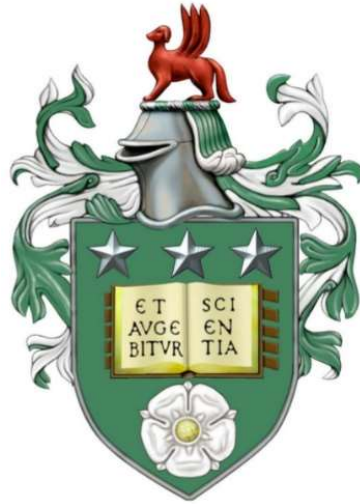


Intelligent Support for Exploration of Data Graphs

Marwan Ahmad Talal Al-Tawil

Submitted in accordance with the requirements

for the degree of Doctor of Philosophy



The University of Leeds

School of Computing

October, 2017

Dedication

To my family

Declaration

The candidate confirms that the work submitted is his/her own, except where work which has formed part of jointly-authored publications has been included. The contribution of the candidate and the other authors to this work has been explicitly indicated below. The candidate confirms that appropriate credit has been given within the thesis where reference has been made to the work of others.

Some of the work in this thesis has been published prior to thesis submission.

The exploratory user study in Chapter 3 has been presented in:

Al-Tawil, M., Thakker, D., Dimitrova, V.: Nudging to Expand User's Domain Knowledge while Exploring Linked Data. In: Proceedings of Intelligent Exploration of Semantic Data @ International Semantic Web Conference (2014).

The KA_{DG} algorithms in Chapter 4 have been presented in:

Al-Tawil, M., Dimitrova, V., Thakker, D., Bennett, B.: Identifying Knowledge Anchors in a Data Graph. In: Proceedings of the 27th ACM Conference on Hypertext and Social Media (2016).

Part of research description in Chapters 1 and 3 has been described in:

Al-Tawil, M.: Intelligent Nudging to Support Interactive Exploration of Big Data Graphs, in: HT 2016 – Doctoral Consortium. 27th ACM Conf. In Hypertext and Social. Media (2016).

Part of the work for identifying KA_{DG} in L4All in Chapter 4 has been presented in:

Poulovassilis, A., Al-Tawil, M., Frosini, R., Dimartino, M., Dimitrova, V.: Combining Flexible Queries and Knowledge Anchors to facilitate the exploration of Knowledge Graphs. In: Proceedings of Intelligent Exploration of Semantic Data @ International Semantic Web Conference (2016).

The evaluation of KA_{DG} algorithms in Chapter 5 has been presented in:

Al-Tawil, M., Dimitrova, V., Thakker, D., Alexandra Poulovassilis.: Evaluating Knowledge Anchors in Data Graphs against Basic Level Objects. In: Proceedings of International conference on Web Engineering (2017).

Part of the future research in Chapter 8 has been presented in:

Al-Tawil, M., Dimitrova, V., Thakker, D.: Using Basic Level Concepts in a Linked Data Graph to Detect User's Domain Familiarity. In: Proceedings of User Modelling Adaptation and Personalisation conference (2015).

This copy has been supplied on the understanding that it is copyright material and that no quotation from the thesis may be published without proper acknowledgement.

The right of Marwan Al-Tawil to be identified as Author of this work has been asserted by him in accordance with the Copyright, Designs and Patents Act 1988.

© 2017 The University of Leeds and Marwan Ahmad Talal Al-Tawil

Acknowledgements

First and foremost, I praise my God Allah who gave me the health, strength and courage to complete this work.

My heartfelt appreciation and deepest gratitude goes to my supervisor Dr. Vania Dimitrova as without her constant support and encouragement this research would not have been accomplished. Her guidance through the years of my study gave me the confidence to pursue my studies and research.

I would like also to thank my Co-supervisors Dr. Dhavalkumar Thakker and Dr. Brandon Bennett for their dedicated efforts and support during my study.

I have no words to describe my gratitude for the support I received from my family. Your encouragement and compassion are precious in all aspects of my life. Special thanks to my parents who made me the way I am. Also, I would like to express my gratitude to my wife and my daughter for their love and support during my journey.

Acknowledgement goes to the University of Jordan who granted my scholarship to complete this research.

Abstract

This research investigates how to support a user's exploration through data graphs generated from semantic databases in a way leading to expanding the user's domain knowledge. To be effective, approaches to facilitate exploration of data graphs should take into account the utility from a user's point of view. Our work focuses on knowledge utility – how useful exploration paths through a data graph are for expanding the user's knowledge. The main goal of this research is to design an intelligent support mechanism to direct the user to 'good' exploration paths through big data graphs for knowledge expansion. We propose a new exploration support mechanism underpinned by the subsumption theory for meaningful learning, which postulates that new knowledge is grasped by starting from familiar concepts in the graph which serve as knowledge anchors from where links to new knowledge are made. A core algorithmic component for adapting the subsumption theory for generating exploration paths is the automatic identification of Knowledge Anchors in a Data Graph (KA_{DG}). Several metrics for identifying KA_{DG} and the corresponding algorithms for implementation have been developed and evaluated against human cognitive structures. A subsumption algorithm which utilises KA_{DG} for generating exploration paths for knowledge expansion is presented and evaluated in the context of a semantic data browser in a musical instrument domain. The resultant exploration paths are evaluated in a controlled user study to examine whether they increase the users' knowledge as compared to free exploration. The findings show that exploration paths using knowledge anchors and subsumption lead to significantly higher increase in the users' conceptual knowledge. The approach can be adopted in applications providing data graph exploration to facilitate learning and sensemaking of layman users who are not fully familiar with the domain presented in the data graph.

Table of Contents

Dedication	ii
Acknowledgements	v
Abstract	vi
Table of Contents	vii
Abbreviations	xi
List of Figures	xii
List of Tables	xiv
Chapter 1 Introduction.....	1
Chapter 2 Related Work	8
2.1 Introduction.....	8
2.2 Background.....	9
2.2.1 The Web Graph.....	9
2.2.2 Linked Data.....	11
2.2.3 Ontology.....	13
2.2.4 SPARQL Query Language.....	15
2.2.5 Graph Databases and Triple Stores	16
2.2.6 Semantic Web Applications.....	18
2.2.6.1 Semantic Data Browsers	19
2.2.6.2 Semantic Data Search Engines.....	21
2.3 Exploration of Data Graphs	23
2.3.1 Visualisation-based Approaches	23
2.3.2 Text-based Approaches.....	25
2.4 Identifying Key Entities in Data Graphs.....	27
2.4.1 Ontology Summarisation Approaches	28
2.4.2 Formal Concept Analysis Based Approaches	29
2.4.3 Approaches that adopt Basic Level Objects.....	30
2.5 Generating Paths in Data Graphs.....	32
2.6 Data Exploration Evaluation Approaches.....	35
2.7 Theories	36
2.7.1 Bloom’s Taxonomy for Approximating Knowledge Utility.....	36
2.7.2 Underpinning Theoretical Model for Data Graph Exploration	39
2.7.2.1 Schema Theory.....	39
2.7.2.2 Subsumption Theory for Meaningful Learning.....	40

2.7.2.3 Underpinning Theoretical Model Identified.....	41
2.7.3 Basic Level Objects in Cognitive Science	42
2.8 Summary	45
Chapter 3 Research Methodology and Scoping.....	46
3.1 Introduction.....	46
3.2 Research Methodology	46
3.3 Formal Definitions	49
3.4 Application Context.....	51
3.4.1 MusicPinta	52
3.4.2 L4All	55
3.5 Exploratory User Study for Research Scoping	57
3.5.1 Exploration Strategies for Knowledge Expansion	58
3.5.2 Study Design.....	60
3.5.3 Results.....	63
3.5.4 Observations.....	68
3.6 Summary.....	69
Chapter 4 Algorithms for Knowledge Anchors.....	70
4.1 Introduction.....	70
4.2 Algorithms for Identifying Knowledge Anchors in a Data Graph.....	71
4.2.1. Distinctiveness Metrics	72
4.2.2. Homogeneity Metrics.....	76
4.3 KA_{DG} Algorithms Implementation.....	77
4.3.1 Implementation in MusicPinta	78
4.3.2 Implementation in L4All.....	79
4.4 Discussion.....	79
4.5 Summary.....	81
Chapter 5 Knowledge Anchors Algorithms Evaluation	82
5.1 Introduction.....	82
5.2 Algorithm for Identifying Human BLO over a Data Graph	83
5.3 Evaluating KA_{DG} in MusicPinta	85
5.3.1 Obtaining Human BLO_{DG} in MusicPinta	85
5.3.2 Evaluating KA_{DG} against Human BLO_{DG}	89
5.4 Evaluating KA_{DG} in L4All	92
5.4.1 Obtaining Human BLO_{DG} in L4All.....	92
5.4.2 Evaluating KA_{DG} against Human BLO_{DG}	95

5.5 Discussion	100
5.5.1 Algorithm for Identifying Human BLO_{DG}	100
5.5.2 Performance of KA_{DG} Metrics	101
5.6 Summary	102
Chapter 6 Exploration Strategies Based on Subsumption	104
6.1 Introduction.....	104
6.2 Algorithm for Generating Exploration Paths in a Data Graph.....	105
6.2.1 Finding the Closest KA_{DG}	105
6.2.2 Subsumption Using the Closest KA_{DG}	108
6.3 Implementation Illustration.....	111
6.3.1 Identify Closest Knowledge Anchors	112
6.3.2 Use Closest Knowledge Anchors to Generate Exploration Paths	113
6.3.2.1 Examples of Exploration Paths in MusicPinta	113
6.3.2.2 Examples of Exploration Paths in L4All.....	118
6.4 Further Improvements.....	122
6.4.1 Algorithm for Identifying the Closest Knowledge Anchor.....	122
6.4.2 Algorithm for Generating Exploration Paths Using the Closest Knowledge Anchor.....	123
6.5 Summary.....	123
Chapter 7 Evaluation of Exploration Paths.....	124
7.1 Introduction.....	124
7.2 Exploration Task Design.....	125
7.3 Experimental Setup.....	129
7.4 Results.....	134
7.4.1 Measuring of Knowledge Utility	134
7.4.2 User Exploration Experience	136
7.5 Discussion.....	138
7.5.1 Study Hypothesis	138
7.5.2 Reflection on Exploration	139
7.5.3 Applicability of Evaluation Approach	140
7.6 Summary.....	141
Chapter 8 Conclusion	142
8.1 Synopsis	142
8.2 Contributions	144
8.3 Future Work.....	148

8.3.1 Immediate.....	148
8.3.2 Long-term.....	148
List of References.....	150
Appendix A Exploratory User Study.....	160
A.1 Exploratory User Study Structure	160
A.1.1 Introduction about the Exploratory User Study	160
A.1.2 Pre-study questionnaire	161
A.1.3 Introduction to MusicPinta (Exploration Example about ‘Table’).....	162
A.1.4 Conduct Your Exploration	163
A.1.5 Post-study Interview.....	164
A.2 Participants’ Information and Familiarity	165
A.3 Knowledge Expansion Results.....	166
A.4 User Exploration Experience	167
Appendix B Output of KA_{DG} Algorithms	168
B.1 KA _{DG} Metrics Normalised Output Values in MusicPinta	168
B.2 KA _{DG} Metrics Normalised Output Values in Occupation Class Hierarchy in L4All	174
B.3 KA _{DG} Metrics Normalised Output Values in Subject Class Hierarchy in L4All	181
Appendix C KA_{DG} Metrics Output of Knowledge Anchors.....	183
C.1 Precision, Recall and F1 values of KA _{DG} Metrics in MusicPinta	183
C.2 KA _{DG} Identified in MusicPinta.....	184
C.3 Precision, Recall and F1 values of KA _{DG} Metrics in Occupation and Subject Class Hierarchies in L4All	187
C.4 KA _{DG} Identified in Occupation Class Hierarchy in L4All	188
C.5 KA _{DG} Identified in Subject Class Hierarchy in L4All.....	191
Appendix D User Study to Evaluate the Subsumption Algorithm.....	192
D.1 Survey for Identifying User Familiarity with Entities in MusciPinta	192
D.2 Participants’ Familiarity.....	194
D.3 Examples of Transition Narratives of Exploration Paths for Biwa and Bansuri	195
D.4 Knowledge Utility.....	201

Abbreviations

BLO	Basic Level Objects
BLO_{DG}	Basic Level Objects in a Data Graph
FCA	Formal Concept Analysis
KA_{DG}	Knowledge Anchors in a Data Graph
RDF	Resource Description Framework

List of Figures

Figure 2.1 Three Webpages from Wikipedia linked via Hyperlinks	10
Figure 2.2 Three Wikipedia Webpages represented as a directed graph	10
Figure 2.3 Extract from DBpedia showing a simple data graph about Guitar.....	11
Figure 2.4 LOD with 12 datasets as in May 2007.	13
Figure 2.5 LOD with 12 datasets as in August 2017.	13
Figure 2.6 Database models which are used to store RDF data [55].....	18
Figure 3.1 Description of the overall research methodology.....	47
Figure 3.2 Description page of the entity 'Harp' in MusicPinta.....	54
Figure 3.3 Semantic Links related to the entity Harp presented in Features and Relevant Information.	54
Figure 3.4 Semantic Links related to the entity Harp presented in the Relevant Information.	55
Figure 3.5 L4All main user interface [154].	56
Figure 3.6 Structure of the user study to examine the performance of three exploration strategies.	61
Figure 3.7 Example of data graph trajectories for the three proposed exploration strategies in MusicPinta.	62
Figure 3.8 Knowledge utility of the three exploration strategies (Density, Familiarity and Unfamiliarity) of the user cognitive processes (<i>median</i> of the knowledge utility of exploration for all users).	64
Figure 3.9 Users' exploration experience of the three exploration strategies. Values show number of user paths rated with the corresponding characteristics.	67
Figure 3.10 Users' subjective perception of the three exploration strategies. Values are based on adapted NASA-TLX questionnaire [160] (median values for all users in range 1-10).....	68
Figure 4.1 A data graph showing entities and relationship types between entities.	73
Figure 4.2 Structure for Implementing the KA_{DG} Algorithms.....	78
Figure 4.3 Extract from the Occupation class hierarchy in L4All showing instances linked to members of the category Air Transport Operatives.....	80
Figure 5.1 An image of Violotta (a leaf in the data graph) was shown to a participant, who named it as Violin (parent of Violotta).	87
Figure 5.2 An image of Fiddle (a category entity in the data graph with two leaf entities) was shown to a user, who named it as Violin (parent of Fiddle).	87
Figure 5.3 An image of Violin (a category entity in the data graph) was shown to a user, who named it as Violin.....	88
Figure 5.4 An image of Bowed String Instrument (a category entity in the data graph) shown to a user, who named it as Violin (category member of Bowed String Instrument).....	88

Figure 5.5 A representation of Housekeeping Occupation (a Category concept in the Occupation class hierarchy with two subclasses) was shown to a user, who identified it as Personal Service Occupation.....	93
Figure 5.6 A representation of Biological Sciences (a category concept in the Subject class hierarchy with four random subclasses) was shown to a user, who identified it as Biological Sciences.	94
Figure 6.1. Extract from the String Instrument class hierarchy in MusicPinta showing exploration path of length 4 generated with first entity Biwa.....	114
Figure 6.2. Extract from the String Instrument class hierarchy in MusicPinta showing exploration path of length 6 generated with first entity Kinnor.....	115
Figure 6.3. Extract from the Wind Instrument class hierarchy in MusicPinta showing exploration path of length 4 generated with first entity Bansuri.....	116
Figure 6.4. Extract from the Wind Instrument class hierarchy in MusicPinta showing exploration path of length 4 generated with first entity Valved Brass Instruments.	117
Figure 6.5. Extract from the Occupation class hierarchy in L4All showing exploration path of length 5 generated with first entity Secondary Education Teaching Professionals.	118
Figure 6.6. Extract from the Occupation class hierarchy in L4All showing exploration path of length 4 generated with first entity Leisure and Other Personal Service Occupations.	119
Figure 6.7. Extract from the Occupation class hierarchy in L4All showing exploration path of length 4 generated with first entity Transport and Mobile Machine Drivers and Operatives.	121
Figure 7.1. Structure of the user study to examine <i>EC</i> against <i>CC</i> in terms of knowledge utility, usability and cognitive load.....	130
Figure 7.2. Example of a transition narrative shown to participants from the exploration path of Biwa.....	131
Figure 7.3. Example of a transition narrative shown to participants from the exploration path of Bansuri.....	132
Figure 7.4. Knowledge utility of the two strategies (<i>EC</i> and <i>CC</i>) of the user cognitive processes (<i>median</i> of the knowledge utility of exploration for all users).	134
Figure 7.5. Knowledge utility of the two strategies (<i>EC</i> and <i>CC</i>) of the user cognitive processes (<i>median</i> of the knowledge utility of exploration for all users).	135
Figure 7.6. Users' exploration experience of the two exploration strategies (<i>EC</i> and <i>CC</i>).....	137
Figure 7.7. Users' subjective perception of the two exploration strategies (<i>EC</i> and <i>CC</i>), based on an adapted NASA-TLX questionnaire [160] (<i>median</i> values for users in range 1-10).....	137

List of Tables

Table 2.1 Three triples about the Entity Guitar extracted from DBpedia.....	12
Table 3.1. Main characteristics of MusicPinta data graph.....	53
Table 3.2. Main characteristics of the L4All data sets.....	57
Table 3.3 Allocation of exploration strategies for the selected musical instruments in MusicPinta.	62
Table 3.4 Statistically significant differences of knowledge utility values (Mann-Whitney, 1-tail, $N_a=N_b=12$)......	64
Table 3.5 Summary of the recognised entities along the users' exploration paths (median values).....	65
Table 4.1 An example of a formal context in FCA represented as sets of objects M and attributes R , where (x) indicates that an object m has attribute r	71
Table 5.1 Precision, recall and F1 values by comparing 60 th percentiles of the KA_{DG} metrics normalised output values over hierarchical (H) and domain-specific (D) relationships, with human BLO_{DG} derived in MusicPinta.	89
Table 5.2 precision, recall and F1 values for comparing human BLO_{DG} and KA_{DG} derived using hierarchical (H) and domain specific (D) relationships in MusicPinta after applying the weighted median.....	90
Table 5.3 Precision, recall and F1 values for comparing the 60 th percentiles of the KA_{DG} metrics normalised output values over the hierarchical (H) and domain-specific (D) relationships with human BLO_{DG} derived, in the Occupation and Subject class hierarchy in L4All data graph.	96
Table 5.4 precision and recall values for comparing human BLO_{DG} and KA_{DG} derived from the metrics using hierarchical (H) and domain-specific (D) relationships in Occupation and Subject class hierarchies after applying weighted median.....	97
Table 6.1. Narrative types of transitions between entities in an exploration path	108
Table 6.2. First entities V_s used in implementation of subsumption algorithm to generate exploration paths	112
Table 6.3. Semantic similarity vales between first entities and KA_{DG} in MusicPinta.....	113
Table 6.4. Semantic similarity between Biwa and KA_{DG} ; and similarity between Bansuri and KA_{DG}	113
Table 6.5. Transition narratives used for generating the exploration path for Biwa	114
Table 6.6. Transition narratives used for generating the exploration path for Kinnor	115
Table 6.7. Transition narratives used for generating exploration path for Bansuri	117
Table 6.8. Transition narratives used for generating the exploration path for Valve Brass Instruments.....	118
Table 6.9. Transition narratives used for generating exploration path for Secondary Education Teaching Professionals	119

Table 6.10. Transition narratives used for generating exploration path for Leisure and Other Personal Service Occupations.....	120
Table 6.11. Transition narratives used for generating exploration path for Transport and Mobile Machine Drivers and Operatives	121
Table 7.1 Task template used in the experimental user study	127
Table 7.2. Examples of entities the participants have visited during their free exploration (<i>CC</i>) of Biwa, compared to generated exploration paths (<i>EC</i>).....	133
Table 7.3. Examples of entities the participants have visited during their free exploration (<i>CC</i>) of Bansuri, compared to generated exploration paths (<i>EC</i>).....	133
Table 7.4. Statistically significant differences of the values in Figure 7.4 (Mann-Whitney, 1-tail, $N_a=N_b=32$).....	134
Table 7.5. Statistically significant differences of the values in Figure 7.5 (Mann-Whitney, 1-tail, $N_a=N_b=16$).....	136
Table 7.6. Questions of NASA-TLX questionnaire used to measure the users' perception of subjective process	138
Table 7.7. Statistically significant differences of users' subjective perception of cognitive process (Mann-Whitney, 1-tail, $N_a=N_b=32$)	138

Chapter 1

Introduction

In the recent years, linked data (in the form of RDF graphs) has emerged as the de facto standard for sharing data on the Web. Consequently, data graphs generated from semantic databases have become widely available on the Web and are being adopted in a range of users facing applications that provide search and exploration tasks. In contrast to regular search where the user has a specific need in mind and an idea of the expected search result [1], exploratory search is open-ended requiring significant amount of exploration [2], has an unclear information need [3], and is used to conduct learning and investigative tasks [4]. Users do exploratory search when they explore resources in a new domain (like in academic research tasks) or browse through large information spaces with many options (like exploring job opportunities, travel and accommodation offers, videos, culture, cuisine and music). An example of a learning task scenario is a student in a class room whom was asked to give a speech about the 'North Pole'. The student doesn't really know what kind of information he/she would like to search for or what will be discovered about this topic. The piece of information the user currently knows is that he/she wants to learn and gain some knowledge about the 'North Pole'.

In many cases, the users will have no (or limited) familiarity with the specific domain. When the users are novices to a domain, the *users' cognitive structures* about that domain are unlikely to match the complex *knowledge structures of a data graph that represent the domain*. This can have a negative impact on the user exploration experience and effectiveness. Users can find themselves in situations where they are not able to formulate knowledge retrieval queries (users do not know what they do not know [3]). Users can face an overwhelming amount of exploration options, not being able to identify which exploration paths are most useful; this can lead to confusion, high cognitive load, frustration and a feeling of being lost.

To overcome these challenges, appropriate ways to facilitate user exploration through data graphs are required. Research on exploration of data graphs has come a long way from initial works on presenting linked data in visual or textual forms [5, 6]. Recent studies on data graph exploration have brought together research from diverse, but related, areas such as Semantic Web, personalisation, adaptive hypermedia, and human-computer interaction; with

the aim of reducing user cognitive load and providing support for knowledge exploration and discovery [7–9]. Several attempts addressed supporting *layman users*, i.e. users who are not domain experts and they have partial knowledge about the domain. Examples include: personalising the exploration path tailored to the user’s interests [10], presenting RDF patterns to give an overview of the domain [11], or providing graph visualisations to support navigation [12].

Although a significant amount of work addresses the problem of facilitating user exploration through data graphs, this has been applied mainly in investigative tasks. There is limited research on supporting learning through data graph exploration. The learning perspective has been studied with regard to providing generic tools for exploration of interlinked open educational resources [13]. We address an important outstanding challenge by focusing on the learning effect of user exploration, i.e. expanding the user’s domain knowledge while he/she is exploring a data graph in an unfamiliar or partially familiar domain. Supporting learning through search is an emerging research area in information retrieval [14, 15]. It argues that “searching for data on the Web should be considered an area in its own right for future research in the context of search as a learning activity” [16]. Motivated by this vision, we investigate how learning can be supported through exploration of data graphs.

Our work opens a new avenue in data graph exploration, which looks at the *knowledge utility*, i.e. expanding one’s domain knowledge while exploring a data graph. This builds on earlier work showing that while exploring data graphs in unfamiliar (or partially familiar) domains, users *serendipitously* learn new things (e.g. facts, concepts, etc) that they were unaware of [17–19]. However, not all exploration paths can be beneficial for knowledge expansion, e.g. paths may not bring new knowledge to the user or may bring too much unfamiliar things so that the user becomes confused and frustrated [18].

The main goal of our research is to develop automatic ways to generate exploration paths that can expand the user’s domain knowledge while exploring a data graph. A scoping user study which investigated several exploration strategies for knowledge expansion [20] pointed us to adapt Ausubel’s subsumption theory for meaningful learning [21] as the underpinning model for the generation of exploration paths (Ausubel’s subsumption theory is discussed in details in Section 2.7.2.2). This theory postulates that human cognitive structures are hierarchically organised with respect to levels of abstraction, generality, and inclusiveness of concepts; hence, familiar and inclusive entities are used as *knowledge anchors* to subsume new knowledge into the users’ cognitive structures. Such entities are crucial for expanding one’s domain knowledge.

Adapting Ausubel’s subsumption theory for meaningful learning, our work addressed the following research questions:

RQ1. *How to develop automatic ways to identify knowledge anchors in a data graph?* This means finding graph entities which correspond to domain concepts that can be familiar to layman users.

RQ2. *How to use knowledge anchors to generate exploration paths to facilitate domain knowledge expansion in a data graph?* This means suggesting to the user graph entity sequences for exploration, which can result in learning new things about the domain.

To address **RQ1**, we utilise Rosch’s notion of Basic Level Objects (BLO) [22] in the context of data graphs. We devise a formal framework that maps Rosch’s definitions of BLO and cue validity to data graphs, and develop several metrics and the corresponding algorithms to identify knowledge anchors in a data graph (KA_{DG}). These anchors represent *concepts which are likely to be familiar to layman users*, and hence can be used as knowledge bridges of familiar entities from where links to new knowledge can be made. To evaluate the KA_{DG} algorithms, we compare the derived KA_{DG} against human cognitive structures. Using an experimental approach that adapts Cognitive Science methods to derive BLO in a domain, we identify human basic level objects in a data graph (BLO_{DG}) used to benchmark the KA_{DG} algorithms. Based on the evaluation, we identify hybridisation heuristics to improve precision and recall.

To address **RQ2**, we develop an algorithm that is underpinned by the subsumption theory for meaningful learning [21] to generate exploration paths for knowledge expansion. This involves two steps – identifying the most appropriate knowledge anchor to start the path, and iterative introduction of new knowledge through subsumption to connected entities. We have conducted a controlled task-driven user study with a semantic data browser in the Music domain to examine the effectiveness of the suggested paths to increase the users’ domain knowledge. The study shows that the generated paths have significantly higher knowledge utility and provide better exploration experience than free exploration paths.

This work in this research contributes to knowledge by providing:

- **Computational methods for identifying knowledge anchors in data graphs.**

We provide formal description and implementation of metrics and the corresponding algorithms for identifying KA_{DG} . Two groups of metrics are developed: (i) *distinctiveness* metrics, where we adapt metrics from Formal Concept Analysis (FCA) [23] to identify differentiated category entities whose members share attributes that are not linked to members of other categories (distinctiveness metrics are discussed in

Section 4.2.1); and (ii) *homogeneity* metrics where we apply set-based similarity metrics [24] to identify categories whose category members share many attributes together (homogeneity metrics are discussed in Section 4.2.2). The KA_{DG} algorithms are generic and can be applied over different application domains represented as data graphs. The algorithms are implemented in two data graphs (MusicPinta, L4All) from two domains (musical instrument and career), respectively.

To evaluate the KA_{DG} algorithms, we compare the derived KA_{DG} against human cognitive structures. Using an experimental approach that adapts Cognitive science methods to derive human BLO in a domain, we identify data graph entities referring to categories that are highly familiar and inclusive in human cognitive structures, used to benchmark KA_{DG} algorithms including hybridisation heuristics to improve performance. The validation of the KA_{DG} algorithm allows utilising the algorithms to derive KA_{DG} used to generate exploration paths for knowledge expansion based on subsumption.

- **Computational methods for generating exploration paths for knowledge expansion.**

We formally describe and implement an automatic approach for generating exploration paths using KA_{DG} adapting the subsumption theory for meaningful learning. We adapt this theory focusing on the subsumption relationship `rdfs:subClassOf` in the data graph and use this relationship to introduce new subclass entities of the knowledge anchor while generating an exploration path. Strictly speaking, we are not applying Ausubels' model of advance organisers [25] which act as previews (usually in the form of written passages) at a higher level of abstraction than new learning materials (i.e. the new subclass entities). These organisers are introduced to help the user to recognise what new elements can be meaningfully linked to relevant knowledge anchors [26] (Ausubel's model of advance organisers is described in Section 2.7.2.2). In our approach, we are not presenting the user with advance organisers prior subsuming subclass entities of the knowledge anchor. Our approach includes two parts:

- (i) Algorithm for identifying the closest knowledge anchor to the first entity of an exploration path using semantic similarity. To start the subsumption process, the user has to be directed from the first entity of an exploration path to a knowledge anchor in the data graph. However, a data graph can have several knowledge anchors, and this algorithm is used to identify the closest knowledge anchor.
- (ii) Algorithm for generating an exploration path consisting of a set of transition narratives using the closest knowledge anchor. The closest knowledge anchor can have many subclass entities that exist at different levels of abstractions in the data graph. Hence, this algorithm identifies which subclasses to subsume and in what order for generating an exploration path. Furthermore, the algorithm uses narrative

scripts between the entities in the exploration path. Providing meaningful narrative scripts between entities can help layman users to create meaningful relationships between familiar entities they already know and the new subsumed entities

The algorithms are applied over two data graphs (MusicPinta and L4All). A controlled task-driven user study over MusicPinta is conducted to examine the effectiveness of the suggested exploration paths compared to free exploration where users freely select entities to visit. The validation showed that our approach can be successfully applied to support the user's exploration in a data graph.

- **Instruments for evaluation of data graph exploration.** Three instruments are developed:
 - (i) **Approximating knowledge utility:** we describe an approach based on Bloom's taxonomy (described in Section 2.7.1) for approximating the changes in the user's cognitive knowledge after exploring a path in a data graph. Our approach for approximating knowledge utility of an exploration path is applied in two experimental user studies which indicates the generality of our approach.
 - (ii) **Algorithm to obtain human BLO_{DG} :** we formally describe an algorithm for identifying human BLO_{DG} which correspond to human cognitive structures over a data graph. The algorithm is applied in two application domains for data exploration (musical instrument and career) using the data graphs from the two domains, which allows two ways of instantiating the algorithm for obtaining human BLO_{DG} :(i) concrete domains (e.g. musical instrument) where objects have digital representations, and (ii) abstract domains (e.g. career) where objects are represented using text labels.
 - (iii) **User-driven exploration task.** We describe a two-step approach to design suitable data exploration task used for evaluating data exploration approaches. The two steps involve designing a task template, and identifying unfamiliar entities in the domain which can plugged into the task template. The task template is designed in the context of a general knowledge quiz which allows its adoption to encourage users to seek knowledge in a given domain represented as a data graph.

Boundaries of our approach. KA_{DG} metrics can be sensitive to the quality of the data graph in terms of the richness of the class entities and the depth of the class hierarchy. Specifically, the metrics tend to pick more anchoring entities with larger graphs that have many classes and high level of depth. Applying the KA_{DG} algorithms over large data graphs may produce high number of KA_{DG} , which requires further filtering approach to reduce the number of KA_{DG} . One possible way to address this issue is to use crowdsourcing to identify the most

familiar entities to the crowd. Furthermore, KA_{DG} metrics may not produce KA_{DG} in shallow class hierarchies (e.g. when the depth of the class hierarchy equals to one or two) especially that if the algorithms use the subsumption relationship `rdfs:subClassOf` to indicate membership of class entities (as discussed in Section 4.4).

In our work, we adapt the subsumption theory for meaningful learning [21] to generate exploration paths for knowledge expansion using KA_{DG} . For this, we formally describe two algorithms.

- *Algorithm for identifying the closest knowledge anchor to the first entity of an exploration path.* It applies a semantic similarity metric based on class hierarchy depth to identify the closest knowledge anchor to the first entity of an exploration path. This similarity algorithm is not applicable in data graphs with shallow class hierarchies (e.g. class hierarchy depth = 1 or 2). In such cases, there will be no common ancestors for the *first entity* and the *knowledge anchors*, and hence the semantic similarity value will be zero (as discussed in Section 6.4.1). Furthermore, two (or more) knowledge anchors in a data graph can have the same semantic similarity value with the first entity. In some cases, the semantic similarity metric can identify closest knowledge anchors which are not superclass, subclass nor sibling to the first entity, which means that the closest knowledge anchor can't be reached directly from the first entity.
- *Algorithm for generating exploration path as a set of transition narratives using the closest knowledge anchor.* The algorithm is underpinned by the subsumption theory for meaningful learning to generate exploration paths for knowledge expansion. The algorithm uses a knowledge anchor to subsume subclass entities while generating transition narratives of an exploration path, and hence the algorithm is dependent on the availability of sub-classes of the selected knowledge anchor (discussed in Section 6.4.2). Accordingly, it may not be possible to generate m transition narratives in the exploration path (where m is the required length of exploration path identified as an input of the algorithm). Our implementation uses only the knowledge anchor, and can result in paths whose length of less than m .

Thesis structure. The thesis is structured into eight Chapters. In this Chapter, we introduced the research motivation, the research questions and the research contributions.

Chapter 2 provides background information to better understand this research context, and positions the work in the relevant literature. It describes cognitive science theories related to this research. An approach for approximating knowledge utility of exploration paths is presented. The underpinning theoretical model for generating exploration paths is described, and notion of BLO is presented.

Chapter 3 discusses the overall research methodology for addressing the research questions and provides definitions that will be used to describe the algorithms in Chapters (4-6). The two application contexts that will be used for experimentations are described. An exploratory user study which examines exploration strategies over a data graph is presented.

Chapter 4 formally describes two groups of metrics for identifying KA_{DG} with the corresponding algorithms for applying the metrics in the context of a data graph. The implementation of the KA_{DG} algorithms over two data graphs from two domains is presented.

Chapter 5 formally describes an algorithm to identify a benchmarking set of basic level objects underpinned by a data graph used to evaluate the KA_{DG} algorithms. Two experimental user studies for evaluating the KA_{DG} algorithms using the derived human BLO_{DG} are presented.

Chapter 6 formally describes a subsumption algorithm used to generate exploration paths using KA_{DG} for knowledge expansion. The implementation of the subsumption algorithm over two data graphs from two domains is presented and several examples of exploration paths from the two domains are illustrated.

Chapter 7 presents an experimental user study to evaluate the knowledge utility of the generated paths against free exploration.

Chapter 8 concludes the thesis by summarising the key achievements, contributions and potential future work.

Chapter 2

Related Work

2.1 Introduction

This Chapter provides background information to present the research context for this thesis. We will position our research with the related work that has been accomplished in three related fields: (i) *data graph exploration* where we will review two broad categories for exploring data graphs: visualisation-based and text-based approaches; (ii) *identifying key entities in data graphs*, where we will review technical approaches for identifying key entities which have been applied in two broad areas: ontology summarisation and formal concept analysis, in order to justify the contribution of our approach for identifying knowledge anchors in a data graph (KA_{DG}); and (iii) *generating exploration paths in data graphs* where we will discuss different approaches used for constructing exploration paths to justify our approach for generating exploration paths using KA_{DG} . We will review relevant evaluation approaches since we will use similar approaches for evaluating our algorithms for identifying KA_{DG} and generating exploration paths for knowledge expansion.

Furthermore, we will review three Cognitive science theories which are related to this research. The three theories are: (i) Bloom's taxonomy [27] for knowledge classification and show how we will adapt it as a question-answering approach for approximating knowledge utility; (ii) the subsumption theory for meaningful learning introduced by David Ausubel [21] since this theory will underpin our approach for generating exploration paths in a data graph; and (iii) the Cognitive since notion of Basic Level Objects (BLO) introduced by Rosch, et al. [22] which we will adopt to develop our algorithms for identifying KA_{DG} . We will review experimental user studies in Cognitive science for identifying BLO, since we will conduct similar studies in two application contexts for data graph exploration to identify human Basic Level Objects in a data graph (BLO_{DG}) used to examine the performance of our algorithms for identifying KA_{DG} .

Next in this Chapter, we will provide background information (Section 2.2). In Section 2.3, we will describe approaches for data graph exploration. In Section 2.4 we will discuss related approaches for identifying key entities in data graphs. In Section 2.5 we will review approaches for generating exploration paths in data graphs. In Section 2.6 we will outline evaluation approaches related to the evaluation that will be conducted in this research. In

Section 2.7 we will describe three theories from Cognitive science that will be adapted in this research and Section 2.8 will summarise the work presented in this Chapter.

