specific relationships in DBpedia using type-based and path-based measures.

[3] We use the infobox properties and ignore the hierarchical relationships.

4.3.1.4 Datasets

We used popular datasets from two domains: (1) movie and (2) book. For movie domain, we used the MovieLens dataset that consists of 1,000,209 ratings for 3,883 movies by 6,040 users and for book domain we used the DBbook[4] dataset that has 72,372 ratings for 8,170 items by 6181 users. Since our recommendation algorithm is a content-based approach, users who have rated very few items significantly impacts the recommendation performance. To deal with this, we eliminated users who have less than 20 ratings as suggested in [Di Noia et al. 2012]. After this filtering step, the MovieLens dataset contains 5,886 users with approximately 0.9M ratings, and the DBbook dataset contains 17,802 ratings by 812 users. For each user, we take 60% of ratings as the training data and the rest 40% as the testing data.

4.3.2 Evaluation Metrics:

We evaluate our approach on three aspects.

- Graph reduction: Graph reduction measures the reduction of domain-specific subgraph DSG_n compared to the n-hop expansion subgraph in terms of the number of nodes, relationships and also the number of reachable paths within n-hops starting with the in-domain entities.
- Impact on accuracy: During the process of reducing the graph, it is important to make sure that the graph reduction does not occur at the cost of the accuracy of the recommendation algorithm. To assess this, we use two measures. First we calculate the standard measure of `precision@n` as used in [Di Noia et al. 2012]. `precision@n` in comparison to other metrics such as recall is the most suitable evaluation metric for recommender systems, particularly where the number of recommended items are pre-obtained [Shani and Gunawardana 2011]. However, `precision@n` only measures binary relevancy of the items up to n^{th} rank but does not capture whether domain-specific subgraph DSG_n replace any highly relevant items selected using n-hop

[4] `http://challenges.2014.eswc-conferences.org/index.php/RecSys#DBbook_dataset`

expansion subgraph with relatively low relevant items. To quantify this, we leverage the ratings provided by users for each item in the gold standard datasets. Ratings provided by the users will give an indicator whether domain-specific subgraph DSG_n replace any highly relevant items from n-hop expansion subgraph.

We take the average of the ratings of the user from the gold standard dataset. Then we calculate the deviation of the rating for each relevant top-n items from the average rating and finally take the mean of the deviation for all relevant items. A higher positive deviation value reflects the better results. This measure `ratingdev` for a given user u is formalized as,

$$ratingdev(u) = \frac{\sum_{r \in R} itemrating_r - avgrating_u}{|R|}$$

where R is the top n relevant items for user u, $itemrating_r$ is the rating given by the user u for item r, and $avgrating_u$ is the average rating for user u.

- Impact on runtime performance: Graph reduction should be able to reduce the time taken to run the recommendation algorithm. The most time consuming module in the recommendation algorithm is to calculate the similarity of all item pairs as it requires traversing the whole KG. Hence, we compare the runtime performance required for this step using the domain-specific subgraph DSG_n and n-hop expansion subgraph. We use the Neo4j[5] graph database to navigate the graph and calculate the item similarities for recommendation.

4.3.3 Evaluation Results

Using the aforementioned metrics we evaluate the quality of domain-specific subgraph extracted by our approach and present the results in this section.

	Path-based			Type-based		
	Relations	**Nodes**	**Paths**	**Relations**	**Nodes**	**Paths**
2-hop	349	1.07M	108.4M	349	1.07M	108.4M
$DSG_{2(15,15)}$	15(95.7%)	0.08M(92.0%)	5.08M(95.3%)	14(95.9%)	0.13M(87.6%)	17M(83.9%)
$DSG_{2(25,25)}$	25(92.8%)	0.13M(87.3%)	17.4M(83.8%)	24(93.1%)	0.63M(40.9%)	61.6M(43.19%)
$DSG_{2(35,35)}$	35(90%)	0.64M(40.7%)	61.64M(43.1%)	32(90.8%)	0.64M(40.7%)	61.62M(43.18%)

Table 4.1: Graph reduction statistics for 2-hop expansion subgraph and DSG_2 on movie domain; M denotes millions

	Path-based			Type-based		
	Relations	**Nodes**	**Paths**	**Relations**	**Nodes**	**Paths**
2-hop	424	1.2M	793.4M	424	1.2M	793.4M
$DSG_{2(15,15)}$	15 (96.5%)	0.09M(92.8%)	159.6M(79.9%)	15(96.5%)	0.09M(92.8%)	159.7M(80%)
$DSG_{2(25,25)}$	25 (94.1%)	0.62M(49.1%)	465.6M(41.5%)	25(94.1%)	0.62M(49%)	464.4M(41.68%)
$DSG_{2(50,50)}$	50 (88.2%)	0.68M(44.8%)	484.4M(39.2%)	38(91.0%)	0.63M(48.5%)	464.6M(41.66%)

Table 4.2: Graph reduction statistics for 2-hop expansion subgraph and DSG_2 on book domain; M denotes millions

	Path-based			Type-based		
	Relations	**Nodes**	**Paths**	**Relations**	**Nodes**	**Paths**
Movie-3-hop	636	2.86M	4885.3M	636	2.86M	4885.3M
$Movie-DSG_{315,25,15}$	30(95.3%)	0.19M(93.2%)	48.8M(98.9%)	24(96.2%)	0.26M(90.9%)	105.5M(97.83%)
$Book-$3-hop	641	3.2M	13852.8M	641	3.2M	13852.8M
$Book-DSG_{315,25,15}$	31(95.2%)	0.18M(94.2%)	1082.6M(92.18%)	21(96.7%)	0.12M(96%)	1062.5M(92.33%)

Table 4.3: Graph reduction statistics for 3-hop expansion subgraph and DSG_3 on movie and book domain; M denotes millions

4.3.3.1 Graph reduction

Tables 4.1, 4.2, and 4.3 show the number of nodes, relationships and paths reached via n-hop expansion subgraph and $DSG_{n(k_1,.,k_n)}$ with the `top-K` relationships at each hop. They also include the reduction percentage from n-hop expansion subgraph to DSG_n in parenthesis. Tables 4.1 and 4.2 show the reduction statistics for the domain-specific subgraphs created with 2-hops for the movie and book domains respectively.

Domain-specific subgraphs created with the `top-15` relationships at each hop reduce the n-hop expansion subgraph by over 80% to 90% (3^{rd} row in Tables 4.1 and 4.2) for both the domains. As we use more relationships to extract the domain-specific subgraph, the reduction percentage is decreased. The graph reduction percentage from both type-based and path-based measures are almost similar except for movie 2-hop graph with `top-25` at each hop in which type-based reduction is less than the path-based reduction.

Table 4.3 shows graph reductions for domain-specific subgraphs created with 3-hops for both the domains.

Domain-specific subgraphs were able to reduce the graph by 90% - 96%. This is slightly higher than the reduction from 2-hop graphs. An important aspect to note here is the significant increase in the number of reachable paths when the hop size changes from 2 to 3.

4.3.3.2 Impact on accuracy

precision@n

We calculate the `precision@n` for different ranks and average it over all the users. Fig. 4.5 shows the `precision@n` for all the users from the MovieLens dataset. We calculate the precision for both 2-hop expansion subgraph and DSG_2. Like the graph reduction, we experiment with different `top-Ks` in selecting the DSG_2.

For the movie domain, DSG_2 selected with the `top-15` relationships at each hop via path-based

[5] `http://neo4j.com/`

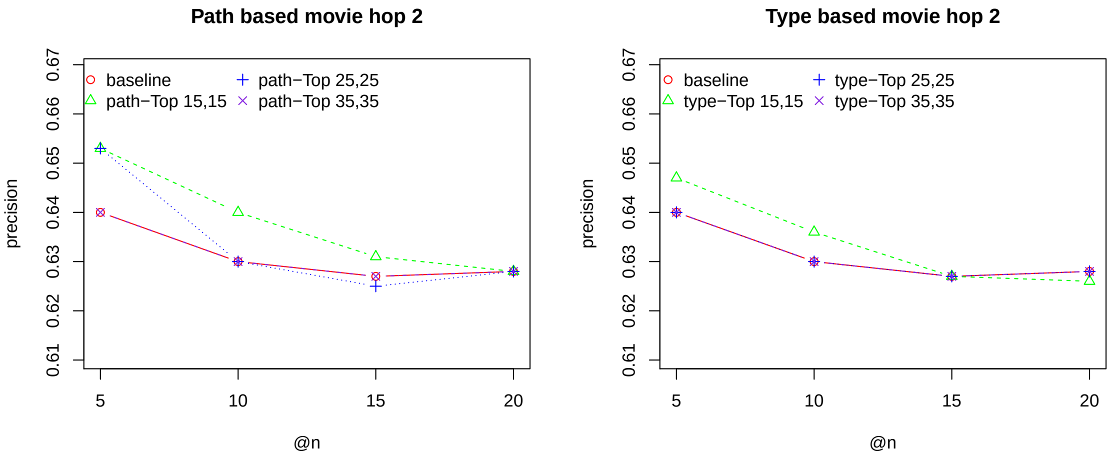


Figure 4.5: Precision for 2-hop expansion subgraph and movie DSG_2 on the movie domain.

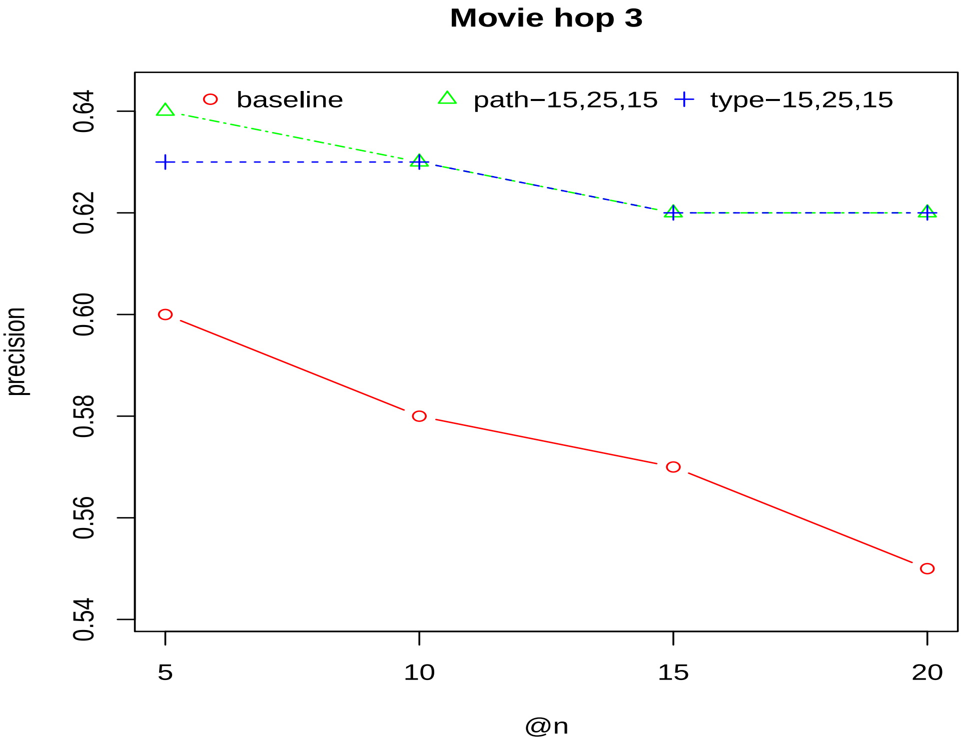


Figure 4.6: Precision for 3-hop expansion subgraph and movie DSG_3 on movie domain

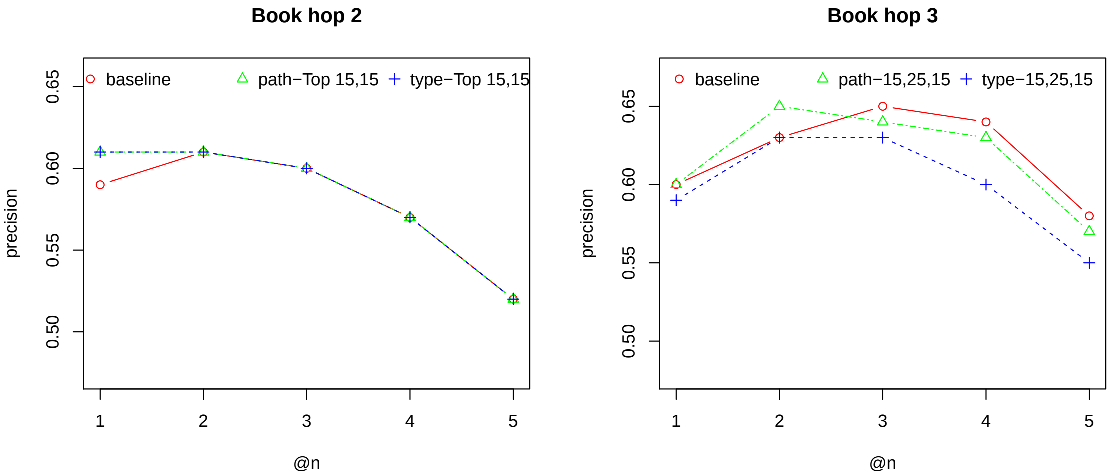


Figure 4.7: Precision for 2 and 3-hop expansion subgraph and DSG on book domain.

scoring technique has the best performance. It performs better than the baseline. This indicates that merely selecting n-hop subgraph can also negatively impact the results. It increased the precision by 2% for `top-5` recommendations. As we increase the number of relationships it decreases the performance and converges with the baseline at `top-35` relationships. Type-based scoring also improves the baseline (1% for top 5 ratings).

For the rest of the evaluation, we pick the domain-specific subgraph created with `top-15`, `25` and `15` at each hop which performed best out of the various `top-K` values and present the results. Fig. 4.6 shows the `precision@n` over all users for the recommendation algorithm with DSG_3 for the movie domain. Fig. 4.7 shows the same results for book domain with DSG_2 and DSG_3.

For the movie domain DSG_3 created with both type and path-based measures, performs equally well and outperforms the baseline. It increased the precision by 6.7% for the top 5 ratings. For the book domain, path-based scoring measure performs slightly better than the type-based scoring measure and is on par with the baseline. However, for the domain-specific subgraph of 3-hops, type-based scoring measure underperforms in comparison to the baseline.