2.2 Background

In this Section, we will provide background information to better understand the research context. In Section 2.2.1, we will provide a brief description of the traditional Web that we know, referred as the Web graph. We will show how the Semantic Web vision [28] evolved the Web graph from a graph of Webpages connected via Hyperlinks into a structured data graph represented as data entities connected via Semantic relationships. We will describe the notion of Linked Data and describe a graph-based data model called the Resource Description Framework (RDF) which is used for publishing structured data in the Web (Section 2.2.2). Furthermore, we will review historically the growth of Linked Data as part of our research problem, since this growth increases the challenge of supporting user's exploration over data graphs where users are exposed to high number of entities and links. In Section 2.2.3, we will describe what is an ontology and we will outline two of the widely used ontology languages. In Section 2.2.4 will describe SPARQL query language, the query language used for querying data graph. In Section 2.2.5, we will review graph databases and triple stores since they are used for storing and managing data graphs. Finally, in Section 2.2.6 we will describe two types of Semantic Web applications which will be part of this research context, namely Semantic data search engines and Semantic data browsers.

2.2.1 The Web Graph

The World Wide Web (WWW), also known as the Web, has become our first choice for information search. The primary units of the Web are the HyperText Markup Language (HTML) documents, also called Webpages. Webpages are connected via un-typed links called Hyperlinks. This view of the Web (i.e. Webpages interconnected via Hyperlinks) is called the Hypertext paradigm [29]. Users in the Hypertext paradigm read sensitive parts of text in Webpages (called Hypertexts), and by pointing and clicking on one Hypertext, the user is then transferred into a corresponding Webpage [29]. For example, Figure 2.1 shows three connected Webpages from Wikipedia¹. The WWW² Webpage has two Hypertexts, namely

¹ https://en.wikipedia.org/wiki/Main_Page

² https://en.wikipedia.org/wiki/World_Wide_Web

*Web browser*³ and *Website*⁴. By clicking on one of the Hypertexts, the user is then transferred to the corresponding Webpage.

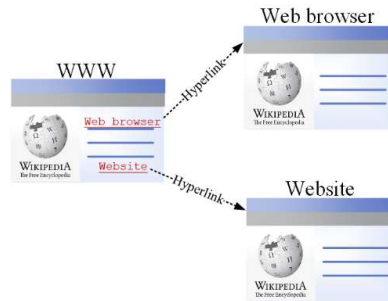


Figure 2.1 Three Webpages from Wikipedia linked via Hyperlinks

The Webpages and Hyperlinks in Figure 2.1 can be viewed as entities and edges of a directed graph, referred as the *Web graph* [30]. The authors in [31] have formalised the view of the traditional Web as a Web graph, where they ignored the content of Webpages (e.g. text and images) and focused on the links between the Webpages. Accordingly, the example from Wikipedia shown in Figure 2.1 can be viewed as a directed Web graph, as in Figure 2.2.

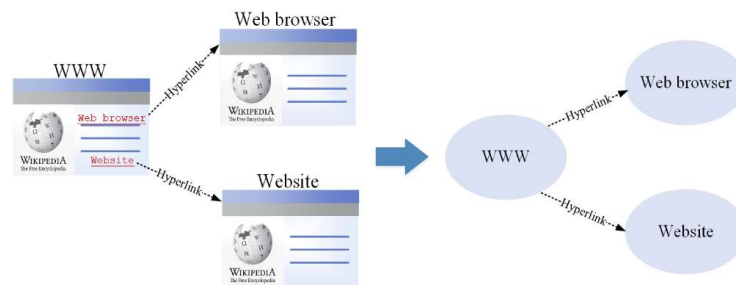


Figure 2.2 Three Wikipedia Webpages represented as a directed graph

However, Web graphs are built as a mediums of Webpages which can be understood by humans [28] where the main task for the machine (e.g. a Web browser in a computer) is to search for key words in the Webpages and present resulting Webpages to the users in a readable format (i.e. content of Webpages is presented in human language and can't be processed by machines). The Semantic Web vision started in 2001 in order to evolve the conventional Web from a global Web graph of Webpages linked via Hyperlinks into a global data space where both documents and data entities are semantically linked in a structured way [32]. In his description of the Semantic Web [33], Tim Berners-Lee, the inventor of the WWW, stated that: "The first step is putting data on the Web in a form that machines can naturally understand, or converting it to that form. This creates what I call a Semantic Web – a web of data that can be processed directly or indirectly by machines". To fulfil the Semantic

³ https://en.wikipedia.org/wiki/Web_browser
⁴ <https://en.wikipedia.org/wiki/Website>

Web vision, the notion of Linked Data emerged in 2006 as a set of best practices for publishing data on the Web [32].

2.2.2 Linked Data

Linked Data refers to a set of best practices for publishing data in the Web in a structured way [32]. It is about using the Web to create typed links (i.e. relationships) between data entities from different sources [34]. Linked Data is building on using traditional Web standards (e.g. Uniform Resource Identifiers (URIs) for identifying any resource on the Web and Hypertext Transfer Protocol (HTTP) used for dereferencing these URIs). A glossary of the terms used in Linked Data practices and its associated vocabularies are available here⁵. While the prime units of the Web graph are the HTML documents, Linked Data relies on a common graph-based data model for publishing structured data on the Web called the Resource Description Framework (RDF) [35]. RDF describes resources (i.e. things) on the Web, where each resource has a unique identifier called the Uniform Resource Identifier (URI) [34]. A resource can be anything such as a person, a movie or a Webpage that we want to make one or more statements about. An RDF statement, referred to as a triple, is represented as a triple of the form *<Subject - Predicate - Object>*. The *Subject* is the resource identified by a URI which we want to make a statement about. An *Object* is either a URI or a string (i.e. text). Each *Predicate* URI denotes a directed relationship between two entities which has the *Subject* as a source entity and the *Object* as a target entity. In this research, the view of RDF triples as data entities (*Subjects* and *Objects*) linked via typed links (*Predicates*) is referred as a data graph. For example, Figure 2.3 shows a simple data graph extracted from DBpedia⁶ (the structured knowledge of Wikipedia), in which the resource ‘Guitar’ is described through three statements (i.e. RDF triples) which are listed in Table 2.1.

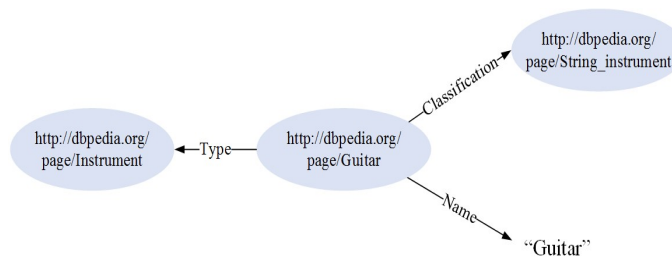


Figure 2.3 Extract from DBpedia showing a simple data graph about Guitar.

The data graph shows three triples about the entity *Guitar*. The three oval entities represent three URIs for three entities in DBpedia. The name *Guitar* is a string *Object* (i.e. not URI).

⁵ <https://www.w3.org/TR/ld-glossary/>

⁶ <http://wiki.dbpedia.org/>.

Table 2.1 Three triples about the Entity `Guitar` extracted from DBpedia

Subject	Predicate	Object
http://dbpedia.org/page/Guitar	Classification	http://dbpedia.org/page/String_instrument
http://dbpedia.org/page/Guitar	Type	http://dbpedia.org/page/Instrument
http://dbpedia.org/page/Guitar	Name	'Guitar'

In 2006, Tim Berners-Lee outlined four principles (i.e. rules) for publishing data on the Web in a structured way [36]. The aim was to support data providers to publish available data as Linked Data so that their data become part of a global open data space containing different data sets interlinked together. The principles for publishing Linked Data are:

- Use Uniform Resource Identifiers (URIs) as global names for identifying things. URIs could be names for real-world object not only the HTML documents.
- Use Hypertext Transfer Protocol (HTTP) URIs to look up things. In other words, allow people to use HTTP as a well-understood retrieval mechanism to look up resources.
- Use standards such as RDF and SPARQL query language. This advocates the use of an agreed single data model for publishing structured data on the Web, as well as using a common language for querying structured data.
- Include links to other URIs to discover more things. This advocates the use of hyperlinks to connect not only Web documents, but to connect any type of thing and describe the relationship between two things.

The most visible example of applying Linked Data principles has been the Linking Open Data (LOD) project⁷. The LOD project has started in 2007 to initiate the Web of Data by identifying available datasets which are under open licenses, converting them to RDF using the Linked Data principles, and then making them available on the Web [32]. Since then, an increasing number of data providers have been adopting Linked Data principles and publishing their data as open data. Accordingly, the LOD has been growing rapidly due to the huge addition in numbers of linked entities, from (12) datasets in 2007 to more than 1100 datasets in 2017⁸ (See Figures 2.4 and 2.5).

Furthermore, Linked Data entities are becoming lengthier in number of entities and links (i.e. triples) the entities are associated with. For example, entities and triples in the English version of DBpedia has been growing rapidly since 2007 (from around 2M entities and 100M triples in 2007 to more than 6.6M entities and 11.5B triples in 2016⁹). This

⁷ <http://lod-cloud.net>

⁸ <https://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>

⁹ <http://wiki.dbpedia.org/datasets/dbpedia-version-2016-10> . Statistics for 2017 are not published yet

increased the average number of links per entity in the English version of DBpedia from 50 links per entity in 2007 to around 1750 links per entity in 2016. This huge increase in number of links per entity increases the challenge of supporting users' exploration through data graphs and represents one of the challenges that motivates this research.

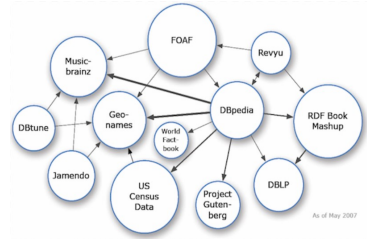


Figure 2.4 LOD with 12 datasets as in May 2007¹⁰.

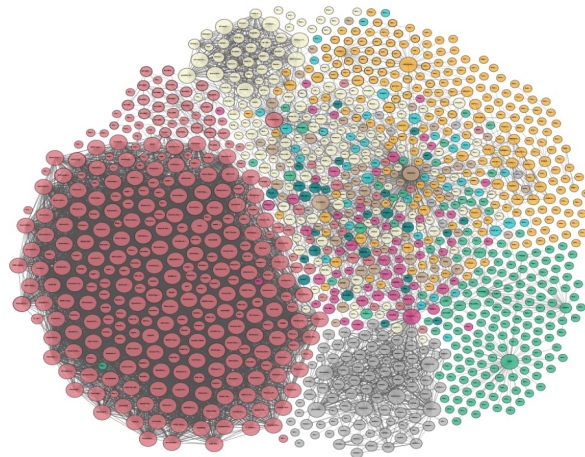


Figure 2.5 LOD with 12 datasets as in August 2017¹¹.

2.2.3 Ontology

Ontologies have played a central role in the development of the Semantic Web [37]. The WWW Consortium (W3C¹²) defines ontology as: the terms used to describe and represent an area of knowledge. A famous definition of ontology was provided by Gruber [38], who defines it as a formal explicit specification of a shared conceptualization. ontology defines a common vocabulary for sharing information within a domain [32]. Hence, whenever an ontology already contain the terms needed to represent a specific dataset, the ontology should be used rather than developing a new one [34]. In Semantic Web, an ontology contains a collection of classes, instances (also called individuals) and properties (i.e. relationships) [32].

Classes: define the main concepts in an ontology (e.g. the class `Guitar` in musical instrument ontology). Characteristics of classes apply to their instances, or individuals (e.g. the fact that the class `Guitar` has `nick` and `strings` can be applied to individuals (instances) of `Guitar` such as `Vietnamese Guitar`). Every ontology has a class hierarchy consisting of all classes linked via the subsumption relationship `rdfs:subClassOf`. Every subclass in the hierarchy inherits the characteristics of its super class. Any class in the class hierarchy can be either (i) root class (i.e. superclass of all class), (ii) category class (i.e. any

¹⁰ <http://lod-cloud.net/versions/2007-05-01/lod-cloud.png>

¹¹ <http://lod-cloud.net/versions/2017-08-22/lod.png>

¹² <https://www.w3.org/TR/webont-req/>

class except the *root class* that has one or more subclasses) or (iii) leaf class (any class that has no subclasses).

Instances: (also called individuals) can be concrete objects such as people, animals, musical instruments; or abstract individuals such as numbers and words (strings). Every instance belongs to at least one class (i.e. one instance can belong to more than one class). An instance inherits the attributes of its class, and has specific values which differentiates them from other individuals in the class. In RDF, the predicate `rdf:type` is used to indicate that an *instance A is a type of class B*.

Relationships: (also called properties) in an ontology are used to define the characteristics of the classes. Relationships have labels and they allow to define links between classes and instances. An important type of relationships between classes is the subsumption relationship `rdfs:subClassOf` which is used to identify the subclass/ superclass relationships in the class hierarchy (also called the subsumption class hierarchy). While the subsumption relationship is a common relationship among different ontologies, however an ontology may have other relationship types which are domain specific (i.e. the relationships are dependant of the represented domain). For example, the musical instrument ontology has a domain-specific relationship `MusicOntology:instrument` to indicate a musical performance where an instrument has been performed.

The criteria for selecting ontologies include four elements [34]:

- Usage and uptake – is the ontology widely used?, and will using this ontology will make RDF data more or less accessible to others?;
- Maintenance and governance – is the ontology actively maintained and updated by its creators?;
- Coverage – does the ontology cover enough concepts of the dataset (or concepts in the domain)?; and
- Expressivity – degree of expressivity in the vocabulary appropriate to the dataset?

Ontology languages (also called schema languages) are formal languages that provide a basis for creating vocabularies that can be used to describe entities in the world and how they are related [32]. These languages define RDF link types to describe relationships between classes. Such links normally describe common things like people, places or movies. There are several ontology languages available for representing ontology information on the Semantic Web. In this research context, we will focus on two well-known and widely used ontology languages [39], which we describe below:

RDF Schema (RDFS): is the simplest ontology definition language which is considered as the basis for other ontology languages [40]. It is an extension of RDF, and enables users to

define hierarchies of concepts and properties using RDF format. It provides a data-modelling vocabulary under the namespace `rdfs` for defining ontologies. A namespace is identified by an IRI where core RDFS modelling includes the following:

- `rdfs:Class` This is the class of resources that are RDF classes.
- `rdfs:Literal` This is the class of literal values such as strings and integers.
- `rdfs:Property` This is the class of RDF properties.

Among many properties used to define relationships in RDFS, the most used properties are:

- `rdf:type` Relates an instance to its class.
- `rdfs:subClassOf` Relates a class to one of its super classes.
- `rdfs:range` Specifies the range of a property (i.e. specifies the *Object* for the property).
- `rdfs:domain` Specifies the domain of a property (i.e. specifies the *Subject* for the property).

Web Ontology Language (OWL): is a part of the W3C's Semantic Web technology stack, which includes RDF, RDFS and SPARQL. OWL language that provides everything that RDFS provides, and it extends RDFS expressivity with additional modelling primitives [41]. For example, OWL defines the primitives `owl:equivalentClass` and `owl:equivalentProperty`. When these two primitives are combined with RDFS primitives (e.g. `rdfs:subClassOf` and `rdfs:subPropertyOf`) they provide powerful mechanisms for providing mappings between concepts from different ontologies, in a way that increases the interoperability of datasets underpinned by different ontologies [34].

Overall, ontologies provide agreed concepts and rules, which facilitates exploring through data graphs. Ontologies are transformed into RDF triples and asserted into a triple store, and become query-able. One of the most common and widely used RDF query language is SPARQL [39], which will be described next.

2.2.4 SPARQL Query Language

SPARQL is an RDF-based query language. It can be used to express queries across diverse data sources. The results of SPARQL queries can be in the form of results sets or RDF graphs. SPARQL is based on matching graph patterns. The simplest graph pattern is a triple, similar to RDF triple where the *Subject*, *Predicate* and *Object* can be replaced with variables [42].

A simple SPARQL query consists of two parts: the *SELECT* clause identifies the variables to appear in the query results, and the *WHERE* clause provides the basic graph pattern to match against the data graph. The result of a query depends on the way in which

the query's graph pattern matches the data, hence, the results could be nothing if there is no match, one or multiple solutions (i.e. one row in the body of the table) that match the query. Furthermore, matching could be to literals. The W3C defined the following structure (in order) for SPARQL queries over RDF data [43]:

- *Prefix declarations*, for abbreviating URIs.
- *Dataset definition*, stating what RDF graph(s) are being queried
- *A result clause*, identifying what information to return from the query
- The *query pattern*, specifying what to query for in the underlying dataset
- *Query modifiers*, slicing, ordering, and rearranging query results

As a simple example, consider the following query:

```
#prefix declaration:
prefix dbp-ont: <http://dbpedia.org/ontology/>
Prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
#result clause: what information to return
SELECT ?Astronaut
#dataset definition
FROM <http://dbpedia.org>
#query pattern
WHERE {
    ?Astronaut rdf:type dbo:Person.           //1st patterns
    ?Astronaut rdf:type dbo:Astronaut.       //2nd pattern
    ?Astronaut dbo:nationality dbr:United_States. //3rd pattern
#query modifier
Order by ? Astronaut asc
```

Running this SPARQL query in DBpedia SPARQL endpoint¹³, will return all Persons (1st pattern) who are Astronauts (2nd pattern) from the United States (3rd pattern), ordered by Astronauts ascending.

2.2.5 Graph Databases and Triple Stores

Being a W3C (WWW Consortium) recommendation, RDF has rapidly gained high popularity among data owners and publishers from different disciplines (e.g. News, Social Media, Medicine, Life Sciences, etc). Consequently, huge amount of RDF data is becoming published or available to be published, which requires physical stores for storing and managing RDF data in an efficient way and provide querying capabilities for the stored data [44, 45].

¹³ <http://live.dbpedia.org/sparql>

Storing RDF data can be viewed from two perspectives: graph-based and database-based perspectives. On the one hand, RDF models information with graph-based structure where basic notions of graph theory such as entity, edge, path, degree, depth etc., can be applied [46]. On the other hand, RDF can be seen as an extension of data models used in the database community, in particular graph database models, where RDF triples can be stored in a single relational database table [47].

The work in [48] described three perspectives for storing RDF data were identified, namely: *relational perspective* (from relational databases), *entity perspective* (from information retrieval), and a *graph-based perspective* (from graph theory and graph databases). The *relational perspective* views RDF data as a particular type of relational databases, and hence techniques developed for storing, indexing and answering queries on relational databases can be applied. In the *entity perspective*, each resource (i.e. *Subject*) is represented as a set of attribute-value (i.e. *Predicate-Object*) pairs. The *graph-based perspective* views RDF data as a classical graph where *Subjects* and the *Objects* of RDF statements form the entities in the graph, and *Predicates* specify directed and labelled edges.

The research in [49] has classified RDF triple stores based on their size into *centralised* (RDF data are stored on a single-machine with limited scalability) and *distributed* (RDF data are stored across multiple machines where query processing is parallelized among them) triple stores. Examples of centralised triple stores are Sesame¹⁴ (now known as RDF4J) [50], Jena (TDB triple store)¹⁵ [51], and Virtuoso¹⁶ [52]. Although these triple stores apply different number of index combinations for significantly reduced response times, however centralised solutions are vulnerable to the growth of the RDF data especially as we entering the era of big data graphs. In order to tackle the big-data challenge, research has recently moved towards utilising distributed RDF data stores which uses indexing schemas and join algorithms are used to create and run SPARQL queries over a large number of triple patterns. Examples of distributed RDF triple stores are 4store [53] and Hadoop distributed file system [54].

The authors in [55] investigated two types of RDF data stores, namely traditional database stores and NoSQL database stores. Figure 2.6 describes the proposed classification of RDF data stores [55].

¹⁴ <http://rdf4j.org/about/>

¹⁵ <https://jena.apache.org/>

¹⁶ <https://virtuoso.openlinksw.com/>

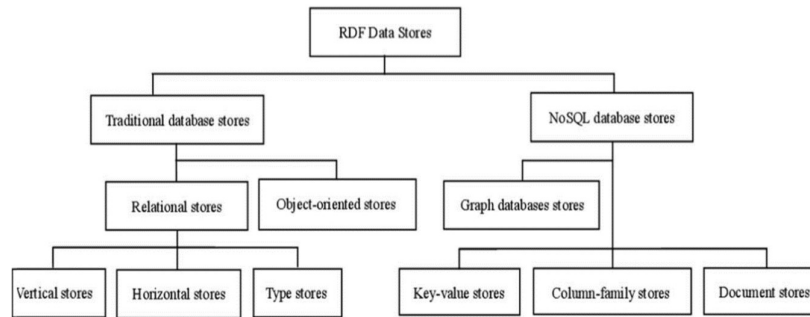


Figure 2.6 Database models which are used to store RDF data [55].

Traditional database stores. Traditional databases are classified into object-oriented databases and relational databases. RDF data in object-oriented databases are stored by modelling resources, properties and values as objects [56]. An RDF statement is graphically represented as a directed edge from the resource object over the property object ending at the value object. As for storing RDF data in a relational database, three types of RDF stores are used:

- Vertical stores: RDF triples are stored in a single relational table with a schema of three attributes (i.e. columns), namely *Subject*, *Predicate* and *Object*, which correspond to the three parts of RDF statements (triples). Accordingly, each RDF triple becomes a single tuple (i.e. row) in the relational table [50, 53]. A well-known and widely used vertical RDF store is *Sesame* triple store [50].
- Horizontal stores: these are predicate oriented stores, where *Subject – Object* relation is directly represented for each predicate of RDF triples (predicates are the column names in the relational tables).
- Type stores: also called property stores, in which one relational table is created for each RDF data type (e.g. two table to store employees and company data) and a relational table contains the properties as columns.

NoSQL (‘not only SQL (Structured Query Language)) database stores. These stores emerged as a commonly used infrastructure for handling big data. An important category of NoSQL database stores is Graph database stores. Graph databases use graphs as their data model, and a graph is used to represent a set of entities, and edges that interconnect the entities [55]. A graph database is defined as a finite, directed, edge-labelled graph [57].

2.2.6 Semantic Web Applications

In this Section, we will review two broad types of Semantic Web applications which are related to this research context: *Semantic data browsers* and *Semantic data search engines* –

we will use a semantic data browser (called MusicPinta [18] – described in Section 3.4.1) in the musical instrument domain, and a semantic data search engine (called L4All [58] – described in Section 3.4.2), for implementing our algorithms and testing our hypothesis.

2.2.6.1 Semantic Data Browsers

There have been several terms used in the literature for naming these browsers. The authors in [5, 37, 59] used the term *Semantic Web browsers*, while the authors in [9, 32, 34, 60] used the term *Linked Data browsers*. Furthermore, the term *Semantic browsers* have been used in [61] and the term *Semantic Data browsers* has been used in [17, 18]. In this research, we will use the later term and we will be calling these browsers as *Semantic data browsers*¹⁷. Semantic data browsers described in this Section will be the underpinning applications for visualisation-based and text-based data graph exploration approaches which will be described in Section 2.3. Semantic data browsers are the first generation of Semantic Web applications emerged to support users while they are exploring data graphs [62]. Similar to the traditional Web browsers (e.g. Firefox¹⁸, Google Chrome¹⁹) which allow users to follow Hyperlinks while navigating Webpages, Semantic data browsers allow users to navigate through entities in a data graph by following typed links, expressed as RDF triples [63–66].

Semantic data browsers are divided into two main types, namely: *visualization-based* and *text-based* browsers. We will review both types, and in particular we will focus on *text-based browsers* since they define the broad context of this research (i.e. our algorithm will be applied and evaluated using text-based semantic data browsers).

Visualization-based browsers. Visualization provides an important tool for exploration that leverages the human perception and analytical abilities to offer exploration trajectories for the users. Visualisation browsing, in addition to intuitiveness, focuses on the need for managing the dimensions in semantic data represented as properties, similarity and relatedness of concepts [67].

Visualization-based browsers use visual or graphic structures, such as images, maps or graphs (individually and in combinations) to represent data graphs for the users. There is a wide range of visualization-based browsing mechanisms [68], such as. *Grid-view*: displays the triples extracted as a table of *Subject-Predicate-Object*; *Tree-view*: by selecting

¹⁷ The term *Semantic browsers* is too generic, and we don't prefer to use the term *Semantic Web browser* to distinguish from the traditional Web browsers (e.g. Google Chrome). Also we don't prefer to use the term *Linked Data browsers*, because the notion of Linked Data represents principles for publishing data on the Web. Therefore, we will use the term *Semantic Data Browsers*.

¹⁸ <https://www.mozilla.org/en-US/firefox/>.

¹⁹ <https://www.google.com/chrome/index.html>.

hierarchically displayed trees, the user can browse through increasing levels of details as moving down towards tree leaves; *Graph-view*: displays the paths existing between two resources of interest in a graph form. An example of a visualization-based browsers is PolyZoom [69] which enables multi-focus exploration of maps for a user to zoom various parts of the map at the same time. A review of visualization-based semantic data browsers can be found [68].

Text-based semantic data browsers. These are the earliest type of semantic data browsers which allow users to express their needs in terms of keywords [8] or via writing SPARQL queries [70]. For instance, to search for an entity (e.g. person, movie, place, etc) in a data graph, a user enters a particular *entity name* as a string in the browser's keyword search area, and then the browser traverses the data graph to retrieve statements (i.e. RDF triples) about this entity and present the results to the user as *Attribute-Value* pairs (i.e. the text browser provides a list of facts about a *Subject* entity in the form of *Predicate-Object* pairs) [68].

We will review two text-based browsing mechanisms which are related to this research, namely *pivoting* and *faceted* browsing. On the one hand, *pivoting* browsing enables the user to start his/her exploration from a single entry point in a data graph. In our work, this entry point is the first entity of an exploration path (Chapter 6 describes the process for generating exploration paths in a data graph). On the other hand, we focus on *faceted* browsing because we will evaluate our algorithms for generating exploration paths (evaluation of exploration paths is presented in Chapter 7) using faceted text-based browser called MusicPinta (described in Section 3.4.1).

Pivoting browsing, also referred as uni-focal browsing [18], set-based browsing [71], or link-based browsing [11] is graph browsing technique that is used to help a user navigate from a set of instances of a single entity in the graph through common links [72]. Pivoting is a widely used browsing mechanism for searching the Web. It was found that around 50% Web search queries pivot around a single entity [73]. Tabulator²⁰ [5] can be considered as the first pivot-based browser. It enables the user to browse textual resources in the data graph by starting from a single resource following semantic links to other resources. Parallax [71] shows to the user a set of entities and provides a list of links for browsing linked entities further, and VisiNav [74] that provides the user with a path exploration option where the user can navigate a path along a link to browse a new set of entities

The work in [72] outlined three common design patterns for pivoting browsers, yet may impose some limitations.

²⁰ <http://dig.csail.mit.edu/2005/ajar/ajaw/tab>

- Browsing is started from a single point (class) in the data graph. In a standard pivoting browser, exploration begins with a single class entity and the user is usually provided with instances of that entity. However, this can reduce the flexibility and ability to quickly browse through entities.
- Browsing is typically supported in a unidirectional fashion (i.e. navigation is enabled only from outgoing links from instances in the current class entity). This can reduce the expressivity of the data graph, and increases users' exploration efforts.
- Exploration and domain overview absence. Exploration is regularly performed without getting familiarity with the domain, which will cause difficulties for users to re-tract exploration steps and make alternative paths.

To address these limitations, multi-pivoting browsing was suggested where users can start exploration from multiple points of interest and perform bi-directional browsing [72]. Visor²¹ [72] is an example of multi-pivoting browsing, where browsing is started by selecting multiple concepts of interest and the user is then suggested with relevant concepts, and the user can explore relationships between the selected and suggested concepts. Furthermore, the users can pivot freely from anywhere in any direction since it provides bi-directional exploration.

Faceted browsing has been seen as an effective and widely used exploration mechanism [75]. It is defined as “a session-based interactive method for query formulation (commonly over a multidimensional information space) through simple clicks that offers an overview of the result set (groups and count information), never leading to empty results sets” [76]. Faceted is an iterative process of querying, browsing and query refinement, and it eases results browsing by iterative drill-down (refinement) or roll-up (generalization) operations [76, 77]. Facets can be created at the class level (e.g. group Action Movies under a facet called ‘Action’) or at the property (i.e. relationship) level (e.g. group Movies by their director using a facet called ‘Directed By’). The main advantage of faceted browsing is that it is a highly familiar exploration paradigm used in e-Commerce applications such as Amazon²².

2.2.6.2 Semantic Data Search Engines

Semantic data search engines are applications that crawl data graphs by following RDF links between data sources, and provide expressive query capabilities [34]. Semantic data search engines are classified into two types: *user-oriented* and *application-oriented indexes* [32].

²¹ <http://visor.psi.enacting.org/>

²² <http://www.amazon.com>

User-oriented semantic data search engines. These search engines follow a similar interaction paradigm to existing Web search tools such as Google Chrome²³ and Yahoo²⁴. This type of semantic data search engines provides keyword-based search interfaces for the users and allows them to enter search keywords related to the item or topic in which the users are interested, and the users are then provided with a list of search results (usually in the form of ranked lists) [8]. In addition to the ranked lists, semantic data search engines provide richer interaction capabilities to the user, such as allowing the user to control the search results by selecting which data graphs to search, or to filter the results using vocabularies from different data sources. Falcons²⁵ [8] can be seen as the earliest semantic data search engine which enables a user to enter a concept name (i.e. class name) in a search box and then filters search results to show entities that are linked to that class via `rdf:Class` and `rdfs:subClassOf` relationships. Falcons also allows the user to filter the results using vocabularies from different data sources. Sig.ma [78] allows the user to enter a concept name in a search box, and then it will present a rich aggregate of information about the concepts. Queries can be about humans, or any other entities in the Web of Data, such as locations, name of documents, etc. Once the user is provided with aggregate information about the concepts, which include links (i.e. predicates) the user can follow these links to visualise the information about the related entities. HAWK [79] is a recent semantic data search engines for entity search over Linked Data and textual data. It takes an input query in the form of a text question and combines knowledge from Linked Data (e.g. DBpedia) and textual data (e.g. Wikipedia).

Application-oriented indexes. These types of semantic data search engines are developed to support applications which are built on top of data graphs (in the form of RDF data). Examples of Application-oriented indexes are Swoogle²⁶ [80], Sindice [81] and Watson [82]. These applications provide Application Programming Interfaces (APIs) through which newly developed semantic Web applications can discover RDF data on the Web. For example, Swoogle [80], helps software agents and knowledge engineers to find semantic Web knowledge encoded in RDF and OWL on the Web. It ranks the importance of semantic Web objects such as documents, concepts, ontologies and RDF graphs, and present results to users. Sindice [81] crawls the sources of the semantic Web and indexes the resources encountered in each source. It provides a simple API to semantic Web application developers to allow them to automatically locate relevant data sources so that they can integrate the data from

²³ <http://ws.nju.edu.cn/falcons/conceptsearch/>

²⁴ <https://www.google.com/chrome/browser/features.html>

²⁵ <https://yahoo.com/>

²⁶ <http://swoogle.umbc.edu/>

these sources into their applications. Watson [82] provides an access point to semantic data in order to support semantic Web application developers in exploiting distributed and heterogeneous semantic data. It supports the developer in finding, selecting, exploiting, and combining online semantic resources.

2.3 Exploration of Data Graphs

Data graphs (in the form of RDF Linked Data) provide users with access to a rich Web of interlinked data, however, this should not be accompanied by technical barriers which are difficult to overcome by layman users [67]. Therefore, approaches to support users' exploration are required. Data graph exploration approaches can be grouped into two broad categories, namely: *visualisation-based* and *text-based* exploration approaches, which correspond to the two semantic data browsers application types discussed in Section 2.2.6.1. Before we start reviewing the different exploration approaches, we will describe the difference between four terms that have been used in the literature: *browsing*, *navigating*, *searching* and *exploration*.

The term *browsing* refers to the process of viewing and navigating Web pages, one page at a time using hyperlinks [31]. Marchionini, et al [83], define browsing as: “an exploratory, information-seeking strategy that depends on serendipity”, and that “it is especially appropriate for ill-defined problems and for exploring new task domains”. According to [30, 31], browsing is appropriate in situations where the users are not certain about their information needs nor they know the appropriate search query. Navigation is similar to exploration with similar techniques. However, navigation implies that a general destination or objective is known, whereas exploration implies that the goal is not clear [9]. *Searching* on the other hand, is different from browsing in that it is the process of entering a search query (e.g. single keyword or text query) and viewing a ranked list of Webpages that matches the search query [31]. The term *exploratory search*, refers to exploration that combines browsing and searching for knowledge acquisition [2]. This is also referred as *exploration*, which is characterised to be generic, open-ended, and the goal is to learn or investigate new knowledge [2, 4].

2.3.1 Visualisation-based Approaches

The state of the art in approaches that harness visualisation as a tool for exploratory discovery and analysis of linked data graphs is presented in [67]. They use Shneiderman's [84] (*overview first, zoom and filter, then details-on-demand*) seminal visual information-seeking mantra as a guiding principle in evaluating the usability and utility of visualisation-driven

approaches. They classified state of the art visualisation approaches into three categories: (i) the role of ontologies in visualisation, (ii) visual querying and (iii) exploratory discovery.

The authors in [85] developed a generic approach (called *AffectiveGraphs*) to support data graph exploration using an interface that is pleasing for the users. The approach combined several features to support exploration for both: *layman* and *expert* users. It provides features such as interactive entity-link graph representation including links about the concept currently explored; contextual information relevant to the concept currently being explored (showing names of data property types and number of instances); show the exploratory query being generated for search, with advanced features to modify the query (for expert users). One of the main outputs from this work, is the need for an intuitive interface that facilitates browsing of large and complex data graphs for layman and experts.

Linked Data Maps [86] is an approach for representing RDF graphs as interactive, map-like visualizations. The visualisation approach presented in [86] uses Sheiderman's seminal work [84] (*overview first, zoom and filter, then details-on-demand*) to provide an interactive visualization approach which utilises cartographic metaphor to represent an RDF data graph in the form of a map. The map displays all instances of a dataset, organised in regions according to their ontological types, and then automatically generating SPARQL queries based on search and interaction with the map.

There are numerous approaches to support visual SPARQL query construction and many of these works [69, 70, 87] have similar target audience, i.e. a layman user who is unfamiliar with the domain of the data graph. These works use visualisation techniques to help layman users browse through large data graphs. For example the work in [87] introduced a graphical interface (called NITELIGHT) for semantic query construction which is based on the specification of SPARQL query language. The interface allows users to create SPARQL queries using a set of graphical notations and editing action. The state of the art approaches to support visual SPARQL query construction is presented in [88]. The focus of their work is in supporting layman users to perform exploratory querying of RDF graphs in space, time, and theme with interactive visual query construction methods [88]. The authors present three design principles to counter the challenges and evaluate them in a usability study on finding maps in a historical map repository. The design principles are: (i) *the interface should place users in control* (e.g. by offering the user with query manipulation overview and provide feedback results into user to help manipulating the query); (ii) *the interface should reduce memory load of users* (e.g. by suggesting the local space of meaningful query construction possibilities, and offering visual cues for possible actions); (iii) *the interface should be consistent* (e.g. use a consistent symbology). Although these SPARQL query construction approaches hide complexity of graph terminologies, their primarily focus on helping layman

users to generate SPARQL queries instead of focusing on the properties of data graphs to guide the exploration process of the users.

The work in [64] is such a work that manipulate the data graph properties to guide exploration with a system called Aemoo that helps users to focus on the most important bit of information about an entity first and then explore other related information. Aemoo utilises encyclopaedic knowledge patterns as relevance criteria for selecting, organising, and visualising knowledge. They are discovered by mining the linking structure of Wikipedia to build entity-centric summaries that can be exploited to help the users in exploratory search tasks. However, this approach is feasible in multi-knowledge domains that are built by humans (e.g. Wikipedia) and may not be feasible in specific domains with complex structures, because Aemoo considers only one level below the root, and does not cover different entities at different level of abstractions.

Current visualisation efforts for exploring data graphs are geared towards helping layman users explore the complex graph structures by hiding the complexity of semantic terminology from the users. However, the effectiveness of the visualisations depends on the user's ability to make sense of the graphical representation which in many cases can be rather complex. Users who are new to the domain may struggle to grasp the complexity of the knowledge presented in the visualisation. Our approach to automatically identify entities that are close to the human cognitive structures can be used as complementary to visualisation approaches to *simplify* the graph by pointing at entities that layman users can be familiar with. However, the prime focus of our approach is on augmenting text-based data browsers by offering exploration paths.

2.3.2 Text-based Approaches

The text-based exploration approaches operate on semantically augmented data (e.g. tagged content) with layout browsing trajectories using relationships in the underpinning ontologies. These are adopting techniques and knowledge from learning, HCI, personalisation and other fields to offer a more sophisticated data exploration experience to its users.

Recent developments are combining faceted and pivoting together to provide better exploration and to enable users to switch between focus entities and provide better filtering capabilities (through facets). A noteworthy variation of the pivoting approach is the use of facets for text-based data browsing of linked datasets. Faceted browsing is the main approach for exploratory search in many applications. The approach employs classification and properties features from linked datasets as a mean to offer facets and context of exploration.

gFacet [89], and tFacet [90], are early efforts in this area. gFacet²⁷ [89] supports relational navigation through the construction of complex facet graphs representing different types of entities and relations between them. It supports pivoting as well since it allows moving from the current path and following another direction. tFacet [90] allows hierarchical faceted exploration of data graphs for layman users. The process involves selecting a class and then presenting objects of that class (called result set). Then it displays all facets that can be used to filter objects in the result set (i.e. facets are related to the direct attributes of the result set), in a tree view, and the user can select individual facets. By selecting one facet, the result set is reduced to objects related to that facet.

More recent attempts include Rhizomer²⁸ [63] which provides data exploration based on an overview, zoom and filter workflow. It combines various types of visualizations such as maps, time- lines, tree-maps, navigation menus and charts, as a way for providing flexible exploration between different classes. Facete [91] a visualization-based exploration tool that offers faceted filtering functionalities. It implements a data exploration paradigm based on three components: (i) faceted filtering and selection; (ii) detecting spatial information (instances) related to the selected fact; (iii) making the map display interact with data sources that contain large amount of special information. Hippalus [92] allows users to rank the facets according to preferences defined directly by the user. Voyager [93] is a mixed-initiative system that couples faceted browsing with visualization recommendation.

Although these approaches provide support for user exploration, layman users who are performing exploratory search tasks to learn or investigate new topic, can be cognitively overloaded especially when facets provide lengthy options (i.e. multiple links) for users to explore. The authors in [11] proposed Sview, a browser that utilises a link pattern-based mechanism for entity-centric exploration over Linked Data. Link patterns describe explicit and implicit relationships between entities and are used to categorise linked entities. A link pattern hierarchy is constructed using Formal Concept Analysis (FCA), and three measures are used to select the top-k patterns from the hierarchy. However, the approach lacked considering the user preference in identifying the link patterns in supporting exploratory search tasks such as learning, which would be challenging especially when the domain is not familiar to the user.

These approaches have become more sophisticated, personalising the exploration to not only hide the complexity of semantics, but also to take into consideration the user's profile and interests. An approach that explicitly targets personalisation in semantic data exploration

²⁷ <http://gfacet.semanticweb.org>

²⁸ <http://rhizomik.net/rhizomer/>

using users' interests is presented in [75]. Linked dataset concepts are dynamically categorised into upper mapping and binding exchange layer concepts using a fuzzy retrieval model. Results with the same concepts are grouped together to form categories, later used during concept-based browsing to align the exploration space to the users' interests. Recent approaches aimed to improve search efficiency over Linked Data graphs by considering user interests [94], or to diversify the user exploration paths with recommendations based on the browsing history [95]. A method for personalised access to Linked Data has been suggested in [96] based on collaborative filtering that estimates the similarity between users, and then produces resource recommendations from users with similar tastes. Similarity between users is calculated by taking into account the commonalities, the informativeness and the connectiveness of the shared resources between the users. More recently, a graph-based recommendation methodology based on a personalised PageRank algorithm has been proposed in [97]. This adopts a non-uniform personalization vector assigning different weights to different nodes to get a bias towards some nodes with user preferences. The personalisation approach in [98] allows the user to rate semantic associations represented as chains of relations that may reveal interesting and unknown connections between different types of entities, which are used to iteratively learn a personalised ranking function.

Personalization approaches suffer from the 'cold start' problem – for a reliable user model to be obtained, the user would have to spend time interacting with the system to provide sufficient information about their interests. To address this problem, conventional recommender systems uses stereotypes to provide generic recommendations for similar users. However, stereotypes have to be constructed, and this can be a difficult task for new domains represented as data graphs with many entities and paths the users can follow. Our approach exploits the structure of the data graph to identify knowledge anchor (i.e. familiar entities) that can be used as stereotypes for a generic layman user. Strictly considered our approach is not personalisation, because we do not dynamically adapt to the user's knowledge as it expands while the user browses through the graph. However, the exploration paths generated are aimed to facilitate the understanding of the graph by using anchoring entities that are likely to be familiar to generic layman users. The knowledge anchors are based solely on the data graph, and hence our approach to identify familiar entities does not suffer the cold start problem.

2.4 Identifying Key Entities in Data Graphs

The problem of identifying key entities (concepts) has been tackled by a number of research fields including: *social networks* (e.g. use node centrality to quantify the structural importance of actors in a network [99]), *Cognitive science* (e.g. identify concepts which correspond to

familiar concepts in human cognitive structures [22]) and recommending systems (e.g. recommending serendipitous entities to the user [19]). Nevertheless, in this Section we will focus on two technical approaches which are most related to the context of this research, namely *ontology summarisation* and *formal concept analysis*. We will describe these technical approaches (Sections 2.4.1 and 2.4.2) and focus on the technical approaches which adopt the Cognitive science notion of basic level objects described in Section 2.7.3, to justify the novelty of our approach for identifying knowledge anchors in a data graph (described in Chapter 4).

2.4.1 Ontology Summarisation Approaches

Ontology summarisation has been seen as an important technology to help ontology engineers to better make sense of an ontology in order to understand, reuse and build new ontologies [100–102]. It is defined as the process of distilling knowledge from an ontology in order to produce an abridged version [103]. The authors in [65] define data graph as a valid RDF Schema (RDFS) graph that includes most representative classes of the ontology, adapted to the corresponding graph instances. They argue that a good summary should be concise, yet it needs to convey enough information to enable a decent understanding of the original graph, yet provide an extensive coverage of the entire ontology [65].

The process of summarising an ontology involves identifying the key concepts in an ontology, hiding (or removing) other concepts which are not key concepts, and then linking between the key concepts [104]. Accordingly, there are two central questions for creating a valid summary are: *how to identify important entities?*, and *how to link them to provide a valid summary?*. In this Section, we will focus on the first question and we will discuss different approaches for identifying key entities.

The work in [103] aimed to identify a new method to automatically summarise an ontology based on RDF Sentence Graph. The work compared three main centrality measures for identifying key concepts, namely: degree centrality, shortest path-based centrality and eigenvector centrality. Each of the centrality measures used to identify 10 entities. semantic Web experts were invited in order to evaluate the produced RDF summaries. The outcomes showed that degree centrality measures and eigenvector centrality gave good performances in identifying key concepts in an ontology.

An algorithm that exploits the semantics and the structure of an RDF schema and the distribution of the corresponding data/instances has been presented in [105] to produce RDF graph summaries. The proposed algorithm uses the notions of relevance based on the relative cardinality and the in/out degree centrality, and the notion of coverage of a node to produce the summaries. Relevance in terms of relative cardinality assumes that class entities with more

number of instances are more important than classes with less instances. The in-degree and out-degree centralities were used to identify the centrality of entities in the RDF schema.

The state of the art ontology summarisation approach is presented in [65]. The approach exploits the structure and the semantic relationships of a data graph to identify the most important entities using the notion of relevance, and then select edges connecting the entities by maximising either locally or globally the importance of the selected edges. The notion of relevance is based on the relative cardinality (i.e. judging the importance of an entity from the instances it contains) and the in/out degree centrality (i.e. the number and type of the incoming and outgoing edges) of an entity.

2.4.2 Formal Concept Analysis Based Approaches

Formal Concept Analysis (FCA) was introduced Rudolf Wille as a mathematical theory for formalising concepts and conceptual thinking [106]. It is a method for analysis of object-attribute data tables, where data is represented as a table describing objects, attributes and binary relationships between the objects and the properties [106]. Formal Concept Analysis (FCA) has been applied in different application areas [107, 108] such as ontology engineering, Web mining and search, Software engineering, and computer-aided learning. In this Section, we focus on the two most relevant fields to this work: *Web mining* and *ontology engineering*.

In *Web mining*, FCA based approaches have been used to improve the quality of search results presented to the end users. For example, the work in [109] developed a personalised domain-specific search system that uses logs of keywords and Web pages previously entered visited by other persons to build a concept lattice. After that, the new queries provided by the users are matched against the lattice and relevant web pages belonging to the most relevant concept are proposed to the user. More recently, FCA has been applied to construct a link pattern hierarchy to organize semantic links between entities in a data graph. The approach includes two steps: constructing a formal context and then generate a link pattern hierarchy. Constructing a formal context can be either single-entity oriented or entity-set oriented [9]. In the single-entity oriented approach there is one focus entity (e.g. Steven Spielberg), set of entities – FCA objects (e.g. movies related to Steven Spielberg), and a set of semantic links between the focus entity and the set of entities (e.g. director, producer). In the entity-set oriented approach there is more than one focus entity (e.g. several movie names), several set of entities (e.g. people related to these movies) linked via semantic links (e.g. actors, directors). Using a formal context, a concept lattice representing a link pattern hierarchy is generated.

In *ontology engineering*, FCA has been used in two topics: ontology construction and ontology refinement. On the one hand, ontology construction approach concern how

ontologies can be constructed in an efficient manner. FCA has been used to extract concepts hierarchies from domains (mostly represented as textual data) to design ontologies (i.e. extract classes of an ontology linked via subsumption relationships). For example, the work in [110] used FCA to construct ad hoc ontologies to help the user to better understand the research domain. On the other hand, ontology refinement approaches concern improving the quality of an ontology. For example the work in [111] described OntoComp, an approach for supporting ontology engineers to check whether an OWL ontology covers all relevant concepts in a domain, and supports the engineers to refine (extend) the ontology with missing concepts. The approach uses FCA to get the complete concept lattice that covers the overall domain and then using this lattice to successive questions to the ontology engineer. More recently, the work in [112] used FCA to check the suitability of an ontology for RDF dataset. The approach uses FCA to build a lattice for an RDF graph and then compare the graph with an ontology schema using the notion of lattice annotation, which associates concepts of the lattice with classes of an ontology.

2.4.3 Approaches that adopt Basic Level Objects

The technical approaches that are most relevant to this research refer to the adoption of the notion of BLO in ontology summarization [103] and formal concept analysis (FCA) [106] techniques.

Ontology summarisation. The closest ontology summarisation approaches that are relevant to the context of our work relates to extracting key concepts as the best representatives of ontology are in [113] and [114]. The work in [113] highlights the value of cognitive natural categories for identifying key concepts in an ontology to aid ontology engineers to better understand the ontology and quickly judge the suitability of an ontology in a knowledge engineering project. The suggested approach aims to identify automatically the key concepts in an ontology that match as much as possible those produced by human experts, by combining cognitive principles, lexical and topological measurements such as density [113]. Regarding the Cognitive principle, the work in [113] applies a name simplicity approach, which is inspired by the Cognitive science notion of BLO [22] as a way to filter entities that have lengthy labels and keep those with simple labels for the ontology summary. The name simplicity approach favours concepts with simple names (i.e. short label names), while penalising complex and compounded names [113]. This is based on assumption that basic categories usually will have simple names (e.g. *Guitar*, *Piano*, *Dog*, and *Chair*) that are easily remembered by layman users. According to this approach, the name simplicity value of such basic categories is 1 (the label is made of only one word), and this value decreases based on the number of words in the label, and the value produced for each concept is in the range of (1,0). Another interesting approach about constructing ontologies from basic level

concepts has been presented in [114]. This work has utilised basic level concepts to extract ontologies from collaborative tags, where tags are given by users to annotate a resource and describe its characteristics. The hypothesis is that ontologies build under the cognitive psychology theory of basic level concepts are more consistent with human thinking and hence easily reused. The work in [114] proposes an algorithm for constructing an ontology using basic level concepts. Such concepts are represented by common tags in the domain context (e.g. the tags ‘software’ and ‘engineering’, together represents a concept about ‘software engineering’). The tags of a concept are inherited by its members and a concept. In this regard, a metric based on the category utility is proposed to identify basic level concepts from tags. In this metric, a concept is the abstraction of a category of instances, where tagged resources are considered to be the instances in the ontology.

Formal Concept Analysis. The two most related psychological approaches to basic level concepts have been formally defined in [23, 115]. The work in [115] demonstrated that the basic level phenomenon may be utilised to select important formal concepts. The suggested approach to identify basic formal concepts considers the cohesion of a formal concept and define three properties required for a formal concept to be identified as a basic level formal concept: (i) has a high cohesion between members; (ii) has larger cohesion than its upper concepts; and (iii) has slightly smaller cohesion than its lower concepts. The cohesion of a formal concept is the average of the accumulated pair-wise similarity values between the concept’s objects based on common attributes. More recently, the work in [23] has reviewed and formalised the main existing psychological approaches to basic level concepts. Several approaches to basic level objects have been formalised with FCA [23]. The approaches utilised the validity of formal concepts to produce informative concepts capable of reducing the user’s overload from a large number of concepts supplied to the user. The three main approaches in [23] include (i) the cue validity approach which based on the formal definition of cue validity provided by Rosch et al. [22]; (ii) the category feature collocation approach which inspired by [116] and uses the so-called collocation of a category and an attribute – the product of cue validity and category validity; and (iii) the category utility approach inspired from [117].

Existing work on ontology summarisation and FCA utilises basic level object with the aim of identifying key concepts in an ontology to help experts to examine and reengineer the ontology, or to reduce significantly their overload due to the usually large number of all concepts in the data supplied by ordinary FCA. However, these approaches lacks applying the formal definitions of basic level objects and cue validity described in [22, 118] in the context of a data graph. In our work, we apply the notion of basic level objects in a data graph to identify *concepts which are likely to be familiar to users who are not domain experts*. We

are operationalizing these definitions by developing several algorithms with the corresponding algorithms for identifying knowledge anchors in a data graph. Our work is the first of its kind in utilizing Rosch's seminal cognitive science work [22] in the context of data exploration of data graphs. The formal framework that maps Rosch's definition of basic level objects and cue validity to data graphs is one of the key contribution of the work presented in this work. Furthermore, we show (in Chapter 6) how to utilise such concepts for generating exploration paths to facilitate the expansion of users' domain knowledge.

2.5 Generating Paths in Data Graphs

In this Section, we will review different approaches used to generate paths in information spaces to justify the contribution of our approach for generating exploration paths in data graphs for knowledge expansion using knowledge anchors.

The problem of generating exploration paths in information spaces has been tackled by several research fields such as education, recommending systems, e-Learning, and data graphs. In its simplest definition, a path is an alternating sequence of nodes and links in an information space, often represented as a sequence of just nodes [119]. An information space can be seen as heterogenous graph containing different types of edges and vertices [120]. Approaches for generating paths in information spaces can be grouped into two categories. The first category concerns in generating paths from a given start entity (e.g. search keyword) and the Second category focuses on generating paths using two or more entities in the information space (e.g. find the relationships between two actors in the movie domain or find relationships between combination drug therapy regimens commonly used to treat a particular disease). However, most attention has been paid to the later type and aimed to discover connections ("associations") between entities by exploring possible paths that link the two entities.

In virtual environments, an approach for generating navigation paths for virtual tour guides been proposed in [121]. The hypothesis was virtual navigation paths can help the users to familiarise themselves with the virtual environment and understand the meanings of its virtual objects. A virtual navigation path is created by linking several navigation landmarks (i.e. objects in the virtual environment) in a well-defined order. The selection of landmarks can be in a freehand-mode where the landmarks are freely selected by the path designer, or via a grid-mode where the path designer specifies the navigation landmarks by selecting a specific sphere of a grid layout. The transition from one landmark to another one will be done by means shortest path.

In recommending systems, path construction has been used to provide serendipitous recommendations between two entities. For example, the work in [122] aimed to generate serendipitous recommendations for the users using mobile applications installed on the user's phone. The main intuition behind the method is that, if there exists a path connecting two applications on a user's phone, then the applications though this path which are not already downloaded by the user's mobile, are good candidates for serendipitous recommendations.

In e-Learning, there have been several definitions for learning paths. In [123], a learning path (also called a curriculum sequence) comprises steps for guiding a student to effectively build up knowledge and skills in an online environment. A learning path of good quality is a sequence of course modules arranged in such a way that can satisfy most/all the knowledge requirements of the involved course modules [124]. According to [125], learning path in online learning systems refers to a sequence of learning objects which are designated to help the students in improving their knowledge or skill in particular subjects or degree courses. The work in [126] aimed to construct learning paths that can help individual learners reduce cognitive overload and disorientation. The approach used ontologies as structured knowledge representations to construct personalised learning paths. It generates an ontology-based concept map which clusters sequences of courses, and then use the concepts-map to generate learning path taking into account the order of prior and posterior courses. The experimental results indicated that the proposed approach can produce high quality learning paths that are likely to reduce learners' cognitive overloads during learning processes.

In education, course selection has been discussed in [127] as a decision problem for students who want to make suitable selections about their future courses. The authors presented CourseNavigator, a course exploration system that identifies possible course selection options, referred to as learning paths, for the students to meet their educational goals. For this, the authors use a learning graph, a directed graph which encodes constraints such as class scheduling and course requirements. Accordingly, the learning graph offers many options for students to follow. To address this challenge, CourseNavigator identifies all possible learning paths for students, and then allows the student to control his/her learning path through a ranking function (e.g. find shortest path, find most reliable path).

In data graphs, the notion of path queries has been used in [128] for constructing paths in graph databases. The work used queries based on regular expressions (i.e. regular words) used to indicate the start and end entities of paths in the graph. For example, in a geographical graph database representing neighbourhoods (i.e. places) as entities and transport facilities (e.g. Bus, Tram) as edges. Then a user who writes a simple query such as "I need to go from Place *a* to Place *b*", then the user will be provided with different transportations facilities

going through different routes (paths) starting from Place *a* to reach the destination Place *b*. Another notion that is widely used in the context of data graph is *property path*. The notion of *property path* (also called property sequence) was used by the W3C to define possible routes between two entities in a data graph. A trivial case of a property path is a triple pattern (i.e. property path of length 1). Property paths have been used in [129] to capture paths in RDF data graph as a sequence of directed edges (i.e. properties). These paths have been used to identify associations between entities in data graphs. An association from entity *a* to entity *b* comprises the labels of the entities and edges [7]. However, in data graphs there are usually high numbers of associations (i.e. possible paths) between entities, and hence ways to refine and filter available paths. The authors in [7] aimed to address this challenge and presented a system called *Exclass*²⁹ for recommending patterns (i.e. paths) between two entities. *Exclass* identifies patterns as a sequence of classes and relationships (edges) used for recommending exploration paths. To suggest suitable patterns, the authors use the frequency of a pattern to reflect its relevance to the query and use informativeness of classes and relationships in the pattern to indicate the Informativeness of the pattern (i.e. informativeness of a pattern is obtained by adding the informativeness of all classes and relationships in the pattern). Informativeness of a class is based on the assumption that a class having fewer instances is more specific and thus more informative. The idea is similar to relationships (i.e. an edge) except that informativeness is considered to the start (*Subject*) entity and target (*Object*) entities of a relationship. Furthermore, *Exclass* considers the overlap between patterns, such that patterns that are highly overlapped are redundant and this will not be recommended together (i.e. overlap used to filter the recommended patterns). *Relfinder*³⁰ [130] is another approach for helping users to get an overview of how two entities are associated together. It reveals all possible paths between the two entities in the RDF graph, which cause high cognitive load for users. Furthermore, one of its usage restrictions for lay users is that the user must supply valid entry points, a SPARQL query endpoint and the repositories to query.

Although a significant amount of work tried to address the problem of supporting users' exploration through paths in data graphs, a solution that considers guiding layman users through paths that will expand their domain knowledge is still missing. None of the approaches outlined above investigates the user's familiarity with the domain, which is the main focus of the approach we present in this research for generating exploration paths, where familiarity is related to understanding and knowledge expansion in the domain. Our uniqueness is the explicit consideration of knowledge utility of exploration paths in data

²⁹ <http://ws.nju.edu.cn/exclass/>

³⁰ <http://relfinder.dbpedia.org>

graphs. This is crucial for the usability of semantic exploration applications, especially when the users are not domain experts. Knowledge utility approximates, to what extent a user expand his/her domain knowledge, while exploring through a path in a data graph. Furthermore, the above approaches lack applying the subsumption theory for meaning learning [21] in the context of a data graph. In this research, we are operationalizing the notion of basic level objects to identify familiar entities to the user that can be used as knowledge bridges to direct the user from familiar to less familiar entities in the graph. We will apply the subsumption theory for meaningful learning and offer a unique use of knowledge anchors in data graphs to generate exploration paths for knowledge expansion. This can facilitate the adoption of data graph exploration in the learning domain. It can also be useful in other applications to facilitate the exploration by users who are not familiar with the domain presented in the graph.

2.6 Data Exploration Evaluation Approaches

Evaluation of data exploration applications usually considers the exploration utility from a user's point of view or analyses the application's usability and performance (e.g. precision, recall, speed etc.) [85]. The prime focus is assessing the usability of semantic Web applications, while assessing how well the applications help the users with their data exploration tasks is still a key challenge [131].

Task driven user studies have been utilised to assess whether a data exploration application provides useful recommendations for accomplishing users exploration tasks [64]. A task driven benchmark for evaluating semantic data exploration has been presented in [131]. The benchmark presents a set of information-seeking tasks and metrics for measuring the effectiveness of completing the tasks. In the context of ontology summarisation, there are two main approaches for evaluating a user-driven ontology summary [100]: gold standard evaluation, where the quality of the summary is expressed by its similarity to a manually built ontology by domain experts, or corpus coverage evaluation, in which the quality of the ontology is represented by its appropriateness to cover the topic of a corpus. The evaluation approach used in [113] included identifying a gold standard by asking ontology engineers to select a number of concepts they considered the most representative for summarizing an ontology. The evaluation approach in [132] aimed to identify whether the simulated exploration paths over information networks are similar to those produced by human exploration.

In this research, we evaluate algorithms for identifying knowledge anchors in data graphs by comparing the algorithms' outputs versus a benchmarking set of basic level objects

identified by humans. To the best of our knowledge, there are no approaches that consider key concepts in data graphs which correspond to cognitive structures of layman users who are not domain experts. We identify such concepts in data graphs through an experimental method following Cognitive Science approaches to derive basic level objects that correspond to human cognitive structures. Furthermore, we evaluate algorithms for generating exploration paths for knowledge expansion by adopting a task-based approach. We approximate the knowledge utility of a path by adapting methods from Education to assess the users' conceptual knowledge, comparing the knowledge utility of generated exploration paths versus free exploration paths.

2.7 Theories

We will review three Cognitive science theories which are adapted in this research.

2.7.1 Bloom's Taxonomy for Approximating Knowledge Utility

The main goal of our research is to develop automatic ways to generate exploration paths that can expand a user's domain knowledge. Our work opens a new avenue which looks at the *knowledge utility* – expanding one's domain knowledge while exploring the data graph. Knowledge utility measures to what extent the user expands his/her domain knowledge while exploring a path in a data graph of a particular domain. To approximate knowledge utility we need a reliable and applicable metric that approximates the changes in the user's cognitive knowledge after exploring a path in a data graph. However, the assessment of learning in the field of information retrieval is considered to be rare [133]. According to [133], there are two main approaches have been used to assess learning after searching a topic:

The first approach uses a set of questions to the search topic, also referred as question answering approach [134]. This approach is considered to be the most direct and abstract way for assessing learning by asking the users questions about a particular topic after completing a search session about that topic. This approach is similar to the schema activation technique used for assessing how students expand their knowledge after reading text [135]. To assess a student's knowledge of a target domain concept, the student is asked to name concepts in a schema activation test before and after reading a text. Comparing students answers before and after reading indicated the knowledge gain.

The second approach uses concept maps [136]. The users at first are asked to draw a concept map (in the form of set of nodes and links) about a topic prior their search session, search about that topic, and then re-draw the concept map of that topic. The differences between nodes and links of the concept maps drawn before and after the search session is used to assess the knowledge gain.

In our work, we are looking to an easy yet reliable way for approximating knowledge utility of an exploration path. For this, we adopt the well-known and established taxonomy for classifying and approximating knowledge introduced by Bloom (known as Bloom's taxonomy) [27], as a question answering approach to approximate the knowledge utility of an exploration path. Recent research in the field of information exploration and search has suggested Bloom's taxonomy as a reliable tool for assessing learning [4, 137]. The taxonomy identifies a set of progressively complex learning objectives that can be used to assess learning experiences over information search tasks, and offers a means of assessing the depth of learning that occurs through search [137]. The taxonomy has also been utilised to assess and support learning in the context of 'search to learn' in exploratory search tasks and applications [4].

The original Bloom taxonomy was introduced in 1956 as a tool for assessing six major cognitive categories in human cognition [138]. The six categories were Knowledge (recalling information), Comprehension (understanding and interpretation of a situation), Application (using concepts in a new application), Analysis (separating concepts into structures), Synthesis (using concepts to build a structure or a pattern), and Evaluation (making judgments about solutions). The categories were ordered from simpler (Knowledge – knowledge of a terminology and some facts about the terminology) to more complex cognitive categories (i.e. Evaluation – evaluation in terms of evidence) [27]. Hence, a person who is functioning at the one cognitive category has also mastered the lower level cognitive categories.

A revised version of Bloom's Taxonomy was presented in 2002 [27]. It changed the original names of the six major cognitive categories from nouns into verbs since learning is described as an active process [27]. The new terms are remember (retrieving relevant knowledge from long-term memory), understand (determining the meaning of instructional messages), apply (carrying out or using a procedure in a given situation), analyse (breaking material into its constituent parts and detecting how the parts relate to one another and to an overall structure or purpose), evaluate (making judgments based on criteria and standards) and create (putting elements together to form a novel, coherent whole or make an original product). Among these cognitive categories, *remember* and *understand* are the two cognitive categories that are directly related to browsing and exploration activities. The remaining categories require deeper learning activities, which usually happen outside a tool, in our case a semantic data browser or a search engine, and hence will not be considered in approximating the knowledge utility of an exploration path. The *remember* cognitive category is about retrieving relevant knowledge from the long-term memory, and includes two steps: (i) recognition (locating the knowledge) and (ii) recall (retrieving it from the memory) [27]. For example, when we see a particular musical instrument such as `Piano`, we might remember a

famous musician who plays Piano (e.g. Sergei Rachmaninoff³¹), a musical performance where Piano was played (e.g. Symphony No.2³²) or remember instruments related to Piano (e.g. Grand Piano).

The *understand* cognitive category is about determining meaning, from which the most relevant cognitive processes from the *understand* category to a semantic browser are two cognitive processes: the cognitive process *categorise* (determining that an entity belongs to a particular category – e.g. the musical instrument Guitar belongs to the String Instrument category) and the cognitive process *compare* (detecting similarities between entity – e.g. the musical instrument Folk Guitar is similar to the musical instrument Classical Guitar) [27].

Approximating knowledge utility of an exploration path. We use a schema activation technique in a question answering format for assessing how users expand their knowledge. To assess the user's knowledge of a target domain concept, the user is asked to name concepts that belongs to and are similar to the concept. The schema activation test is conducted before an exploration (i.e. pre-test) and after an exploration (i.e. post-test), using questions related to the *cognitive processes of remember, categories, and compare* of Bloom's taxonomy described in Section 2.7.1:

- **Q1 [remember]** *What comes in your mind when you hear the word X?*;
- **Q2 [categorise]** *What categories does X belong to?*;
- **Q3 [compare]** *What entities are similar to X? .*

The number of *accurate* concepts named (e.g. naming an entity with its *exact name*, or with a *parent* or with a *member* of the entity) by user before and after exploration is counted, and the *difference indicates the knowledge utility of the exploration*. For example, let us consider the following question for approximating the knowledge utility on the cognitive process *compare* about the musical instrument Biwa:

What musical instruments are similar to Biwa ?

If a user could name correctly *two* musical instruments similar to the musical instrument Biwa (Q3) before his/her exploration and then the user could name correctly *six* names of musical instruments similar to the instrument Biwa after his/her exploration, then the effect of the exploration on the cognitive process *compare* is indicated as 4 (i.e. as a result of the exploration the user learned 4 new similar musical instruments to the musical instrument

³¹ https://en.wikipedia.org/wiki/Sergei_Rachmaninoff

³² [https://en.wikipedia.org/wiki/Symphony_No._2_\(Rachmaninoff\)](https://en.wikipedia.org/wiki/Symphony_No._2_(Rachmaninoff))

Biwa). If a user named one instrument after his/her exploration (i.e. knowledge utility is -1), in such cases the knowledge utility equal is to zero.

2.7.2 Underpinning Theoretical Model for Data Graph Exploration

Based on the observations from the exploratory user study outlined above, we aimed to identify a suitable underpinning theoretical model to generate exploration paths for knowledge expansion in data graphs. Since our focus is knowledge expansion (i.e. assessing user learning), therefore we investigated two well-known learning theories, namely schema and subsumption theory for meaningful learning.

2.7.2.1 Schema Theory

Schema theory focuses on explaining how humans develop their cognitive structures (schemas) [139]. The underlying hypothesis of schema theory is that comprehension and learning of new concepts is based on relevant prior knowledge or schema [140]. This theory has been a major force in the development of reading models and had an important influence on research reading comprehension and learning [139]. The theory helped researchers and teachers to understand how knowledge is organised in memory and the role of an individual's prior knowledge in comprehension and learning while reading texts [139].

The term schema represents a cognitive structure that organises large amounts of information into a meaningful system [141]. A schema reflects a knowledge of the co-occurrence of elements, such as behaviours, features and objects that the individual has acquired through experience [142]. Rumelhart [143] highlighted several important characteristics of schemas: schemas represent knowledge rather than definitions, schemas represent knowledge at all levels of abstraction; and schemas are active (i.e. changeable) rather than being static.

According to schema theory, to learn new concepts (i.e. connect new concepts to existing schema), a person's relevant schema is first activated (called schema activation) and then modified by bringing new concepts to it (schema modification) [144]. Schema activation is described as a continuous retrieval of relevant schemas from memory, and schema modification is an application of the activated schemas in new contexts or creation of new schemas [145]. According to [146] the process of schema modification includes three main learning tasks, namely *accretion* (when an existing schema from the prior knowledge is directly used to interpret a new concept), *tuning* (when an existing schema has to be slightly changed in order to understand a new concept), and *restructuring* (when an existing schema has to be significantly modified to create a new schema with new concepts).

2.7.2.2 Subsumption Theory for Meaningful Learning

David Ausubel, a Professor of Educational Psychology, aimed to help teachers to organise and convey learning materials to students in a meaningful way [147]. He argued that each of the academic disciplines has a structure of concepts, and there is a parallel between the way this structure is organised and the way humans organise their knowledge units in their existing cognitive structures [148].

Ausubel's model of advance organisers served as a practical guide for assessing teachers in selecting, ordering and presenting new information to their students [25]. The advance organisers provide previews (usually in the form of written passages) at a higher level of abstraction than new learning materials, which are introduced to the students before bringing new concepts to. These organisers help the students to recognise what elements of new material can be meaningfully linked to relevant concepts in existing cognitive structure [26]. The underlying hypothesis of the advanced organiser model was that learning and retention of unfamiliar but meaningful (i.e. relevant to existing cognitive structures) learning material could be facilitated by the advance introduction of subsuming concepts [25].

The process of linking new learned material to pre-existing segments of knowledge in a cognitive structure is referred to as subsumption [149]. The subsumption theory for meaningful learning has been based on the premise that existing cognitive structure (i.e. individual's organization, stability, and clarity of knowledge in a particular subject matter field) is the principal factor influencing the learning and retention of new material in a meaningful way. This theory for meaningful learning postulates that the human cognitive structure is hierarchically organised with respect to levels of abstraction and inclusiveness of concepts, where abstract and familiar concepts are deliberately introduced to the user prior to bringing new concepts to learn [25, 150]. The new concepts then become incorporated (anchored) under the relevant more abstract subsuming concepts in with insightful relationships, leading to meaningful learning [21, 25].

Ausubel in [150] argues that once the basic organising concepts (i.e. knowledge anchors) are identified³³, attention can be directed towards identifying the presentation and sequential arrangement of the new subsumed content (i.e. how to present the new learning material and in what order to the learner). With this regard, Ausubel hypothesised two principles *progressive differentiation* and *integration reconciliation*. On the one hand, the *progressive differentiation* postulates that an individual organisation of the content of a

³³ According to Ausubel, subject matter experts and talented teachers are able to identify the basic organising concepts (i.e. knowledge anchors) in the subject field

particular domain consists of a hierarchical structure in which the most inclusive concepts exist at the most abstract level (apex) of the and subsume progressively less inclusive concepts. Ausubel described this process as a sphere of knowledge from regions of greater to lesser inclusiveness, each linked to the next higher step in the hierarchy through subsumption [25]. On the other hand, the integration reconciliation principle indicates that subsuming concepts explicitly indicate in what ways previously learned, related concepts in cognitive structure are either basically similar to or essentially different from the new concept.

Ausubel distinguished between four processes for meaningful learning [26], namely derivative subsumption, correlative subsumption and superordinate learning and combinatorial learning. Derivative subsumption occurs when the new learned material is an instance or example of an existing concept in the human cognitive structure (e.g. a person who learned that Vietnamese Guitar is an example of Guitar). Correlative subsumption occurs when the new learned material is a modification or elaboration of existing concepts (e.g. a person sees Acoustic Guitar with 12 strings. This will alter the person's knowledge about Guitar to include the possibility that a Guitar may have 12 strings). Superordinate learning occurs when an individual learn that learned concepts (e.g. Apple, Orange) may all be subsumed under the new term Fruit. Finally, combinatorial learning occurs when new material can't be subsumed through a subordinate relationship nor a superordinate relationship to a particular relevant idea.

2.7.2.3 Underpinning Theoretical Model Identified

Schema theory (as outlined above) provides general description of how human cognitive structures are organised and developed. Whereas, the subsumption theory for meaningful learning provides more detailed description of how knowledge anchors in a data graph can be used to introduce and learn new concepts. Furthermore, Ausubel describes two types of meaningful relationships used in the first three processes for meaningful learning (outlined above), namely superordinate and subordinate relationships. We argue that these relationships can be aligned with the subsumption relationships (e.g. `rdfs:subClassOf`) in the context of a data graph. In other words, the subsuming entities (i.e. the knowledge anchors) and the subsumable entities (e.g. subclasses of the knowledge anchors) represent class entities in the data graph that are linked via subsumption relationship `rdfs:subClassOf`. Accordingly, the subsumption process can be applied in the context of a data graph. Therefore, our work will adopt the subsumption theory for meaningful learning as the underpinning theoretical model for generating exploration paths in data graphs to facilitate knowledge expansion. The theory will be applied over the two data graph (MusicPinta and L4All) for generating exploration paths in Chapter 7.

2.7.3 Basic Level Objects in Cognitive Science

The notion of Basic Level Objects (BLO) was introduced in Cognitive Science research, illustrating that domains of concrete objects include familiar categories that exist at an inclusive level of abstraction in humans' cognitive structures (called the *basic level*), more than categories at the *superordinate* level (i.e. above the *basic level*) or categories at the *subordinate* level (i.e. below the *basic level*) [22, 118], where a human cognitive structure is hierarchically organised with respect to levels of abstraction, generality, and inclusiveness of concepts [150].