	2-hop		3-hop	
	Baseline	$DSG_{215,15}$	**Baseline**	$DSG_{315,25,15}$
5	0.822	0.823	0.807	0.823
10	0.814	0.816	0.806	0.815
15	0.810	0.811	0.806	0.811
20	0.806	0.807	0.805	0.806

Table 4.4: Deviation from average rating for Movie

	2-hop		3-hop	
	Baseline	$DSG_{215,15}$	**Baseline**	$DSG_{315,25,15}$
1	0.592	0.584	0.533	0.558
2	0.599	0.604	0.571	0.579
3	0.601	0.614	0.569	0.579
4	0.606	0.617	0.595	0.595
5	0.610	0.620	0.596	0.6

Table 4.5: Deviation from average rating for Book

ratingdev

As *precision@n*, we calculate the *ratingdev* for different ranks and average it over all the users. We pick the path-based measures to present the results for *ratingdev* as it shows the better performance in comparison to the type-based measures. Table 4.4 shows the deviation from the average ratings for movie domain with the domain-specific subgraphs $DSG_{215,15}$ and $DSG_{315,25,15}$ in comparison to the n-hop expansion graph. Table 4.5 shows the same results for book domain.

For both the domains, deviation from the average rating performs equally well or outperforms

	Movie			Book		
	n-hop Expansion	DSG		n-hop Expansion	DSG	
		Path Rank	Type Rank		Path Rank	Type Rank
2-Hop	72 s	5s	11.2 s	10.15 m	1.3 m	1.4 m
3-Hop	2h 35 m	76 s	3.2 m	7 h 40 m	15.2 m	27 m

Table 4.6: Time performance improvement; s - seconds, m - minutes and h - hours

the baseline except for 1^{st} rank for book 2-hop graph. This indicates that our approach did not replace highly relevant items from the $n-$hop expansion subgraph and also was able to identify more relevant items using the domain-specific subgraph. For example, the top-5 items recommended by the algorithm using n-hop expansion subgraph to a user in MovieLens dataset, have only 3 relevant items with ratings 3, 3 and 2. But using DSG the user was given 5 relevant items with ratings 4, 4, 5, 3 and 4. $ratingdev$ was increased by 2.5% and 3.7% for movie and book domain respectively at their highest ranks (top-5 for movies and top-1 for book) for the 3-hop domain-specific subgraph. This shows our approach will particularly performs well as the hop increases.

4.3.3.3 Impact on runtime performance

Table 4.6 shows the time taken to calculate the item similarity which is the most time consuming component of recommendation system detailed in Section 4.3.1.1.

Results indicate a tenfold decrease in the amount of time taken to run the algorithm with a domain-specific subgraph in all cases. The reduction significance is higher as we increase the number of hops to cover the subgraph. As given in the Table 4.6, for the 3-hop subgraphs, time is reduced from hours to minutes.

To summarize, the evaluation results suggest that the domain-specific subgraph decreases the size of the graph by an average of 80% to 90% and still performs comparable to and in some cases better than the original subgraph. Graph reduction results in tenfold reduction in time to calculate the item similarity.

In some cases, path-based scoring measures perform better than type-based scoring. The quality of type-based scoring depends on the type assignments of the entities. Some entities are not assigned to the most specific class, and instead assigned to a more generic class in the schema. For example, DBpedia contains only 6,591 entities of type actor while there are 80,837 unique entities which are linked to movies via `starring` relationship.

4.4 Conclusion

In this work, we presented a novel approach to extract the domain-specific subgraph from a large, generic knowledge graph by ranking relationships to capture their domain specificity. The approach considers relationships as first-class elements and utilizes the semantics of non-hierarchical relationships to capture their domain specificity. We experimented with two novel measures that utilize intermediate entity types and intermediate relationships from the in-domain entities to determine the domain specificity. By restricting the KG to only domain-specific relationships and entities, we demonstrated its effectiveness for a recommendation use case on two domains, movie and book. Both our domain specificity measures were able to reduce the graph size by more than 80% which led to a tenfold decrease in computation time of the recommendation algorithm. The results also showed that there was no compromise in the accuracy of recommendations rather found more accurate results.

5

Identifying domain-specific subgraph with hierarchical relationships

5.1 Overview

Hierarchical relationships are one of the key components of knowledge graphs (KGs). They induce a structure of generalization and specialization of concepts[1] in a KG. For example, Fig. 5.1 shows a subgraph of the Wikipedia Category Hierarchy (WCH) comprising of entities and categories. In the Fig. 5.1, the entity *Franco Zeffirelli* connects to the category *Italian Film Directors* using the hierarchical relationship *broader*, and the category *Italian Film Directors* connects to the category *European Film Directors* using the same relationship.

Commonly used KGs such as DBpedia, Yago, and Freebase contain a significant number of facts expressed with hierarchical relationships. Fig. 5.2 shows the dominance of the number of facts expressed with hierarchical relationships compared to the facts expressed with other relationships

[1] Concepts represent both entities (articles) and categories on Wikipedia [Lehmann et al. 2015].

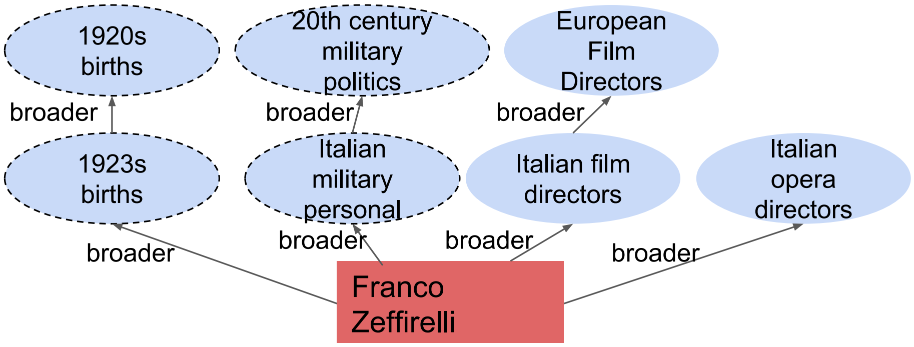


Figure 5.1: hop-based path navigation. Entities and Categories are indicated by rectangles and ovals respectively. Out of domain categories for the movie domain are indicated by ovals with dotted lines.

within 150 most frequently used DBpedia relationships. They play a major role in intelligent applications such as personalization [Kapanipathi et al. 2014], question answering [Welty et al. 2012], and recommendation systems [Ostuni et al. 2013] [Musto et al. 2014] [Di Noia et al. 2012] [Piao and Breslin 2016][Cheekula et al. 2015].

These applications face two primary challenges in consuming KGs: (1) they are computationally intensive due to the large number of facts in the KGs [Lalithsena et al. 2016]; and (2) many of these KGs are generic and comprise of facts representing multiple domains. On the other hand, most applications are domain-specific and may require knowledge that is only specific to the domain of interest. For example, a movie recommendation system may require knowledge related to only movies.

For example, Fig. 5.1 shows a subgraph extracted by traversing up to 2-hops from the entity *Franco Zeffirelli* in WCH. According to the subgraph, entity *Franco Zeffirelli* belongs to categories *Italian Film Directors*, *Italian Opera Directors*, *Italian military personal*, and *1923s births*. Out of these categories, the first two categories are relevant to the movie domain and the last two are irrelevant and might not be necessary for a movie recommendation system. Furthermore, Table 5.1 shows the exponential growth of the number of paths reached of a movie-specific subgraph created

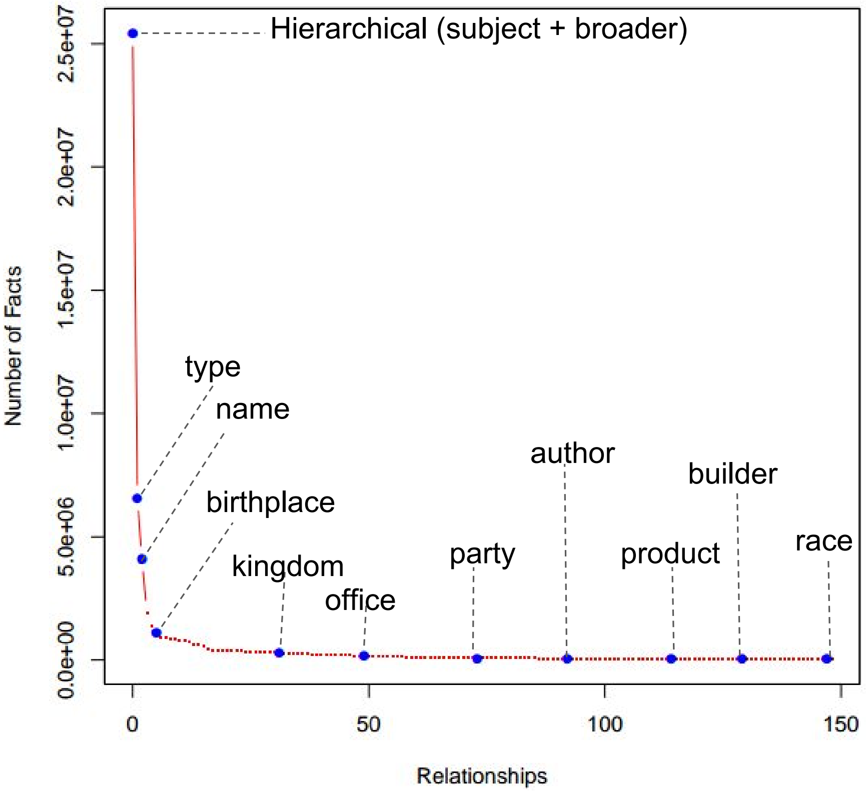


Figure 5.2: Number of facts for each relationship.

by navigating up to 4-hops in WCH, starting from 3, 121 movie entities in the MovieLens[2] dataset. Hence, the simple n-hop expansion technique to extract a domain-specific subgraph has not shown to be effective either in reducing size or noise from a large KG. Therefore, this chapter discusses how to extract a domain-specific hierarchical knowledge graph that reduces the size of the KG without compromising the accuracy of the applications for a given domain of interest.

We propose [Lalithsena et al. 2017] an evidence-based approach for extracting a domain-specific subgraph from a hierarchical KG. We selected WCH as the testbed to develop and evaluate our proposed approach due to its prominence and usage in existing domain-specific applications. WCH is the main hierarchical knowledge source for many KGs such as DBpedia and YAGO and consists of

[2] https://grouplens.org/datasets/movielens/

Table 5.1: Number of paths reached via seed movie enties; M - million; B - billion.

hops	Paths Reached
1	50953
2	18M
3	345M
4	72B

7.5 million entities and categories connected via 25 million hierarchical relationships.[3] Our approach collects evidence for supporting or opposing a category's relevance to a given domain in the KG. These pieces of evidence are based on type, lexical, and structural semantics of the categories. To systematically combine the different sources of evidence and to manage the uncertainty associated with each of the sources, we use the probabilistic soft logic (PSL) framework [Bach et al. 2015]. PSL is proven to work well in such environments and has generated state-of-the-art results [Kouki et al. 2015] [Pujara et al. 2013].

To demonstrate the effectiveness of the subgraphs extracted by our approach, we pick a recommendation use case which is an important application leveraging KGs. We show that it is possible to extract a domain-specific subgraph from a hierarchical knowledge graph, reducing around 40% - 50% of the paths compared to the subgraph created with an n-hop based navigation technique. This is done without compromising the accuracy of the recommendation algorithm and in most cases, domain-specific subgraphs extracted by our approach also improve the accuracy of the recommendation system. We also demonstrate that the recommendation results obtained by the subgraph created with our approach outperform the results obtained by a subgraph created with a supervised learning technique.

[3] http://wiki.dbpedia.org/downloads-2016-04

5.2 Approach

In order to extract a domain-specific hierarchical subgraph, our approach considers a set of domain entities as the input. These domain entities represent the domain of interest. For example, to create a movie-specific hierarchical subgraph a set of movie entities would be the input to our approach. Given the domain entities and the hierarchical graph, a domain-specific hierarchical subgraph is extracted by expanding the domain entities to only the categories that are specific to the domain. Existing approaches [Ostuni et al. 2013] consider connectedness to the domain entities in the hierarchical graph as the proxy to estimate the domain-specificity. For example, Fig. 5.3 shows a subgraph extracted by navigating 2 to 3 hops starting from five movies in WCH. While, this subgraph contains categories which describe the genre of the movie (*American LGBT related films*, *America spy films*) and the director of the movie (*Films Directed by Doug Liman*), it also contains out-of-domain categories such as *Museums in Popular Culture*, *LGBT Culture in US*, *Education*, etc. With respect to this subgraph, the domain-specific hierarchical subgraph should retain categories describing genre and director of the movies and eliminate the out-of-domain categories. In order to assess and retain the domain-specific categories, we have identified three different types of semantics of categories in the hierarchy that can be quantified and systematically leveraged, they are; 1) type semantics, 2) lexical semantics, and 3) structural semantics. These provide evidence for assessing the domain-specificity of categories.

We discuss these three types of semantics in Section 5.2.1 and Section 5.2.2 describes how the evidences obtained via these types of semantics are aggregated to calculate the domain-specificity of the categories using a probabilistic framework.

5.2.1 Evidence types towards domain-specificity

A category on Wikipedia can span across multiple topics. Our intuition is that the domain-specificity of a category can be estimated based on the relevancy of its associated topics to the domain. The

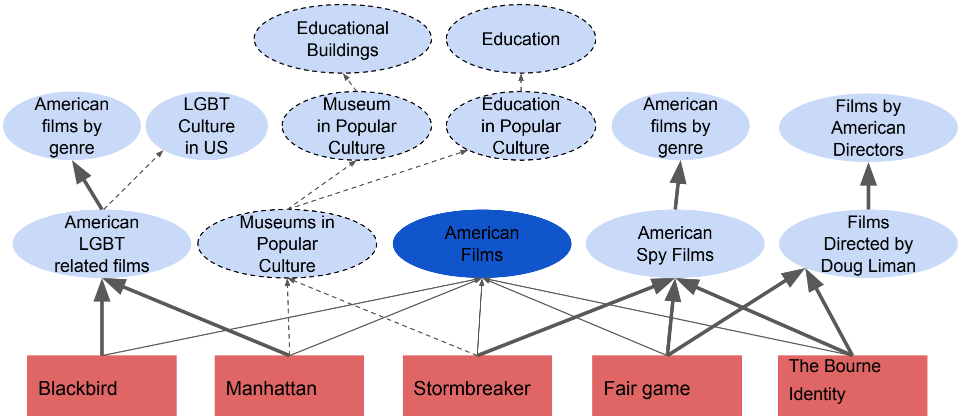


Figure 5.3: n-hop expansion subgraph. The graph is created by navigating 2 to 3-hops from five movies (marked in boxes). The in-domain and out-of-doman categories are marked by ovals with a solid lines and dotted lines respectively.

relevancy can be determined through the type and lexical semantics of the category label.

5.2.1.1 Type semantics

The topics of a category can be made explicit by identifying the types of the entities mentioned in the category label. For example as shown in Fig. 5.4(a) the category 1) *Films Directed by James Cameron* has types *Film* and *Film Director*; 2) *Documentary films about horror* has types *Documentary Films* and *Horror Film*; and 3) *Kingdom and countries of Austria-Hungary* has types *Monarchy* and *Country*.

The identified types can be utilized to show that first two categories are specific to the movie domain while third category is less/not specific to the movie domain.

5.2.1.2 Lexical semantics

The topics of a category label can also be derived using their lexical semantics. For example, categories *1997 anime* and *biographical work* have topics *animation films* and *biographical films* respectively. However, they cannot be identified via type semantics as current computational tech-

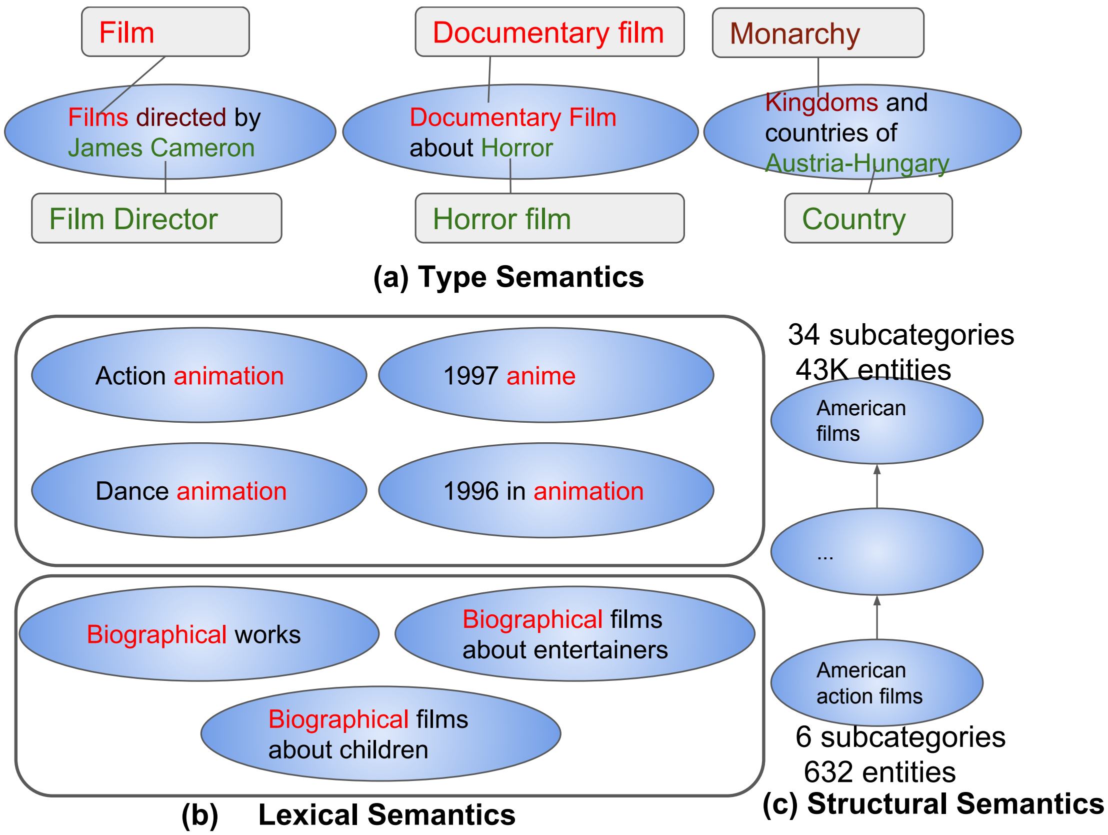


Figure 5.4: Evidence types for categories.

niques fall short in identifying entities within them. The lexical semantics of these labels can be used to group such categories with more descriptive categories and collectively identify their topics. Fig. 5.4(b) shows such grouping where category label *1997 anime* and *biographical works* are grouped with other category labels which share a similar lexical pattern. This grouping enrich the semantics of these categories and enable the identification of their respective topics (i.e *animation films* and *biographical film*). The domain-specificity calculated for these topics are considered as reflective to the domain-specificity of the categories in the group.

Section 5.2.2.2 details the techniques used to identify type and lexical semantics and calculate domain-specificity of these topics.

5.2.1.3 Structural semantics

Abstract categories are generally shown to have less importance in most domain-specific applications that utilize knowledge graphs [Tonon et al. 2013] [Welty et al. 2012]. For example, in a recommendation system, categories such as *American films* or *English-language films* may provide less or no impact in recommendations whereas specific categories such as *American action films* or *War epic films* are important for better recommendations. We utilize the structural semantics of hierarchies to quantify the abstractness of a category [Resnik 1995] [Seco et al. 2004]. Specifically, we consider the outdegree of a category to determine its abstractness. For example, Fig. 5.4(c) shows that the category *American films* is linked to 43k entities and 34 sub categories in the graph and *American action films* is only linked to 632 entities and 6 subcategories. Hence, measuring the specificity of a category using structural features provides another piece of evidence regarding the domain-specificity of a given category.

5.2.2 PSL Framework for category ranking

The three types of semantics can provide complementary and contrasting signal towards determining the domain-specificity of a category. Fig. 5.5 shows evidences collected for a few categories. The

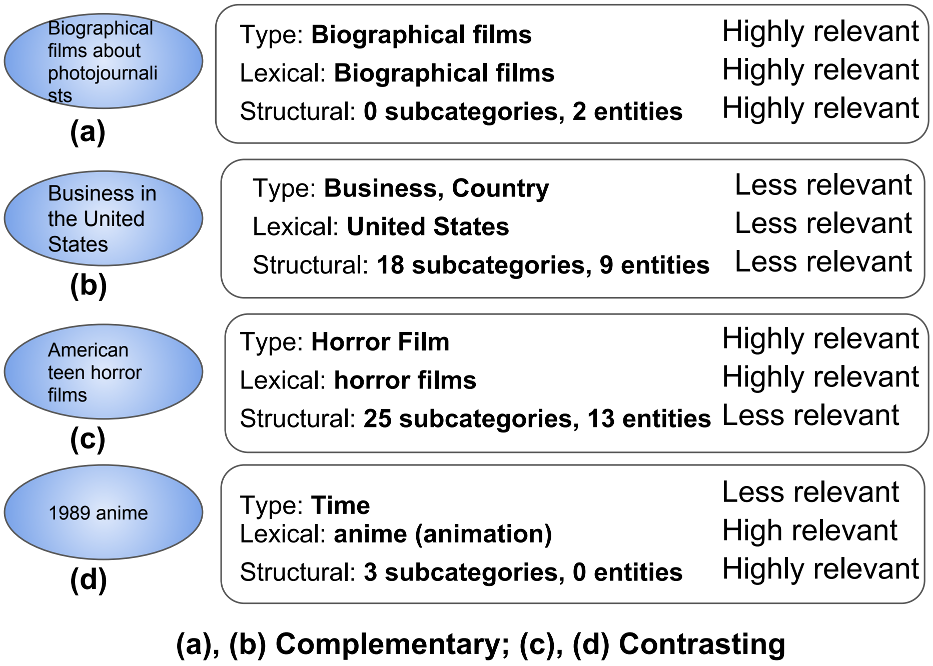

Biographical films about photojournali sts
(a)
Type: Biographical films
Highly relevant
Lexical: Biographical films
Highly relevant
Structural: 0 subcategories, 2 entities
Highly relevant
Business in the United States
(b)
Type: Business, Country
Less relevant
Lexical: United States
Less relevant
Structural: 18 subcategories, 9 entities
Less relevant
American teen horror films
(c)
Type: Horror Film
Highly relevant
Lexical: horror films
Highly relevant
Structural: 25 subcategories, 13 entities
Less relevant
1989 anime
(d)
Type: Time
Less relevant
Lexical: anime (animation)
High relevant
Structural: 3 subcategories, 0 entities
Highly relevant
(a), (b) Complementary; (c), (d) Contrasting

Figure 5.5: Complementary and contrasting evidences.

evidences collected for both (a) and (b) categories are complementary. The type and lexical semantic evidence collected for (c) are complimentary while structural evidence provides a contrasting signal to them. The type semantic evidence provides a contrasting signal to other two types of evidences collected for (d). Our approach uses a probabilistic framework to aggregate these complementary and contrasting evidences in a principled way. Specifically, we use probabilistic soft logic (PSL), which is a statistical relational learning framework with a declarative language as the interface. Section 5.2.2.1 provides a brief overview of PSL and Section 5.2.2.2 describes the category ranking using PSL.

5.2.2.1 A Brief Introduction to PSL

PSL is a declarative language which uses first order logic to specify probabilistic models. PSL mainly has: (1) predicates - P, (2) atoms - P(A, B), and (3) weighted rules. A weighted rule will be of the form $w : X \to Y$, where X can be a conjunction, disjunction or an individual atom. Eq. 5.1 shows a simple rule where P is a predicate, A, B, and C are variables, and w is the weight.

$$w : P(A,B)\tilde{\wedge}P(B,C) \rightarrow P(A,C) \tag{5.1}$$

The grounding of each atom happens when the variables are instantiated with individuals. A grounded atom $P(a,b)$, where a and b are the individuals for A and B, can take a continuous value ranging from 0 to 1. The capability to deal with continuous values instead of a boolean value for atoms makes PSL more useful and practically applicable for many scenarios [Kouki et al. 2015]. The value of each atom can either be observed or unknown.

PSL uses lukasiewicz t-norm [Bach et al. 2015] to provide a relaxation for the logical connectives $\wedge$ and $\vee$. It assigns truth values to the logical connectives as follows.

$$p\tilde{\wedge}q = max(0, p+q-1) \tag{5.2}$$

$$p\tilde{\vee}q = min(1, p+q) \tag{5.3}$$

Using lukasiewicz t-norm, PSL assigns a distance to satisfaction for each grounded rule. The distance to satisfaction to the grounded rule for the rule in Eq. 5.1 will be $max((P(a,b)\tilde{\wedge}P(b,c)) - P(a,c), 0)$.

PSL rules with grounded atoms and continuous values to represent probabilities for each grounded atom define features for Markov networks, which are probabilistic graphical models used for collective inferencing. In particular, the PSL implementation used in this work employs Hinge-loss Markov Random Fields (HL-MRFs), which are probabilistic models over continuous random variables. HL-MRFs infer the values for the unknown variables in the model by looking at the most probable explanation (MPE). MPE tries to find an interpretation I (the most possible assignment of the soft truth values) such that it minimizes the distance to satisfaction for grounded rules R. The probability distribution over interpretation I for a set of grounded rules R can be formulated as,

$$f(I) = \frac{1}{Z}\exp[-\sum_{r\in R} w_r(d_r(I))^p] \tag{5.4}$$

where w_r is the weight of rule r, d_r is the rules's distance to satisfaction, p is the distance exponent, which can be 1 or 2, and Z is a normalization constant. For a detailed description of PSL, see [Bach et al. 2015].

5.2.2.2 PSL rules for scoring domain-specificity of categories

The proposed framework combines type, lexical, and structural semantic evidence obtained from categories using PSL to assess the domain-specificity of a category. Here, we have expressed these in the form of rules in the PSL model. The domain-specificity of categories will be the value to be inferred (unknown) and the types of evidence will be the observed (known) values. Now, we present the rules and their expansions obtained using different computational techniques.

PSL rules for type semantics

Type semantics harnesses the semantics of topics mentioned in the category labels. In order to extract the topics, we perform entity annotation on labels, specifically using the DBpedia Spotlight annotation tool [Mendes et al. 2011].[4] We consider the types (*rdf:type*) of the annotated entities as topics. When the type of the annotated entity is *owl:Thing*, we consider the entity itself as the topic. The domain-specificity of a category is calculated by averaging the domain-specificity of these topics.

The domain-specificity of a topic is calculated by measuring its similarity to the domain term. The domain term is the *rdfs:label* of the class representing the domain in DBpedia. For example, for the movie domain, we consider the DBpedia class *dbo:Film* and use its label, which is *film*, as the domain term. The similarity between domain term and the topics extracted is calculated using two most prominent state-of-the-art techniques; 1) word2vec similarity [Mikolov et al. 2013]: embedding-based semantic similarity measure and 2) UMBC similarity [Han et al. 2013]: hybrid semantic similarity measure. The hybrid similarity measure uses information derived from both the knowledge base and the corpus. The rules defined using type semantics and their expansion using two similarity measures are shown in Equations 5.5 and 5.6,

[4] As DBpedia Spotlight does not capture the temporal annotations, we implement a regular expression based temporal annotator to identify the time mentions in a category label

$$\begin{aligned} &semtype(Cat, TypeSet) \tilde{\wedge} semtypesim_{uc}(TypeSet, Dom) \\ &\qquad \rightarrow domainspec(Cat, Dom) \end{aligned} \tag{5.5}$$

$$\begin{aligned} &semtype(Cat, TypeSet) \tilde{\wedge} semtypesim_{w2v}(TypeSet, Dom) \\ &\qquad \rightarrow domainspec(Cat, Dom) \end{aligned} \tag{5.6}$$

where *semtype*, *semtypesim*, and *domainspec* are predicates. *Cat* and *Dom* denote the category variable and domain variable respectively, *uc* and *w*2*v* denotes two similarity measures. *semtype* predicate defines the relationship between a category and a set of topics. *semtypesim* and *domainspec* predicates capture the similarity of set of topics to the domain and membership value of a category to the domain respectively. The above rules state that the membership of the category to the domain is determined by its topics and similarity of those topics to the domain term.

PSL rules for lexical semantics

Lexical semantics groups the categories based on their labels and then identify topics for each group. These topics act as a reference for measuring the domain-specificity of the members of that group. In order to perform the grouping, we use k-means clustering [MacQueen et al. 1967] to cluster the category labels. The category labels are represented as vectors and these vectors are obtained by averaging the word2vec embedding vectors of the words present in the category. This simple averaging method has proven to be a strong baseline for multiple tasks [Kenter et al. 2016].

The topics for each group are obtained using two methods. First, we take the most frequent entity/type obtained via DBpedia Spotlight annotations in a given cluster as its topic. Second, we extract the most frequent bigram of a given cluster as topics. Finally, we measure the semantic similarity of these two topics to the domain term using UMBC and word2vec similarity measures. PSL rules derived from lexical semantics are shown in Equations 5.7, 5.8, 5.9, and 5.10,

$$\begin{split} lexclutype(Cat, Clus) \tilde{\wedge} lexclussim_{db,uc}(TClus_{Clus}, Dom) \rightarrow \\ domainspec(Cat, Dom) \end{split} \tag{5.7}$$

$$\begin{split} lexclutype(Cat, Clus) \tilde{\wedge} lexclussim_{db,w2v}(TClus_{Clus}, Dom) \rightarrow \\ domainspec(Cat, Dom) \end{split} \tag{5.8}$$

$$\begin{split} lexclutype(Cat, Clus) \tilde{\wedge} lexclussim_{bi,uc}(TClus_{Clus}, Dom) \rightarrow \\ domainspec(Cat, Dom) \end{split} \tag{5.9}$$

$$\begin{split} lexclutype(Cat, Clus) \tilde{\wedge} lexclussim_{bi,w2v}(TClus_{Clus}, Dom) \rightarrow \\ domainspec(Cat, Dom) \end{split} \tag{5.10}$$

where *db* denotes the DBPedia Spotlight, *bi* denotes the bigram, $Clus$ denotes a cluster and $TClus_{Clus}$ denotes the topic for the cluster $Clus$. The predicate *lexclutype* represents the membership of a category to the cluster $Clus$. $lexclussim_{bi,uc}$ predicate is used to represent the similarity of the topic obtained via bigram to the domain term using UMBC similarity. The above rules state that the membership of the category to the domain is determined by the cluster that it belongs and similarity between the topics derived for that cluster and the domain term.