Rosch, et al. [22], define BLO as: categories that “carry the most information, possess the highest category cue validity, and are, thus, the most differentiated from one another”. Crucial for identifying basic level categories is calculating *cue validity*: “the validity of a given cue x as a predictor of a given category y (the conditional probability of y/x) increases as the frequency with which cue x is associated with category y increases and decreases as the frequency with which cue x is associated with categories other than y increases” [22].

Most people are likely to recognise and identify objects at the basic level. An example from the experimental studies from Rosch et al. [22] of a BLO in the musical instrument domain is Guitar. The BLO Guitar represents a familiar category that is neither too generic (e.g. musical instrument—root entity of musical instrument taxonomy) nor too specific (e.g. Folk Guitar—subclass of Guitar). According to Rosch et al. [22], Guitar is at a level within the musical instrument taxonomy where its members (e.g. Folk Guitar, Classical Guitar) share attributes that are different from the attributes of members (e.g. Grand Piano, Upright Piano) of another object at the same level of abstraction (i.e. at the basic level) such as Piano. This indicates that members of a BLO share many features (attributes) together, and hence they have high similarity values in terms of the feature the BLO members share. Accordingly, two approaches can be applied to identify BLO in a domain taxonomy.

- **Distinctiveness (highest cue validity).** *Identifies most differentiated category objects.* A differentiated category object has most (or all) of its cues (i.e. attributes) linked to the category's members (e.g. subclasses of the category) only, and not linked to other category objects in the domain taxonomy. In other words, each entity that is linked through an attribute (i.e. edge label in a data graph) to members of the category will have a single validity value used as a predictor of the distinctiveness of the category among other category objects in the taxonomy. The aggregation of all validity values will indicate the distinctiveness of the category object.

- **Homogeneity (highest commonality between category members).** Identifies category objects whose members have high similarity values. The higher the similarity between category members, the more likely it is that the category object is at the basic level of abstraction. This is complementary with the distinctiveness feature. A category object with high cue validity will usually have high number of entities common to its members.

Cognitive science experimental approaches for deriving BLO

Most layman users are familiar with Basic Level Objects (BLO) categories (i.e. objects at the basic level) such as the musical instruments `Guitar` and `Piano` from the research by Rosch et al. [22]. However, layman users who are not experts in the domain (e.g. musical instrument) are unlikely to recognise members of the BLO categories (e.g. `Folk Guitar` member of the BLO `Guitar` and `Grand Piano` member of the BLO `Piano`) at the *subordinate level* (i.e. below the basic level) and name them with their exact names (i.e. layman users are unlikely to see an image of the instrument `Folk Guitar` and name it as ‘Folk Guitar’ or see the image of the instrument `Grand Piano` and name it as ‘Grand Piano’). Instead, layman users may consider such objects equivalent to the BLO categories at the *basic level* (i.e. name `Folk Guitar` as ‘Guitar’ and name `Grand Piano` as ‘Piano’) rather than naming them as ‘Musical Instrument’ at the *superordinate level* (i.e. above the basic level).

Experimental studies in Cognitive science have shown that objects in domain taxonomies such as `Musical instrument`, `Fruit`, `Vehicle` and `Furniture` taxonomies can be identified by layman users through different modalities [22, 118, 151, 152]. For example, `Guitar` in the musical instrument domain can be identified by its *sound*, its *image*, or simply by reading its *text name* (i.e. label) ‘Guitar’. Several experimental studies have been conducted in Cognitive science considering different modalities to identify BLO. For instance, the experimental study in [151] investigated how accurately listeners who are not familiar with the musical instrument domain, identify visual instruments when they hear the sounds of those instruments. In this experimental study, children and adults were listening to passages of musical compositions of different musical instruments and the participants were asked to select images of musical instruments they thought were producing the sound [151].

Rosch et al. [22] conducted several experimental studies comprising *free-naming tasks* testing the hypothesis that *object names at the basic level should be the names by which objects are most generally designated by adults*. In a free-naming task, concrete objects in a domain taxonomy are shown to a participant as a series of images in fixed portions of times, and the participant is asked to identify the names of the objects shown in the images as quickly

as possible. Three types of packets of images were shown to the participants [22]: those in which one picture from each superordinate category appeared; one in which one image from each basic level category appeared; and one in which all images appeared in an envelope. The selection of object names used in the free-naming tasks in [22] was based on the population of categories of concrete nouns in common use in English. Every noun with a word frequency of 10 or greater from a sample of written English [153] was selected as a basic level object. A superordinate category was considered in common use if at least four of its members met this criterion. The participants overwhelmingly used names at the basic level while naming objects in the images [22]. The experimental study presented in [152] has utilised free-naming tasks (similar to the approach in Rosch et al. [22]) to identify BLO in four domains (Fruit, Vegetable, Clothing, and Furniture). For each domain, three categories at the basic level were chosen plus three category members at the subordinate level were chosen. The stimuli included 24 images of objects drawn by professionals. A list of 28 words that included the names of all objects (4 for the domain names, 12 for categories at the basic level, and 12 objects for category members at the subordinate level). Participants were familiarised with the list of words by asking them to read it, and then they were shown images and asked to name the objects in the images.

In the experimental studies outlined above, *accuracy* and *frequency* were considered to identify BLO. *Accuracy* refers to naming an entity correctly by the user. It considers whether a user names an entity with its *exact name*, or with a *parent* (superclass) or with a *category member* (subclass) of the entity. *Frequency* indicates *how many times a particular category was accurately named by different users*. In the example of Guitar, when participants were shown members of Guitar such as Folk Guitar, Classical Guitar in a packet of images, they named them with their parent Guitar (Guitar is seen as accurate name) at the *basic level* more frequently than with names at the *superordinate level* (e.g. Musical instrument) or with their exact names (e.g. Folk Guitar, Classical Guitar) at the subordinate level.

However, Cognitive science experimental studies for identifying BLO cannot be applied directly in the context of a data graph. The principal difference is that we need to constrain the human cognitive structures upon the data graph, as opposed to using a bag of words from popular dictionaries as in [22]. This is because a data graph presents a lesser number of concepts from a domain, which belong to the graph scope, and there can be concepts that have been omitted, or not sufficiently presented. Moreover, the Cognitive science studies included concrete domains where images of the objects could be shown to participants. Whereas, many semantic web applications utilise data graphs which include more abstract concepts for which images cannot be reliably shown to users (e.g. medical

illnesses, environmental concepts, professions). We adopt the Cognitive science experimental approach for deriving human Basic Level Objects over a data graph (BLO_{DG}) to take into account the domain coverage of a data graph, which is applicable to any domain presented with a data graph.

2.8 Summary

This Chapter provided background information to present the research context for this thesis. Three main research fields were examined in detail focusing on approaches for: (i) *exploration of data graphs*, (ii) *identifying key entities in data graphs*, and (iii) *generating exploration paths in data graphs*. State of the art approaches were identified and used to justify our research contributions on these areas.

Exploration of data graphs: we have reviewed different exploration approaches for supporting user exploration over data graph, including visualisation and text-based approaches. Our work adds to data graph exploration approaches by opening a new avenue which looks at the knowledge utility – expanding one’s domain knowledge while exploring the data graph.

Identifying key entities in data graphs: we have discussed state of art technical approaches in the fields of ontology summarization and formal concept analysis which utilise basic level objects for identifying key concepts to support domain experts to better understand and reuse an ontology or a formal context, respectively. However, these approaches do not apply the formal definitions of basic level objects and cue validity in the context of a data graph. In this work, we apply the notion of basic level objects in a data graph to identify *concepts which are likely to be familiar to users who are not domain experts*.

Generating exploration paths in data graphs: we have reviewed approaches for generating paths in information spaces in different research fields. The path generation approaches were mainly based on user’s preferences (e.g. approaches allow users to control their paths by selecting interesting entities, or present suggested queries and select desired query), based on expert’s design (e.g. educational tutors design learning paths of courses for their students) or based on the structure of the information space (e.g. use query paths, use informative entities in a path, or identify shortest path between two entities). To the best of our knowledge, this research is the first work on generating exploration paths for knowledge expansion using familiar entities in human cognitive, used as knowledge anchors to subsume and learn new concepts. We will adopt the subsumption theory for meaningful learning and offer a unique use of knowledge anchors in a data graph to support the user’s exploration through paths for knowledge expansion.

Chapter 3

Research Methodology and Scoping

3.1 Introduction

We started this research by building on earlier work investigating obstacles related to users' exploration of data graphs. It was shown that while exploring a data graph in unfamiliar (or partially familiar) domains, layman users *serendipitously* learn new things in the domain that they were unaware of [17, 18]. However, not all exploration paths are beneficial for knowledge expansion. Therefore, ways for influencing the user's exploration pointing at paths that can expand the user's knowledge can be beneficial for learning.

Consequently, this research focuses on *knowledge utility* – expanding the user's domain knowledge while exploring a data graph. Our ultimate goal is to develop an automatic approach for generating exploration paths for knowledge expansion in data graphs. In this Chapter, we will present the methodology that underpins the overall research and addresses the research questions. We will also scope the research using a small scale exploratory user study with 12 participants. The study outcomes pointed us to adapt the Ausubel's subsumption theory for meaningful learning (described in Section 2.7.2.2) as the underpinning model for generating exploration paths for knowledge expansion.

Next in this Chapter, we will give an overview of the research methodology (Section 3.2). Then in Section 3.3 we will provide main definitions that will be used throughout the thesis. Section 3.4 will describe two different application contexts that will be used for developing and examining the algorithms developed in this thesis. After that in Section 3.5 we will present an exploratory user study to scope the research, leading to our underpinning theoretical model for generating exploration paths in a data graph. Finally, in Section 3.6 we will provide a summary of the work presented in this Chapter.

3.2 Research Methodology

Figure 3.1 outlines the overall research methodology of this thesis. It illustrates three main phases that depict the overall research and answer the research questions.

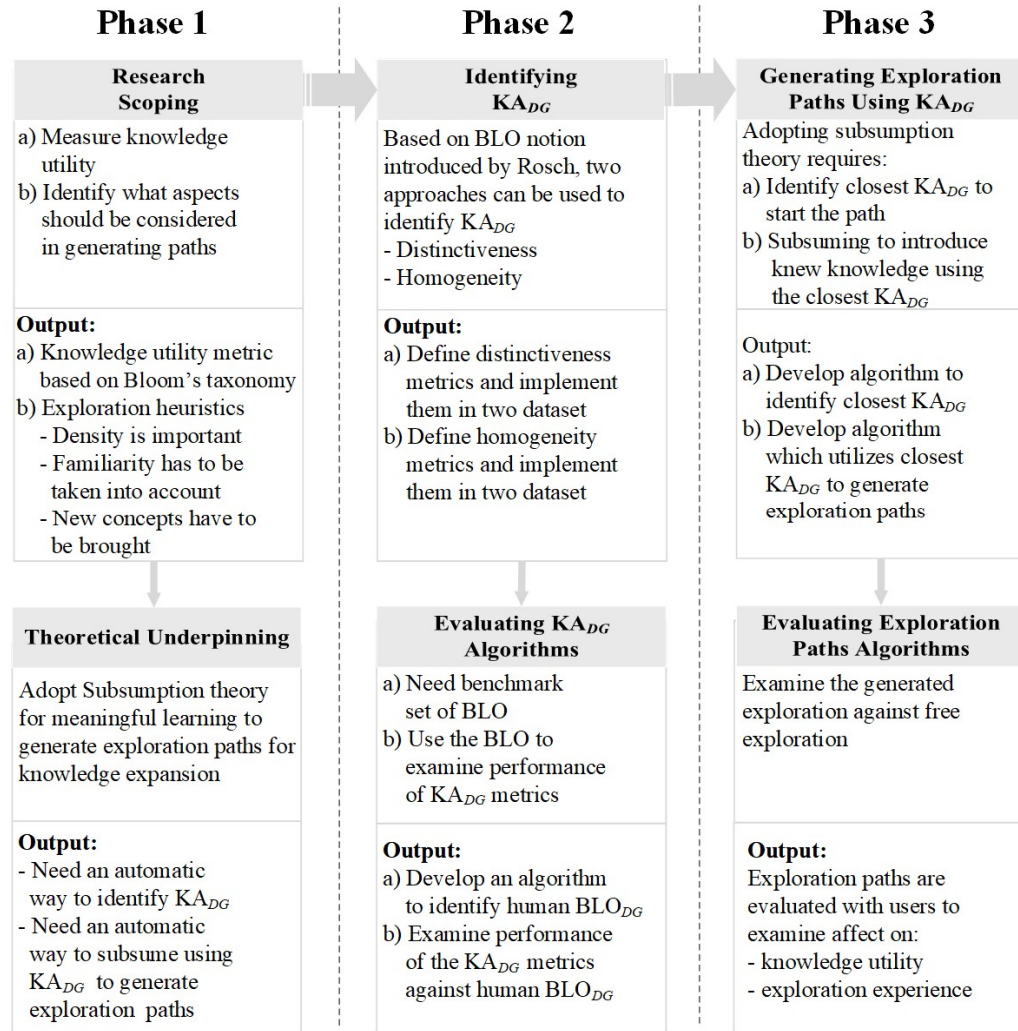


Figure 3.1 Description of the overall research methodology

Phase 1: Research Scoping (this Chapter). The work in this phase helped us to identify the main research goal and formulate the research questions. In this phase, we conducted a small scale exploratory user study with 12 participants examining exploration strategies through a data graph (reported in Section 3.5) which helped us to identify (i) a measure for knowledge utility based on Bloom's taxonomy [27] (described in Section 2.7.1), and (ii) exploration heuristics about designing exploration paths for knowledge expansion. It was shown that density of entities in the data graph is important and can be used as underpinning strategy for exploration. It was also shown that users' familiarity in the domain is important and has to be taken into account. Users were able to learn more things when they explored new entities linked to familiar entities they know.

The outcomes from the exploratory user study indicated that learning occurred when users saw familiar things and then expanded to new knowledge. This pointed us to adapt

Ausubel's subsumption theory for meaningful learning [21] as the underpinning theoretical model for generating exploration paths to facilitate knowledge expansion. However, to adopt this theory we need two things: (i) automatic approaches to identify knowledge anchors in a data graph KA_{DG} (Phase 2), and (ii) automatic approach that uses KA_{DG} to generate exploration paths that can facilitate the expansion of users' domain knowledge (Phase 3).

Phase 2: Identifying KA_{DG} (Chapters 4 and 5). The work in this phase addresses *RQ1* outlined in Chapter 1. To identify KA_{DG} , we utilise Rosch's notion of basic level objects (BLO) [22] in the context of data graphs. This notion illustrates that domains of concrete objects include familiar categories that exist at an inclusive level of abstraction in humans' cognitive structures (called the *basic level*), more than categories at the *superordinate level* (i.e. above the *basic level*) or categories at the *subordinate level* (i.e. below the *basic level*) [22, 118]. We devise a formal framework that maps Rosch's definitions of BLO and cue validity to data graphs, developing two groups of metrics and the corresponding algorithms for identifying KA_{DG} (Chapter 4): (i) *distinctiveness* metrics, where we adapt metrics from Formal Concept Analysis (FCA) [23] to identify differentiated categories whose members share attributes that are not linked to members of other categories, and (ii) *homogeneity* metrics where we apply set-based similarity metrics [24] to identify categories whose category members share many attributes together.

To evaluate the KA_{DG} algorithms, we compare human Basic Level Objects in a data graph (BLO_{DG}) that represent familiar concepts in human cognitive structures with the automatically derived KA_{DG} . To identify human BLO_{DG} , we present a systematic approach that adapts experimental methods from Cognitive science to derive human BLO_{DG} underpinned by a data graph. The approach considers two ways for deriving human BLO_{DG} , namely free-naming (using images) and category verification (using abstract text labels). We use the proposed approach to evaluate KA_{DG} algorithm in the two data graphs (MusicPinta and L4All) from two domains (musical instrument and career), respectively (Chapter 5).

Phase 3: Generating exploration paths using KA_{DG} (Chapters 6 and 7). The work in this phase addresses *RQ2* outlined in Chapter 1. We develop an automatic approach underpinned by the subsumption theory for meaningful learning [21] to generate exploration paths for knowledge expansion (Chapter 4). However, adopting this theory brings forth two challenges: (i) *how to find the closest knowledge anchor to the first entity of an exploration path?* and (ii) *how to use the closest knowledge anchor to subsume new entities for generating an exploration path.* To address the First challenge, we develop an algorithm which uses semantic similarity to identify the closest knowledge anchor to the first entity. To address the Second challenge, we define a subsumption algorithm which uses the closest knowledge

anchor to subsume (i.e. introduce) new knowledge to users. An exploration path consists of a set of transition narrative, where each transition has *Source* and *Target* entities and a script between the entities. A controlled task-driven which examines the effectiveness of the generated exploration paths to increase the users' knowledge compared to free exploration, is presented in Chapter 8. The evaluation shows that users who followed the generated exploration paths have expanded their knowledge more than users who freely explored the data graph.

3.3 Formal Definitions

In this Section, we provide the main definitions that will be used in the formal description of context and algorithms throughout the thesis.

Data graphs define the broad context of this research. They are built using traditional Web standards (e.g. Uniform Resource Identifiers (URIs) and Hypertext Transfer Protocol (HTTP) and use a common data graph model, the Resource Description Framework (RDF). RDF describes entities and attributes (edges) in the data graph, represented as RDF statements. Each statement is a triple of the form $\langle \textit{Subject} - \textit{Predicate} - \textit{Object} \rangle$ [32]. The *Subject* and *Predicate* denote entities in the data graph. An *Object* is either a URI or a string (literal). Each *Predicate* URI denotes a directed attribute (edge) with *Subject* as a source entity and *Object* as a target entity.

Definition 1 [Data graph]. Formally, a data graph is a labelled directed graph $DG = \langle V, E, T \rangle$, depicting a set of RDF triples where:

- $V = \{v_1, v_2, \dots, v_n\}$ is a finite set of entities;
- $E = \{e_1, e_2, \dots, e_m\}$ is a finite set of edge labels;
- $T = \{t_1, t_2, \dots, t_k\}$ is a finite set of triples where each triple is a proposition in the form of $\langle v_u, e_i, v_o \rangle$ with $v_u, v_o \in V$, where v_u is the *Subject* (the source entity) and v_o is the *Object* (the target entity); and $e_i \in E$ is the *Predicate* (edge label).

In our analysis of data graphs, the set of entities V will mainly consist of the concepts of the ontology and can also include individual objects (notable instances of certain concepts). The edge labels will correspond to semantic relationships between concepts and individual objects. These labels will always include the subsumption relationship `rdfs:subClassOf` and may also include the `rdf:type` relationship. For a given entity v_i , we will be interested primarily in its direct and inferred subclasses, and instances.

The set of entities V can be divided further by using the subsumption relationship `rdfs:subClassOf` (denoted as \subseteq) and following its transitivity inference. This includes:

- Root entity (r) which is superclass for all entities in the domain;
- Category entities ($C \subseteq V$) which is the set of all inner entities (other than the root entity r), that have at least one subclass, and may also include some individual objects (notable instances of certain concepts);
- Leaf entities ($L \subseteq V$) which is the set of entities that have no subclasses, and may have one or more individuals.

Starting from the root entity r , the class hierarchy in a data graph is the set of all entities linked via the subsumption relationship `rdfs:subClassOf`. The set of entities in the class hierarchy include the root entity r , Category entities C and Leaf entities L .

The set of edge labels E is divided further considering two types of relationships:

- Hierarchical relationships (H) is a set of subsumption relationships between the *Subject* and *Object* entities in the corresponding triples.
- Domain-specific relationships (D) represent relevant links in the domain, other than hierarchical links, e.g. in a Music domain, instruments used in the same *performance* are related.

Definition 2 [Data Graph Trajectory]. A trajectory J in a data graph $DG = \langle V, E, T \rangle$ is defined as a sequence of entities and edge labels within the data graph in the form of $J = \langle v_1, e_1, v_2, \dots, v_n, e_n, v_{n+1} \rangle$, where:

- $v_i \in V, i = 1, \dots, n + 1$;
- $e_j \in E, j = 1, \dots, n$;
- v_1 and v_{n+1} are the first and the last entities of the data graph trajectory J , respectively;
- n is the length of the data graph trajectory J .

Definition 3 [Entity depth]. The depth of an entity $v \in C \cup L$ is the length of the shortest data graph trajectory from the entity v to the root entity r in the class hierarchy of the data graph.

Definition 4 [Exploration Path]. An exploration path P in a data graph DG is defined as a sequence of finite set of transition narratives generated in the form of $P = \langle \langle v_1, n_1, v_2 \rangle, \langle v_2, n_2, v_3 \rangle, \dots, \langle v_m, n_m, v_{m+1} \rangle \rangle$, where:

- $v_i \in V, i = 1, \dots, m+1$;
- v_1 and v_{m+1} are the first and last entities of the exploration path P , respectively;
- m is the length of the exploration path P ;
- $n_i, i = 1, \dots, m$ is a *text string* that represents a narrative script;
- $\langle v_i, n_i, v_{i+1} \rangle$ presents a transition from v_i to v_{i+1} , which is enabled by the narrative script n_i . Note that an exploration path P is different from a data graph trajectory J in that v_i and v_{i+1} in P may not be directly linked via an edge label, i.e. the transition from v_i to v_{i+1} in P can be either via direct link, an edge, or through an implicit link, a trajectory.

Our ultimate goal is to provide an automatic way, starting from an entity $v \in V$, referred as first entity of an exploration path, and a data graph $DG = \langle V, E, T \rangle$ to generate an exploration path P in DG . The definitions provided here will be used in the definitions of the metrics and algorithms in Chapters 4-6.

3.4 Application Context

We utilise two application contexts for data graph exploration - semantic browsing and semantic search for implementing our algorithms and testing our hypothesis. Semantic data browsers operate on semantically tagged content and present browsing trajectories using relationships in the underpinning ontologies [18, 62], supporting uncertain or complex information needs [7]. They enable the users to initiate a data exploration session from a single entry point in the graph and move through entities by following RDF links [18]. Semantic data search engines [34] allow the users to enter search queries through keyword-based search interfaces and provide the users with a list of search results (usually in the form of ranked recommendations) obtained by using queries automatically generated by the system [8].

Semantic data browser in the musical instrument domain, called MusicPinta [18]. MusicPinta enables users to navigate through musical instruments extracted from DBpedia, and get information about these instruments together with musical performances and artists using these instruments. MusicPinta provides context for studying the Cognitive science notion of basic level object (BLO) in a concrete domain, as users can see images of musical instruments (as in [22, 151]). which illustrates that domain taxonomies (i.e. the subsumption class hierarchy of entities linked via the subsumption relationship `rdfs:subClassOf` in a data graph) are hierarchal organised such that there exists one level of abstraction called the basic level where most familiar concepts exist.

Semantic search engine is represented in this thesis in career guidance domain, called L4All [58]. L4All is a proprietary semantic search application which enables learners to explore various career options to plan their career progression [58]. The career domain provides suitable context for experimentation due to the richness of its ontological structures and the fact that the identification of knowledge anchors can facilitate users' exploration of such structures. L4All provides abstract domain for studying BLO, as it enables users to read entity names (labels), rather than images. We describe both application contexts. They are external systems developed outside this thesis. We used the corresponding data graphs and application context to run our algorithms and conduct corresponding evaluation strategies.

3.4.1 MusicPinta

MusicPinta provides a uni-focal interface for users to navigate through musical instrument information extracted from various semantic databases [18]. The MusicPinta data graph comprise datasets extracted from the following resources:

- *DBpedia*³⁴: for musical instruments and artists. This dataset is extracted from dbpedia.org/sparql³⁵ using CONSTRUCT queries. These queries along with a programming wrapper and additional coding are made available as open source at the [sourceforge](https://sourceforge.net/p/pinta/code/38/tree/)³⁶.
- *DBTune*³⁷: for music-related structured data made available by the DBTune.org in linked data fashion. Among the datasets on DBTune.org we utilise: (i) Jamendo which is a large repository of Creative Commons licensed music; (ii) Megatune is an independent music label; and (iii) MusicBrainz is a community-maintained open source encyclopaedia of music information.

The dataset coming from DBTune.org (such as MusicBrainz, Jamendo and Megatunes) already contains the `owl:sameAs` links between them for linking same entities. The `owl:sameAs` links provided by DBpedia are used to link MusicBrainz and DBpedia datasets. In this way, DBpedia is linked to the rest of the datasets from DBTune.org, enabling exploration via rich interconnected datasets. All datasets are available as a linked RDF data graph and the Music ontology³⁸ is the ontology used as the schema to interlink them. MusicPinta dataset can be found here³⁹. The music ontology provides sufficient class

³⁴ <http://dbpedia.org/About>

³⁵ <http://live.dbpedia.org/sparql>

³⁶ <http://sourceforge.net/p/pinta/code/38/tree/>

³⁷ <http://dbtune.org/>

³⁸ <http://musicontology.com/>

³⁹ <https://doi.org/10.5518/304>

hierarchy for traversing the user to different areas of the graph. For instance, the class hierarchies `String Instrument` and `Wind Instrument` have depth of 7 (i.e. 7 edges from the root class in the data graph), which is considered appropriate for applying the cognitive science notion of basic level objects [22] on data graphs, as this notion states that objects within a hierarchy are classified at least three different levels of abstraction (superordinate, basic, subordinate). Table 3.1 shows the main characteristics of the MusicPinta data graph. MusicPinta dataset can be found here

Table 3.1. Main characteristics of MusicPinta data graph.

The data graph includes five class hierarchies. Each class hierarchy has number of classes linked via the subsumption relationship `rdfs:subClassOf` (e.g. there are 151 classes in the `String Instrument` class hierarchy). DBpedia categories are linked to classes in a hierarchy via the `dcterms:subject` relationship, and classes are linked via the domain-specific relationship `MusicOntology:instrument` to musical performances. The depth of each class hierarchy is the maximum depth value for of the entities in that class hierarchy.

Class hierarchy	No. of classes	Depth	No. of DBpedia categories	No. of music performances
<code>String Instrument</code>	151	7	255	348
<code>Wind Instrument</code>	108	7	161	1539
<code>Percussion Instrument</code>	82	5	182	127
<code>Electronic Instrument</code>	16	1	7	11
<code>Other Instruments</code>	7	1	0	2

Among five class hierarchies in the MusicPinta data graph, the `String Instrument`, `Wind Instrument` and `Percussion Instrument` class hierarchies are the richest class hierarchies in terms of class classification (i.e. number of classes) and depth in the hierarchy. These three hierarchies enable adopting the notion of basic level objects (BLO) [22] which postulates that a domain taxonomy (i.e. the subsumption class hierarchy of entities linked via the subsumption relationship `rdfs:subClassOf` in a data graph) have at least three levels abstractions where objects do exist in a taxonomy, namely the *basic level*, the *superordinate level* (above basic level) and the *subordinate level* (below basic level). MusicPinta provides an adequate setup since it is fairly large data graph, yet of a manageable size for experimentation. It has 2.4M entities and 38M triple statements, taking 1.5GB physical space. Figures 3.2 - 3.4 show examples of the user interfaces in the MusicPinta for describing instrument `Harp`.

The screenshot shows the MusicPinta website interface. At the top left is the MusicPinta logo with the tagline "Social . Semantic . Music". To the right is a "Logout" link and the "dicode" logo. Below this is a navigation bar with "Home", "Semantic Search", "Contribute", and "Help". The breadcrumb trail is "Home > Semantic Search > Harp". The main heading is "Harp". Below the heading is a tabbed interface with "Description", "Features", "Relevant Information", "Reviews", and "Link History". The "Description" tab is active, showing a text box with the message "Description is extracted from Wikipedia when available." Below this is an image of a harp and a text block: "The harp is a multi-stringed instrument which has the plane of its strings positioned perpendicularly to the soundboard. Organologically, it is in the general category of chordophones (stringed instruments) and has its own sub category (the harps). All harps have a neck, resonator and strings. Some, known as frame harps, also have a pillar; those without the pillar are referred to as open harps." At the bottom of the page, there is a copyright notice for the University of Leeds, logos for DBpedia and MusicBrainz, and a link to DBTune.org.

Figure 3.2 Description page of the entity 'Harp' in MusicPinta

Descriptions of the musical instruments in MusicPinta were extracted from DBpedia using CONSTRUCT SPARQL queries. Images of the musical instruments were also extracted from DBpedia.

This screenshot shows the "Features" and "Relevant Information" tabs of the "Harp" page. The "Features" tab is active, displaying the text "Features are items extracted from the data sources and immediately related to the search term." Below this, it shows "Harp is:" followed by a button labeled "Instrument". The "Relevant Information" tab is also visible, showing "Harp belongs to:" followed by several buttons representing semantic classes: "Celtic_musical_instruments", "Composite_chordophones", "Harps", "Instrument", "Irish_musical_instruments", "Lyre", "National_symbols_of_Ireland", "Plucked_string_instruments", and "String_instruments".

Figure 3.3 Semantic Links related to the entity Harp presented in Features and Relevant Information.

Features include the semantic relationships `rdf:type` (e.g. Harp is an instrument), and semantic relationships `rdfs:subClassOf` (e.g. the subClass Harp belongs to the superClass Plucked String Instrument) and `dcterms:subject` relationship that links an entity to its DBpedia category (e.g. Harp belongs to the category Irish musical instruments).



Figure 3.4 Semantic Links related to the entity Harp presented in the Relevant Information.

Relevant information represents subClasses of Harp extracted via the `rdfs:subClassOf` relationship (e.g. the instrument Concert harp is a subClassOf the instrument Harp).

We will use MusicPinta for:

- Conduct an exploratory user study to scope the research (Section 3.5).
- Apply algorithms to identify KA_{DG} (Chapter 4)
- Validation of KA_{DG} algorithms (Chapter 5)
- Illustration of exploration paths (Chapter 6)
- Evaluation of exploration paths (Chapter 7)

3.4.2 L4All

The data graph is drawn from the ‘LifeLong Learning in London for All’ (L4All) project [58, 154]. The project⁴⁰ was developed at Birkbeck, University of London and we used the L4All data graph which was provided by the project team at Birkbeck. The L4All project aimed to provide lifelong learners with access to information and resources that would support them in exploring learning and career opportunities and in planning and reflecting on their learning, bringing together experts from lifelong learning and career guidance, content providers, and groups of students and tutors. Figure 3.5 (from [154]) illustrates interface of L4All.

⁴⁰ UK/JISC project L4All

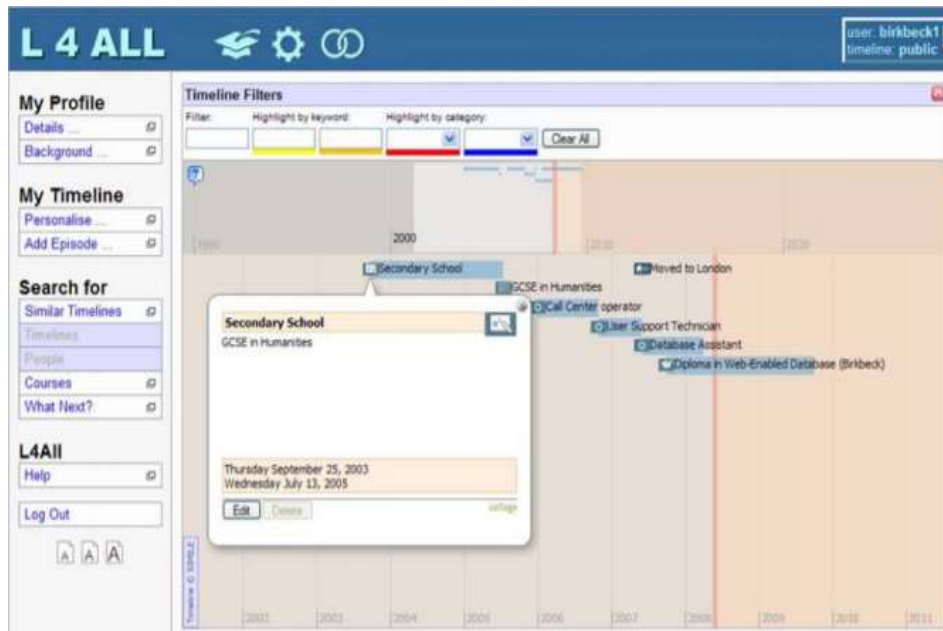


Figure 3.5 L4All main user interface⁴¹ [154].

At its centre is a visual representation of the user's timeline, and the system functionalities are organised around this. Each episode of learning or work is displayed in chronological order, depicted by an icon specific to its type and a horizontal block representing its duration. Details of an episode can be viewed by clicking on the block representing it, which pops-up more detailed information about the episode (dates, description), as well as access to edit and deletion functions. Users' timelines are encoded in the form of RDF/S. Some types of episode can be annotated by the user with a primary and possibly a secondary classification. These classifications are drawn from standard occupational and educational taxonomies of the UK Office for National Statistics. In particular, all educational episodes are classified by a subject from the Subject of Degree classification and a qualification level from the National Qualifications Framework; and all occupational episodes are classified by an occupation from the Standard Occupational Classification and an industry activity sector from the Standard Industrial Classification.

The L4All data graph uses the ontology developed by the L4All project, and users' data collected during the project (anonymised for privacy). Table 3.2 shows the main characteristics of the L4All data graph.

⁴¹ Note that L4All is used as a Second data graph for evaluation but not from the interface provided by the system. The evaluation in this thesis develops interactions outside L4All

Table 3.2. Main characteristics of the L4All data sets.

The data graph includes five class hierarchies. Each class hierarchy has number of classes linked via the subsumption relationship `rdfs:subClassOf` (e.g. there are 464 classes in the `Occupation` class hierarchy). Classes in the hierarchies are linked via the domain-specific relationships `l4all:job` and `l4all:qualify` to *job occupations* (in `Occupation` class hierarchy) and *qualification* (in `Subjects` class hierarchy). Depth of each class hierarchy is the maximum depth value for the entities in that class hierarchy.

Class hierarchy	Depth	No. of Classes	No. of Instances
Occupation	5	464	3737
Subject	3	160	2194
Episode	2	9	8800
National Qualification Framework	2	12	12
Activity Sector	1	1	4863

Among the five class hierarchies in the L4All data graph, the `Occupation` and `Subject` class hierarchies are the richest class hierarchies in terms of class classification and depth, especially that these two hierarchies enable adopting the notion of BLO [22], which postulates that a domain taxonomy (i.e. the subsumption class hierarchy of entities linked via the subsumption relationship `rdfs:subClassOf` in a data graph) have at least three levels abstractions where objects do exist in a taxonomy, namely the *basic level*, the *superordinate level* (above the basic level) and the *subordinate level* (below the basic level).

We will use the L4All data graph for:

- Apply algorithms to identify KA_{DG} (Chapter 4)
- Validation of KA_{DG} algorithms (Chapter 5)
- Illustration of exploration paths (Chapter 6)

3.5 Exploratory User Study for Research Scoping

The beginning of this research was exploratory in nature and aimed to identify the research goal and formulate the research questions. Earlier studies examining the role of semantic links and their effect on exploration through data graphs [22, 23] showed that while exploring data graphs in unfamiliar (or partially familiar) domains, users *serendipitously* learn new things that they were unaware of. However, the learning effect of exploration through data graphs has not been investigated and is usually unsupported. Furthermore, not all exploration paths are beneficial for knowledge expansion in the data graph, and there are known usability drawbacks. For example, while edge labels (i.e. semantic relationships) in the graph provide a structure for user exploration, they can also give an overwhelming amount of options to explore leading to high cognitive load, confusion, frustration, and sense of being lost in the data graph. This calls for intelligent support mechanisms to facilitate users' exploration

through exploration paths in a data graph which can bring some benefit (utility) for the user (e.g. efficiency, effectiveness, motivation, and knowledge expansion).