PSL rules for graph structural features

In order to measure the structural specificity of a category, we use the inverse of the out-degree of a category. Here, the out-degree refers to the sum of the number of entities assigned to the category and the number of sub-categories subsumed by the category. This is shown in the PSL rule 5.11,

$$graphspec(Cat) \rightarrow domainspec(Cat, Dom) \tag{5.11}$$

where *graphspec* is the predicate which captures the specificity value. Hence, the structural specificity directly determines the domain-specificity of the category.

Any additional types of evidence can be easily incorporated into the PSL model. PSL rules require a weight for each of the rules and these weights can either be pre-specified or can be learned. In order to learn the weights, PSL needs training data for each rule. However, it is hard to create training data in a generic way which works for any domain. So, we determine the weights of the rules via an empirical experiment, as describe in the evaluation section.

The domain-specific subgraph is created by restricting the subgraph created by navigating n-hops to the top-K domain-specific categories determined by the PSL model.

5.3 Evaluation with a Recommendation Usecase

Following the related work on domain-specific subgraph extraction [Lalithsena et al. 2016], we use recommendation systems as a use case for evaluating the effectiveness of domain-specific hierarchical subgraphs. We adapt evaluation setup in [Lalithsena et al. 2016] to evaluate domain-specific hierarchical subgraphs where we use an existing KG-based recommendation algorithm.

5.3.1 Evaluation Setup

5.3.1.1 Recommendation algorithm

A KG-based recommendation algorithm is implemented for the evaluation [Di Noia et al. 2012]. The algorithm in [Di Noia et al. 2012] recommends items based on its similarities determined using a similarity function on the KGs. Detailed algorithm is described in Section 4.3.1.1.

5.3.1.2 Baseline: Recommendations with ***EXP-DSHG***$_n$

Existing content-based recommendation systems using hierarchical KGs create domain-specific subgraphs by navigating n-hops from domain entities [Ostuni et al. 2013] [Di Noia et al. 2012]. Hence, we consider the recommendation algorithm presented in Section 4.3.1.1 with the domain-specific hierarchical subgraph created with n-hop navigation on WCH as our baseline. We experimented with $n = 2$ and $n = 3$ for the evaluation.

5.3.1.3 Our Approach: Recommendations with *PSL-DSHG*$_n$

For our approach, we use the n-hop domain-specific hierarchical subgraph of the WCH created using our PSL framework. One of the parameters in our algorithm is the set of weights for the identified rules in Section 5.2.2.2. In order to find the optimal weights for the PSL rules, we conducted an user study with three users who empirically evaluated the generated ranked list of categories for different weight combinations. In order to reduce the complexity of this experiment, we assume that rules only differ due to the similarity measure being used, have same weight value. Hence, the goal of this experiment was to find the weights for; 1) the type semantic rules, 2) the lexical semantic rules derived using DBpedia annotations, 3) the lexical semantic rules derived using bigrams, and 4) the structural semantic rule. This experiment is conducted in an incremental manner. First step fixes the weights of rules 2 and 3 and vary the weights of rules 1 and 4 between 0 and 10 with step size 1. The next step vary the weight of rule 2 with the best combination of weights found for rules 1 and 4. Last step repeats this for rule 3 after finding the weights for rules 1, 2, and 4. The results of this experiment showed that the ranked list of categories generated with weight values 8 for type semantic rules, 6 for structural semantic rules, 6 for lexical semantic rules with DBpedia features, and 1 for lexical semantic rules with bigram features is better than the other ranked lists. Hence, we used the subgraph created with these rule weights for the experiments presented here.

5.3.1.4 Supervised Approach: Recommendations with *SUP-DSHG*$_n$

An existing approach [Mirylenka et al. 2015] on bootstrapping a domain ontology from Wikpedia category hierarchy uses a binary classifier to determine the domain relevance of Wikipedia categories. Given a root category (category Film) that represents the domain, their approach performs a breadth first graph traversal from the root category up to a given max depth and identifies relevant and irrelevant categories to the domain of interest. They showed that the model trained for determining the domain relevancy for a particular domain can be used to determine the domain relevancy of a another domain.

Hence, we have used the provided training data on computing domain and derived the domain-specific subgraphs for movie and book domains. We pick the max depth in a way that it reaches all the domain entities in our evaluation datasets. The depth values used for the movie and book are 7 and 8 respectively. Again, we use the recommendation algorithm outlined in Section 4.3.1.1 on the subgraph created with this supervised approach SUP-DSHG$_n$.

5.3.1.5 Datasets

In order to evaluate the effectiveness of our algorithm for multiple domains, we used well-known datasets from two domains. (1) MovieLens dataset [Herlocker et al. 2000] for the movie domain and DBbook[5] dataset for the book domain. Detailed description of the datasets can be found in Section 4.3.1.4.

5.3.2 Evaluation Metrics

Our primary goal is to reduce a large KG to a domain-specific subgraph while preserving the performance of the application that utilizes the domain-specific subgraph. Based on this, we evaluate our approach on the two aspects described below for the recommendation use case.

- Graph reduction: Graph reduction measures the reduction of *PSL-DSHG$_n$* and *SUP-DSHG$_n$* compared to *EXP-DSHG$_n$* in terms of the number of categories, and the number of reachable paths within n-hops starting from domain entities.

- Impact on accuracy: We used two measures to calculate the accuracy of the recommendation algorithm. First, we calculate the standard measure of `precision@n` as used in [Di Noia et al. 2012]. As `precision@n` only measures the binary relevancy of the items up to the n^{th} rank, it does not capture whether recommendations obtained with *PSL-DSHG$_n$* or *SUP-DSHG$_n$* replaces

[5] http://challenges.2014.eswc-conferences.org/index.php/RecSys

any highly relevant items selected using *EXP-DSHG*$_n$ with relatively less relevant items. To quantify this, we came up with a measure `ratingdev` by leveraging the ratings provided by users for each item in the gold standard datasets. Equation for the `ratingdev` and its interpretation is detailed in Section 4.3.2.

5.3.3 Evaluation Results on *PSL-DSHG*$_n$ with *EXP-DSHG*$_n$

In this section, we present the results obtained by *PSL-DSHG*$_n$ in comparison to the EXP-DSHG$_n$ using the evaluation metrics detailed above.

5.3.3.1 Graph reduction

Tables 5.2, 5.3, 5.4, and 5.5 summarize the graph reduction results. These results are generated on 2 and 3 hop subgraphs for the movie and book domain. Our approach retains the `Top-`K domain-specific categories in the subgraph. If there are more than one category with the same domain-specific score as the K^{th} rank category, all those categories are included in the subgraph. Hence in Table 4.1 for *PSL-DSHG*$_2$*(*4500*)* where K is $4,500$, the overall movie subgraph extracted contains $4,782$ categories.

Table 5.2 shows the number of categories and paths reached in *EXP-DSHG*$_2$ and three *PSL-DSHG*$_2$s extracted by selecting different `top-`K for the movie domain. The reduction percentage from *EXP-DSHG*$_2$ to *PSL-DSHG*$_2$ is presented in parentheses beside the raw number of categories and paths. Tables 5.3, 5.4, and 5.5 portray the results for movie 3-hop, book 2-hop, and book 3-hop graphs.

In Table 5.2, for the movie 2-hop subgraphs, we can see that reducing the number of categories by 25% has led to a reduction of the total number of paths by 43% (the last row in Table 5.2), and in the case of movie 3-hop subgraphs, reducing the number of categories by 27% has led to a reduction of the total number of paths by 52% (the last row in Table 5.3). As expected, reducing K in `top-`K to extract the domain-specific subgraph, increases the reduction percentage.

	Categories	**Paths**
EXP-DSHG$_2$	6413	18M
PSL-DSHG$_2$*(3500)*	3844(40%)	1.62M(91%)
PSL-DSHG$_2$*(4000)*	4315(33%)	10.8M(44%)
PSL-DSHG$_2$*(4500)*	4782(25%)	10.26M(43%)

Table 5.2: Graph reduction statistics of *PSL-DSHG*$_2$*(K)* in comparison to *EXP-DSHG*$_2$ in the movie domain; M denotes millions, K denotes `top-`K categories.

	Categories	**Paths**
EXP-DSHG$_3$	12348	320M
PSL-DSHG$_3$*(6500)*	6534(47%)	106M(67%)
PSL-DSHG$_3$*(7500)*	7508(39%)	115M(64%)
PSL-DSHG$_3$*(9000)*	9015(27%)	151M(52%)

Table 5.3: Graph reduction statistics of *PSL-DSHG*$_3$*(K)* in comparison to *EXP-DSHG*$_3$ in the movie domain; M denotes millions, K denotes `top-`K categories.

	Categories	**Paths**
EXP-DSHG$_2$	8603	2.2M
PSL-DSHG$_2$*(5000)*	5155(40%)	1.4M(36%)
PSL-DSHG$_2$*(5500)*	5847(32%)	1.6M(27%)
PSL-DSHG$_2$*(6000)*	6297(27%)	1.8M(18%)

Table 5.4: Graph reduction statistics of *PSL-DSHG*$_2$*(K)* in comparison to *EXP-DSHG*$_2$ in the book domain; M denotes millions, K denotes `top-`K categories.

	Categories	**Paths**
EXP-DSHG$_3$	18680	22M
PSL-DSHG$_3$*(6500)*	6868(63%)	9M(73%)
PSL-DSHG$_3$*(7500)*	7504(60%)	12M(45%)
PSL-DSHG$_3$*(8500)*	8916(52%)	14M(35%)

Table 5.5: Graph reduction statistics of *PSL-DSHG*$_3$*(K)* in comparison to *EXP-DSHG*$_3$ in the book domain; M denotes millions, K denotes `top-`K categories.

The book 2-hop subgraph was only able to reduce the number of paths by 18% when reducing the number of categories by 27% as shown in Table 5.4. This is comparatively lesser than the movie domain. This is due to the cardinality of links between the domain entities and categories for each domain on Wikipedia. Specifically, there are fewer links between nodes in the book domain in comparison to the movie domain. On an average, a movie entity is connected to 2 categories ($3,121$ entities – $6,413$ categories) whereas a book entity is connected to 1.1 categories ($7,632$ entities – $8,603$ categories) on Wikipedia. However, as we increase the number of hops for the book domain, domain-specific subgraphs were able to increase the reduction. The book 3-hop subgraph was able to reduce the number of paths by 35% by reducing the number of categories by 52% as shown in Table 5.5. An important aspect to note in both the domains is the significant increase in the number of reachable paths when the hop size changes from 2 to 3.

5.3.3.2 Accuracy

precision@n

The reduction percentages reported above are only meaningful if the domain-specific subgraphs (*PSL-DSHG*$_n$) do not compromise the accuracy of the application in comparison to the n-hop expansion domain-specific subgraph (*EXP-DSHG*$_n$). Therefore, to assess this, we compare the performance by calculating the precision for `top-`n recommendations provided for each user and then averaging it over all the users in the dataset.

Fig. 4.5 shows the `precision@n` for all the users from the MovieLens dataset. We calculate the precision for both *EXP-DSHG*$_2$ and *PSL-DSHG*$_2$. We experimented with same `top-Ks` presented for graph reduction in selecting the *PSL-DSHG*$_2$. Movie 2-hop subgraph extracted with `top-`4500 categories show the best performance. It increased the precision by 1.5% for the `top-5` and `top-10` recommendations over the baseline. This indicates that merely selecting n-hops to create a subgraph can sometimes negatively impact the results. In other words, selecting the domain-specific categories and paths can lead to better performance of the overall application.

Fig. 5.6 shows the `precision@n` for all the users from the MovieLens dataset. We calculate the precision for both *EXP-DSHG*$_2$ and *PSL-DSHG*$_2$. We experimented with same `top-Ks` presented for graph reduction in selecting the *PSL-DSHG*$_2$. Movie 2-hop subgraph extracted with `top-`4500 categories show the best performance. It increased the precision by 1.5% for the `top-5` and `top-10` recommendations over the baseline. This indicates that merely selecting n-hops to create a subgraph can sometimes negatively impact the results. In other words, selecting the domain-specific categories and paths can lead to better performance of the overall application.

Fig. 5.7 shows the results for the 3-hop subgraphs. The best performance for the subgraph extracted is seen when selecting `top-`9000 categories and it also outperforms the baseline.

The recommendation results obtained for the 2-hop subgraphs on book domain are shown in the Fig. 5.8. The subgraph extracted with the `top-`5500 categories shows the similar performance as the baseline subgraph.

With respect to the book 3-hop subgraphs shown in Fig. 5.9, the subgraph extracted with `top-`7500 categories shows the best performance with an increase in precision by 3.3% for `top-1` recommendations. To conclude, all the domain-specific subgraphs extracted by our approach perform better or equally well when compared to the subgraph created by expanding set of domain entities by navigating n-hops.

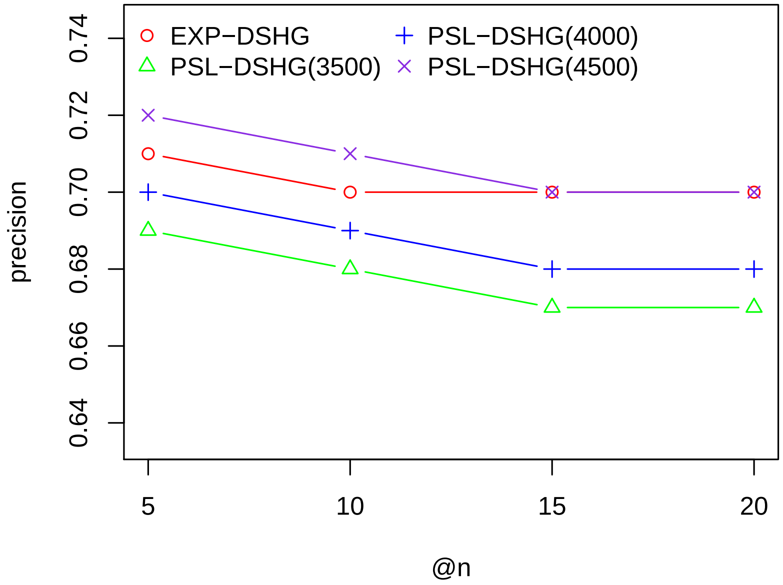


Figure 5.6: Precision@n for *PSL-DSHG*$_2$*(K)* and *EXP-DSHG*$_2$ on the movie domain; K denotes `top-`K categories.

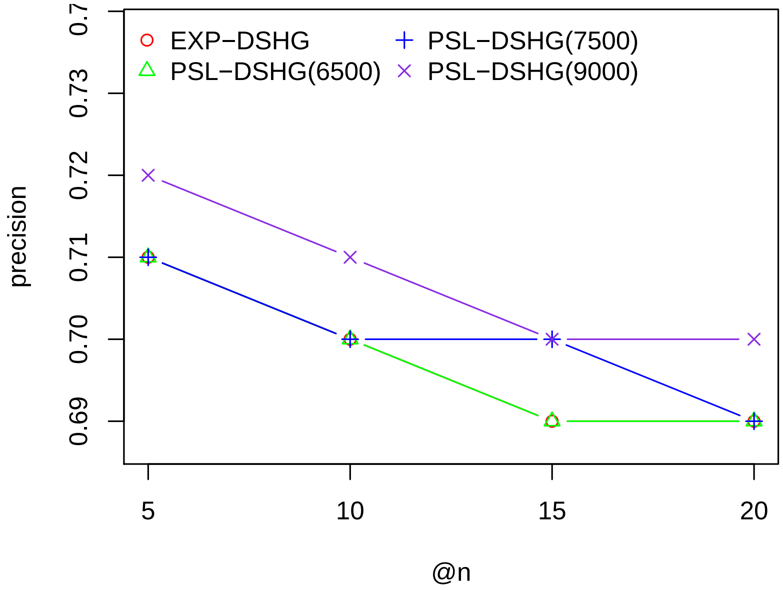


Figure 5.7: Precision for *PSL-DSHG*$_3$*(k)* and *EXP-DSHG*$_3$ on the movie domain; K denotes `top-`K categories.

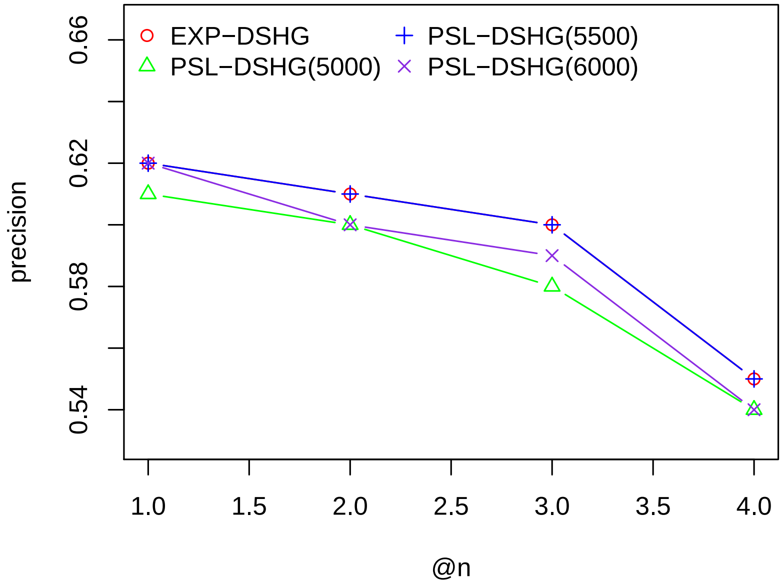


Figure 5.8: Precision for *PSL-DSHG*$_2$*(k)* and book *EXP-DSHG*$_2$ on the book domain; K denotes `top-`K categories.

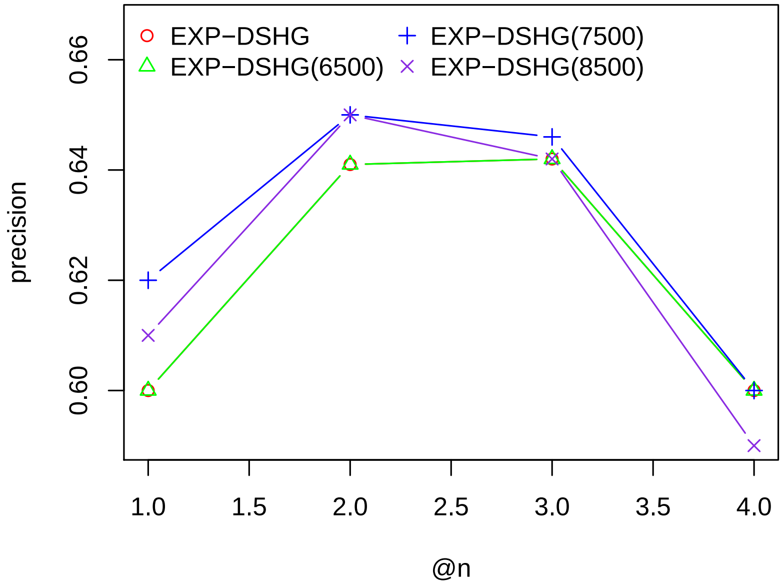


Figure 5.9: Precision for *PSL-DSHG*$_3$*(k)* and *EXP-DSHG*$_3$ on the book domain; K denotes `top-`K categories.

	2-hop		**3-hop**	
	EXP-DSHG$_2$	***PSL-DSHG***$_2$	***EXP-DSHG***$_3$	***PSL-DSHG***$_3$
5	0.87	0.876	0.87	0.87
10	0.852	0.857	0.851	0.852
15	0.842	0.849	0.842	0.844
20	0.837	0.842	0.836	0.84

Table 5.6: Deviation from average rating for Movie for *EXP-DSHG* and best performing *PSL-DSHG*; *PSL-DSHG*$_2$*(*4500*)* and *PSL-DSHG*$_3$*(*9000*)*

ratingdev

We calculate the *ratingdev* for different ranks and average it over all the users. We pick the best performing *PSL-DSHG*$_n$ in terms of `precision@n` to present the results in comparison to the *EXP-DSHG*$_n$. Tables 5.6 and 5.7 show the deviation from the average ratings for the movie and book domains calculated with 2-hop and 3-hop subgraphs. For both the domains, deviation from the average rating performs equally well or outperforms the results obtained with *EXP-DSHG*$_n$, except for top 3 results for the book 3-hop graph. Hence, we can conclude that our approach does not compromise the quality of recommendations by replacing highly rated recommendations. In most cases, our approach improves the recommendations by replacing low rated recommendations with better rated ones.

To summarize, the best performing movie-specific *PSL-DSHG*$_2$ outperforms the *EXP-DSHG*$_2$, with a 43% reduction in the number of paths, and the *PSL-DSHG*$_3$ outperforms the *EXP-DSHG*$_3$, with a 52% reduction in number of paths. The best performing book-specific *PSL-DSHG*$_2$ performs equally well with a 27% reduction in the number of paths and *PSL-DSHG*$_3$ outperforms with a 45% reduction in the number of paths with respect to the corresponding *EXP-DSHG*$_n$ subgraphs.

	2-hop		**3-hop**	
	EXP-DSHG$_2$	***PSL-DSHG***$_2$	***EXP-DSHG***$_3$	***PSL-DSHG***$_3$
1	0.619	0.623	0.613	0.632
2	0.617	0.617	0.622	0.629
3	0.610	0.617	0.632	0.625
4	0.613	0.613	0.627	0.627

Table 5.7: Deviation from average rating for Book for *EXP-DSHG* and best performing *PSL-DSHG*; *PSL-DSHG*$_2$*(*5500*)* and *PSL-DSHG*$_3$*(*7500*)*

5.3.4 Evaluation Results on *PSL-DSHG*$_n$ with the *SUP-DSHG*$_n$

In this section, we compare the reduction and `precision@n` results for subgraphs generated by our approach and a supervised approach.

5.3.4.1 Graph Reduction

In our approach, *PSL-DSHG* is generated by identifying the domain-specific categories from a n-hop expansion subgraph from the domain entities. While *SUP-DSHG* follows a similar procedure, its expansion graph is generated by navigating n-hops starting from a category representing the domain (e.g., category Film) rather than the domain entities. In order to compare this approach to ours, the graph expanded has to comprise of all the domain entities for recommendation. Therefore, we set the n to 7 and 8 for movie and book domains where the expanded graph contains all the domain entities in our evaluation datasets. The graph reduction statistics for *PSL-DSHG* and *SUP-DSHG* is presented in Table 5.8. For 2-hop subgraphs in both movie and book domains, we pick the best performing *PSL-DSHG* to report the results. As shown in the Table 5.8, movie path reductions are significantly higher in the *PSL-DSHG* in comparison to *SUP-DSHG*. But, book path reductions are higher in the *SUP-DSHG*.

	PSL approach		**Supervised approach**	
	Categories	Paths	Categories	Paths
Movie				
EXP-DSHG$_n$	6413	18M	77033	17M
**x-DSHG*	4782(25%)	10.2M(43%)	10576(86%)	16M(6%)
Book				
EXP-DSHG$_n$	8603	2.2M	45784	2.0M
**x-DSHG*	5847(31%)	1.6M(27%)	8521(81%)	1.0M(50%)

Table 5.8: Graph reduction statistics for *PSL-DSHG*$_2$ and *SUP-DSHG* with expansion graphs for movie and book domain; M denotes millions; *x refers either to PSL or SUP (depends on the column title).

5.3.4.2 Accuracy - precision@n

We compare the `precision@n` results of *PSL-DSHG* in comparison to the *SUP-DSHG* and the expansion graph of *SUP-DSHG* created by starting at the domain category (*EXP-DSHG*). The results are portrayed in Fig. 5.10 where *PSL-DSHG*$_2$ performs the best with an improvement of 3% for both `top-`5 and `top-`15 movie recommendations in comparison to *SUP-DSHG*. The performance of the recommendation system that utilize *SUP-DSHG* not only deteriorates in comparison to *PSL-DSHG* but also in comparison to its own expansion subgraph. This is also corroborated with the results shown in Fig.5.11 for the book domain. *PSL-DSHG*$_2$ performs the best with an improvement of 17% and 19% for `top-`3 and `top-`4 in comparison to the *SUP-DSHG*. *SUP-DSHG* is a good example that emphasizes the importance of the evaluation metrics selected and the requirements stated for a good graph reduction problem in this work. To reiterate, while reducing the graph to a domain-specific graph is the primary goal of the work, it is also important not to compromise the performance of the application that utilizes the reduced graph (in our case, the recommendation

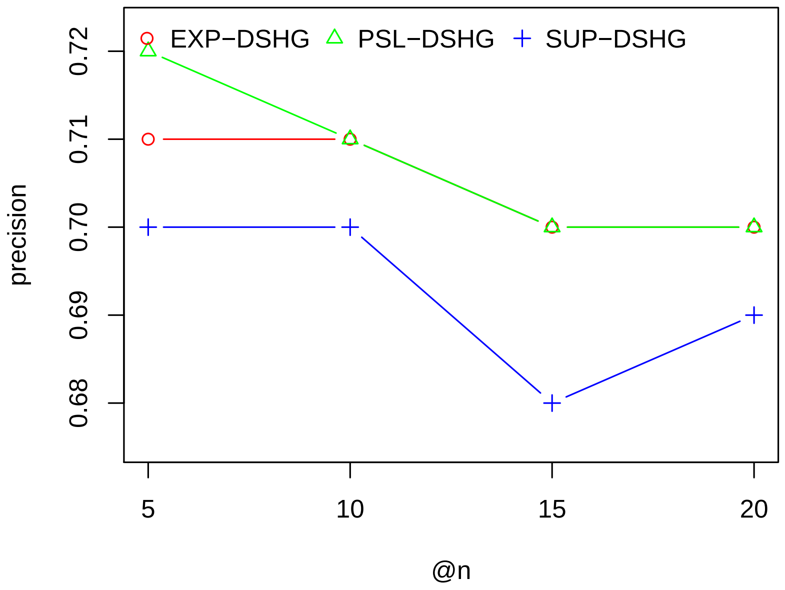


Figure 5.10: Precision@n for *PSL-DSHG*, *SUP-DSHG*, and *EXP-DSHG* for movie 2-hop.

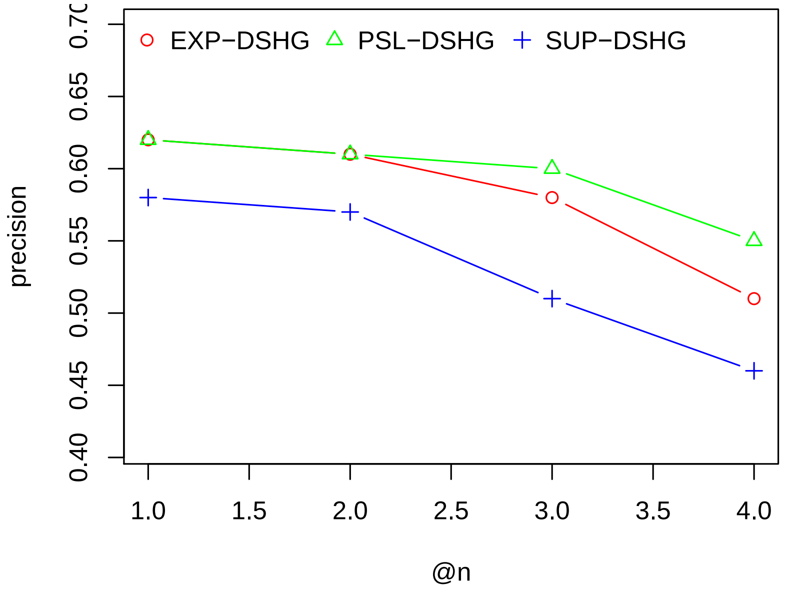


Figure 5.11: Precision@n for *PSL-DSHG*, *SUP-DSHG*, and *EXP-DSHG* for book 2-hop.

system). *SUP-DSHG* performs better in reducing the graph for the book domain, however, the approach fails to provide equivalent or better recommendation performance in comparison to the baseline. On the other hand, our approach, shows significant graph reductions without compromising (and in most cases improving) the performance of the recommendation systems.