Following this, we aim to identify *aspects that should be considered in order to enhance users' domain knowledge while exploring data graphs*. Consequently, we conduct an exploratory user study by looking at the data graph's structure (e.g. considering the density of entities in the graph) and user's familiarity with the domain (e.g. entities are either familiar or unfamiliar to the user). In this study, we examine the effect of three suggested exploration strategies on users' knowledge expansion, using the measure for knowledge utility described in Section 2.7.1. We aim to identify key benefits and limitations for each of the suggested exploration strategies, and suggest ways to combine and further improve them. As a use case for the user study we used MusicPinta semantic data browser and data graph (described in Section 3.4.1), since it is rich in content and enables users to easily explore diverse facts about musical instruments.

3.5.1 Exploration Strategies for Knowledge Expansion

We propose three exploration strategies based on two elements: the structure of the data graph and the user familiarity with the domain.

Strategies based on the structure of the data graph (referring the user to dense entities in the graph). This follows an early work presented in [18] suggesting that users can be signposted in a data graph based on the entities' density – density in the context of data graphs is associated with the level of knowledge details in the representation of an entity, and can indicate importance of that entity. Similarly, the work in [113] highlighted the importance of density to identify important concepts in an ontology – density highlights entities which are richly characterised with properties and hierarchical relationships.

Strategies based on user familiarity with the domain (referring the user to entities that are either familiar or unfamiliar to the user). User familiarity has been an important element for characterising information exploration [2] – users who are involved in exploratory search sessions are usually unfamiliar with the exploration domain (or topic). Furthermore, different users have different domain familiarities and different needs towards expanding their domain knowledge while exploring a particular domain. Consequently, users behave differently and follow different exploration paths while traversing between entities linked via semantic relationships.

Accordingly, three exploration strategies are suggested based on structure of the data graph and the user familiarity with the domain:

- **Density Strategy [D-Strategy]** – directing the user to category entities $v \in C$ in the data graph with the highest density. Density in the context of data graphs is associated with the level of knowledge details in the representation of an entity, and can indicate importance of that entity.
- **Familiarity Strategy [F-Strategy]** – selecting candidate entities which are familiar to the user. Familiarity is generally considered to be related to user's domain awareness and ability to recognise entities in the domain.
- **Unfamiliarity Strategy [U-Strategy]** – selecting candidate entities which are unfamiliar to the user. This strategy assumes that going to unfamiliar entities would have an impact on knowledge utility, as the user will be directed to explore new concepts.

Implementation of the Density Strategy (D-Strategy). To implement the *D-strategy*, the degree centrality measure [155] is applied over the MusicPinta data graph – degree centrality is based on the idea that entities with higher connections are more important in the graph [156]. The implementation of the *D-Strategy* includes three main steps.

First step. Extract the sub-graph of all entities linked to a focus entity $v \in V$ using Sesame⁴². Starting from the focus entity in a given data graph DG , we first extract all entities that can be reached directly from the focus entity via the edge labels in the graph such as `rdfs:subClassOf`⁴³, `rdf:type`⁴⁴, `dcterms:subject`⁴⁵, and `MusicOntology:instrument`⁴⁶ (examples of edge labels in MusicPinta are illustrated in Figures 3.2 - 3.4). Repeat this process for the extracted entities. In our implementation, the process is repeated five times starting from the focus entity, producing a sub-graph with radius of six entities. This allows collecting mostly musical instruments and performances reviews. The iterative repetition of five times is based on Miller's Law [157], which indicates the number of objects that an average human can hold in working memory is 7 ± 2 . In other words, the iterative repetition of five times will allow us to generate trajectories where people can explore between 5 to 7 entities. The output of this step provides two tables for entities and edges produced, respectively.

Second step. Upload the sub-graph for the focus entity into Gephi⁴⁷ using the tables for entities and edges. The statistical outputs for the degree centrality algorithm provided by

⁴² <http://www.openrdf.org/>

⁴³ <https://www.w3.org/TR/rdf-schema/>

⁴⁴ <https://www.w3.org/TR/2004/REC-rdf-schema-20040210/>

⁴⁵ <http://musicontology.com/specification/#term-Performance>

⁴⁶ <http://dublincore.org/documents/2012/06/14/dcmi-terms/?v=terms#subject>

⁴⁷ <https://gephi.org/>

Gephi are filtered to include the *highest density candidate entities to be explored starting from the focus entity*, ranked according to their degree centrality metrics produced by Gephi.

Third step. Use the degree centrality values produced by Gephi for each entity, we generate a data graph trajectory of length 4 (i.e. the trajectory has 4 edges – 5 entities) starting from the focus entity (e.g. in Figure 3.7 the focus entity in the D-Strategy is the instrument `Oud`). Densest entities are used to create the trajectory. For example, in Figure 3.7, the densest entity that can be reached from the focus entity `Oud` is `String instrument`, and the densest entity that can be reached from `String instrument` is `Guitar`. This process is repeated 4 times, producing a data graph trajectory of length 4.

Implementation of the Familiar Strategy (F-Strategy) and the Unfamiliar Strategy (U-Strategy)

These strategies require identifying a user’s familiarity with the candidate entities which can be done implicitly (e.g. from the user’s interactive exploration) or explicitly (by asking the user to specify their familiarity during their exploration). In this study, *we are examining whether familiarity could be useful for knowledge expansion*. Hence, we are explicitly asking the user to select candidate entities in the data graph that are familiar or unfamiliar, respectively. In both strategies, the data graph trajectories will be of same length with the trajectory generated using the density strategy (all trajectories will be of length four – four edge labels and 5 entities).

User interaction with MusicPinta with the exploration strategies. We adopt a ‘Wizard of Oz’ [158] style of experimental design using MusicPinta (i.e. the strategies are not implemented directly in the system but are simulated with the help of a human). The user is ‘guided’ to select candidate entities based on the calculated density (implemented outside MusicPinta) or the stated user familiarity (as declared by the user). Each strategy is followed *independently*, i.e. a trajectory follows either *D-strategy*, *F-strategy*, or *U-strategy*. This allows us to isolate the strategies in order to examine their strengths and limitations. To ensure ‘equal’ start for each strategy, the user is directed to start from the *most dense entity* (i.e. 1st step of exploration in Figure 3.7). Then, the user follows one of the selected strategies.

3.5.2 Study Design

Participants. Twelve international postgraduates (age 18-50, mean=25) – non-native English speakers living in the UK - were recruited on a voluntary basis (a compensation of £10 Amazon voucher was offered). Users varied in Gender (7 males and 5 females) and cultural

background (1 Chinese, 1 Greek, 2 Jordanian, 2 Indian, 1 Iranian, Malaysian, 1 Mexican, 1 Polish, and 1 Saudi Arabian).

Method. Each participant was given a study form (provided in Appendix A.1). At the beginning of the form, the participants were provided with an introduction about the study (the introduction that was provided to the participants can be found in Appendix A.1.1). Each participant was provided with individual access session to MusicPinta via a URL⁴⁸. Every session was *conducted separately* with one participant and observed by the author. All participants were asked to provide feedback before, during, and after the interaction with MusicPinta to explore the paths resulted from the exploration strategies. Figure 3.6 shows the overall structure of the user study for examining three exploration strategies in terms of knowledge utility, usability and cognitive load.

Pre-study questionnaire [5 min] - collected information about participant's profiles, and their familiarity with the musical instrument class hierarchies (i.e. musical instruments families), focusing on the class hierarchies which would be explored – three class hierarchies: String Instrument, Wind Instrument, Percussion Instrument (the pre-study questionnaire is available in Appendix A.1.2). The participants' familiarity varied from none, low, medium, and high. The overall information about the participants is presented in Appendix A.2).

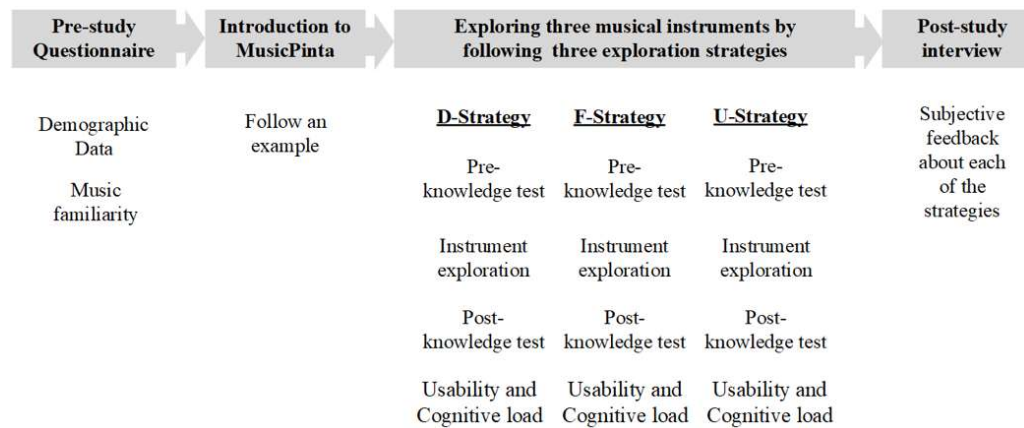


Figure 3.6 Structure of the user study to examine the performance of three exploration strategies.

The three exploration strategies (*D-Strategy*, *F-Strategy* and *U-Strategy*) are examined in terms of knowledge utility, usability and cognitive load, where each strategy correspond to one musical instrument in MusicPinta.

⁴⁸ <http://imash.leeds.ac.uk/services/pinta/app/u/n:user18 pass:musicpinta18>

Introduction to MusicPinta [5 min] - the participants followed a script which introduced the main features of the system using the musical instrument *Tabla* - an Arab musical instrument from the *Percussion* instrument class hierarchy. The example scripts for exploring *Tabla* can be found in Appendix A.1.3.

Exploring three musical instruments [45min] - the users explored three musical instruments by following the three exploration strategies in Section 3.5.1. Each instrument belongs to a particular instrument family and originates from a national culture; the GLOBE cultural clusters [159] for the national cultures was used (summary of the musical instruments is provided in Table 3.3). The order of conditions was balanced to counter balance the impact on the results. An example of the information provided for the participants for exploring the musical instrument *Oud* using the *D-Strategy* is illustrated in Appendix A.1.4.

Table 3.3 Allocation of exploration strategies for the selected musical instruments in MusicPinta.

The three instruments belong to three different class hierarchies in the MusicPinta data graph. Also, each instrument belongs to a different GLOBE cultural cluster [159].

Exploration Strategy	Instrument Name	Instrument class Hierarchy	GLOBE Cultural Cluster
Density	Oud	String Instrument	Arab Cultures
Familiarity	Bansuri	Wind Instrument	Southern Asia
Unfamiliarity	Xylophone	Percussion Instrument	Eastern Europe

Figure 3.7 shows examples of the three strategies, as used by the participants in the user study. For each strategy, we measured the degree of participants' cognitive processes of *remember*, *categorise*, and *compare* before and after the completion of each exploration, indicating the knowledge utility of the path (as discussed in Section 2.7.1). In addition, we considered the degree of recognition made by the participants for each entity in the path to have an indication about participant's familiarity with the exploration domain.

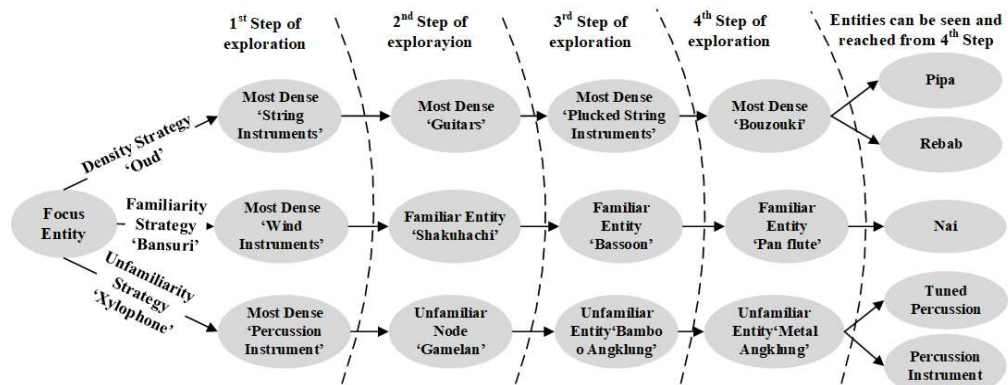


Figure 3.7 Example of data graph trajectories for the three proposed **exploration** strategies in MusicPinta.

After each exploration, the participants were asked to fill a questionnaire about their exploration experience and the cognitive load (based on a modified version of the NASA-TLX questionnaire [160]). Participants were asked to think aloud; the experimenter kept comments made by the participants during their exploration.

Post-study interview [5 min] - each participant was interviewed at the end of their session about their subjective feedback on the exploration strategies (see Appendix A.1.5).

3.5.3 Results

To identify the strengths and limitations of each of the exploration strategies and suggesting possible ways to combine/improve these strategies, the *user knowledge expansion*, and *user exploration experience* (informed by usability aspects associated with the users exploration sessions) are analysed.

- **Approximating knowledge utility**

To compare the *knowledge utility* of the exploration paths resulting from each strategy, the user knowledge was approximated before and after each exploration using the three questions that correspond to Bloom's cognitive processes of *remember*, *categorise* and *compare*. The process for approximating knowledge utility is described in Section 2.7.1.

- **User's exploration experience**

After each exploration, the participants were asked to fill a questionnaire about their exploration experience and the cognitive load they have experienced (based on a modified version of the NASA-TLX questionnaire [161]). Furthermore, the participants were asked to think aloud; notes of all comments were kept.

For each exploration strategy, the user knowledge was measured before and after the exploration using the three questions correspond to the cognitive processes of *remember*, *categorise* and *compare* (using the approach for measuring knowledge utility discussed in Section 2.7.1) related to the focus entity (i.e. Oud, Bansuri, Xylophone). The number of different entities mentioned in each user answer was counted. The difference between these numbers for each question before and after exploration is taken as an indication of the *knowledge utility of the exploration path* on the corresponding cognitive process. The knowledge utility of the three exploration strategies is shown in Figure 3.8. Before exploration, the median values for the three questions was zero, as most users were not able to articulate items linked to the three musical instruments.

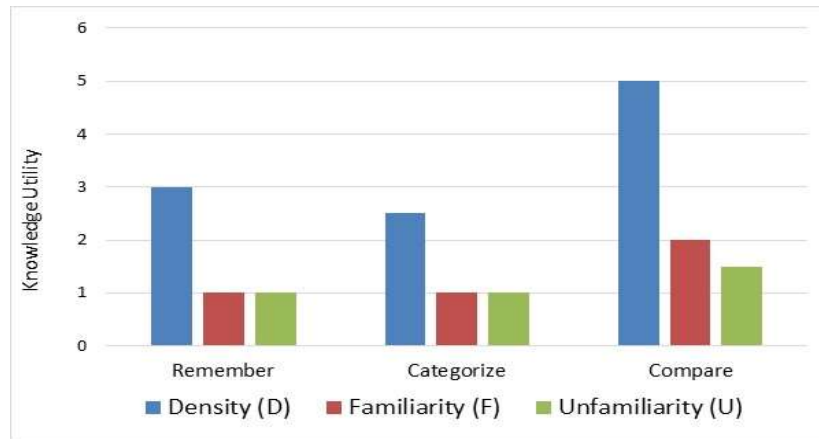


Figure 3.8 Knowledge utility of the three exploration strategies (Density, Familiarity and Unfamiliarity) of the user cognitive processes (*median* of the knowledge utility of exploration for all users).

The results in Figure 3.8 show that the knowledge utility of the *D-strategy* on the three cognitive processes (*remember*, *categorise* and *compare*) was higher than the effect of *F-strategy* and *U-strategy*; and this difference is significant as shown in Table 3.4.

Table 3.4 Statistically significant differences of knowledge utility values (Mann-Whitney, 1-tail, $N_a=N_b=12$).

Notably, for the cognitive processes *categorise* and *compare* the bigger effect on exploration of the *D-strategy* over the other strategies is highly significant ($p<0.001$).

Knowledge utility values	Cognitive Process	U	p
Density > Familiar	Remember	35.5	$P<0.050$
	Categorise	14	$P<0.001$
	Compare	18	$P<0.001$
Density > Unfamiliar	Remember	34	$P<0.050$
	Categorise	15	$P<0.001$
	Compare	27	$P<0.001$

The results showed that all participants were able to *remember*, *categorise* and *remember* entities with the *D-strategy*. Whereas not all participant could *remember*, *categorise* and *compare* new entities after they have finished their exploration with *F-strategy* (there was one participant that could not *categorise* entities, and one participant that could not *compare* entities) or *U-strategy* (there were two participants that could not *remember* entities, one participant that could not *categorise* entities, and two participants that could not *compare* entities), as shown in Appendix A.3. To further analyse what caused that the *D-strategy* was better than the other two strategies, we looked at the data collected *during each user's exploration*. At every entity in an exploration path, the user was asked to the click on both Features and Relevant Information in the MusciPinta interface (Figures

3.2, 3.3) and name the entities he/she recognised from all entities in MusicPinta associated with the that entity. All recognised entities were recorded by the author.

The overall number of entities recognised along the user exploration paths is summarised in Table 3.5. The recognition along the whole path, which involves the initial search, the suggested first click (which was always the most dense entity to ensure the users started with the same conditions), and the following three clicks where the users explicitly followed the specified strategy (strategy-related part).

Table 3.5 Summary of the recognised entities along the users’ exploration paths (median values).

There is a statistically significant strong positive correlation between the number of recognised entities during the whole path and the effect of exploration on the *compare* cognitive process (Spearman; $R=0.67$; $p<0.0001$).

Exploration Strategy	Recognised Entities	
	Whole Path	Strategy-related Part
Density (D)	40.5	21
Familiar (F)	20.5	7
Unfamiliar (U)	15	6
ALL	24	10

There is a weak correlation between the recognition and the effect of the exploration on the *categorise* and *remember* cognitive processes (Spearman; $0<R<0.3$; $p>0.5$ in both cases). Hence, the *compare* cognitive process (i.e. indicating similar instruments) is influenced by the number of entities the users recognise during the exploration. To further analyse why the *F-strategy* and *U-strategy* had smaller effect on the *compare* cognitive process than the *D-strategy*, we linked for each strategy the effect of exploration on the cognitive processes with the recognition only along the strategy-related path (given in the second column in Table 3.5). There is a statistically significant strong positive correlation between the recognition when the users followed the *D-strategy* and the effect on *remember* (Spearman; $R=0.82$; $p<0.001$) and *compare* (Spearman; $R=0.68$; $p<0.01$), while the observed moderate correlation for *categorise* was not statistically significant (Spearman; $R=0.43$; $p=0.08$). There was no correlation between the recognition based on the strategy and the effect on exploration (Spearman; $0<R<0.3$; $p>0.5$ in all). Hence, users recognised more during the *D-strategy* which led to a positive effect on the cognitive processes *remember* and *compare*.

Exploration cases with low knowledge utility. Further analysis of the individual cases when the effect of exploration was low identified several interesting situations:

Exploring familiar entities in a familiar domain. The three users who were from the Southern Asia GLOBE cluster (i.e. familiar with the instrument *Bansuri*) and followed the

F-strategy (which was allocated to `Bansuri`) did not improve their scores for remember, categorise and compare, despite the fact that many entities were recognised along the path. Hence, being familiar with the domain and sticking to familiar items had low knowledge utility, as the users did not notice new things.

Exploring familiar entities in an unfamiliar domain. Two users who were not familiar with the Southern Asia GLOBE cluster and followed the *F-strategy* (for `Bansuri` – Indian instrument) had poor scores for the three cognitive processes, as they stayed within the scope of what they knew and did not make any connection to any of the new things they were seeing on the exploration path. This indicates that even if the user is presented with something new, they may not be able to learn it as they may not be able to contextualise it.

Exploring unfamiliar entities in an unfamiliar domain. Four of the users did not improve much their knowledge when following the *U-strategy* (which was allocated to `Xylophone`). An analysis of the profiles of these users revealed that they had no knowledge of the class hierarchy `Percussion instruments` and were not from the corresponding the Eastern Europe GLOBE cluster (`Xylophone` is Greek instrument). It was noted that the users recognised a fair bit of entities during the exploration path, yet they were not able to associate to `Xylophone`.

Exploring dense entities in a familiar domain. As a whole, the exploration paths which followed the *D-strategy* had the highest knowledge utility. However, there was one specific case when a user did not gain much about `Oud` (the entity for the *D-strategy*) from the exploration. A close examination of the profile of this user showed that she was both familiar with the instrument class hierarchy `StringInstrument` and lived in the Arab Cultures GLOBE cluster where `Oud` is played. This is similar to first case – although the user was recognising many things, she did not noticing new things and not expanding their knowledge.

One user gained most from all her exploration paths disregarding the strategy she followed. She was looking for links between familiar and entities (e.g. starting from her national culture and picking instruments she did not know). This suggests that encouraging users to seek connections and form associations may increase the knowledge utility of their exploration paths.

User Exploration Experience

After each exploration path, the participants' feedback on the exploration experience during the path was collected including exploration complexity (adapted from NASA-TLX) and exploration usability (referring to aspects related to exploratory search over linked data, observed earlier). Figures 3.9 and 3.10 give a summary of the feedback.

D-strategy. The exploration paths following density strategy were seen as *most interesting*. Most of the users noticed musical instruments from diverse cultures which enabled them to make connection and associations between musical instruments that were originating from their culture with musical instruments from other cultures. For example, one of the users stated that “*I saw new string instruments from China, India, Arabic world, Greek and Africa, and this cultural variation was very interesting for me*”. Also, users found *D-strategy* interesting since it led them to a mix of familiar and unfamiliar instruments, e.g. exploring `String Instrument` users noticed entities with familiar instruments which led them to unfamiliar instruments. Overall, the users also found the *D-strategy* *least boring* and *most informative* (but these differences were not statistically significant). However, on a few occasions the users found the *D-strategy* confusing or frustrating, as it directed them to information spaces with many unfamiliar instruments.

F-strategy. The paths following familiarity strategy were also found *informative* since once directed `Wind Instrument` (start from a dense entity) the users were able to see many familiar instruments and make connections. However, two users found *F-strategy* *boring*, as they only explored familiar things and did not find new things.

U-strategy. More paths following unfamiliarity strategy were rated as frustrating comparing the other two strategies. Furthermore, the *U-strategy* was rated as *least informative*. In addition, half of the users indicated that they were confused since it was difficult for them to understand descriptions of unfamiliar instruments.

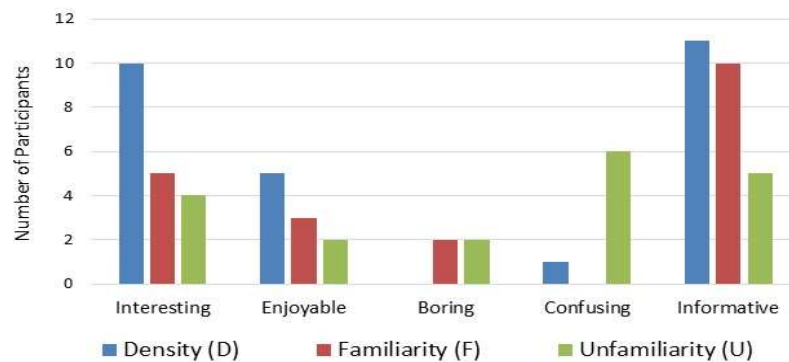


Figure 3.9 Users’ exploration experience of the three exploration strategies. Values show number of user paths rated with the corresponding characteristics.

The exploration experience was the most informative with *D-strategy* (only one participant indicated his exploration experience with *D-strategy* as not informative), and least informative with *U-strategy* (only five participants indicated their exploration with *U-strategy* as informative). The participants also found the exploration with the *D-strategy* to be the most interesting and enjoyable (only two participants indicated their exploration with *D-strategy*

as not interesting). On few occasions, the users indicated their experience with *D-strategy* as boring or confusing.

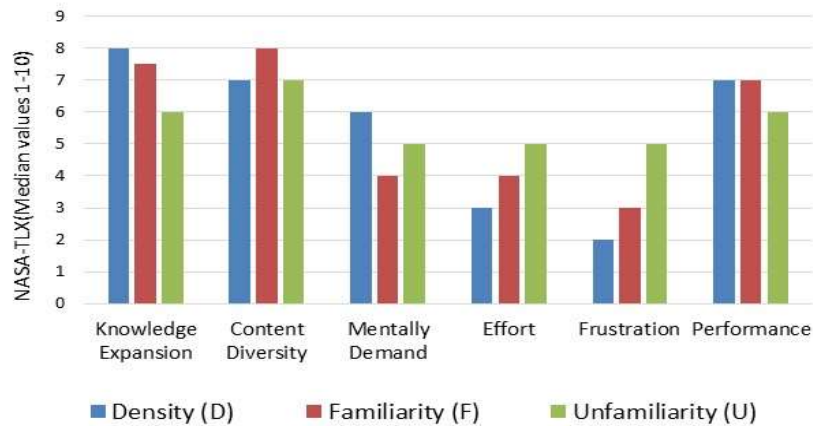


Figure 3.10 Users' subjective perception of the three exploration strategies. Values are based on adapted NASA-TLX questionnaire [160] (median values for all users in range 1-10).

User Feedback

The individual interviews at the end of each user session provided additional feedback about possible ways to overcome observed problems and combine the three strategies. The users confirmed their preference for being directed to dense places, as they could see both familiar and unfamiliar things. The users elaborated several useful points:

- When the exploration goes through too familiar spaces, the user should be directed to something new.
- Newness is associated with unfamiliar entities, not seen during the exploration.
- Offering new things should be based on some aspects from the domain, e.g. in the case of MusicPinta the users suggested that new entities could be offered based on the cultural cluster or the instrument family. For example, a user stated '*I would like to put Bansuri within Arabic Wind Instruments so it becomes easy to understand what bansuri is and to make useful associations.*'
- When the exploration goes through too unfamiliar spaces which can cause frustration and confusion, and hence the users should be helped to make a connection between new things and what they are familiar with.

3.5.4 Observations

Several observations about the strategies can be drawn from the study results.

D-Strategy used as the underpinning strategy. D-strategy has statistically significant higher cognitive effect on exploration compared to the other two strategies. In particular, participants were able to remember, recognise and compare more things when using D-

strategy. Hence, Density strategy can play a key role as underpinning (default) strategy, which can be complimented/extended using the other strategies.

Recognition is a key enabler for user knowledge expansion. The study found that the more the participants recognised entities, the higher the effect on the cognitive processes was. Hence, nudges for triggering recognition (e.g. directing the user to familiar entities) should be provided after the user is directed to dense nodes.

Encourage new connections. The cases when the F-Strategy and U-strategy performed poorly indicated interesting situations which could be detected. When the user stays mainly in familiar places, the user may miss to notice new things. When such situations are detected, the user should be directed to explore something new. Similarly, when the user explores mainly unfamiliar spaces, a connection with something familiar can be pointed to increase the knowledge utility.

User profile to detect user's domain familiarity. To detect situations when prompts can be added, in addition to exploration history, a mechanism for deriving a user profile is needed.

3.6 Summary

In this Chapter, we discussed the overall research methodology. We presented an exploratory user study that investigates three exploration strategies to aid the users when exploring data graphs. The exploration strategies were devised based on the graph structure (directing the user to dense entities in the data graph) and user familiarity with the domain (i.e. directing the user to entities that are either familiar or unfamiliar to the user, respectively). We identified a reliable and applicable measure based on Bloom's taxonomy, that measures the changes in the user's cognitive knowledge after exploring a path in a data graph. The measure for knowledge expansion was applied to examine the devised exploration strategies.

The observations from the study findings indicated that learning occurred when users saw familiar things and then expanded to new knowledge [20]. These observations directed us to investigate Ausubel's subsumption theory for meaningful learning [21] and adopt it as our underpinning model for generating exploration paths to facilitate knowledge expansion. According to this theory, the human cognitive structure is hierarchically organised with respect to levels of abstraction, generality, and inclusiveness of concepts, where familiar and inclusive entities are used as *knowledge anchors* to subsume new knowledge into the users' cognitive structures. Therefore, an important algorithmic component to adopt this theory for generating exploration paths in data graphs is the automatic generation of knowledge anchors. This will be the focus of the work presented Chapters 4 and 5.

Chapter 4

Algorithms for Knowledge Anchors

4.1 Introduction

Our observations in the exploratory user study (presented in the previous Chapter) examining exploration strategies through a data graph, directed us to adapt Ausubel’s subsumption theory for meaningful learning [21] as the underpinning theoretical model for generating exploration paths for knowledge expansion. According to this theory, familiar and inclusive entities in a data graph are used as knowledge anchors to subsume and learn new knowledge in users’ cognitive structure. Therefore, a core algorithmic component for adopting the subsumption theory for generating exploration paths is the automatic identification of knowledge anchors in a data graph (KA_{DG}), that correspond to familiar concepts in users’ cognitive structures (this formulates RQ2). We address this challenge in two steps: first, we develop and implement algorithms to automatically identify knowledge anchors in a data graph (KA_{DG}), and then we examine the performance of KA_{DG} algorithms by comparing their output of KA_{DG} against benchmarking set of familiar concepts as identified by humans.

The purpose of this Chapter is to develop algorithms to automatically identify KA_{DG} . For this, we adopt the Cognitive science notion of Basic Level Objects (BLO) introduced by Rosch, et al. [22] in the context of data graphs. We devise a formal framework that maps Rosch’s definitions of BLO and cue validity to data graphs, developing two groups of metrics for identifying KA_{DG} , together with the algorithms for implementing these metrics.

- *Distinctiveness* metrics. Identify differentiated categories whose members share attributes that are not linked to members of other categories; and
- *Homogeneity* metrics. Identify categories whose category members share many attributes together (i.e. a category entity with high similarity between its members).

Next in this Chapter, we will present two groups of metrics for identifying KA_{DG} with the corresponding algorithms for applying the metrics in the context of a data graph (Section 4.2). After that, in Section 4.3 we will describe the implementation of the KA_{DG} algorithms over two data graphs from two application contexts for data graph exploration – semantic browsing (in the musical instrument domain – MusicPinta data graph) and semantic search (in the career domain – L4All data graph). In Section 4.4 we will discuss the findings from the KA_{DG} implementation. Finally, in Section 4.5 we will summarise the work in this Chapter.

4.2 Algorithms for Identifying Knowledge Anchors in a Data Graph

We formally adopt the Cognitive science notion of BLO introduced in Section 2.7.3 to develop two groups of metrics for identifying knowledge anchors in data graphs KA_{DG} , with the corresponding algorithms for implementation. The set of all knowledge anchors in a data graph DG is denoted as KA_{DG} . Knowledge anchors refer to inclusive and familiar concepts in human cognitive structures from where links to introduce and learn new knowledge can be made. Any category entity $v \in C$ in a data graph DG , except the root entity r and the set of leaf entities L , could potentially be identified as a knowledge anchor. We follow the *distinctiveness* and *homogeneity* approaches described in Section 2.7.3 of this Chapter to define two groups of metrics for identifying KA_{DG} . The definitions of the KA_{DG} metrics are adapted from FCA approaches in [23] and adapt them in the context of identifying KA_{DG} (FCA approaches are outlined in Section 2.4.2).

Formal Concept Analysis. Formal Concept Analysis (FCA) was presented by Rudolf Wille in 1982 as a method for data analysis and knowledge representation [162]. It is a method for analysis of object-attribute data tables, where data is represented as a table describing objects, attributes and binary relationships between the objects and the properties [106]. The two main notions in FCA relevant to the approaches presented in this Section are: *formal context* and *formal concept*. A formal context X is represented by a triple $\langle M, R, I \rangle$, where M is a set of objects, R is a set of attributes and I is a binary relation $I \subseteq M \times R$. For an object $m \in M$ and formal attribute $r \in R$, is read as: the object m has the attribute r . Table 4.1 presents an example of a formal context.

Table 4.1 An example of a formal context in FCA represented as sets of objects M and attributes R , where (x) indicates that an object m has attribute r .

set of Objects M	set of Attributes R				
	r_1	r_2	r_3	r_4	r_5
m_1	x		x		x
m_2	x	x	x	x	x
m_3		x			

For a given formal context X , let $A \subseteq R$ and $B \subseteq M$. The pair $\langle A, B \rangle$ is called a formal concept where $A = B'$ (B' is the set of objects having all the attributes belonging to A), and $B = A'$ (A' is the set of attributes applying to all objects belonging to B). An example from the formal context X in Table 4.1 for a formal concept is $\langle A, B \rangle = \langle \{m_1, m_2\}, \{r_1, r_3, r_5\} \rangle$ - the

concept objects m_1, m_2 are conceptually clustered based on the three shared attributes r_1, r_3, r_5 . In our work, we adapt the FCA notions of *formal context* and *formal concept* to data graph DG and category entity $v \in C$, respectively. Objects M and attributes R of a formal context $\langle M, R, I \rangle$ comprise entities V and edge labels E in a data graph DG , respectively. The objects $B \subseteq M$ of a formal concept $\langle A, B \rangle$ comprise members $v' : v' \subseteq v$ of a category entity $v \in C$, and the attributes A in the formal concept represent attribute entities v'_e linked to members $v' : v' \subseteq v$ of the category entity $v \in C$ via an edge label $e \in E$ in a data graph DG .

4.2.1. Distinctiveness Metrics

This group of metrics aims to identify the most differentiated category entities whose members are linked to distinctive entities that are shared amongst the members of the category entities but are not shared to members of other categories. Each entity $v \in V$ that is linked through an edge label e to members $v' \subseteq v$ of the category entity $v \in C$ will have a single validity value used to distinctive the category entity $v \in C$ among other category entities in the data graph). Three distinctiveness metrics, namely *Attribute Validity (AV)*, *Category Attribute Collocation (CAC)* and *Category Utility (CU)*, are developed which follow three cognitive science approaches for identifying basic formal concepts in FCA [23].

Attribute Validity (AV). The attribute validity definition corresponds to the cue validity definition in [22] (presented in Section 2.7.3) and adapts the cue validity metric from [23]. We use ‘attribute validity’ to indicate the association with data graphs - ‘cues’ in data graphs are attributes of the entities and are represented as relationships in terms of triples. The attribute validity value of a category entity $v \in C$ is calculated with regard an edge label e , as the aggregation of the attribute validity values for all entities v'_e linked to subclasses $v' : v' \subseteq v$ via an edge label e (i.e. the attribute validity entities v'_e are the *Subject* entities, the set of subclass entities v' are the *Object* entities, and e is the *Predicate*). The attribute validity value of v'_e increases, as the number of edge labels of type e between v'_e and the subclasses $v' : v' \subseteq v$ increases; whereas the attribute validity value of v'_e decreases as the number of edge label of type e between v'_e and *all entities* in the data graph increases.

We define the set of entities $W(v, e)$ related as *Subjects* to the subclasses $v' : v' \subseteq v$, via an edge label of type e :

$$W(v, e) = \{ v'_e : \exists v' [v' \subseteq v \wedge \langle v'_e, e, v' \rangle \in T] \} \quad (1)$$

Formula (2) defines the attribute validity metric for a given entity v with regard to an edge label of type e ,

$$AV(v, e) = \sum_{v'_e \in W(v, e)} \frac{|\{\langle v'_e, e, v' \rangle : v' \subseteq v\}|}{|\{\langle v'_e, e, v_a \rangle : v_a \in V\}|} \quad (2)$$

where v is a category entity $v \in C$ in the data graph, e is an edge label whose type can belong to either the hierarchical H or domain-specific D relationships and $v_a \in V$ is any entity in the data graph DG . For example, in Figure 4.1, the AV value for category entity v_2 with regard the domain-specific relationship D is the aggregation of the AV values of the (*Subject* entities e_3, e_4, e_5, e_6) linked to members of the category entity v_2 (*Object* entities $v_{21}, v_{22}, v_{23}, v_{24}$) via the edge label (i.e. *Predicate*) D . The AV value for the entity e_3 equals the number of the triples between the *Subject* entity e_3 and members of the category v_2 (*Object* entities v_{21}, v_{22}) via the relationship D (2 triples), divided by the number of triples between the *Subject* entity e_3 and all *Object* entities in the graph (*Object* entities v_{12}, v_{21}, v_{22}) via the relationship D (3 triples). Hence the AV value for e_3 equals $2/3 = 0.66$. The aggregation AV values for entities e_3, e_4, e_5, e_6 will identify AV value for v_2 .