5.4 Campaign-specific Hierarchical Subgraph Extraction - An Use Case

Social media platforms such as Twitter, Facebook, and LinkedIn have become a necessary tool in the modern world as a way for people and organizations to share content. According to the statistics reported in 2016, 69% of the total US population visit social networking sites.[6] Twitter has around 330 million active users contributing to 500 million tweets per day.[7] Many applications leverage this vast amount of data being generated by social media users. Twitris [Sheth et al. 2014] is a social data analytics platform to analyze the real world events. To name a few, it has been used to study political events including election [Chen et al. 2012], brands [Purohit et al. 2012], crisis [Bhatt et al. 2014], and drug abuse [Lamy et al. 2016]. Social media text is short and noisy, and it also lacks the context. This is challenging for NLP algorithms and hence has become a very active research area [Derczynski et al. 2015]. Studies [Perera et al. 2015] [Perera et al. 2016] [Brambilla et al. 2017] have shown that background knowledge about the domain can play a key role to understand social media text better.

Creating domain knowledge that can support social media analytics can be very challenging. The main source for creating the domain knowledge is the social media content. This content cover diverse and highly dynamic events happening anywhere at anytime. For example, twitter campaigns covered topics such as election campaigns, cancer studies, political events, and brand advertising.

[6] http://www.pewinternet.org/fact-sheet/social-media/

[7] https://www.omnicoreagency.com/twitter-statistics/

With the vast amount of information being generated by social media with these diverse topics, it is challenging to filter the domain relevant content [Kapanipathi et al. 2011] to create the domain-specific knowledge. Furthermore, [Szekely et al. 2015] discuss the challenges caused by informal language in creating a domain-specific KG for human trafficking domain. This emphasizes the that relying purely on the social media content to create the domain-specific KGs can result in noisy extraction. There have not been many efforts in leveraging large existing KGs to bootstrap the domain-specific knowledge graphs. Even though the existing knowledge graphs cannot completely capture the diverse real-time social media events, the domain-specific subgraphs extracted from large knowledge graphs can be the base domain knowledge which can lead to further improvement. For example, the domain-specific subgraph extracted from a large KG can be used to reduce the noisy extraction from social media content. In fact, prior studies have shown that distant supervision methods [Ren et al. 2017] which use a curated KG can enhance the accuracy of the fact extraction from textual data. Then, the facts extracted from social media content can be used to improve the domain-specific subgraph extracted from a large KG, and this can iterate over several times until we capture complete knowledge.

The techniques discussed in this dissertation on domain-specific subgraph extraction cannot be directly applicable to campaign-specific subgraph extraction. We use domains such as movie and book to show the effectiveness of our approach. These domains have cohesive sets of entity types such as movies, actors, directors, and producers which are tightly coupled together. However, if we consider a domain which is interesting to social media such "US Presidential Election", it covers diverse sets of entity types ranging from politician and policies to funding and business. Also, even though every actor in the world relevant in the movie domain, every politician in the world might not be relevant for the "US Presidential Election". This indicates that dealing with fine-grained domains such as "US Presidential Election" is not straightforward compared to domains such as movie and book. At the same time, existing techniques leverage a large number of known seed domain entities and an entity type to guide the extraction. For example movie-specific subgraph used around 3000 movies and entity type *Film* in DBpedia. In the case of "US Presidential Election", there are only 58 *US Presidential Election* entities in DBpedia, and most of these entities might not be relevant as social studies typically want study temporally important prior elections when analyzing an election campaign. Also, if we stick only to the entity type *US Presidential Election*, we will miss out on an important entity such *US Foreign Policy* as *US Foreign Policy* cannot be reached by simple 3 or 4 hop expansion from *US Presidential Election*.

In this chapter, we discuss how we leverage a description of the domain "US Presidential Election" to extract a domain-specific subgraph using the techniques proposed in Section 5.2

5.4.1 Schema-based Subgraph Extraction for US Presidential Election Campaign

US Presidential Election Schema

"US Presidential Election" domain is a fine-grained domain and hence can not confine to a single entity type in existing KG. We use a domain description about the "US Presidential Election". While this description can be any structured or unstructured source which provides the necessary initial coverage for the domain, we develop a schema/ontology to describe the "US Presidential Election" domain. Fig. 5.12 represents the entity types (classes) and relationships between these entity types captured in the "US Presidential Election" domain. For our study, we select these entity type with the help of domain experts who work on analyzing the social media content including US presidential election campaign in the Twitris platform. The selected entities are: 1) US Presidential Election, 2) State, 3) County, 4) Presidential Candidate, 5) Vice President Running Mate, 6) Debate, 7) Debate Poll, 8) Election Campaign, 9) Candidate Finance, 10) Campaign Funding, 11) Party Nominee, 12) Politician, 13) Supporter, 14) Political Party, 15) Political Convention, and 16) Political Topic.

N-hop Expansion Subgraph

To create the domain-specific subgraph, we first need to create the n-hop expansion subgraph. Prior approaches use domain entities (e.g., movie entities in the movie domain) to create the n-hop expansion subgraph. However, coming up with the domain entities for the "US Presidential Election" domain is not straightforward due to the diversity of the domain. In order to collect a diverse set of domain entities, we extract domain entities from the Wikipedia page about "US Presidential Election" domain.[8] This page contains important entities such as `Political Parties in the US`, `Electoral College`, `President of the United States`, `Vice President of the United State`, and `United States Presidential Primary`. As we conducted this study during the US election 2016, we also extracted domain entities from the Wikipedia page about 2016 US presidential election domain.[9] This led to adding domain entities such as `2016 Democratic National Convention`, `Donald Trump`, `Hilary Clinton`, `Bernie Sanders`, and `Swing State`. Out to all the domain entities in these two Wikipedia pages, we manually pick 75 domain entities as seed entities in the n-hop expansion subgraph. Finally, starting with these domain entities, we traverse the Wikipedia category graph up to 3-hop towards both up and down in the hierarchy. Fig. 5.13 shows a portion of the subgraph we extracted from the domain entity `Donald Trump`.

In extracting the domain-specific subgraph from n-hop expansion graph, we leverage the PSL-based framework proposed in Section 5.2.2. We assess the domain-specificity of a category using the type, lexical and structural semantics. However, in applying the PSL rules 5.5 and 5.6 for type semantics and PSL rules 5.7, 5.8, 5.9, and 5.10 for lexical semantics require to calculate $semtypesim_x(TypeSet, Dom)$ and $lexclussim_{y,x}(TClus_{Clus}, Dom)$ constructs. In the case of movie and book domains, we use the rdfs:label of the class representing the domain in DBpedia as the domin term to calculate these similarities. In this scenario, we calculate $semtypesim_x(TypeSet, Dom)$ and $lexclussim_{y,x}(TClus_{Clus}, Dom)$ with each domain term listed above in the "US Presidential Elec-

[8] https://en.wikipedia.org/wiki/United_States_presidential_election

[9] https://en.wikipedia.org/wiki/United_States_presidential_election,_2016

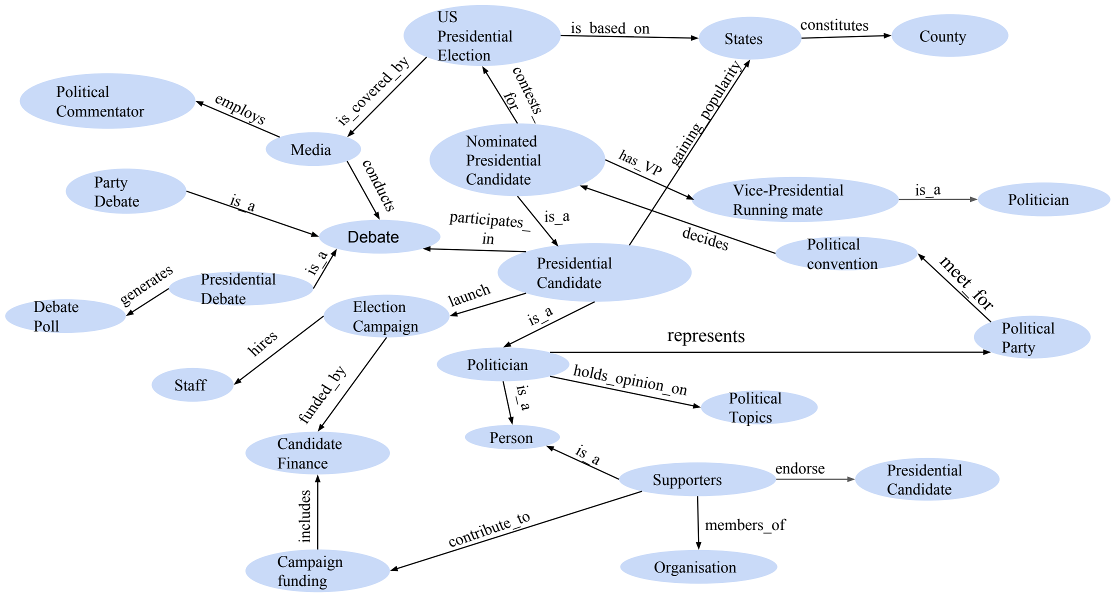


Figure 5.12: *US Presidential Election Domain*

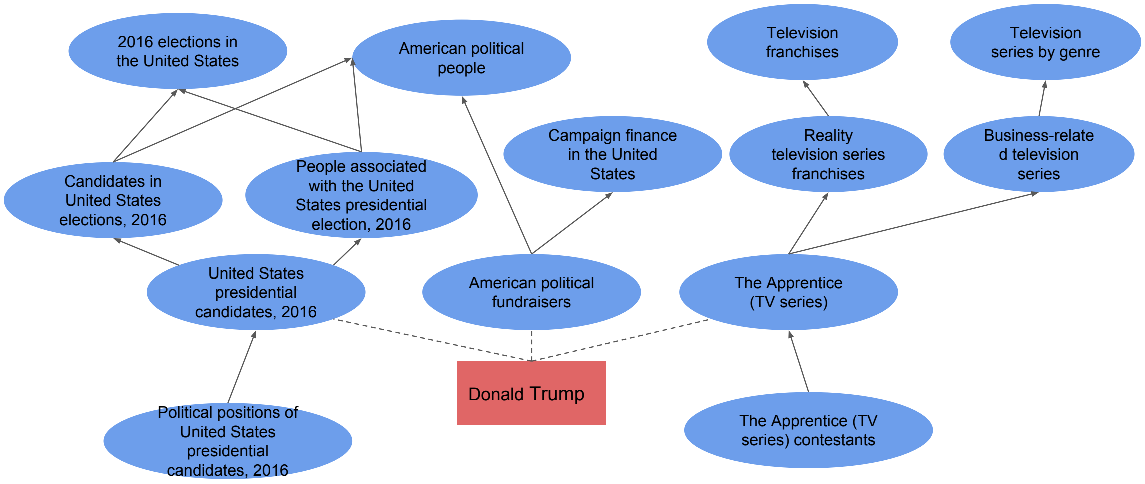

2016 elections in the United States
American political people
Television franchises
Television series by genre
Campaign finance in the United States
Reality television series franchises
Business-related television series
Candidates in United States elections, 2016
People associated with the United States presidential election, 2016
United States presidential candidates, 2016
American political fundraisers
The Apprentice (TV series)
Donald Trump
Political positions of United States presidential candidates, 2016
The Apprentice (TV series) contestants

Figure 5.13: *N-hop expansion subgraph for US Presidential Election Domain*

tion" domain and average it over all the domain terms to calculate the final value.

5.5 Evaluation Results

We ranked the categories in the n-hop expansion subgraph using PSL rules as outlined above. Our n-hop expansion subgraph consisted of 19371 categories. We extract the top-K ranked categories as domain specific and use only those categories to extract the domain-specific subgraph. We select top-5000 ranked categories as relevant using the rankings from the PSL framework.

We follow the evaluation set up in [Mirylenka et al. 2015] which is the state of the art technique for bootstrapping ontologies from Wikipedia category graph. We manually annotate 1000 random sample from the 19371 categories in the n-hop expansion subgraph. Three annotators independently annotate whether a given category is "relevant" or "irrelevant" to the "US Presidential Election" domain. We consider a category as "relevant" if at least two of the three annotators indicated the category is "relevant". This gold standard contains 264 relevant categories and 760 irrelevant categories.

We implement a baseline using the approach outlined in [Mirylenka et al. 2015] and compare its performance with the domain-specific ranking from our approach. We use the accuracy of the prediction and the F1 scores of each class (relevant and irrelevant) as performance metrics. Table 5.9 shows the performance of this task. As shown in Table 5.9, our approach performs well in comparison to the baseline. It increased the accuracy by 13%, F1 relevant by 21%, and F1 irrelevant by 18% with a reduction of 74% of the categories.

5.6 Conclusion

We proposed an approach to extract a domain-specific subgraph from a generic hierarchical knowledge graph. We used Wikipedia category hierarchy as the test bed. Our approach uses type, lexical, and structural semantics of Wikipedia categories as evidences and aggregate them using PSL to

	Accuracy	**F1 relevant**	**F1 irrelevant**
Baseline	0.71	0.532	0.786
PSL	**0.8**	**0.645**	**0.861**

Table 5.9: Performance of category relevancy

determine the domain-specificity of a category. To demonstrate that our approach can work on multiple domains, we used datasets from two diverse domains, i.e., movie and book. We showed that our approach is able to reduce the size of the subgraph by 40% - 50% in terms of number of paths compared to the subgraph created by simple n-hop navigation-based approach. Furthermore, to show the effectiveness on applications using KGs, we evaluated the quality of the domain-specific subgraph extracted with a recommendation use case. Our evaluation showed that the recommendation results improved in majority of the scenarios which demonstrated that harnessing relevant, domain-specific information in KGs can in turn improve the performance of the applications in comparison to using the entire KGs. We also compared our approach with a state-of-the-art domain-specific subgraph extraction approach which uses a supervised learning technique and showed that our approach outperforms accuracy of recommendation results obtained via supervised technique with a significant graph reduction. We use the same state-of-the-art domain-specific subgraph extraction approach to evaluate the subgraph created for ”US Presidential Election domain” and have shown that our approach outperforms their approach.

We believe that this work has major impact in utilizing knowledge graphs for domain-specific applications, specially with the exclusive growth in the creation of knowledge graphs. The reduction in size with no compromise in the performance of applications will lead to fast and quick processing of KGs for corresponding applications.

6

Conclusion and Future Work

6.1 Summary

Structured data on the Web frequently referred to as the Web of Data consists of a large number of knowledge graphs representing diverse domains. Widely used commercial applications such as entity recommendation, search, question answering and knowledge discovery use these knowledge graphs as their knowledge source. Majority of these applications have a particular domain of interest, hence require only the fragment of the Web of data representing that domain (e.g., movie, biomedical, sports). In fact, leveraging the entire Web of data for a domain-specific application is not only computationally expensive, but also the irrelevant portions negatively impact the accuracy of the application. Hence, finding the relevant portion of the Web of data for domain-specific applications has become an important research challenge.

This dissertation addressed the problem of extracting relevant Web of Data for domain-specific applications. We categorize the problem of identifying relevant portion of the Web of Data in to two subproblems; 1) Find the relevant knowledge graphs that contain knowledge about the domain of interest, and 2) extract domain-specific subgraphs from the knowledge graphs that represent multiple domains (e.g., DBpedia, YAGO, Freebase). This chapter summarizes our findings with respect to these subproblems and discusses interesting future research directions.

6.1.1 Automatic domain identification from the Web of Data

In this dissertation, we propose a solution to automatically identify the domain(s) of Web of Data which can be used to automatically identify the relevant knowledge graphs for a given domain-specific application. We have shown that existing crowd-sourced knowledge sources can be leveraged to automatically identify the domains of the knowledge graphs. The main intuition behind our approach is to derive a probability distribution over a well-specified list of domains organized at different abstraction levels for each knowledge graph in a data-driven manner. We leverage knowledge sources in the Web of data as a vocabulary to come up with a well-specified list of domains organized at different abstraction levels. We use Freebase hierarchy (`Freebase domain` and `Freebase type`) as our vocabulary. In a nutshell, our approach aligns the instances of a source knowledge graph with freebase instances. It then uses the type information in Freebase to create a hierarchy as a domain representation of the source knowledge graph with a weight computed for each category in the hierarchy using the instance matching statistics.

We evaluate our approach on 30 LOD datasets using a user study involving twenty users. The user study shows that 50% of the users agreed with 73% of our assignments. We also present a search application with the domains identified using our approach and compare our approach with well-known existing knowledge graph search applications: CKAN, LODStats, and Sindice. Our approach performs better than LODStats and Sindice and is nearly as effective as CKAN which uses manual tagging given to knowledge graphs. We believe our approach performs well due to the use of Freebase which has a category hierarchy with good coverage of different domains in comparison to approaches which utilize traditional indexing and manual tagging based approaches.

Automatic domain identification from the Web of Data is an essential step to improve the searchability of knowledge graphs. As the number of knowledge graphs in the Web of Data is rapidly increasing, easy access to these knowledge graphs will certainly be helpful to improve the utilization of knowledge graphs for domain-specific applications.

6.1.2 Identifying domain-specific subgraph

Structured data on the web produced large-scale knowledge graphs such as DBpedia, YAGO, Freebase, NELL, and Google Knowledge Graph. Relationship connecting two entities play a key role in identifying the domain-specific subgraph from these large knowledge graphs. We categorize the relationships into two main categories; 1) Hierarchical relationships which use inheritance to connect entities, and 2) non-hierarchical relationships which represent diverse named relationships to connect entities. We propose solutions to deal with both types of relationships in this dissertation.

For non-hierarchical relationships, we propose techniques to identify non-hierarchical relationships specific to a given domain, and then use only those relationships to extract the domain-specific subgraph. For example, recognizing that relationships `dbprop:starring`, `dbprop:director`, and `dbprop:award` are specific to the movie domain whereas `dbprop:spouse` and `dbprop:deathdate` are less specific would help to generate a better movie-specific subgraph. The specificity of a relationship to its domain is calculated by measuring the strength of association of the relationship to its seed domain entities using statistical techniques. Our approach identifies two characteristics, type and path, in the KG to measure the strength of association of relationships to domain entities and propose two novel measures based on these characteristics. As hierarchical relationships have uniform semantics (inheritance or ISA), we measure the specificity of entities to derive the domain-specific subgraph. We measure the domain specificity of an entity in a hierarchy using the type, lexical and structural semantics of the entities. To systematically combine different semantics towards domain specificity of the category, we use the probabilistic soft logic (PSL) framework which is a statistical framework to collect different forms of evidence.

To demonstrate the effectiveness of the proposed methodology, we evaluate the domain-specific subgraphs created with our approach in the context of a recommendation application. Evaluation results indicate that our approach was able to reduce the subgraph by 80% for non-hierarchical relationships and by 50% for hierarchical relationships without compromising on the accuracy of the recommendation algorithm. We perform these experiments on two domains: movie and book. In addition to the domains such as movie and book, we also evaluate our work in the context of a more fine-grained domain such as US presidential election. We have shown that the subgraph extracted by our approach is better than the state-of-the-art domain-specific subgraph extraction techniques which use supervised learning.

As we have outlined above, Linked Data based recommendation algorithms can improve both its performance and accuracy by extracting domain-specific subgraphs. Linked data based recommendation algorithms commonly use graph-based algorithms such as page rank to rank items to be recommended [Nguyen et al. 2015] [Musto et al. 2017]. These algorithms typically require a number of iterations to converge to its final ranking and hence cause performance issues. The presented subgraph extraction techniques can be a great way to deal with these kinds of performance issues for any graph-based algorithm. Even though we demonstrate the effectiveness of domain-specific extraction techniques with a recommendation use case, this can be useful for applications such as named entity disambiguation, question answering, and graph-based querying. To disambiguate an entity, existing disambiguation techniques first select candidate entities for a given input word. Domain-specific subgraphs can help to reduce the number of candidate entities by selecting only domain relevant entities. This can contribute to improve both accuracy and performance. Graph-based querying (e.g. SPARQL) can use domain-specific subgraphs to improve its response time.

To conclude, we believe that both automatic domain identification and domain-specific subgraph extraction contribute in utilizing knowledge graphs for domain-specific applications by identifying the relevant piece of knowledge.

6.2 Future Work

6.2.1 Automatic domain identification from Web of Data

We believe the proposed technique to identify domains has a potential to be a basis for creating a search catalog for a large number of knowledge graphs out there, and hence assisting in finding relevant knowledge graph for a given domain-specific application. Our approach uses the richness of Freebase as the domain vocabulary for our work. In future, it will be important to discuss any alternatives for Freebase such as KBpedia [1] given that Freebase is no longer being actively maintained. The proposed techniques are generic enough to be adaptable to any other knowledge source. At the same time, it is impractical to assume that a single vocabulary has the capability to address diverse domains represented by current knowledge graphs. Hence, another future direction worth investigating would be to come up with mapping from one vocabulary to other. For example, if we can map higher level concept in UMLS to generic medical concept in Freebase, this technique will be able to provide a fine-grained description for medical domain.

In order to make sure of the sustainability of these kinds of data profiling techniques, we need to create a community effort for indexing the knowledge graphs in this way by asking data providers to provide these descriptions along with the knowledge graphs. This will make it easier to integrate these domain descriptions with other data profiling techniques such as VOID and LODStats. Integrated domain descriptions can be used by various search and indexing services to improve the searchability of the knowledge graphs.

6.2.2 Identifying domain-specific subgraph

In this dissertation, we propose domain-specific subgraph extraction techniques which go beyond from simple n-hop expansion techniques. However, these techniques do not address the extraction of temporally relevant subgraphs. Incorporating spatial-temporal relevance to the domain-specific

[1] http://kbpedia.com/

subgraph extraction will be an important criteria and future research direction in this area (such as social campaigns during US presidential election). As of now, our techniques facilitate domains described as an entity type or a set of entity types. Some applications might benefit from extracting subgraphs when the domain description is not only limited to entity types but also to relationships types. At the same time, there can be certain relationships which have different interpretations depending on the context. For example, even though `spouse` relationship is generally not considered as relevant to the movie domain if the `spouse` relationship is specified between two characters in a movie that will be a relevant relationship for the movie domain. Tackling these different interpretations of the same relationship will be a challenging future research direction.

With respect to the technical advancements to the proposed approaches, one limitation in the current approaches is that of selecting top-k entities and relationships after the ranking is done via experimental analysis. An approach to determine the right k value with some measures of reduction will be an important contribution to the proposed approaches. Furthermore, while in non-hierarchical relationships have only focused on relationships and hierarchical relationships on entities, investigating that combining both entities and relationships for domain-specific ranking can be an interesting discussion. Certainly, this will require more computation power as the number of entities are much higher than the number of relationships. In this case, an analysis which discusses the trade-offs between the accuracy improvement versus scalability of the approach can be highly valuable. It will also be interesting to study different applications which can be benefited from the subgraph extraction in addition to the recommendation applications.

References

Adomavicius, G. and Tuzhilin, A. 2015. Context-aware recommender systems. In *Recommender systems handbook*. Springer, 191–226.

Aleman-Meza, B., Halaschek-Weiner, C., Arpinar, I. B., Ramakrishnan, C., and Sheth, A. P. 2005. Ranking complex relationships on the semantic web. *IEEE Internet computing 9,* 3, 37–44.

Alexander, K. and Hausenblas, M. 2009. Describing linked datasets-on the design and usage of void, thevocabulary of interlinked datasets. In *In Linked Data on the Web Workshop (LDOW 09), in conjunction with 18th International World Wide Web Conference (WWW 09*. Citeseer.

Anyanwu, K., Maduko, A., and Sheth, A. 2005. Semrank: ranking complex relationship search results on the semantic web. In *Proceedings of the 14th international conference on World Wide Web*. ACM, 117–127.

Arumugam, M., Sheth, A., and Arpinar, I. 2002. Towards peer-to-peer semantic web: A distributed environment for sharing semantic knowledge on the web. *WWW2002 Workshop on Real World RDF and Semantic Web Applications*.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*. Springer, 722–735.

Auer, S., Demter, J., Martin, M., and Lehmann, J. 2012. Lodstats–an extensible framework for high-performance dataset analytics. In *International Conference on Knowledge Engineering and Knowledge Management*. Springer, 353–362.

Bach, S. H., Broecheler, M., Huang, B., and Getoor, L. 2015. Hinge-loss markov random fields and probabilistic soft logic. *arXiv:1505.04406 [cs.LG]*.

Bach, S. H., Broecheler, M., Huang, B., and Getoor, L. 2017. Hinge-loss markov random fields and probabilistic soft logic. *Journal of Machine Learning Research 18,* 109, 1–67.

Belleau, F., Nolin, M.-A., Tourigny, N., Rigault, P., and Morissette, J. 2008. Bio2rdf: towards a mashup to build bioinformatics knowledge systems. *Journal of biomedical informatics 41,* 5, 706–716.

Berners-Lee, T., Fischetti, M., and Foreword By-Dertouzos, M. L. 2000. *Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor*. HarperInformation.

Bhatt, S. P., Purohit, H., Hampton, A., Shalin, V., Sheth, A., and Flach, J. 2014. Assisting coordination during crisis: a domain ontology based approach to infer resource needs from tweets. In *Proceedings of the 2014 ACM conference on Web science*. ACM, 297–298.

Bizer, C., Heath, T., and Berners-Lee, T. 2009. Linked data-the story so far. *Semantic Services, Interoperability and Web Applications: Emerging Concepts*, 205–227.

Blei, D. M., Ng, A. Y., and Jordan, M. I. 2003. Latent dirichlet allocation. *Journal of machine Learning research 3,* Jan, 993–1022.

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*. ACM, 1247–1250.

Bouma, G. 2009. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, 31–40.

Brambilla, M., Ceri, S., Della Valle, E., Volonterio, R., and Acero Salazar, F. X. 2017. Extracting emerging knowledge from social media. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 795–804.

Burke, R. 2002. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction 12,* 4, 331–370.

Cai, L. and Hofmann, T. 2004. Hierarchical document categorization with support vector machines. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*. ACM, 78–87.

Cameron, D., Kavuluru, R., Rindflesch, T. C., Sheth, A. P., Thirunarayan, K., and Bodenreider, O. 2015. Context-driven automatic subgraph creation for literature-based discovery. *Journal of biomedical informatics 54*, 141–157.

Cheekula, S. K., Kapanipathi, P., Doran, D., Jain, P., and Sheth, A. P. 2015. Entity recommendations using hierarchical knowledge bases.

Chen, L., Wang, W., and Sheth, A. P. 2012. Are twitter users equal in predicting elections? a study of user groups in predicting 2012 us republican presidential primaries. In *International Conference on Social Informatics*. Springer, 379–392.

Church, K. W. and Hanks, P. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics 16,* 1, 22–29.

d'Aquin, M. and Motta, E. 2011. Watson, more than a semantic web search engine. *Semantic Web 2,* 1, 55–63.

de Oliveira, H. R., Tavares, A. T., and Lóscio, B. F. 2012. Feedback-based data set recommendation for building linked data applications. In *Proceedings of the 8th International Conference on Semantic Systems*. ACM, 49–55.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science 41,* 6, 391.

Derczynski, L., Maynard, D., Rizzo, G., van Erp, M., Gorrell, G., Troncy, R., Petrak, J., and Bontcheva, K. 2015. Analysis of named entity recognition and linking for tweets. *Information Processing & Management 51,* 2, 32–49.

Devine, D. J. and Kozlowski, S. W. 1995. Domain-specific knowledge and task characteristics in decision making. *Organizational Behavior and Human Decision Processes 64,* 3, 294–306.

Di Noia, T., Mirizzi, R., Ostuni, V. C., Romito, D., and Zanker, M. 2012. Linked open data to support content-based recommender systems. In *Proceedings of the 8th International Conference on Semantic Systems.* ACM, 1–8.

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 *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 601–610.

Erxleben, F., Günther, M., Krötzsch, M., Mendez, J., and Vrandečić, D. 2014. Introducing wikidata to the linked data web. In *International Semantic Web Conference.* Springer, 50–65.

Feigenbaum, E. A., Buchanan, B. G., and Lederberg, J. 1970. On generality and problem solving: A case study using the dendral program.

Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A. A., Lally, A., Murdock, J. W., Nyberg, E., Prager, J., et al. 2010. Building watson: An overview of the deepqa project. *AI magazine 31,* 3, 59–79.

Figueroa, C., Vagliano, I., Rocha, O. R., and Morisio, M. 2015. A systematic literature review of linked data-based recommender systems. *Concurrency and Computation: Practice and Experience 27,* 17, 4659–4684.

Finin, T., Ding, L., Pan, R., Joshi, A., Kolari, P., Java, A., and Peng, Y. 2005. Swoogle: Searching for knowledge on the semantic web. In *Proceedings of the National Conference on Artificial Intelligence*. Vol. 20. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 1682.

Flati, T., Vannella, D., Pasini, T., and Navigli, R. 2014. Two is bigger (and better) than one: the wikipedia bitaxonomy project. In *ACL (1)*.

Frosterus, M., Hyvönen, E., and Laitio, J. 2011. Datafinlanda semantic portal for open and linked datasets. In *Extended Semantic Web Conference*. Springer, 243–254.

Gabrilovich, E. and Markovitch, S. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJcAI*. Vol. 7. 1606–1611.

Görlitz, O. and Staab, S. 2011. Splendid: Sparql endpoint federation exploiting void descriptions. In *Proceedings of the Second International Conference on Consuming Linked Data-Volume 782*. CEUR-WS. org, 13–24.

Gruber, T. R. 1995. Toward principles for the design of ontologies used for knowledge sharing? *International journal of human-computer studies 43,* 5, 907–928.