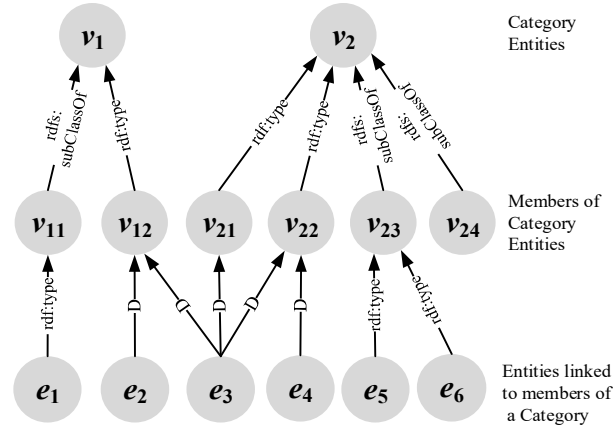


Figure 4.1 A data graph showing entities and relationship types between entities.

The steps for extracting similar data graph, as described in Algorithms 4.1 and 4.2 below, include two steps: (i) starting from a category entity in a data graph such as v_2 (correspond to line 1 in Algorithms 4.1 and 4.2), extract all *Subject* members of the category ($v_{21}, v_{22}, v_{23}, v_{24}$) linked to the category entities via subsumption relationships, such as e.g. *rdfs: subClassOf* and *rdf:type* (correspond to line 2 in Algorithms 4.1 and 4.2); and (ii) for each member of the category entity (e.g. the member v_{22}), extract all *Subject* entities, such as the entities (e_3, e_4) linked to the category member via a particular edge label, such as the edge label D (correspond to line 3 in Algorithm 4.1 and lines 3-5 in Algorithm 4.2).

Category Attribute Collocation (CAC). This approach was used in [33] to improve the cue validity approach by adding the so called category-feature collocation measure which takes into account the frequency of the attribute within the members of the category. This gives preference to ‘good’ categories that have many attributes shared by their members. In our case, a good category will be a category entity $v \in C$ with high number of relationships of type e between v' and the subclasses $v': v' \subseteq v$, relative to the number of its subclasses. Formula (3) defines the category-attribute collocation metric for a given category entity v with regard to a relationship type e ,

$$CAC(v, e) = \sum_{v' \in W(v, e)} \frac{|\{\langle v', e, v' \rangle : v' \subseteq v\}|}{|\{\langle v', e, v_a \rangle : v_a \in V\}|} \cdot \frac{|\{\langle v', e, v' \rangle : v' \subseteq v\}|}{|V'|} \quad (3)$$

where v is a category entity $v \in C$ in the data graph, e is an edge label whose type can belong to either the hierarchical H or domain-specific D relationships and $v_a \in V$ is any entity in the data graph DG . Considering the above example for identifying the AV value for the entity e_3 , and considering the relationship D , the CAC adds a weight of (the number of the triples the *Subject* between e_3 and the members of v_2 (*Object* entities v_{21}, v_{22}) via the relationship D (2 triples) divided by (the number of members of the category entity v_2 (i.e. FOUR members: $v_{21}, v_{22}, v_{23}, v_{24}$)). Hence the CAC of entity e_3 will be the AV value of e_3 multiplied by (2/4). The CAC value for entity e_3 equals [(2/3) * (2/4) = 0.33]. The aggregation CAC values for entities e_3, e_4, e_5, e_6 will identify the CAC value for v_2 .

Category Utility (CU). This approach was presented in [30] as an alternative metric for obtaining categories at the basic level object. The metric takes into account that a category is useful if it can improve the ability to predict the attributes for members of the category, i.e. a good category will have many attributes shared by its members (as mentioned in the category-attribute collocation metric). At the same time, it possess ‘unique’ attributes that are not related to many other categories (efficiency of category recognition). We adapt the formula in [26] for a data graph,

$$CU(v, e) = \frac{|V'|}{|V|} \sum_{v' \in W(v, e)} \left(\frac{|\{\langle v', e, v' \rangle : v' \subseteq v\}|}{|V'|} \right)^2 - \left(\frac{|\{\langle v', e, v_a \rangle : v_a \in V\}|}{|V|} \right)^2 \quad (4)$$

where v is a category entity $v \in C$ in the data graph, e is an edge label whose type can belong to either the hierarchical H or domain-specific D relationships and $v_a \in V$ is any entity in the data graph DG . Following the previous examples for calculating the AV and the CAC values for entity e_3 (see Figure 4.1), in addition to the proportion of used by the CAC

value (2/4), the CU will include the proportion of all triples between the *Subject* entity e_3 and all *Object* entities in the graph (THREE *Object* entities v_{12}, v_{21}, v_{22}) linked via relationship D (i.e. 3 triples) over the number of entities linked via the subsumption relationships (e.g. `rdfs:subClassOf` and `rdf:type`) in the graph (11 entities). Hence the CU value for entity e_3 will be: $[(2/4)^2 - (3/11)^2 = 0.177]$. The aggregation CU values for entities e_3, e_4, e_5, e_6 , multiplied by the number of members of category v_2 (FOUR members) divided by number of entities in the graph linked via the subsumption relationships (11 entities) will identify the CU value for the category entity v_2 .

Algorithm 4.1 describes calculating the distinctiveness metrics. The algorithm takes a data graph DG and an edge label type (hierarchical H or domain-specific D relationship) as input and returns values for the three distinctiveness metrics for each category entity $v \in C$.

Algorithm 4.1: Distinctiveness Metrics

Input: $DG = \langle V, E, T \rangle, e \in E$

Output: three distinctiveness values AV_v, CAC_v, CU_v for every category entity $v \in C$

1. **for all** $v \in C$ **do** *//all category entities in the data graph*
 2. $V' :=$ the set of all $v' : v' \subseteq v$ *//all members of the category v*
 3. **for all** $v'_e : \exists \langle v'_e, e, v' \rangle$ **do**
 4. $N_e :=$ set of all $\langle v'_e, e, v' \rangle : v' \in V'$
 5. $M_e :=$ set of all $\langle v'_e, e, v_a \rangle : v_a \in V$
 6. $AV_{v'_e} := |N_e| / |M_e|$
 7. $CAC_{v'_e} := (|N_e| / |M_e|) \cdot (|N_e| / |V'|)$
 8. $CU_{v'_e} := (|N_e| / |V'|)^2 - (|M_e| / |V|)^2$
 9. $AV_v := AV_v + AV_{v'_e}$
 10. $CAC_v := CAC_v + CAC_{v'_e}$
 11. $CU_v := CU_v + CU_{v'_e}$
 12. **end for**
 13. $CU_v := \frac{|V'|}{|V|} \cdot CU_v$
 14. **end for**
-

In (line 1), all category entities $v \in C$ are retrieved via SPARQL query:

```

SELECT distinct ?category
WHERE {
    ?category rdfs:subClassOf root_entity.
    ?subclass rdfs:subClassOf ?category. }

```

After that, for a category entity $v \in C$, all members are retrieved (line 2). Members of an entity can be subclasses or instances. In the current implementation, we are using the subsumption relationship `rdfs:subClassOf` to retrieve the entity's members via the following SPARQL query:

```
SELECT distinct ?subclass
WHERE {
    ?subclass rdfs:subClassOf v . }
```

For each entity v'_e linked to one or more subclass entities $v' : v' \subseteq v$ via an edge label e (i.e. connected via the triple $\langle v'_e, e, v' \rangle$) in (line 3), several steps are conducted: retrieving all triples with *Subject* v'_e and *Object* any subclass $v' : v' \subseteq v$ (line 4); retrieving all triples with *Subject* v'_e and *Object* any graph entity v (line 5); applying the formulas for calculating the AV , CAC , and CU metrics for v'_e (lines 6-8); and aggregating values for v'_e to the overall values for v (lines 9-11).

4.2.2. Homogeneity Metrics

These metrics aim to identify categories whose members share many entities among each other (i.e. identify categories that have high similarity values between their members). In this work, we have utilised three well-known set-based similarity metrics [24] : *Common Neighbours (CN)*, *Jaccard (Jac)*, and *Cosine (Cos)*. These metrics consider the pair-wise similarity between members of a category entity $v \in C$. The homogeneity value for a category entity is identified as the average of the accumulated pair-wise similarity values between the categories members with regard to an edge label e . For instance (see Figure 4.1), the *Jaccard* similarity between the pair-wise members (v_{21}, v_{22}) of the entity v_2 considering the edge label D is the number of intersected *Subject* entities (ONE entity e_3) linked to *Objects* v_{21}, v_{22} via edge label D , divided by the number of union *Subject* entities (TWO entities e_3, e_4) linked to *Objects* v_{21}, v_{22} via edge label D . Hence the *Jaccard* similarity between members (v_{21}, v_{22}) is $(1/2)$.

Algorithm 4.2 describes implementation of the metrics. The algorithm takes a data graph and a relationship type (hierarchical or domain-specific relationship) as input and returns values for the three homogeneity metrics for each entity $v \in C$.

Algorithm 4.2: Homogeneity Metrics

Input: $DG = \langle V, E, T \rangle, e \in E$

Output: CN_v, Jac_v, Cos_v for all $v \in C$

1. **for all** $v \in C$ **do** *// check all category entities if they are KA_{DG}*
 2. V' := the set of all $v' : v' \subseteq v$ *//get the members for each category entity*
 3. **for all** $(v', v'') : v' \in V' \wedge v'' \in V'$ **do**
 4. $V'_e := \{v'_e : \exists \langle v'_e, e, v' \rangle\}$
 5. $V''_e := \{v''_e : \exists \langle v''_e, e, v'' \rangle\}$
 6. $I := V'_e \cap V''_e$;
 7. $U := V'_e \cup V''_e$;
 8. $CN_{v',v''} := |I|$; *//Common Neighbours Similarity*
 9. $Jac_{v',v''} := |I| / |U|$; *//Jaccard Similarity*
 10. $Cos_{v',v''} := |I| / (\sqrt{|V'_e|} \cdot \sqrt{|V''_e|})$; *//Cosine Similarity*
 11. $CN_v := CN_v + CN_{v',v''}$;
 12. $Jac_v := Jac_v + Jac_{v',v''}$;
 13. $Cos_v := Cos_v + Cos_{v',v''}$;
 14. **end for**
 15. $CN_v := CN_v / (|V'| \cdot (|V'| - 1) / 2)$;
 16. $Jac_v := Jac_v / (|V'| \cdot (|V'| - 1) / 2)$;
 17. $Cos_v := Cos_v / (|V'| \cdot (|V'| - 1) / 2)$;
 18. **end for**
-

In (line 1), all category entities are retrieved via the same SPARQL query used in (Algorithm 4.1 / line 1). Then, for every category entity v , all members are retrieved using the subsumption relationship (line 2) using the same SPARQL query in (Algorithm 4.1 / line 2). For each pair of subclass entities v' and v'' (line 3), several steps are conducted: retrieving all entities linked via triples with v' and v'' (lines 4-5); calculating their intersection and union (lines 6-7); applying the formulas for calculating the similarity metrics CN , Jac , and Cos (lines 8-10); and aggregating these values to the overall values for v (lines 11-13); and averaging the aggregated values (lines 15-17).

4.3 KA_{DG} Algorithms Implementation

We implement the KA_{DG} algorithms over different data graphs in different application contexts to show the applicability and generality of the algorithms. Furthermore, it is

important to implement the KA_{DG} algorithms to identify their outputs of KA_{DG} , used to examine the algorithms' performance.

Figure 4.2 describes the overall structure for implementing the KA_{DG} algorithms. The algorithms have been implemented in Java as a set of classes and Application Programming Interfaces (API), used for constructing SPARQL queries in RDF repositories in a triple store (a triple store can have more than one repository), and store the retrieved RDF triples in a MySQL database.

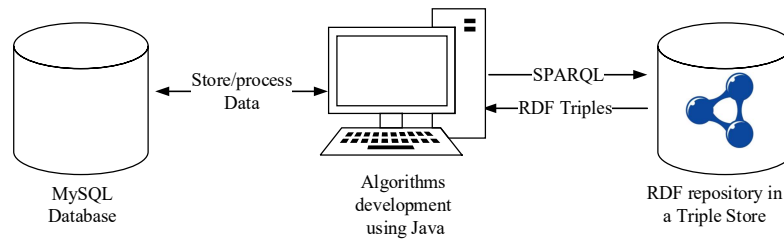


Figure 4.2 Structure for Implementing the KA_{DG} Algorithms

4.3.1 Implementation in MusicPinta

The implementation includes the three following Java classes:

- (i) Java class for constructing the SPARQL queries that are described in Algorithms 4.1 and 4.2. The class uses OpenRDF Sesame API⁴⁹ to open a two way interface with the MusicPinta triple store. The SPARQL queries are run in the triple store, and then the results of the queries are retrieved and stored in MySQL database.
- (ii) Java class that uses the output of the SPARQL queries stored in MySQL for calculating the values of the *distinctiveness* metrics for each category entity in the data graph; and
- (iii) Java class that uses the output of the SPARQL queries stored in MySQL for calculating the values of the *homogeneity* metrics for each category entity.

Examples of Java classes are available here⁵⁰.

The metrics output values for each category entity in the data graph are identified. Appendix B.1 lists the normalised output values in the range [0,1] for three distinctiveness metrics –attribute validity (*AV*), category attribute collocation (*CAC*), category utility (*CU*); and three homogeneity metrics – common neighbours (*CN*), Jaccard similarity and Cosine similarity. All metrics were run over two types of relationships in MusicPinta – two hierarchical relationships (`rdfs:subclassOf` and `dcterms:subject`) and one domain-specific (`MusicOntology: instrument`).

⁴⁹ <http://archive.rdf4j.org/javadoc/sesame-2.7.12/>.

⁵⁰ <https://doi.org/10.5518/304>

4.3.2 Implementation in L4All

The implementation in L4All uses the same Java classes described in Section 4.4.1 above. The only difference is that the triple store in L4All was created locally on the machine using a Graph database tool called GraphDB⁵¹.

The metrics output values for each category entity in `Occupation` and `Subject` class hierarchies were identified. Appendices B.2, B.3 show the normalised output values of in the range [0,1] for three distinctiveness metrics –attribute validity (*AV*), category attribute collocation (*CAC*), category utility (*CU*); and three homogeneity metrics –common neighbours (*CN*), Jaccard similarity and Cosine similarity, in `Occupation` and `Subject` class hierarchies respectively. All metrics were run over two types of relationships – hierarchical (`rdfs:subClassOf` and `rdf:type`) and domain specific (`L4All:job` for `Occupation` and `L4All:qualify` for `Subject`).

4.4 Discussion

The KA_{DG} algorithms presented in Section 4.2 are generic and can be applied over different application domains represented as data graphs. The algorithm is applied in two application domains for data exploration, musical instrument and career, using the data graphs from two semantic exploration applications.

By inspecting the output values of the Category Utility (*CU*) distinctiveness metric and the three homogeneity metric Common Neighbours (*CN*), Jaccard and Cosine) for `Occupation` and `Subject` class hierarchies in L4All data graph (Appendices B2, B3 respectively), we noticed that these metrics gave zero values for category entities in `Occupation` and `Subject` class hierarchies using the domain-specific relationships (for `Occupation` and `Subject`) and using the hierarchical relationships (for `Subject`).

The *CU* metric produced zero values since it multiplies the ratio [number of instances of a category entity divided by number of all entities, classes and instances] with the total *CU* values for members of a category. Hence, the *CU* value will be decreased to almost zero values, especially when there are 1000s of entities (i.e. classes and instances) in the graph. For instance, in the `Occupation` class hierarchy, the *CU* ratio for the category entity `Sales Related Occupation` is: 87 instances divided by 4201 (464 classes + 3737 instances in the `Occupation` class hierarchy), reducing the *CU* value for `Sales Related Occupation`.

⁵¹ <https://ontotext.com/products/graphdb/>.

The *three homogeneity metrics* had zero similarity values since each category entity has instances that are linked to one instance only via a domain-specific relationship (e.g. relationship `l4all:Job` in `Occupation`). Hence, the categories will have no intersections among their instances, producing zero values in the *homogeneity* metric. For example, in Figure 4.3, the category entity `Air Transport Operatives` has five members linked via the `rdf:type` relationship, and these members do not shared instances (i.e. $I = \{\}$ in line 6 in Algorithm 4.2) between each other, and hence the homogeneity values between the category members will be zero (lines 8-10 in Algorithm 4.2). This indicates that homogeneity metrics can have similar performance, and one metric can be for future applications.

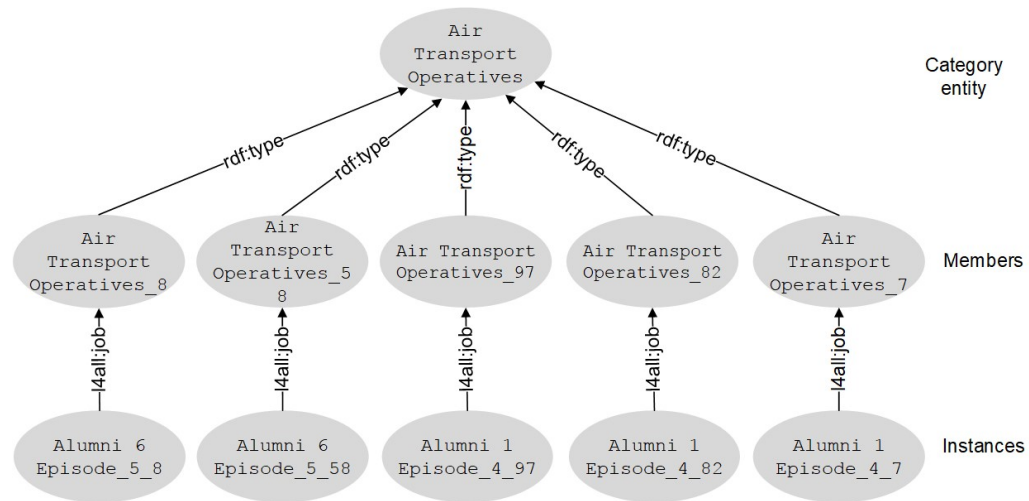


Figure 4.3 Extract from the `Occupation` class hierarchy in `L4All` showing instances linked to members of the category `Air Transport Operatives`

In our implementation of the KA_{DG} algorithms, we noticed that it is important to identify the edge labels that are used to indicate membership of a category entity in a data graph (i.e. the edge labels used to identify members of a category entity as in line 2 in Algorithms 4.1, 4.2). This step is considered important as it will affect the metrics values identified for each category entity. For example, in Figure 4.1 members of the category entity v_2 are linked through two edge label types (`rdfs:subClassOf` and `rdf:type`), and choosing both or either of them will affect which entities will be involved in calculating the KA_{DG} metrics values of v_2 . For instance, choosing the `rdfs:subClassOf` will indicate v_{23} and v_{24} as members of v_2 , and hence, only the entities e_5 and e_6 will be taken into account in calculating the KA_{DG} metrics values of v_2 , whereas the entities e_3 and e_4 will not be taken into account in calculating the metrics values of the category v_2 . Hence,

inspection of the data graph before running the KA_{DG} algorithms to identify which edge labels will be used to indicate membership of category entities, is important.

The KA_{DG} algorithms presented in this Chapter have many potential applications. For example, the algorithms can be applied to ontology summarization where the knowledge anchors from the data graph allows capturing a layman user's view of the domain. Furthermore, KA_{DG} can be also used to solve the key problem of 'cold start' in personalization and adaptation. One of the popular choices for addressing the cold start problem in a dialogue system with the user where knowledge anchors can be used to identify the user familiarity in probing dialogue.

4.5 Summary

In this Chapter, we have uniquely utilised Rosch's definitions of Basic Level Objects (BLO) and cue validity to develop two group of metrics and the corresponding algorithm for identifying knowledge anchors in a data graph (KA_{DG}). *Distinctiveness* metrics identify differentiated categories whose attributes are shared amongst the category members but are not associated to members of other categories. *Homogeneity* metrics identify categories whose members share many attributes together.

The formal framework presented in this Chapter, that maps Rosch's definition of basic level objects and cue validity to data graphs is a key contribution of our work. The developed KA_{DG} algorithms are generic and can be applied over different application domains represented as data graphs. We have implemented the algorithms over two applications for data exploration, semantic browsing (in the musical instrument domain) and semantic search (in the career domain), using the data graphs from the two applications, *MusicPinta* and *L4All*, respectively.

The next step in this research, is to evaluate the performance of the KA_{DG} algorithms against human cognitive structures that correspond to the data graphs from the two application contexts. This will be the focus of the next Chapter.

Chapter 5

Knowledge Anchors Algorithms Evaluation

5.1 Introduction

We discussed in Chapter 3 how the findings from the exploratory user study directed us to adopt the subsumption theory for meaningful learning [21], as our underpinning theoretical model for generating exploration paths for knowledge expansion. According to this theory, familiar entities in a data graph act as knowledge anchors from where links to learn new entities can be made. Therefore, to adopt this theory, it was important for us to develop an automatic way for identifying knowledge anchors in a data graph (KA_{DG}). Accordingly, in Chapter 4, we have presented two groups of metrics for identifying KA_{DG} and the corresponding algorithms for implementation. The algorithms have been implemented over two data graphs (MusicPinta, L4All) from two domains (musical instrument and career), respectively.

The aim of this Chapter is to assess the performance of the KA_{DG} algorithms by considering the user perspective and the application context. We will do this by comparing human Basic Level Objects in a data graph (BLO_{DG}) that represent familiar concepts in human cognitive structures with automatically derived KA_{DG} (i.e. output of the KA_{DG} algorithms) in a data graph. For this, we will present a systematic approach that adapts experimental methods from Cognitive science to derive human BLO_{DG} underpinned by a data graph. The approach considers two ways for deriving human BLO_{DG} , namely free-naming (using images) and category verification (using abstract text labels). We will use the proposed approach to evaluate KA_{DG} algorithm in the two domains – musical instrument and career.

Next in this Chapter, we will present an algorithm for identifying a benchmarking set of human BLO_{DG} underpinned by a data graph used to evaluate the KA_{DG} algorithms (Section 5.2). After that, in Sections 5.3 and 5.4 we will describe experimental studies where we apply the algorithm for identifying human BLO_{DG} using data graphs of two application contexts for data graph exploration - semantic browsing (musical instrument – MusicPinta) and semantic search (career – L4All), respectively. The human BLO_{DG} identified in the two application contexts will be used to evaluate the derived KA_{DG} . In Section 5.5 we discuss the findings from the evaluation, and in Section 5.6 we will summarise the work presented in this Chapter.

5.2 Algorithm for Identifying Human BLO over a Data Graph

Following Cognitive science experimental studies outlined in Section 2.7.3, two strategies with the corresponding algorithm for identifying human BLO_{DG} , are presented.

- **Strategy 1.** Takes into account whether a leaf entity $v \in L$ in a given data graph $DG = \langle V, E, T \rangle$, that has no subclasses is presented and named with one of its parents (i.e. a superclass). An example for naming a leaf entity in the musical instrument domain are presented in Section 5.3.1 (Figure 5.1).
- **Strategy 2.** Takes into account whether a category entity $v \in C$ in a given data graph, that has one or more subclasses is presented and named with its exact name, or with the name of a category parent (i.e. a superclass) or with the name of a category member (i.e. subclass that is not a leaf). Examples for naming category entities in the musical instrument and the career domains are presented in Section 5.3.1 (Figures 5.2 – 5.4) and Section 5.4.1 (Figures 5.5, 5.6), respectively.

Algorithm 5.1 describes the two strategies for identifying human BLO_{DG} . Algorithm 5.1 uses *accuracy* and *frequency* described above (Section 5.2) in identifying human BLO_{DG} . The algorithm takes a data graph $DG = \langle V, E, T \rangle$ as input and returns two sets of human BLO_{DG} that correspond to the two strategies. For a class entity $v \subseteq V$ in DG (line 1), we identify the number of participants to be asked to name the entity (line 2). For *Strategy 1* (lines 3-7), we consider accurate naming of a category entity (a parent) when a leaf entity $v \in L$ that is a member of this category is seen. For *Strategy 2* (lines 8-14), we consider naming a category entity $v \in C$ with its exact name (lines 10, 11) or a name of its superclasses (parents) or subclasses (members) (lines 12-13). In each strategy, we use a representation function $show(v)$ to create a representation of an entity v to be shown to the user. The representation of a leaf entity $v \in L$ (*Strategy 1*) will consider the leaf itself (e.g. show a *single text label* or a *single image* for the leaf entity), while the representation of a category entity $v \in C$ (*Strategy 2*) will consider all (or some) of the category's leaf entities (e.g. showing a random listing of a set of text labels of leaf entities or showing a group of images of leaf entities as a collage). The set of leaf entities that are used in the representation of a category entity $v \in C$ is identified via the following SPARQL query:

```
SELECT ?leaf ?leaf_label
WHERE {
  ?leaf rdfs:subClassOf v .
  ?leaf rdfs:label ?leaf_label.
FILTER NOT EXISTS
{
  ?member rdfs:subClassOf ?leaf.}}
```

Algorithm 5.1: Identifying Human Basic Level Objects Over a Data Graph

Input: $DG = \langle V, E, T \rangle$

Output: two sets of entities: **Set1** for human BLO_{DG} identified from *Strategy 1*, and **Set2** for human BLO_{DG} identified from *Strategy 2*. The **Union** of the **Set1** and **Set2** identifies the final set of Human BLO_{DG}

```

1. for a set of entities  $v \subseteq V$  do
2.   for ( $i := 1; i \leq n; i++$ )                               //show an entity  $v$  to  $n$  different participants
3.     if  $v \in L$  then                                       //Strategy1
4.       show ( $v$ ) and ask a user to name  $v$ 
5.       if  $answer(v) \in parent(v)$  then //check if answer is an accurate name of a parent
6.          $count_a++$ ;                                       //increase frequency
7.       end if;
8.     else if  $v \in C$  then                                   //Strategy2
9.       show ( $v$ ) and ask a user to name  $v$ 
10.      if  $answer(v) = label(v)$  then                       //check if answer is the same name
11.         $count_a++$ ;                                       //increase frequency
12.      else if  $answer(v) \in \{parent(v) \cup member(v)\}$  then //check accurate naming
13.         $count_a++$ ;                                       //increase frequency
14.      end if;
15.    end if;
16.  end for;
17. end for;
18. Set1 =  $\{answer(v) : v \in L \wedge count_a \geq k\}$          //K is frequency of different users
19. Set2 =  $\{answer(v) : v \in C \wedge count_a \geq k\}$          //K is frequency of different users

```

The two strategies in Algorithm 5.1 for obtaining human BLO_{DG} are applied as follows:

Strategy 1. When a user is shown a representation of a leaf entity $v \in L$ (line 4), the following steps are conducted:

- The function $answer(v)$ assigns a user's answer to the leaf entity v .
- The function $parent(v)$ returns a set of labels (i.e. names) of the parent(s) of the leaf entity v , via the following SPARQL query:

```

SELECT ?parent ?parent_label
WHERE {
  v rdfs:subClassOf ?parent.
  ?parent rdfs:label ?parent_label.}

```

- The algorithm in (line 5) checks if the user named the leaf entity v with one of its parents. If an accurate name of a parent was provided, then the frequency of the parent entity will be increased by one (line 6).

Strategy 2. When a user is shown a representation of a category entity $v \in C$ (line 9), the following steps are conducted:

- The function $answer(v)$ assigns a user's *answer* to the category entity v .
- The function $parent(v)$ returns a set of labels of parent(s) of the category entity v via SPARQL queries, similar to *Strategy 1* above.
- The function $member(v)$ returns a set of labels of *member*(s) of the category entity v , via the following SPARQL query:

```
SELECT ?member_label
WHERE {
    ?member rdfs:subClassOf v .
    ?member rdfs:label ?member_label .}
```

- The function $label(v)$ returns the *label* (i.e. name) of the category entity v via the following SPARQL query:

```
SELECT ?label
WHERE {
    v rdfs:label ?label .}
```

The algorithm in (lines 10, 12) checks if the user named the category entity v with its exact name, or a name of its parents or its members. If there was accurate naming of the category entity, a parent or a member, the frequency of the category name (line 11), the parent name or the member name (line 13) will be increased by one.

5.3 Evaluating KA_{DG} in MusicPinta

In order to evaluate KA_{DG} metrics, we compare the outputs of the KA_{DG} metrics over the MusicPinta data graph (KA_{DG} metrics output are presented in Section 4.3.1) versus a benchmarking set of human BLO_{DG} from the data graph categories, as identified by humans.

5.3.1 Obtaining Human BLO_{DG} in MusicPinta

To obtain human BLO_{DG} , we conducted a user study following Algorithm 5.1.

Participants. 40 participants, university students and professionals, age 18–55 (mean = 30), were recruited on a voluntary basis. None of participants in the study had expertise in Music.

Method. The participants were asked to freely name objects that were shown in image stimuli, under limited response time (10 seconds). Overall, 364 taxonomical musical instruments (i.e. 364 classes in the ontology) were extracted from the MusicPinta data graph by running SPARQL queries over the MusicPinta triple store to get all musical instrument concepts linked via the subsumption relationship `rdfs:subClassOf`. The extracted entities included: *leaf entities* $v \in L$ (total 256) and *category entities* $v \in C$ (total 108). Applying the two strategies in Algorithm 5.1, for each leaf entity, a representative image was collected from the Musical Instrument Museums Online (MIMO)⁵² to ensure that pictures of high quality were shown⁵³. For a category entity, *all leaves* from that category entity were shown as a group in a single image (similarly to a packet of images in [22]). Ten online surveys⁵⁴ were run. The surveys are divided into two groups that correspond to the two strategies for identifying human BLO_{DG} :

- leaf entities (*Strategy 1*): eight surveys presented 256 leaf entities (each survey showed 32 leaf entities);
- category entities (*Strategy 2*): two surveys presented 108 category entities (each survey showed 54 category entities).

Free-naming task. Each image in the developed surveys was shown for 10 seconds on the *participant's* screen (a new image flips every 10 seconds on the participant's screen). The participant was asked to type the name of the given object (for leaf entities) or the category of objects (for category entities). The image allocation in the surveys was random. Every survey had four respondents from the participants (corresponds to line 2 in Algorithm 5.1). Each participant was allocated to one survey (either *Strategy 1* - leaf entities or *Strategy 2* category entities). By applying Algorithm 5.1 over MusicPinta, two sets of human BLO_{DG} are identified. Figures 5.1 - 5.4 show example instrument images and the corresponding participant answers (Figure 5.1 from *Strategy 1*, Figures 5.2 -5.4 from *Strategy 2*).

- *Set1 (Strategy 1)* was derived from presenting leaf entities to participants. We consider accurate naming of a category entity (a parent of the leaf entity) when a leaf entity that belongs to this category is seen. For example in Figure 5.1, a participant was shown the image of the instrument `Violotta`, a leaf entity in the data graph, and the participant named it with its parent category entity `Violin`.

⁵² <http://www.mimo-international.com/MIMO/>

⁵³ MIMO provided pictures for most musical instruments. In the rare occasions when an image did not exist in MIMO, Wikipedia images were used instead.

⁵⁴ The study was conducted with Qualtrics (www.qualtrics.com). Example from the surveys is available at: https://login.qualtrics.com/jfe/preview/SV_cHhHPPthBFO5r6d?Q_CHL=preview



What is the name of this object ?

Violin

Figure 5.1 An image of *Violotta* (a leaf in the data graph) was shown to a participant, who named it as *Violin* (parent of *Violotta*).

This will be counted as an accurate naming and will increase the count for the category *Violin* by one. The overall count for *Violin* will include all cases when participants named *Violin* while seeing any of its leaf members.

- *Set2 (Strategy 2)* was derived from presenting category entities. We consider naming a category entity with its exact name or a name of its parent or subclass member. For example in Figure 5.2, a participant saw the category *Fiddle* and named its parent category *Violin*; this will increase the count for *Violin*.



What is the name of this object ?

Violin

Figure 5.2 An image of *Fiddle* (a category entity in the data graph with two leaf entities) was shown to a user, who named it as *Violin* (parent of *Fiddle*).

In Figure 5.3 a participant was shown the image of category *Violin* and named it with its exact name; this will also increase the count for *Violin*.

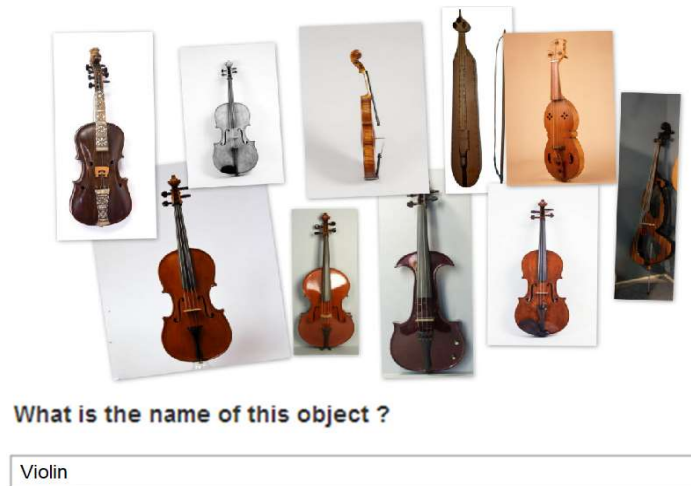


Figure 5.3 An image of Violin (a category entity in the data graph) was shown to a user, who named it as Violin.

In Figure 5.4 a participant saw the category Bowed String Instrument and named it as its member category Violin; this will also increase the count for Violin.

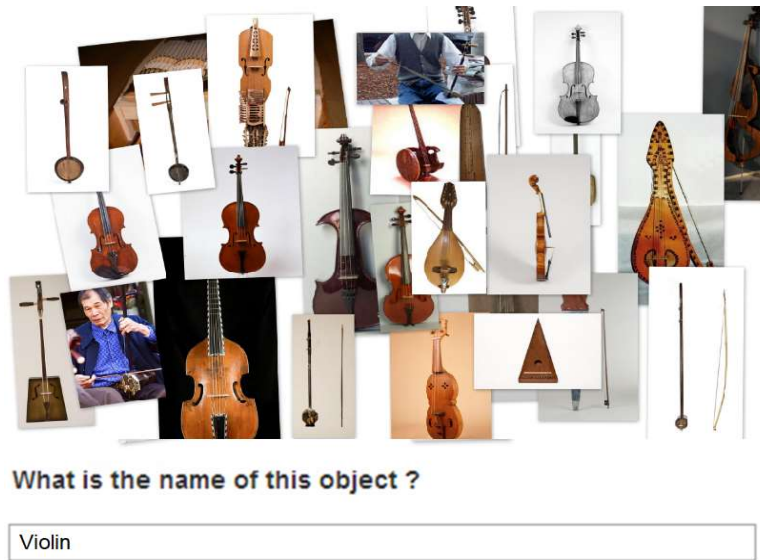


Figure 5.4 An image of Bowed String Instrument (a category entity in the data graph) shown to a user, who named it as Violin (category member of Bowed String Instrument).

In each of the two sets (*Set1* and *Set2*), entities with frequency equal or above *two* (i.e. named by at least two different participants) were identified as potential human BLO_{DG} . The union of *Set1* and *Set2* in Algorithm 5.1 gives the set of human BLO_{DG} . The set of human BLO_{DG} obtained from MusicPinta includes the musical instruments:

Accordion, Banjo, Bell, Bouzouki, Cello, Clarinet, Drum, Electric Piano, Flute, Gong, Guitar, Harmonica, Harp, Lute, Lyre, Organ, Recorder, Saxophone, String Instrument, Trombone, Trumpet, Tuba, Violin and Xylophone.