Guha, R. V. and Lenat, D. B. 1993. Cyc: A midterm report. In *Readings in knowledge acquisition and learning*. Morgan Kaufmann Publishers Inc., 839–866.

Han, L., Kashyap, A., Finin, T., Mayfield, J., and Weese, J. 2013. Umbc ebiquity-core: Semantic textual similarity systems. In *Proc. of the Second Joint Conference on Lexical and Computational Semantics*. 44–52.

Hao, P.-Y., Chiang, J.-H., and Tu, Y.-K. 2007. Hierarchically svm classification based on support vector clustering method and its application to document categorization. *Expert Systems with applications 33,* 3, 627–635.

Harper, F. M. and Konstan, J. A. 2016. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS) 5,* 4, 19.

Harth, A., Hose, K., Karnstedt, M., Polleres, A., Sattler, K.-U., and Umbrich, J. 2010. Data summaries for on-demand queries over linked data. In *Proceedings of the 19th international conference on World wide web*. ACM, 411–420.

Hartig, O., Bizer, C., and Freytag, J.-C. 2009. Executing sparql queries over the web of linked data. In *International Semantic Web Conference*. Springer, 293–309.

He, J. and Chu, W. W. 2010. A social network-based recommender system (snrs). In *Data mining for social network data*. Springer, 47–74.

Herlocker, J. L., Konstan, J. A., and Riedl, J. 2000. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*. ACM, 241–250.

Hoffart, J., Suchanek, F. M., Berberich, K., and Weikum, G. 2013. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence 194*, 28–61.

Hofmann, T. 1999. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 50–57.

Hulpuş, I., Prangnawarat, N., and Hayes, C. 2015. Path-based semantic relatedness on linked data and its use to word and entity disambiguation. In *International Semantic Web Conference*. Springer, 442–457.

Jadhav, A. S., Purohit, H., Kapanipathi, P., Anantharam, P., Ranabahu, A. H., Nguyen, V., Mendes, P. N., Smith, A. G., Cooney, M., and Sheth, A. P. 2010. Twitris 2.0: Semantically empowered system for understanding perceptions from social data. In *Semantic web application challenge at ISWC, Shanghai*. 5–11.

Jain, P., Hitzler, P., Sheth, A. P., Verma, K., and Yeh, P. Z. 2010. Ontology Alignment for Linked Open Data. In *Proceedings of the 9th International Semantic Web Conference, ISWC 2010, Shanghai, China, November 7-11, 2010*. Vol. 6496. Springer-Verlag, 402–417.

Jiang, P., Hou, H., Chen, L., Chen, S., Yao, C., Li, C., and Wang, M. 2013. Wiki3c: exploiting wikipedia for context-aware concept categorization. In *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM, 345–354.

Kapanipathi, P., Jain, P., Venkataramani, C., and Sheth, A. 2014. User interests identification on twitter using a hierarchical knowledge base. In *European Semantic Web Conference*. Springer, 99–113.

Kapanipathi, P., Orlandi, F., Sheth, A. P., and Passant, A. 2011. Personalized filtering of the twitter stream.

Kenter, T., Borisov, A., and de Rijke, M. 2016. Siamese cbow: Optimizing word embeddings for sentence representations. *arXiv preprint arXiv:1606.04640*.

Kobilarov, G., Scott, T., Raimond, Y., Oliver, S., Sizemore, C., Smethurst, M., Bizer, C., and Lee, R. 2009. Media meets semantic web–how the bbc uses dbpedia and linked data to make connections. In *European Semantic Web Conference*. Springer, 723–737.

Konrath, M., Gottron, T., Staab, S., and Scherp, A. 2012. Schemexefficient construction of a data catalogue by stream-based indexing of linked data. *Web Semantics: Science, Services and Agents on the World Wide Web 16*, 52–58.

KOUKI, P., FAKHRAEI, S., FOULDS, J., EIRINAKI, M., AND GETOOR, L. 2015. Hyper: A flexible and extensible probabilistic framework for hybrid recommender systems. In *Proc. of the 9th ACM Conference on Recommender Systems*.

LALITHSENA, S., HITZLER, P., SHETH, A., AND JAIN, P. 2013. Automatic domain identification for linked open data. In *Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)-Volume 01*. IEEE Computer Society, 205–212.

LALITHSENA, S., KAPANIPATHI, P., AND SHETH, A. 2016. Harnessing relationships for domain-specific subgraph extraction: A recommendation use case. In *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 706–715.

LALITHSENA, S., PERERA, S., KAPANIPATHI, P., AND SHETH, A. 2017. Domain-specific hierarchical subgraph extraction: A recommendation use case. In *Big Data (Big Data), 2017 IEEE International Conference on*. IEEE, 666–675.

LAMY, F. R., DANIULAITYTE, R., SHETH, A., NAHHAS, R. W., MARTINS, S. S., BOYER, E. W., AND CARLSON, R. G. 2016. those edibles hit hard: Exploration of twitter data on cannabis edibles in the us. *Drug & Alcohol Dependence 164*, 64–70.

LASSILA, O. AND SWICK, R. R. 1999. Resource description framework (rdf) model and syntax specification.

LEAL, J. P. 2013. Using proximity to compute semantic relatedness in rdf graphs. *Computer Science and Information Systems 10,* 4, 1727–1746.

LEHMANN, J., ISELE, R., JAKOB, M., JENTZSCH, A., KONTOKOSTAS, D., MENDES, P. N., HELLMANN, S., MORSEY, M., VAN KLEEF, P., AUER, S., ET AL. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web 6,* 2, 167–195.

Lenat, D. B., Prakash, M., and Shepherd, M. 1985. Cyc: Using common sense knowledge to overcome brittleness and knowledge acquisition bottlenecks. *AI magazine 6,* 4, 65.

Liu, X., Song, Y., Liu, S., and Wang, H. 2012. Automatic taxonomy construction from keywords. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1433–1441.

Lops, P., De Gemmis, M., and Semeraro, G. 2011. Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*. Springer, 73–105.

Lu, J., Wu, D., Mao, M., Wang, W., and Zhang, G. 2015. Recommender system application developments: a survey. *Decision Support Systems 74*, 12–32.

MacQueen, J. et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. Oakland, CA, USA., 281–297.

McBride, B. 2004. The resource description framework (rdf) and its vocabulary description language rdfs. In *Handbook on ontologies*. Springer, 51–65.

McGuinness, D. L., Van Harmelen, F., et al. 2004. Owl web ontology language overview. *W3C recommendation 10,* 10, 2004.

Mena, E., Illarramendi, A., Kashyap, V., and Sheth, A. P. 2000. Observer: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. *Distributed and parallel Databases 8,* 2, 223–271.

Mendes, P. N., Jakob, M., García-Silva, A., and Bizer, C. 2011. Dbpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th international conference on semantic systems*. ACM, 1–8.

Meusel, R., Bizer, C., and Paulheim, H. 2015. A web-scale study of the adoption and evolution of the schema. org vocabulary over time. In *Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics*. ACM, 15.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Miller, G. A. 1995. Wordnet: a lexical database for english. *Communications of the ACM 38,* 11, 39–41.

Mirylenka, D., Passerini, A., Serafini, L., et al. 2015. Bootstrapping domain ontologies from wikipedia: A uniform approach. In *IJCAI*. 1464–1470.

Mitchell, T. M., Cohen, W. W., Hruschka Jr, E. R., Talukdar, P. P., Betteridge, J., Carlson, A., Mishra, B. D., Gardner, M., Kisiel, B., Krishnamurthy, J., et al. 2015. Never ending learning. In *AAAI*. 2302–2310.

Musto, C., Basile, P., Lops, P., De Gemmis, M., and Semeraro, G. 2014. Linked open data-enabled strategies for top-n recommendations. In *CBRecSys@ RecSys*. 49–56.

Musto, C., Semeraro, G., de Gemmis, M., and Lops, P. 2017. Tuning personalized pagerank for semantics-aware recommendations based on linked open data. In *European Semantic Web Conference*. Springer, 169–183.

Newell, A., Shaw, J. C., and Simon, H. A. 1959. Report on a general problem solving program. In *IFIP congress*. Vol. 256. 64.

Nguyen, P., Tomeo, P., Di Noia, T., and Di Sciascio, E. 2015. An evaluation of simrank and personalized pagerank to build a recommender system for the web of data. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 1477–1482.

Nikolov, A., dAquin, M., and Motta, E. 2011. What should i link to? identifying relevant sources and classes for data linking. In *Joint International Semantic Technology Conference*. Springer, 284–299.

Nokes, T., Schunn, C., and Chi, M. 2010. Problem solving and human expertise. In *Elsevier Ltd.*

Ostuni, V. C., Di Noia, T., Di Sciascio, E., and Mirizzi, R. 2013. Top-n recommendations from implicit feedback leveraging linked open data. In *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 85–92.

Passant, A. 2010. dbrecmusic recommendations using dbpedia. In *International Semantic Web Conference*. Springer, 209–224.

Perera, S., Mendes, P., Sheth, A., Thirunarayan, K., Alex, A., Heid, C., and Mott, G. 2015. Implicit entity recognition in clinical documents. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*. 228–238.

Perera, S., Mendes, P. N., Alex, A., Sheth, A. P., and Thirunarayan, K. 2016. Implicit entity linking in tweets. In *International Semantic Web Conference*. Springer, 118–132.

Piao, G. and Breslin, J. G. 2016. Measuring semantic distance for linked open data-enabled recommender systems. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*. ACM, 315–320.

Pirrò, G. 2015. Explaining and suggesting relatedness in knowledge graphs. In *International Semantic Web Conference*. Springer, 622–639.

Ponzetto, S. P. and Strube, M. 2011. Taxonomy induction based on a collaboratively built knowledge repository. *Artificial Intelligence*.

Prud, E., Seaborne, A., et al. 2006. Sparql query language for rdf.

Pujara, J., Miao, H., Getoor, L., and Cohen, W. 2013. Knowledge graph identification. In *Proceedings of the 12th International Semantic Web Conference*. ISWC '13. Springer-Verlag New York, Inc., New York, NY, USA, 542–557.

Purohit, H., Ajmera, J., Joshi, S., Verma, A., and Sheth, A. P. 2012. Finding influential authors in brand-page communities. In *ICWSM*.

Ren, X., Wu, Z., He, W., Qu, M., Voss, C. R., Ji, H., Abdelzaher, T. F., and Han, J. 2017. Cotype: Joint extraction of typed entities and relations with knowledge bases. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1015–1024.

Resnik, P. 1995. Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007*.

Russell, S., Norvig, P., and Intelligence, A. 1995. A modern approach. *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs 25,* 27, 79–80.

Schmitz, M., Bart, R., Soderland, S., Etzioni, O., et al. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, 523–534.

Schuhmacher, M. and Ponzetto, S. P. 2014. Knowledge-based graph document modeling. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 543–552.

Schwarte, A., Haase, P., Hose, K., Schenkel, R., and Schmidt, M. 2011. Fedx: Optimization techniques for federated query processing on linked data. In *International Semantic Web Conference*. Springer, 601–616.

SECO, N., VEALE, T., AND HAYES, J. 2004. An intrinsic information content metric for semantic similarity in wordnet. In *Proceedings of the 16th European conference on artificial intelligence*. IOS Press, 1089–1090.

SHAH, I. AND SHETH, A. 1999. Infoharness: managing distributed, heterogeneous information. *IEEE Internet Computing 3,* 6, 18–28.

SHANI, G. AND GUNAWARDANA, A. 2011. Evaluating recommendation systems. In *Recommender systems handbook*. Springer, 257–297.

SHETH, A., ARPINAR, I. B., AND KASHYAP, V. 2004. Relationships at the heart of semantic web: Modeling, discovering, and exploiting complex semantic relationships. In *Enhancing the Power of the Internet*. Springer, 63–94.

SHETH, A., AVANT, D., AND BERTRAM, C. 2001. System and method for creating a semantic web and its applications in browsing, searching, profiling, personalization and advertising. US Patent 6,311,194.

SHETH, A., BERTRAM, C., AVANT, D., HAMMOND, B., KOCHUT, K., AND WARKE, Y. 2002. Managing semantic content for the web. *IEEE Internet Computing 6,* 4, 80–87.

SHETH, A., JADHAV, A., KAPANIPATHI, P., LU, C., PUROHIT, H., SMITH, G. A., AND WANG, W. 2014. Twitris: A system for collective social intelligence. In *Encyclopedia of social network analysis and mining*. Springer, 2240–2253.

SHORTLIFFE, E. H. 1974. A rule-based computer program for advising physicians regarding antimicrobial therapy selection. In *Proceedings of the 1974 annual ACM conference-Volume 2*. ACM, 739–739.

SZEKELY, P., KNOBLOCK, C. A., SLEPICKA, J., PHILPOT, A., SINGH, A., YIN, C., KAPOOR, D., NATARAJAN, P., MARCU, D., KNIGHT, K., ET AL. 2015. Building and using a knowledge graph to combat human trafficking. In *International Semantic Web Conference*. Springer, 205–221.

Thomas, C., Mehra, P., Brooks, R., and Sheth, A. 2008. Growing fields of interest-using an expand and reduce strategy for domain model extraction. In *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08. IEEE/WIC/ACM International Conference on.* Vol. 1. IEEE, 496–502.

Tonon, A., Catasta, M., Demartini, G., Cudré-Mauroux, P., and Aberer, K. 2013. Trank: Ranking entity types using the web of data. In *International Semantic Web Conference*. Springer, 640–656.

Tran, T., Zhang, L., and Studer, R. 2010. Summary models for routing keywords to linked data sources. In *International Semantic Web Conference*. Springer, 781–797.

Tummarello, G., Cyganiak, R., Catasta, M., Danielczyk, S., Delbru, R., and Decker, S. 2010. Sig. ma: Live views on the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web 8,* 4, 355–364.

Tummarello, G., Delbru, R., and Oren, E. 2007. Sindice. com: Weaving the open linked data. In *The Semantic Web*. Springer, 552–565.

Usbeck, R., Ngomo, A.-C. N., Röder, M., Gerber, D., Coelho, S. A., Auer, S., and Both, A. 2014. Agdistis-graph-based disambiguation of named entities using linked data. In *International Semantic Web Conference*. Springer, 457–471.

Welty, C., Murdock, J., Kalyanpur, A., and Fan, J. 2012. A comparison of hard filters and soft evidence for answer typing in watson. *The Semantic Web–ISWC 2012*, 243–256.

Zhang, Z., Lin, H., Liu, K., Wu, D., Zhang, G., and Lu, J. 2013. A hybrid fuzzy-based personalized recommender system for telecom products/services. *Information Sciences 235*, 117–129.

Zwicklbauer, S., Seifert, C., and Granitzer, M. 2015. From general to specialized domain: Analyzing three crucial problems of biomedical entity disambiguation. In *International Conference on Database and Expert Systems Applications*. Springer, 76–93.