5.3.2 Evaluating KA_{DG} against Human BLO_{DG}

Quantitative Analysis. We used the set of human BLO_{DG} identified in Section 5.3.1 above to examine the performance of the KA_{DG} metrics – the three distinctiveness metrics: *Attribute Validity (AV)*, *Category Attribute Collocation (CAC)*, *Category Utility (CU)*, and the three homogeneity metrics: *Common Neighbours (CN)*, *Jaccard (Jac)*, *Cosine (Cos)*, applied over the hierarchical (*H*) and domain-specific (*D*) relationships. The KA_{DG} metrics normalised output values over *H* and *D* relationships for MusicPinta are listed in Appendix B.1. To compare the outputs of the KA_{DG} metrics versus a benchmarking set of human BLO_{DG} , we need to identify a cut-off threshold point for the KA_{DG} metrics normalised output values. Therefore, we examined the performance of the KA_{DG} by considering the 60th, 70th, 80th and 90th percentiles of the metrics normalised output lists, and compare them to the set of human BLO_{DG} identified. We noticed that the three homogeneity metrics (*Common Neighbours*, *Jaccard*, *Cosine*) gave the same lists of KA_{DG} , therefore one metric is used when reporting the results, namely Jaccard similarity (Jaccard similarity was chosen since it is a widely applied similarity metric, and was used in a similar context which is identifying basic formal concepts in the context of formal concept analysis [115]). For each metric, the entities identified using the hierarchical relationships – the subsumption relationship `rdfs:subClassOf` and the `dcterms:subject` relationship that links musical instruments to DBpedia categories, were aggregated using union. One domain-specific relationship was used – the `MusicOntology:instrument` relationship that links a musical instrument to a performance. Overall, the KA_{DG} metrics evaluated included the three distinctiveness metrics plus the *Jaccard* homogeneity metric over the hierarchical (*H*) and domain-specific (*D*) relationships (Refer to Appendix C.1). We noticed that the metrics performed best (using F1 value) when the 60th percentile of the normalised lists was used as a cut-off point. Therefore, we use the 60th percentile as our cut-off threshold point for identifying the metrics output of KA_{DG} over the hierarchical (*H*) and domain-specific (*D*) relationships in the MusicPinta data graph (Refer to Appendix C.2). The metrics precision, recall and F1 values by comparing human BLO_{DG} and KA_{DG} derived using 60th percentile are shown in Table 5.1.

Table 5.1 Precision, recall and F1 values by comparing 60th percentiles of the KA_{DG} metrics normalised output values over hierarchical (*H*) and domain-specific (*D*) relationships, with human BLO_{DG} derived in MusicPinta.

Relationship types	Precision				Recall				F1			
	AV	CAC	CU	Jac	AV	CAC	CU	Jac	AV	CAC	CU	Jac
<i>H</i>	0.31	0.32	0.34	0.37	0.66	0.71	0.67	0.54	0.42	0.44	0.46	0.44
<i>D</i>	0.31	0.29	0.31	0.32	0.54	0.50	0.54	0.36	0.39	0.36	0.39	0.35

Precision values (using 60th percentile) were poor (ranging from 0.29 to 0.37), and recall values were mixed (ranging from 0.36 to 0.71). To inspect what caused the low precision, the False Positive (FP) entities were inspected and it was noticed that the metrics were picking FP entities which had long label names, such as:

Archaic and other Bowed String Instrument, Plucked String instrument, Valved Brass Instruments, and Slide Brass Instruments.

Layman users who are not domain experts tend to remember simple domain concepts with simple names such as Dog, Chair or Guitar. In this regard, Rosch et al. [22] conducted several experiments to identify BLO in several domain taxonomies (Musical Instruments, Fruit, Tool, Clothing, Furniture, Vehicle, Tree, Fish and Bird), and all BLO extracted from these domain taxonomies had single label names (Guitar, Apple, Hammer, Pants, Table, Car, Oak, Salmon and Eagle, respectively). This was also observed in the user study in Section 5.3.1 for obtaining human BLO_{DG} in MusicPinta – in 89% (1292 answers) of all answers (1456 answers), the participants used single words to name musical instruments in the free naming task, whereas around 10% (151 answers) used two words, and only 1% (13 answers) used three or more words to name musical instruments. As in the ontology summarization approach [113] discussed in Section 4.3.2, a name simplicity strategy was applied to reduce noise when calculating key concepts. The strategy treats each concept separately by considering its label length. The name simplicity approach we use is solely based on the data graph. We identify the *weighted median* for the length of the labels of all data graph entities $v \subseteq V$ and filter out all entities whose name length is higher than the weighted median. The weighted median for the length of the labels of all data graph entities is 1.2, and hence we only included entities which consist of one word. Table 5.2 illustrates precision, recall and F1 values by comparing human BLO_{DG} and KA_{DG} derived using hierarchical (*H*) and domain-specific (*D*) relationships, after applying the weighted median. We filtered the set of human BLO_{DG} and KA_{DG} by removing entities with labels of two (or more) words. For example, KA_{DG} entities such as Plucked String Instrument and human BLO_{DG} entities such as String Instrument were removed when identifying the metrics performance.

Table 5.2 precision, recall and F1 values for comparing human BLO_{DG} and KA_{DG} derived using hierarchical (*H*) and domain specific (*D*) relationships in MusicPinta after applying the weighted median.

Relationship types	Precision				Recall				F1			
	AV	CAC	CU	Jac	AV	CAC	CU	Jac	AV	CAC	CU	Jac
<i>H</i>	0.60	0.62	0.62	0.60	0.68	0.73	0.73	0.55	0.64	0.67	0.67	0.57
<i>D</i>	0.55	0.53	0.55	0.62	0.50	0.45	0.50	0.36	0.52	0.49	0.52	0.46

Applying the weighted median has improved the performance of the KA_{DG} metrics – precision values (ranging from 0.53 to 0.62), recall values (ranging from 0.36 to 0.73) and F1 values (ranging from 0.46 to 0.67).

Qualitative analysis and hybridization. Further analysis of the False Positive (FP) and False Negative (FN) entities indicated that the KA_{DG} metrics had different performance on the different taxonomical levels in the data graph. In other words, we compared the metrics output of KA_{DG} and the set of human BLO_{DG} at each hierarchical level in the data graph, where each level indicates a particular depth from the root entity in MusicPinta. This led to the following heuristics for *hybridization*.

- **Heuristic 1.** Use Jaccard similarity metric with hierarchical relationships for the most specific categories in the graph (i.e. the category entities at the bottom quartile of the class hierarchy). There were FP entities such as the entities `Shawm` and `Oboe` returned by distinctiveness metrics using the domain-specific relationship `MusicOntology:Performance`, because these entities are highly associated with musical performances (e.g. the entity `Shawm` is linked to 99 performances and the entity `Oboe` is linked to 27 performance). Such entities may not be good knowledge anchors for exploration, as their class hierarchy is flat. The best performing metric at the specific level was *Jaccard* for hierarchical attributes - it excluded entities which had no (or a very small number of) hierarchical attributes.
- **Heuristic 2.** Take the majority voting for all other taxonomical levels. Most of the entities at the middle and top taxonomical level will be well represented in the graph hierarchy and may include domain-specific relationships. Hence, combining the values of all algorithms is sensible. Each algorithm represents a voter and provides two lists of votes, each list corresponding to hierarchical or domain-specific associated attributes (*H*, *D*). At least half of the voters should vote for an entity for it to be identified in KA_{DG} .

The list of KA_{DG} identified by applying the above hybridization heuristics includes the musical instruments:

Accordion, Bass, Bell, Bouzouki, Brass, Castanets, Clarinet, Drum, Flute, Guitar, Harp, Huqin, Lute, Lyre, Organ, Reeds, Saxophone, Tambura, Tuba, Violin, Woodwind, Xylophone and Zurna.

Applying the hybridization heuristics presented above improved *Precision value* to **0.65** (average Precision in Table 5.2 = 0.59), *Recall value* to **0.68** (average Recall in Table 5.2 = 0.56), and F1 value to **0.66** (average F1 in Table 5.2 = 0.57).

5.4 Evaluating KA_{DG} in L4All

The career domain is a suitable domain for studying BLO due to the richness the *Occupation* and *Subject* class hierarchies in terms of class classification and depth (See Table 3.2). These two hierarchies enable adopting the notion of BLO [22], which postulates that a domain taxonomy (i.e. the subsumption class hierarchy of entities linked via the subsumption relationship `rdfs:subClassOf` in a data graph) have at least three levels abstractions where objects do exist in a taxonomy, namely the *basic level*, the *superordinate level* (above the basic level) and the *subordinate level* (below the basic level). In order to evaluate KA_{DG} algorithms in L4All, we compare the outputs of the implementation of the KA_{DG} algorithms over the L4All data set versus a benchmarking set of human BLO_{DG} , as identified by humans.

5.4.1 Obtaining Human BLO_{DG} in L4All

To enable impartial comparison of the outputs of the KA_{DG} metrics and human BLO_{DG} , we conducted a user study in the career domain following Algorithm 5.1.

Participants. 28 participants, university students and professionals, age 25–64 (mean = 35), recruited on a voluntary basis. Most of them had computing background.

Method. The experimental study for evaluating knowledge anchors in the L4All data graph included categories from the *Occupation* and *Subject* class hierarchies, as these class hierarchies have the richest class representation and depth compared to other class hierarchies, and they enable adopting the notion of BLO [22], as discussed in Section 3.4.2. Category entities were represented to participants (corresponding to the $show(r, v)$ function in Algorithm 5.1) using names (i.e. labels) of the category's leaves. Overall, 624 class entities were extracted from the two class taxonomies (464 for *Occupation* class hierarchy and 160 for *Subject* class hierarchy) by running SPARQL queries to get all class entities linked via the subsumption relationship `rdfs:subClassOf`. The extracted entities included: leaves (349 for *Occupation* and 141 for *Subject*) and categories (115 for *Occupation* and 19 for *Subject*). Seven online surveys⁵⁵ were developed. Six surveys presented the 115 category entities of the *Occupation* class hierarchy (five surveys each showing 19 categories and one survey showing 20 categories), and one survey presented the 19 categories of the *Subject* class hierarchy. The category allocation in each survey was random. Every

⁵⁵ The study was conducted with Qualtrics (www.qualtrics.com). Examples from the surveys are available at: https://login.qualtrics.com/jfe/preview/SV_aidgbaUYm3DZwSp?Q_CHL=preview

survey had four respondents from the study participants. Each participant was allocated *only to one survey*.

Category identification task. A representation of each category entity was shown on the participant's screen and he/she was asked to identify the category name. The representation included a list of leaves' names of that category (at most four leaf names were randomly selected and shown on the participant's screen). The participant was provided with four different categories as candidate answers (including the correct category answer which the leaves belong to) and the participant was asked to select one category that he/she thinks the leaf entities belong to. The three additional candidate categories covered three levels of abstraction, namely: a parent from the *superordinate* level, a member from the *subordinate* level, and a *sibling* at the same category level. In cases where no parents or members could be added to the candidate answers, siblings were used instead.

Applying Strategy 2 in Algorithm 5.1 over the *Occupation* and *Subject* class hierarchies in the L4All data graph, we considered naming a category entity with its exact name or a name of its parents or its non-leaf subclass members shown to the participants. Figures 5.5 and 5.6 show two examples of the category identification task from the *Occupation* and *Subject* class hierarchies, respectively.

Select the Career category that all of the following Job titles belong to?

- Housekeepers and Related
- Caretakers



Figure 5.5 A representation of Housekeeping Occupation (a Category concept in the Occupation class hierarchy with two subclasses) was shown to a user, who identified it as Personal Service Occupation.

For example in Figure 5.5, the participant saw two leaves (the category has two leaves only) of the category Housekeeping Occupation and the participant identified the category's parent Personal Service Occupation, which he/she thinks that the leaves belong to. This will increase the frequency for the category Personal Service Occupation. In Figure 5.6, a participant was shown the leaf names of the category

Biological Sciences (four random leaves were selected among 9) and selected its exact name. This will increase the count for the category Biological Sciences.

Select the Subject category that all of the following Subject titles belong to?

- Psychology
- Microbiology
- Zoology
- Molecular Biology, Biophysics and Biochemistry



NEXT

Figure 5.6 A representation of Biological Sciences (a category concept in the Subject class hierarchy with four random subclasses) was shown to a user, who identified it as Biological Sciences.

Category entities in the Occupation and Subject class taxonomies with frequency equal or above two (i.e. categories named by at least two different users) were identified as potential human BLO_{DG} . The lists of human BLO_{DG} obtained from Occupation and Subject class hierarchies in L4All data graph include:

List of human BLO_{DG} obtained from the Occupation class hierarchy:

Administrative and Secretarial Occupations;
Administrative Occupations;
Administrative Occupations: General;
Administrative Occupations: Government and Related Organisations;
Assemblers and Routine Operatives;
Associate Professional and Technical Occupations;
Business and Finance Associate Professionals;
Business and Public Service Associate Professionals;
Business and Public Service Professionals;
Construction Operatives;
Corporate Managers;
Culture, Media and Sports Occupations;
Customer Service Occupations;
Customer Service Occupations and Related;
Draughtspersons and Building Inspectors;
Engineering Professionals;
Functional Managers;
Health and Social Services Managers;
Information and Communication Technology Professional;
IT Service Delivery Occupations;

Librarians and Related Professionals;
Managers and Senior Officials;
Personal Services Occupations N.E.C.;
Process, Plant and Machine Operatives;
Process, Plant and Machine Operatives and Related;
Process Operatives;
Professional Occupations;
Research Professionals;
Sales and Customer Service Occupations;
Sales Assistants and Retail Cashiers;
Sales Related Occupations;
Science and Engineering Technicians;
Science and Technology Associate Professionals;
Science and Technology Professionals;
Science Professionals;
Teaching and Research Professionals;
Transport and Mobile Machine Drivers and Operatives;
Transport Drivers and Operatives;

List of human BLO_{DG} obtained from the Subject class hierarchy:

Architecture, Building and Planning;
Biological Sciences;
Business and Administrative Studies;
Creative Arts and Design;
East, Asiatic, African, American and Australian
Languages, Literature;
Education;
Engineering;
European Language, Literature and related subjects;
Historical and Philosophical Studies;
Law;
Linguistics, Classics and related subjects;
Mathematical and Computer Sciences;
Medicine and Dentistry;
Social Studies;
Subjects allied to Medicine;
Veterinary Science, Agriculture and related subjects.

5.4.2 Evaluating KA_{DG} against Human BLO_{DG}

Quantitative Analysis. We used the set of human BLO_{DG} identified in Section 5.4.1 above, to examine the performance of six KA_{DG} metrics – three distinctiveness metrics: *Attribute Validity (AV)*, *Category Attribute Collocation (CAC)*, *Category Utility (CU)*, and three homogeneity metrics: *Common Neighbours (CN)*, *Jaccard (Jac)*, *Cosine (Cos)*, applied over the hierarchical (*H*) and domain-specific (*D*) relationships. The KA_{DG} metrics normalised output values over *H* and *D* relationships for Occupation and Subject class hierarchies are listed in Appendices B.2 and B.3, respectively.

At the beginning of the analysis we aimed to identify a cut-off threshold point for evaluating the KA_{DG} metrics in the Occupation and Subject class hierarchies in L4All

data graph (similar to MusicPina analysis in Section 5.3.2). For this we examined the performance of the metrics by taking the 60th, 70th, 80th and 90th percentiles of the metrics normalised output lists, and compare them to the set of human BLO_{DG} identified for Occupation and Subject class hierarchies (the normalised output lists for Occupation and Subject class hierarchies are listed in Appendices B.2 and B.3, respectively). It was noticed that the three homogeneity metrics (*CN*, *Jaccard*, and *Cosine*) had the same output lists of KA_{DG}. Therefore, we choose *Jaccard* similarity metric (similar to MusicPinta) to report the results. For each metric, the entities identified using the hierarchical relationships – the subsumption relationships (*rdfs:subClassOf* and *rdf:type*), were aggregated using union. One domain-specific relationship was used by the metrics in each class hierarchy – the *L4All:Job* relationship for Occupation class hierarchy and *L4All:qualify* relationship for Subject class hierarchy. Overall, the KA_{DG} metrics evaluated included the three distinctiveness metrics plus the *Jaccard* homogeneity metric over *H* and *D* relationships. Appendix C.3 illustrate precision, recall and F1 values of the metrics at the different percentiles for Occupation and Subject class hierarchies. We noticed that the metrics performed better (in terms of F1 values) when the 60th percentile was used as a cut-off threshold point. Therefore, we consider the 60th percentile as our cut-off threshold point for identifying the metrics output of KA_{DG} over the hierarchical (*H*) and domain-specific (*D*) relationships in Occupation and Subject class hierarchies. The metrics output of KA_{DG} using the 60th percentile as a cut-off point for Occupation and Subject class hierarchies are shown in Appendices C.4 and C.5, respectively. The metrics precision, recall and F1 values by comparing human BLO_{DG} and KA_{DG} derived using 60th percentile in Occupation and Subject class hierarchies are shown in Table 5.3.

Table 5.3 Precision, recall and F1 values for comparing the 60th percentiles of the KA_{DG} metrics normalised output values over the hierarchical (*H*) and domain-specific (*D*) relationships with human BLO_{DG} derived, in the Occupation and Subject class hierarchy in L4All data graph.

Class Hierarchy	Relationship type	Precision				Recall				F1			
		AV	CAC	CU	Jac	AV	CAC	CU	Jac	AV	CAC	CU	Jac
Occupation	<i>H</i>	0.59	0.59	0.74	0.74	0.92	0.92	0.45	0.45	0.72	0.72	0.56	0.56
	<i>D</i>	0.77	0.76	0.00	0.00	0.95	0.92	0.00	0.00	0.85	0.83	0.00	0.00
Subject	<i>H</i>	1	0.89	0.00	0.00	0.56	0.50	0.00	0.00	0.72	0.64	0.00	0.00
	<i>D</i>	1	1.00	0.00	0.00	0.56	0.56	0.00	0.00	0.72	0.72	0.00	0.00

To analyse the results, we inspected the False Positive (FP) entities and we noticed that the metrics were picking FP entities which had long label names, such as: Managers and Proprietors in Hospitality and Leisure Services and Health and Social Welfare Associate Professionals in the Occupation class hierarchy; and East, Asiatic, African, American and Australian Languages, Literature in the Subject class hierarchy).

Therefore, a name simplification approach was applied using the *weighted medians* for the length of the labels of class entities in the Occupation and Subject class hierarchies to filter out entities whose name length is higher than the median (similar to the approach used in MusicPinta data graph). The weighted median for Occupation class hierarchy was 3.2 and for Subject class hierarchy was 2.8. Hence, entities with name length greater than 3 were excluded (the names of the two class taxonomies - Occupation and Subject - and conjunctions, e.g. 'and', were not taken into account in counting the name length of entities). Tables 5.4 illustrates precision, recall and F1 values comparing human BLO_{DG} and KA_{DG} derived using hierarchical (H) and domain-specific (D) relationships in Occupation and Subject class hierarchies, after applying the weighted median. Using the hierarchical relationships ($rdfs:subClassOf$ and $rdf:type$), precision and recall values were good for Occupation (precision ranging from 0.74 to 0.79 and recall from 0.44 to 0.92) and very mixed for Subject (precision ranging from 0.00 to 0.89 and recall from 0.00 to 0.73). For the domain-specific relationships, the precision and recall were mixed for Occupation (precision ranging from 0.00 to 0.75 and recall from 0.00 to 0.96) and Subject (precision ranging from 0.00 to 0.89 and recall from 0.00 to 0.73).

Table 5.4 precision and recall values for comparing human BLO_{DG} and KA_{DG} derived from the metrics using hierarchical (H) and domain-specific (D) relationships in Occupation and Subject class hierarchies after applying weighted median.

Class Hierarchy	Relationship type	Precision				Recall				F1			
		AV	CAC	CU	Jac	AV	CAC	CU	Jac	AV	CAC	CU	Jac
Occupation	H	0.61	0.61	0.65	0.65	0.92	0.92	0.44	0.44	0.73	0.73	0.52	0.52
	D	0.73	0.72	0.00	0.00	0.96	0.92	0.00	0.00	0.83	0.82	0.00	0.00
Subject	H	0.89	0.89	0.00	0.00	0.73	0.73	0.00	0.00	0.80	0.80	0.00	0.00
	D	0.89	0.89	0.00	0.00	0.73	0.73	0.00	0.00	0.80	0.80	0.00	0.00

By inspecting what caused the zero precision and recall values for the Category Utility (CU) distinctiveness metric and Jaccard (Jac) similarity metric, we noticed that none of these metrics picked False Negative (FN) entities (i.e. potential KA_{DG}) using the domain-specific relationships (for `Occupation` and `Subject`) and using the hierarchical relationships (for `Subject`).

There were two reasons why the CU metric did not pick any FN entities using the domain-specific relationships. On the one hand, the metric multiplies the ratio [number of instances of a category divided by number of all entities, classes and instances] with the total CU values for members of a category, which will decrease the CU value especially when there are 1000s of entities (i.e. classes and instances) in the graph. For instance, in the `Occupation` class hierarchy, the CU ratio for the FN category `Sales Related Occupation` is: 87 instances divided by 4201 (464 classes + 3737 instances in the `Occupation` class hierarchy), reducing the CU value for `Sales Related Occupation` to become less than the 60th percentile cut-off point. On the other hand, members of category entities in `Occupation` and `Subject` class hierarchies are associated with one instance only ($N_e = 1$ in Algorithm 4.1) via a domain-specific relationship (an example from `Occupation` class hierarchy is illustrated in Figure 4.3), which all also reduce the CU value. This also caused not picking FN entities by the Jac similarity metrics since the category entities in `Occupation` and `Subject` class hierarchies have members which are linked with one instance only via a domain-specific relationship (e.g. `l4all:job` relationships in `Occupation` class hierarchy, and `l4all:qualify` relationships in the `Subject` class hierarchy). Therefore, the categories will have no intersections among their members, producing zero similarity values between the members. The CU and Jac metrics did not pick FN entities in the `Sub Subject` class hierarchy using the hierarchical relationships, since it has shallow hierarchy of two levels, where members of category entities are not associated with entities via hierarchical relations.

Qualitative analysis and hybridization. Analysis of the False Positive (FP) and False Negative (FN) entities indicated that the algorithms had different performance on the different taxonomical levels in the L4All data graph, which is formulated in the two heuristics below.

- **Heuristic 1.** *Use the AV and CAC distinctiveness metrics with hierarchical relationships for the categories at the bottom quartile of the class hierarchy.* There were FN entities (e.g. `Assemblers` and `Routine Operatives`) that were not identified as KA_{DG} by the CAC homogeneity metrics using the domain-specific relationship `Job`, because such entities have a low number of instances (e.g. `Assemblers` and `Routine`

Operatives has 20 instances and Science and Engineering Technicians has 50 instances; whereas the median of instances per entity is 144).

- **Heuristic 2.** *Take the majority voting for all other taxonomical levels.* Most of the entities at middle and top taxonomical level are well represented in the class hierarchy of the data graph. Each metric represents a voter and provides two lists of votes, each list corresponding to hierarchical or domain-specific relationships. At least half of the voters should vote for an entity for it to be identified as KA_{DG} .

The KA_{DG} identified by applying the above hybridization heuristics for Occupation and Subject class hierarchies are:

For the *Occupation* class hierarchy:

Administrative and Secretarial Occupations;
Administrative Occupations;
Administrative Occupations: General;
Administrative Occupations: Records;
Assemblers and Routine Operatives;
Associate Professional and Technical Occupations;
Caring Personal Service Occupations;
Corporate Managers;
Culture, Media and Sports Occupations;
Customer Service Occupations;
Design Associate Professionals;
Elementary Occupations;
Engineering Professionals;
Functional Managers;
IT Service Delivery Occupations;
Librarians and Related Professionals;
Managers and Senior Officials;
Personal Service Occupations;
Plant and Machine Operatives
Process Operatives;
Professional Occupations;
Research Professionals;
Sales and Customer Service Occupations;
Sales Occupations;
Sales Related Occupations;
Science and Engineering Technicians;
Science and Technology Professionals;
Science Professionals;
Secretaries and Related;
Skilled Trades Occupations;
Teaching and Research Professionals;
Teaching Professionals;
Transport Drivers and Operatives;

For the *Subject* class hierarchy:

Architecture, Building and Planning;
Biological Sciences;
Business and Administrative Studies;

Creative Arts and Design;
Education;
European Language, Literature and related subjects;
Linguistics, Classics and related subjects;
Mathematical and Computer Sciences;

Applying the hybridization heuristics improved the performance for Occupation and Subject class hierarchies, as follows:

- for Occupation, *Precision* value improved to **0.68** (average Precision in Table 5.4 = 0.49), *Recall* value improved to **0.92** (average Recall in Table 5.4 = 0.58), and *F1* value improved to **0.78** (average F1 in Table 5.4 = 0.52);
- for Subject, *Precision* value improved to **1.00** (average Precision in Table 5.4 = 0.45), *Recall* value improved to **0.73** (average Recall in Table 5.4 = 0.37), and *F1* value improved to **0.84** (average F1 in Table 5.4 = 0.4).

5.5 Discussion

In this Chapter, we presented a systematic evaluation approach to validate KA_{DG} metrics against human BLO_{DG} . In this Section, we focus our discussion on the developed algorithm for identifying human BLO_{DG} which correspond to familiar entities in human cognitive structures, and the evaluation of the KA_{DG} metrics by comparing the metrics output with the set of human BLO_{DG} .

5.5.1 Algorithm for Identifying Human BLO_{DG}

The human BLO_{DG} algorithm presented in Section 5.2 is generic and can be applied over different application domains represented as data graphs. The algorithm is applied in two application domains for data exploration, musical instrument and career, using the data graphs from two semantic exploration applications.

Applying the human BLO_{DG} algorithm over the two domains allows us to *illustrate two ways of instantiating the algorithm for obtaining human BLO_{DG}* . MusicPinta describes concrete objects - musical instruments - that can have digital representations (e.g. image, audio, video). An image stimulus was used to represent musical instruments, and free-naming tasks included showing image representations of graph entities and asking the users to quickly name the entities they see. In contrast, L4All comprises of abstract career categories, such as Occupation and Subject class hierarchies which have text representations (i.e. labels of entities) but no clearly distinguishable images. In this case, a category verification task was used to obtain human BLO_{DG} by showing text representations of graph entities and asking the user to identify the matching entity given some answers.

An important component for applying the human BLO_{DG} algorithm is to identify appropriate and reliable stimuli. They are used for representing the data graph entities and showing them to humans in either a free-naming task (e.g. showing images) or in a category verification task (e.g. showing text labels). One of the main factors that affects choosing appropriate stimuli is how well the stimuli cover the entities in the data graph. In other words, the chosen stimuli should have representations for all entities in the graph hierarchies. For instance, the stimuli for MusicPinta were images - taken from an established source (MIMO⁵). The chosen stimuli have to be close enough to users' cognitive structures, so the users can understand the representation of entities in the graph.

Applying the human BLO_{DG} algorithm over shallow graph hierarchies has some limitations. For instance, most categories (15 categories out of 19) in the `Subject` class hierarchy of the L4All ontology were identified as human BLO_{DG} . In a category verification task over a shallow hierarchy, finding candidate answers to be presented to users is challenging, especially when the shallow hierarchy does not contain the three levels of abstraction (basic, subordinate and superordinate). Furthermore, the identified human BLO_{DG} in data graphs can have confusing category labelling which reflect insufficiently articulated scope; for instance, vague names (e.g. 'European Language, Literature and related subject') or combining two categories in one (e.g. 'Mathematical and Computer Sciences'). Hence, the human BLO_{DG} algorithm is sensitive to the quality of the ontology. This points at another possible application of human BLO_{DG} – peculiarities in the output can indicate deficiencies of the ontology which can provide insights for re-engineering the ontology. An area of future work is to improve the L4All ontology by modifying the class labels and better articulating their scope.

5.5.2 Performance of KA_{DG} Metrics

The identified human BLO_{DG} were used to examine the performance of the KA_{DG} metrics. Our analysis found that hybridization of the metrics notably improved performance. The hybridization heuristics for the upper level of the graph hierarchies tend to be the same – combine the KA_{DB} metrics using majority voting. However, the hybridization heuristics for the bottom level of the hierarchy differed depending on how instances at the bottom of the graph were associated through domain-specific relationships. The performance is sensitive to the appropriateness of the domain-specific relationships captured in the data graph. Examining the FP and FN entities for the hybridization algorithms for KA_{DG} led to the following observations:

Missing basic level entities due to unpopulated areas in the data graph. We noticed that none of the metrics picked FN entities belonging to the bottom quartile of the taxonomies

and having a small number of members (such as `Cello` in `MusicPinta` and `Construction Operatives` in the `Occupation` class hierarchy in `L4All` - `Cello` has only one subclass and `Construction Operatives` has 10 instances – mean number of instances in `Occupation` is 184). While these entities belong to the cognitive structures of humans and were therefore added to the human BLO_{DG} sets, one could question whether such entities would be useful knowledge anchors because of their relatively small number of members. These entities could lead the user to ‘dead ends’ within unpopulated areas of the data graph which may be confusing. We therefore see such FN cases as ‘good misses’ by the KA_{DG} metrics.

Selecting entities that are superordinates of entities in human BLO_{DG} . The FP included entities (such as `Reeds` in `MusicPinta` and `Secretarial` and `Related Occupation` in the `Occupation` class hierarchy in `L4All`) which are well represented in the graph (`Reeds` has 36 subclasses linked to 60 DBpedia categories; `Secretarial` and `Related Occupation` has 8 subclasses and 800 instances). Although these entities are not close to human cognitive structures, they provide direct links to entities in human BLO_{DG} (`Reeds` links to `Accordion`; `Secretarial` and `Related Occupation` links to `Administrative` and `Secretarial Occupation`). We therefore see such FP as ‘good picks’, as they provide *bridges* to human BLO_{DG} entities.

5.6 Summary

In this Chapter we adapted Cognitive Science experimental approaches for deriving the BLO , and presented an algorithm to capture the human BLO_{DG} that correspond to human cognitive structures over a data graph. We implemented the BLO algorithm in the same data graphs used in the implementation of the KA_{DG} algorithms in Chapter 4. We used the obtained set of human BLO_{DG} to evaluate the KA_{DG} algorithms. The evaluation validates the KA_{DG} algorithms, which enables their adoption over different domains and application contexts.

The evaluation approach presented in this Chapter contributes to adopting KA_{DG} algorithms to develop usable semantic data graph exploration applications by providing:

- formal description of an algorithm for identifying human BLO_{DG} which correspond to human cognitive structures over a data graph;
- implementation of the human BLO_{DG} algorithm and utilization to evaluate KA_{DG} algorithms over the two application contexts for data graph exploration - semantic browsing (in musical instrument domain) and semantic search (in career domain);
- analysis of the performance of KA_{DG} algorithms including hybridisation heuristics, using the benchmarking sets of human BLO_{DG} identified by humans.

The obtained sets of human BLO_{DG} and KA_{DG} can have three broad implications: (i) to improve users' exploration of large data graphs; (ii) to facilitate search by suggesting core concepts in the domain; and (iii) to reengineer the ontology to better align with human cognitive structures. We are focusing on the first implication, and are devising exploration paths to expand users' knowledge while exploring a data graph, which will be discussed in the next Chapters.

Chapter 6

Exploration Strategies Based on Subsumption

6.1 Introduction

In the previous Chapters, we discussed the importance of identifying knowledge anchors in a data graph (KA_{DG}) to apply the subsumption theory for meaningful learning for generating exploration paths for knowledge expansion in data graphs. Accordingly, we utilised the Cognitive science notion of Basic Level Objects (BLO) introduced by Rosch, et al. [22], to develop two groups of metrics for identifying KA_{DG} , with the corresponding algorithms for implementing these metrics. The KA_{DG} algorithms were implemented and evaluated over two application contexts – semantic browsing (in musical instrument domain) and semantic search (in career domain). Based on quantitative and qualitative analysis, the strengths and limitations of the KA_{DG} metrics were assessed, and a hybridization approach to enhance the performance of the metrics approach was proposed. This allowed as to identify the final set of knowledge anchors – set of KA_{DG} in MusicPinta and set of KA_{DG} in L4All.

The aim of this Chapter is to develop an automatic approach for generating exploration paths in data graphs which is underpinned by the subsumption theory for meaningful learning [21] utilising knowledge anchors. This brings two challenges:

- How to find the closest knowledge anchor to the first entity of an exploration path?. When a user starts his/her exploration from a single entity, referred as the first entity, that is not a knowledge anchor, we need a mechanism for identifying the most appropriate knowledge anchor, especially that there may be many knowledge anchors in the graph; and
- How to use the closest knowledge anchor to subsume new entities for generating an exploration path?. After identifying the closest knowledge anchor, the user will be directed to that anchor. Then we need a mechanism for selecting which entities to subsume and in what order, to generate an exploration path.

Next in this Chapter, we will describe an algorithm which calculates semantic similarity values between the first entity of an exploration path and each of the knowledge anchors in the data graph, used to identify the closest knowledge anchor to the first entity (Section 6.2). Also in this Section, we will describe a subsumption algorithm which utilises the closest

knowledge anchor to subsume entities used to generate an exploration path in the data graph. After that in Section 6.3, we will describe implementation illustrations of the subsumption algorithm over MusicPinta and L4All data graphs to generate exploration paths. In Section 6.4 we will discuss limitations and possible improvements of the developed algorithms and Section 6.5 will summarise the work in this Chapter.

6.2 Algorithm for Generating Exploration Paths in a Data Graph

We formally adopt the subsumption theory for meaningful learning introduced in Section 2.7.2 to develop an algorithm for generating exploration paths for knowledge expansion. *This includes two parts:*

Part 1: *finding the closest knowledge anchor to the first entity of an exploration path.* In unifocal browsing (pivoting), a user starts his/her exploration from a single entity in the graph, also referred as *first entity* v_s of an exploration path. In order to start the subsumption process, the user has to be directed from this first entity v_s to a knowledge anchor in the data graph, from where links to learn new entities can be made. There can be several knowledge anchors in a data graph, requiring a mechanism to identify the most appropriate knowledge anchor in the data graph v_{KA} . For this, we will focus on the closest knowledge anchor to the first entity of the path. ‘Closest’ will be measured with semantic similarity.

Part 2: *using the chosen knowledge anchor to subsume new entities.* Chosen knowledge anchor v_{KA} can have many subclass entities that exist at different levels of abstractions. Hence, it is important to identify which subclasses to subsume and in what order for generating an exploration path for the user. Furthermore, we will also include appropriate narrative scripts between the entities in the exploration path. Providing meaningful narrative scripts between entities can help layman users to create meaningful relationships between familiar entities they already know in their cognitive structures and the new subsumed entities.

6.2.1 Finding the Closest KA_{DG}

Let v_s be the first entity of an exploration path. The first entity v_s can be any class entity in the graph (i.e. any entity in the subsumption class hierarchy of all entities linked via the subsumption relationship `rdfs:subClassOf`). If v_s is a knowledge anchor ($v_s \in KA_{DG}$), then there is no need to identify the closest knowledge anchor (i.e. find another anchor in the data graph), and the subsumption process can start immediately from v_s . However, if v_s is not a knowledge anchor ($v_s \notin KA_{DG}$), then v_s can be *superordinate*, *subordinate*, or *sibling*

of one or more knowledge anchors. In this case, an automatic approach for identifying the closest knowledge anchor v_{KA} to v_s is required. To find the closest knowledge anchor, we calculate the semantic similarity between v_s and every knowledge anchor in the data graph $v_i \in KA_{DG}$. The semantic similarity between two entities in the class hierarchy is based on their distances (i.e. length of data graph trajectory between both entities). Due to the fact that class hierarchies exist in most data graphs, we adopt the semantic similarity metric from [163] and apply it in the context of a data graph. The semantic similarity between v_s and a knowledge anchor v_i is calculated as:

$$sim(v_s, v_i) := \frac{2 \cdot depth(lca(v_s, v_i))}{depth(v_s) + depth(v_i)} \quad (5)$$

where, $lca(v_s, v_i)$ is the *least common ancestor* of v_s and v_i , and $depth(v)$ is a function for identifying the depth of the entity v in the class hierarchy. Algorithm 6.1 describes how the semantic similarity metric is applied to identify the closest knowledge anchor v_{KA} to v_s .

Algorithm 6.1: Identifying Closest KA_{DG}

Input: $DG = \langle V, E, T \rangle$, $v_s \in V$, $KA_{DG} = \{v_1, v_2, \dots, v_i\}$

Output: v_{KA} – the closest knowledge anchor with the highest semantic similarity value to v_s

1. **if** $v_s \in KA_{DG}$ **then** *// v_s is a knowledge anchor*
 2. $v_{KA} := v_s$;
 3. **else** *// v_s is NOT a knowledge anchor*
 4. $S := \{ \}$; *//list for storing semantic similarity values*
 5. **for all** $v_i \in KA_{DG}$ **do** *//for all knowledge anchors*
 6. $CA := \{ \}$; *//list for storing common ancestors of v_s and v_i*
 7. $L := \{ \}$; *//list for storing trajectory lengths*
 8. $CA \leftarrow common_ancestors(v_s, v_i)$; *//get all common ancestors of v_s and v_i*
 9. **for all** $v_{ca} \in CA$ *//for all common ancestors*
 10. $L \leftarrow length(v_{ca}, v_i)$; *//get length of the trajectory between v_{ca} and v_i*
 11. **end for** ;
 12. $v_{lca} := v_{ca}$ with least length in L ; *//get least common ancestor of v_s and v_i*
 13. $S \leftarrow \frac{2 \cdot depth(v_{lca})}{depth(v_s) + depth(v_i)}$; *//calculate semantic similarity between v_s and v_i*
 14. **end for** ;
 15. $v_{KA} := v_i$ with maximum semantic similarity value in list S .
 16. **end if** ;
-

The algorithm takes a data graph, the first entity v_s of an exploration path and a set of knowledge anchors KA_{DG} as an input, and identifies the closest knowledge anchor $v_{KA} \in$

KA_{DG} with highest semantic similarity value to v_s . If the first entity v_s belongs to the set of knowledge anchors $v_s \in KA_{DG}$ (line 1), then the first entity v_s is identified as the closest knowledge anchor v_{KA} (line 2). However, if the first entity v_s does not belong to the set of knowledge anchors in the data graph $v_s \notin KA_{DG}$ (line 3), the following steps are conducted:

- The algorithm initialises a list S to store semantic similarity values between the v_s and every knowledge anchor $v_i \in KA_{DG}$ (line 4).
- For every knowledge anchor $v_i \in KA_{DG}$ (line 5), the algorithm initiates two lists: list CA for storing the common ancestors (i.e. common superclasses) of v_s and v_i (line 6), and list L for storing the trajectory lengths between the common ancestors in list CA and the knowledge anchor v_i (line 7).
- The algorithm in (line 8) uses a function *common_ancestors* (v_s, v_i) which retrieves all *common ancestors* of v_s and v_i in the class hierarchy, and stores them in list CA . The function uses the following SPARQL query to retrieve common ancestors of v_s and v_i :

```
SELECT distinct ?common_ancestor
WHERE {
    v_s rdfs:subClassOf ?common_ancestor.
    v_i rdfs:subClassOf ?common_ancestor }.
```

- For every common ancestor in list CA (line 9), the algorithm identifies the *length* of the data graph trajectories between v_i and each of the common ancestors v_{ca} in CA , via the following SPARQL query:

```
SELECT (count(?intermediate)-1 as ?length)
WHERE {
    v_i rdfs:subClassOf ?intermediate.
    ?intermediate rdfs:subClassOf v_ca .}
```

- The common ancestor v_{ca} with least trajectory length with v_i in list L is identified as the *least common ancestor* v_{lca} (line 12).
- After that, the semantic similarity metric (Formula 5) is applied (line 13). The metric includes identifying depths of v_s , v_i , and v_{lca} . Depth of entity v is identified using the following SPARQL query:

```
SELECT (count(?intermediate)-1 as ?depth)
WHERE {
    v rdfs:subClassOf ?intermediate.
    ?intermediate rdfs:subClassOf r .}
```

where r is the root entity in the data graph.

- The semantic similarity value is then inserted into the list S (line 13), and the knowledge anchor with the highest similarity value to the first entity v_s will be identified as the closest knowledge anchor v_{KA} (line 15). Implementation example can be found here⁵⁶.

6.2.2 Subsumption Using the Closest KA_{DG}

The closest knowledge anchor v_{KA} is used to subsume new entities and to generate transition narratives in the exploration path. Table 6.1, describes the types of transition narratives and corresponding scripts used between entities of an exploration path.

Table 6.1. Narrative types of transitions between entities in an exploration path

Narrative Type	From_entity	To_entity	Description	Output script (From_entity, Narrative type, To_entity)	Illustrative example
N_1	v_s	v_{KA}	v_s is subclass of v_{KA}	"You may find it useful to know that v_s belongs to a familiar and well-known class – v_{KA} . Let's explore v_{KA} ."	
N_2	v_s	v_{KA}	v_s is superclass of v_{KA}	"You may find it useful to know that there is a well-known class – v_{KA} that belongs to v_s . Let's explore v_{KA} ."	
N_3	v_s	v_{KA}	v_s and v_{KA} are siblings	"You may find it useful to know that v_s is similar to a well-known class – v_{KA} . Let's explore v_{KA} ."	
N_4	v_{KA}	v'_{KA}	v'_{KA} is subclass of v_{KA} and superclass of v_s	"You may find it useful to know that v'_{KA} belongs to v_{KA} , and v_s belongs to v'_{KA} . Let's explore v'_{KA} ."	
N_5	v_{KA}	v''_{KA}	v''_{KA} is subclass of v_{KA} and not superclass of v_s	"You may find it useful to know that v''_{KA} belongs to v_{KA} . Let's explore v''_{KA} ."	

⁵⁶ <https://doi.org/10.5518/304>

Algorithm 6.2 describes our approach for generating exploration paths following Ausubels' subsumption theory for meaningful learning (described in Section 2.7.2), where the closest knowledge anchor is used to subsume subordinate entities (i.e. subclass of v_{KA}). The narrative types (N_1, N_2, \dots, N_5) used correspond to the narratives types in Table 6.1.

Algorithm 6.2: Subsumption Using the Closest KA_{DG}

Input: $DG = \langle V, E, T \rangle$, $v_s \in V$, $v_{KA} \in KA_{DG}$, $m = \text{length of exploration path}$, $e = \text{rdfs:subClassOf}$

Output: an exploration path P of length m .

1. $P := \{\}$; *//empty exploration path P*
 2. **if** $\exists \langle v_s, e, v_{KA} \rangle$ **then** *//v_s is subclass of v_{KA}*
 3. $P \leftarrow \langle v_s, \text{script}(N_1), v_{KA} \rangle$; *//insert transition narrative from v_s to v_{KA} using N₁*
 4. $m --$; *//reduce length of P by one*
 5. V'_{KA} is set of all $v'_{KA} : \exists \langle v_s, e, v'_{KA} \rangle \wedge \exists \langle v'_{KA}, e, v_{KA} \rangle$ *//intermediate classes between v_s and v_{KA}*
 6. $Q' := \{\}$; *//list for storing intermediate classes between v_s and v_{KA}*
 7. $Q' \leftarrow \text{sortDepth}(V'_{KA})$; *//sort classes in V'_{KA} in ascending order from least depth*
 8. **for** ($i := 1$; $i \leq |Q'| \wedge m \geq 0$; $i ++$) *//start subsuming classes that are in Q'*
 9. $P \leftarrow \langle v_{KA}, \text{script}(N_4), Q'[i] \rangle$; *//insert transition narrative from v_{KA} to Q'[i] using N₄*
 10. $m --$; *//reduce length of P by one*
 11. **end for**;
 12. **else if** $\exists \langle v_{KA}, e, v_s \rangle$ **then** *//v_s is superclass of v_{KA}*
 13. $P \leftarrow \langle v_s, \text{script}(N_2), v_{KA} \rangle$; *//insert transition narrative from v_s to v_{KA} using N₂*
 14. $m --$; *//reduce length of P by one*
 15. **else if** $\exists \langle v_s, e, v_i \rangle \wedge \exists \langle v_{KA}, e, v_i \rangle$ **then** *//v_s and v_{KA} are siblings*
 16. $P \leftarrow \langle v_s, \text{script}(N_3), v_{KA} \rangle$; *//insert transition narrative from v_s to v_{KA} using N₃*
 17. $m --$; *//reduce length of P by one*
 18. **end if**;
 19. V''_{KA} is set of all $v''_{KA} : \exists \langle v''_{KA}, e, v_{KA} \rangle \wedge v''_{KA} \notin Q'$ *//subclasses of v_{KA} that are not in list Q'*
 20. $Q'' := \{\}$; *//list for storing subclasses of v_{KA} that are not in list Q'*
 21. $Q'' \leftarrow \text{sortDepthDensity}(V''_{KA})$; *//sort classes in V''_{KA} based on depth and density*
 22. **for** ($j := 1$; $j \leq |Q''| \wedge m \geq 0$; $j ++$) **do** *//start subsuming classes that are in Q''*
 23. $P \leftarrow \langle v_{KA}, \text{script}(N_5), Q''[j] \rangle$; *//insert transition narrative from v_{KA} to Q''[j] using N₅*
 24. $m --$; *//reduce length of P by one*
 25. **end for**;
-

The algorithm takes a data graph DG , the first entity v_s , the closest knowledge anchor v_{KA} , the length of the exploration path (m), and the edge label $e = \text{rdfs:subClassOf}$ as an input, and generates an exploration path P of length m . The algorithm starts by initialising

an empty exploration path P (line 1) used to store m transition narratives. Then the algorithm starts identifying the relationship type between v_s and v_{KA} .

If v_s is subclass of v_{KA} (lines 2), then the following steps are conducted:

- The transition narrative $\langle v_s, \text{script}(N_1), v_{KA} \rangle$ from v_s to v_{KA} is inserted into P (line 3) where the function $\text{script}(N_1)$ retrieves the *script output* for narrative type N_1 from Table 6.1. The length of exploration path m is decreased by one (line 4).
- The algorithm in (line 5) identifies the set of intermediate class entities (V'_{KA}) between v_s and v_{KA} , using the following SPARQL query:

```
SELECT distinct ?intermediate
WHERE {
    v_s rdfs:subClassOf ?intermediate .
    ?intermediate rdfs:subClassOf v_{KA} .}
```

- A list Q' is created in (line 6), and the function $\text{sortDepth}()$ sorts the class entities $v'_{KA} \in V'_{KA}$ based on their depths starting from the least depth class entity (i.e. direct subclass of v_{KA}) to highest depth in ascending order, and inserts the sorted class entities into list Q' (line 7). The function $\text{sortDepth}()$ identifies the depth of a class entity v'_{KA} via the following SPARQL query:

```
SELECT (count(?intermediate)-1 as ?depth)
WHERE {
    v'_{KA} rdfs:subClassOf ?intermediate.
    ?intermediate rdfs:subClassOf r .}
```

where r is the root entity in the data graph.

- The algorithm in (lines 8 – 11) uses the closest knowledge anchor v_{KA} to subsume the class entities Q' . The transition narrative $\langle v_{KA}, \text{script}(N_4), Q'[i] \rangle$ from v_{KA} to $Q'[i]$ (i.e. v'_{KA}) is inserted into P (line 9) where the function $\text{script}(N_4)$ retrieves the *script output* for narrative type N_4 from Table 6.1. The length of exploration path m is decreased by one (line 10).

If v_s is superclass of v_{KA} (lines 12), then the transition narrative $\langle v_s, \text{script}(N_2), v_{KA} \rangle$ from v_s to v_{KA} is inserted into P (line 13) where the function $\text{script}(N_2)$ retrieves the *script output* for narrative type N_2 from Table 6.1. The length of exploration path m is decreased by one (line 14). If v_s and v_{KA} are siblings (i.e. share same superclass) (line 15), then the

transition narrative $\langle v_s, \text{script}(N_3), v_{KA} \rangle$ from v_s to v_{KA} is inserted into P (line 16) where the function $\text{script}(N_3)$ retrieves the *script output* for narrative type N_3 from Table 6.1. The length of exploration path m is decreased by one (line 17).

After identifying the relationship between v_s and v_{KA} , the following steps are conducted:

- Identify the set of subclass entities (V''_{KA}) of v_{KA} which do not belong to Q' (line 19). A list Q'' is created in (line 20), and the function $\text{sortDepthDensity}()$ sorts the class entities $v''_{KA} \in V''_{KA}$ based on two their depths and density, as the following steps:
 - (i) Identify the depth of the class entities (similar to function $\text{sortDepth}()$ described above).
 - (ii) Identify the density (using degree centrality) of the class entities based on number of subclasses.
 - (iii) Sort the class entity starting from the least depth (i.e. direct subclasses of v_{KA}) to highest depth in ascending order, and from highest density to least density (i.e. first sort using depth, if two or more entities are at the same depth, then sort these entities based on their density from highest to lowest). The sorted class entities are inserted into list Q'' (line 21).
- The algorithm in (lines 22 – 25) uses the closest knowledge anchor v_{KA} to subsume the class entities in Q'' . The transition narrative $\langle v_{KA}, \text{script}(N_5), Q''[j] \rangle$ from v_{KA} to $Q''[j]$ (i.e. v''_{KA}) is inserted into P (line 23) where the function $\text{script}(N_5)$ retrieves the *script output* for narrative type N_5 from Table 6.1. The length of exploration path m is decreased by one (line 24). Implementation example of the algorithm is here⁵⁷.

In the next section, we will provide several illustration examples for generating exploration paths in the data graphs of the two application context of this research: MusicPinta and L4All.

6.3 Implementation Illustration

To generate exploration paths, it is important to identify the start entity v_s . The first entity v_s in an exploration path can be any class entity in a data graph. It can be a leaf entity $v \in L$ or a category entity $v \in C$. Furthermore, v_s can be related to the closest knowledge

⁵⁷ <https://doi.org/10.5518/304>

anchor V_{KA} in different ways (e.g. V_s is subclass of V_{KA} ; V_s is superclass of V_{KA} ; V_s and V_{KA} are siblings). We selected V_s from MusicPinta and L4All data graphs which reflect the above cases. Table 6.2 describes the selected V_s . Overall 7 V_s were selected (4 V_s from `String Instrument` and `Wind instrument` class hierarches in MusicPinta, and 3 V_s from `Occupation` class hierarchy in L4All). The reason for selecting V_s from these three class hierarchies is since they have the richest class representation in terms of the number of classes and the hierarchy depth among other hierarchies (as discussed in Sections 3.4.1, 3.4.2, respectively). Furthermore, these class hierarchies have the highest number of knowledge anchors among other class hierarchies (for MusicPinta: there are 9 anchors in the `String Instrument` class hierarchy and 10 anchors in the `Wind Instrument` class hierarchy – out of 23 anchors in MusicPinta data graph; for L4All: there are 33 anchors in the `Occupation` class hierarchy compared to only 8 anchors in the `Subject` class hierarchy).

Table 6.2. First entities V_s used in implementation of subsumption algorithm to generate exploration paths

Data Graph	Class hierarchy	V_s	V_{KA}	m	Relationship between V_s and V_{KA}	Entity type
MusicPinta	String Instrument	Biwa	Lute	4	V_s is subclass of V_{KA}	Leaf entity
		Kinnor	Harp	6	V_s and V_{KA} are siblings	Category entity
	Wind Instrument	Bansuri	Flute	4	V_s is subclass of V_{KA}	Leaf entity
		Valved brass instruments	Tuba	5	V_s is superclass of V_{KA}	Category entity
L4All	Occupation	Secondary Education Teaching Professional	Teaching Professional	5	V_s is subclass of V_{KA}	Leaf entity
		Transport and Mobile Machine Drivers and Operatives	Transport Drivers and Operatives	4	V_s is superclass of V_{KA}	Category entity
		Leisure and Other Personal Service Occupations	Caring Personal Service Occupations	4	V_s and V_{KA} are siblings	Category entity

To illustration examples for generating exploration paths, we apply Algorithms 6.1 and 6.2 for every V_s in Table 6.2.

6.3.1 Identify Closest Knowledge Anchors

We applied Algorithm 6.1 to identify the closest knowledge anchor to first entity. Using this algorithms, we identified the semantic similarity between every first entity in Table

6.2 and each of the KA_{DG} in MusicPinta and L4All. Tables 6.3 and 6.4 list the highest 5 semantic similarity values for every first entity and the knowledge anchors in MusicPinta and L4All, respectively.

Table 6.3. Semantic similarity vales between first entities and KA_{DG} in MusicPinta

Biwa		Kinnor		Bansuri		Valved brass instruments	
Knowledge Anchor	Similarity Value	Knowledge Anchor	Similarity Value	Knowledge Anchor	Similarity Value	Knowledge Anchor	Similarity Value
Tambura	0.72	Harp	0.75	Flute	0.40	Tuba	0.57
Lute	0.60	Lyre	0.57	Reeds	0.40	Woodwind	0.40
Guitar	0.44	Bouzouki	0.57	Accordion	0.36	Brass	0.40
Lyre	0.44	Guitar	0.57	Saxophone	0.33	Organ	0.40
Bouzouki	0.44	Lute	0.50	Zurna	0.33	Reeds	0.33

Table 6.4. Semantic similarity between Biwa and KA_{DG} ; and similarity between Bansuri and KA_{DG}

Secondary Education Teaching Professionals		Transport and Mobile Machine Drivers and Operatives		Leisure and Other Personal Service Occupations	
Knowledge Anchor	Similarity Value	Knowledge Anchor	Similarity Value	Knowledge Anchor	Similarity Value
Teaching Professionals	0.75	Transport Drivers and Operatives	0.66	Caring Personal Service Occupations	0.80
Teaching and Research Professionals	0.57	Process Operatives	0.66	Administrative and Secretarial Occupations	0.50
Science and Technology Professionals	0.57	Sales and Customer Service Occupations	0.50	Sales and Customer Service Occupations	0.50
Engineering Professionals	0.50	Professional Occupations	0.50	Professional Occupations	0.50
Librarians and Related Professionals	0.50	Personal Service Occupations	0.50	Personal Service Occupations	0.50

6.3.2 Use Closest Knowledge Anchors to Generate Exploration Paths

We apply Algorithm 6.2 to illustrate examples for exploration paths for every v_s in Table 6.2 using the chosen closest knowledge anchor v_{KA} .

6.3.2.1 Examples of Exploration Paths in MusicPinta

Example 1. Exploration path for Biwa

Figure 6.1 shows an example for generating the exploration path for the instrument Biwa (first entity in the exploration path) using the closest knowledge anchor Lute.

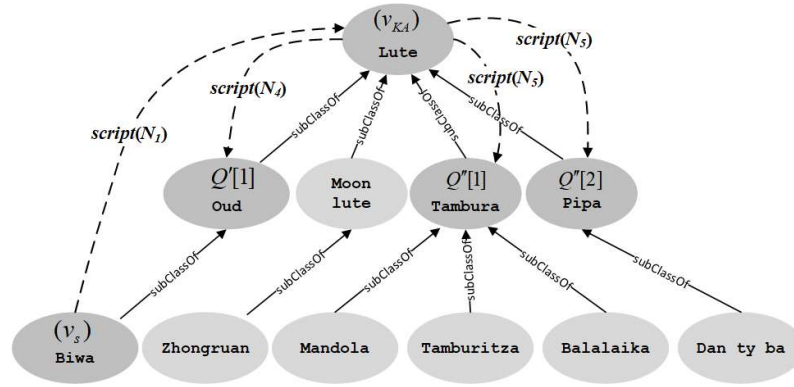


Figure 6.1. Extract from the `String Instrument` class hierarchy in MusicPinta showing exploration path of length 4 generated with first entity `Biwa`.

The exploration path for `Biwa` (has length of $m = 4$) and was generated using the closest knowledge anchor `Lute` and narrative types N_1, N_4, N_5 given in Table 6.1. The first transition narrative in the exploration path is between the `Biwa` and the closest knowledge anchor `Lute` (using N_1 in Table 6.1). After that, the closest knowledge anchor `Lute` is used to subsume new entities and generate transition narratives in the exploration path using narrative types N_4 (subsume intermediate class entities between `Lute` and `Biwa` – line 9 in Algorithm 6.2) and N_5 (subsume subclasses of `Lute` other than class entities subsumed using N_4 – line 23 in Algorithm 6.2). The transition narratives that were followed in generating the exploration path for `Biwa` are listed in Table 6.5.

Table 6.5. Transition narratives used for generating the exploration path for `Biwa`

Transition Narratives for path of <code>Biwa</code>	Narrative Script
$\langle v_s, \text{script}(N_1), v_{KA} \rangle$	You may find it useful to know that ‘ <i>Biwa</i> ’ belongs to a familiar and well-known instrument called ‘ <i>Lute</i> ’. Let’s explore ‘ <i>Lute</i> ’.
$\langle v_{KA}, \text{script}(N_4), Q'[1] \rangle$	You may also find it useful to know that ‘ <i>Oud</i> ’ belongs to ‘ <i>Lute</i> ’, and ‘ <i>Biwa</i> ’ belongs to ‘ <i>Oud</i> ’. Let’s explore ‘ <i>Oud</i> ’.
$\langle v_{KA}, \text{script}(N_5), Q''[1] \rangle$	You may also find it useful to know that ‘ <i>Tambura</i> ’ belongs to ‘ <i>Lute</i> ’. Let’s explore ‘ <i>Tambura</i> ’.
$\langle v_{KA}, \text{script}(N_5), Q''[2] \rangle$	You may also find it useful to know that ‘ <i>Pipa</i> ’ belongs to ‘ <i>Lute</i> ’. Let’s explore ‘ <i>Pipa</i> ’.

Example 2. Exploration path for `Kinnor`

Figure 6.2 shows an example for generating the exploration path for the instrument `Kinnor` (first entity in the exploration path) using closest knowledge anchor `Harp`.

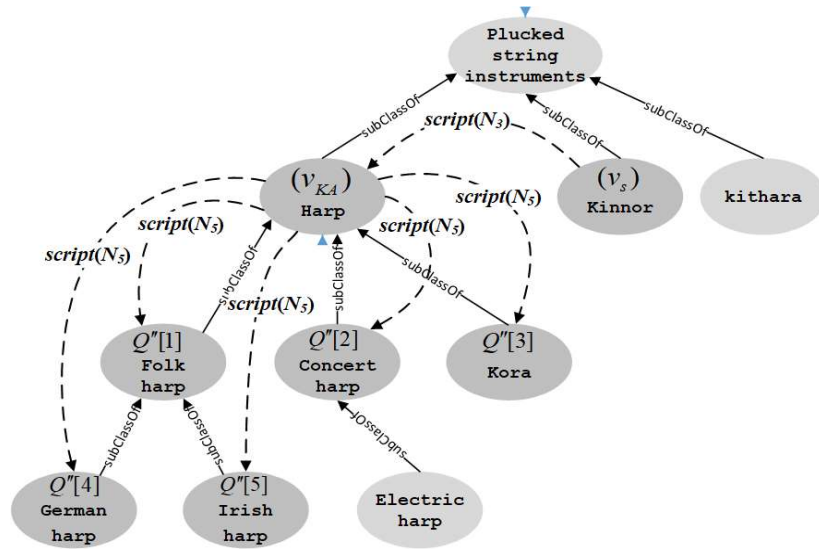


Figure 6.2. Extract from the String Instrument class hierarchy in MusicPinta showing exploration path of length 6 generated with first entity Kinnor.

The exploration path for Kinnor (has length of $m = 6$) and was generated using the closest knowledge anchor Harp and narrative types N_3, N_5 given in Table 6.1. The first transition narrative is between Kinnor and closest knowledge anchor Harp (using N_3 in Table 6.1). After that, the closest knowledge anchor Harp is used to subsume new entities and to generate transition narratives in the exploration path using narrative type N_5 (subsume subclass entities of Harp – line 23 in Algorithm 6.2). Subclasses of harp were subsumed from least depth (i.e. direct subclasses ,e.g. Folk Harp) to highest depth (e.g. German Harp). The transition narratives that were followed in generating the exploration path for Kinnor are listed in Table 6.6.

Table 6.6. Transition narratives used for generating the exploration path for Kinnor

Transition Narratives for path of Kinnor	Narrative Script
$\langle v_s, script(N_3), v_{KA} \rangle$	You may find it useful to know that ‘Kinnor’ is similar to a familiar and well-known instrument called ‘Harp’. Let’s explore ‘Harp’.
$\langle v_{KA}, script(N_5), Q''[1] \rangle$	You may also find it useful to know that ‘Folk Harp’ belongs to ‘Harp’. Let’s explore ‘Folk Harp’.
$\langle v_{KA}, script(N_5), Q''[2] \rangle$	You may also find it useful to know that ‘Concert Harp’ belongs to ‘Harp’. Let’s explore ‘Concert Harp’.
$\langle v_{KA}, script(N_5), Q''[3] \rangle$	You may also find it useful to know that ‘Kora’ belongs to ‘Harp’. Let’s explore ‘Kora’.
$\langle v_{KA}, script(N_5), Q''[4] \rangle$	You may also find it useful to know that ‘German Harp’ belongs to ‘Harp’. Let’s explore ‘German Harp’.
$\langle v_{KA}, script(N_5), Q''[5] \rangle$	You may also find it useful to know that ‘Irish Harp’ belongs to ‘Harp’. Let’s explore ‘Irish Harp’.

Example 3. Exploration path for Bansuri

Figure 6.3 shows an example for generating the exploration path for the instrument Bansuri (first entity in exploration path) using closest knowledge anchor Flute.

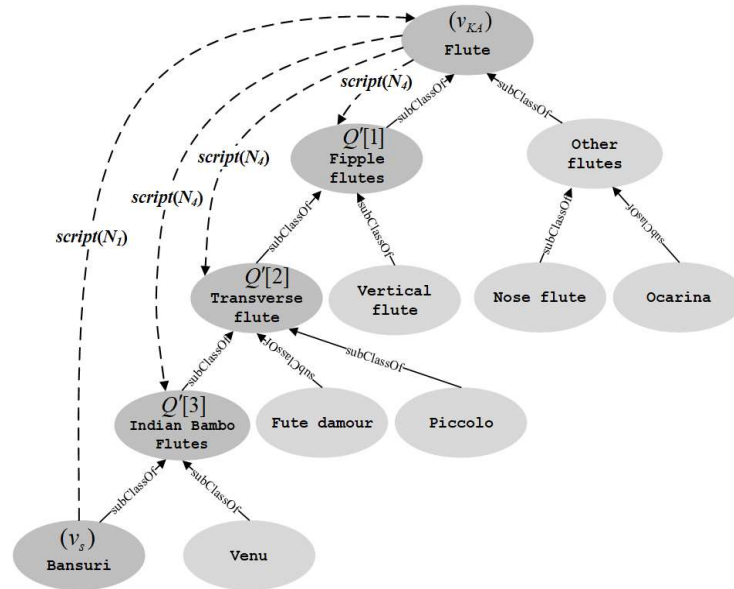


Figure 6.3. Extract from the Wind Instrument class hierarchy in MusicPinta showing exploration path of length 4 generated with first entity Bansuri .

From the results in Table 6.3, it was noticed that the knowledge anchors Flute, and Reeds have the same semantic similarity value with the first entity Bansuri. By examining the relationship between Bansuri and each of the anchors (Flute, Reeds) we noticed that Flute is a direct subclass of Bansuri, whereas there is no direct relationship (subclass, superclass, sibling) between Bansuri and Reeds. Therefore, Flute is identified as the closest knowledge anchor.

The exploration path for Bansuri (has length of $m = 4$) was generated using the closest knowledge anchor Flute and narrative types N_1, N_4 given in Table 6.1. The first transition narrative is between the first entity Bansuri and the closest knowledge anchor Flute (using N_1 in Table 6.1). After that, the closest knowledge anchor Flute is used to subsume new entities and to generate transition narratives in the exploration path using narrative type N_4 (subsume intermediate entities between Flute and Bansuri line 9 in Algorithm 6.2). Transition narratives that were followed in generating the exploration path for Bansuri are listed in Table 6.7.

Table 6.7. Transition narratives used for generating exploration path for Bansuri

Transition Narratives for path of Bansuri	Narrative Script
$\langle v_s, script(N_1), v_{KA} \rangle$	<i>You may find it useful to know that 'Bansuri' belongs to a familiar and well-known instrument called 'Flute'. Let's explore 'Flute'.</i>
$\langle v_{KA}, script(N_4), Q'[1] \rangle$	<i>You may also find it useful to know that 'Fipple Flute' belongs to 'Flute', and 'Bansuri' belongs to 'Fipple Flute'. Let's explore 'Fipple Flute'.</i>
$\langle v_{KA}, script(N_4), Q'[2] \rangle$	<i>You may also find it useful to know that 'Transverse Flute' belongs to 'Flute', and 'Bansuri' belongs to 'Transverse Flute'. Let's explore 'Transverse Flute'.</i>
$\langle v_{KA}, script(N_4), Q'[3] \rangle$	<i>You may also find it useful to know that 'Indian Bamboo Flutes' belongs to 'Flute', and 'Bansuri' belongs to 'Indian Bamboo Flutes'. Let's explore 'Indian Bamboo Flutes'.</i>

Example 4. Exploration path for Valve Brass Instruments

Figure 6.4 shows an example for generating the exploration path for the instrument Valve Brass Instruments (the first entity in the exploration path) using the closest knowledge anchor Tuba.

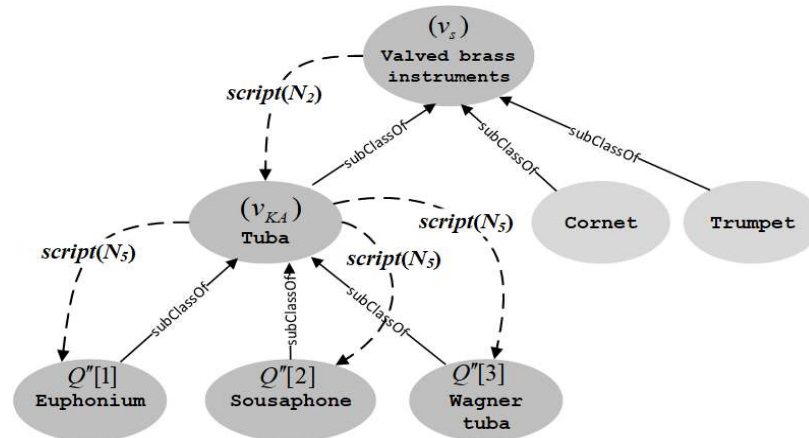


Figure 6.4. Extract from the Wind Instrument class hierarchy in MusicPinta showing exploration path of length 4 generated with first entity Valved Brass Instruments.

Although we aimed to generate an exploration path of length ($m = 5$) for Valved Brass Instruments, however the generated exploration path length was 4 since there were only 3 subclasses to subsume. The exploration path was generated using the closest knowledge anchor Tuba and narrative types N_2, N_3 given in Table 6.1. The first transition narrative is between the first entity Valve Brass Instruments and the closest knowledge anchor Tuba (using N_2 in Table 6.1). After that, the closest knowledge anchor Tuba is used to subsume new Q' entities and to generate transition narratives in the exploration path using narrative type N_3 (subsume subclasses of Tuba – line 23 in Algorithm 6.2). The transition narratives that were followed in generating the exploration path for Valve Brass Instruments are listed in Table 6.8.

Table 6.8. Transition narratives used for generating the exploration path for Valve Brass Instruments

Transition Narratives for path of Valve Brass Instruments	Narrative Script
$\langle v_s, script(N_2), v_{KA} \rangle$	You may find it useful to know that a familiar and well-known instrument called ‘Tuba’ belongs to the instrument ‘Valve Brass Instruments’. Let's explore ‘Tuba’.
$\langle v_{KA}, script(N_5), Q''[1] \rangle$	You may also find it useful to know that ‘Euphonium’ belongs to ‘Tuba’. Let's explore ‘Euphonium’.
$\langle v_{KA}, script(N_5), Q''[2] \rangle$	You may also find it useful to know that ‘Sousaphone’ belongs to ‘Tuba’. Let's explore ‘Sousaphone’.
$\langle v_{KA}, script(N_5), Q''[3] \rangle$	You may also find it useful to know that ‘Wagner Tuba’ belongs to ‘Tuba’. Let's explore ‘Wagner Tuba’.

6.3.2.2 Examples of Exploration Paths in L4All

Example 1. Exploration path for Secondary Education Teaching Professionals

Figure 6.5 shows an example for generating the exploration path for the instrument Secondary Education Teaching Professionals (first entity in exploration path) using closest knowledge anchor Teaching Professionals.

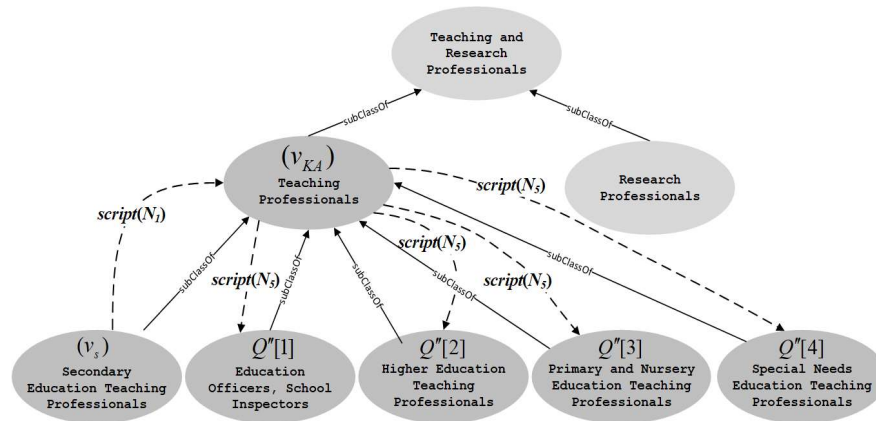


Figure 6.5. Extract from the Occupation class hierarchy in L4All showing exploration path of length 5 generated with first entity Secondary Education Teaching Professionals.

The exploration path for Secondary Education Teaching Professionals (has length of $m = 5$) was generated using the closest knowledge anchor Teaching Professionals and narrative types N_1, N_5 given in Table 6.1. The first transition narrative is between the first entity Secondary Education Teaching Professionals and the closest knowledge anchor Teaching Professionals (using N_1 in Table 6.1). After that, the closest knowledge anchor Teaching Professionals is used to subsume new entities and to generate transition narratives in the exploration path using narrative type N_5 (subsume subclass entities of Teaching Professionals – line 23 in Algorithm

6.2). Transition narratives that were followed in generating the exploration path for Secondary Education Teaching Professionals are listed in Table 6.9.

Table 6.9. Transition narratives used for generating exploration path for Secondary Education Teaching Professionals

Transition Narratives for path of Secondary Education Teaching Professionals	Narrative Script
$\langle v_s, script(N_1), v_{KA} \rangle$	You may find it useful to know that ' <i>Secondary Education Teaching Professionals</i> ' belongs to a familiar and well-known instrument called ' <i>Teaching Professionals</i> '. Let's explore ' <i>Teaching Professionals</i> '.
$\langle v_{KA}, script(N_5), Q''[1] \rangle$	You may also find it useful to know that ' <i>Education Officers, School Inspectors</i> ' belongs to ' <i>Teaching Professionals</i> '. Let's explore ' <i>Education Officers, School Inspectors</i> '.
$\langle v_{KA}, script(N_5), Q''[2] \rangle$	You may also find it useful to know that ' <i>Higher Education Teaching Professionals</i> ' belongs to ' <i>Teaching Professionals</i> '. Let's explore ' <i>Higher Education Teaching Professionals</i> '.
$\langle v_{KA}, script(N_5), Q''[3] \rangle$	You may also find it useful to know that ' <i>Primary and Nursery Education Teaching Professionals</i> ' belongs to ' <i>Teaching Professionals</i> '. Let's explore ' <i>Primary and Nursery Education Teaching Professionals</i> '.
$\langle v_{KA}, script(N_5), Q''[4] \rangle$	You may also find it useful to know that ' <i>Special Needs Education Teaching Professionals</i> ' belongs to ' <i>Teaching Professionals</i> '. Let's explore ' <i>Special Needs Education Teaching Professionals</i> '.

Example 2. Exploration path for Leisure and Other Personal Service Occupations

Figure 6.6 shows an example for generating exploration path for instrument Leisure and Other Personal Service Occupations (first entity in exploration path) using closest knowledge anchor Caring Personal Service Occupations.

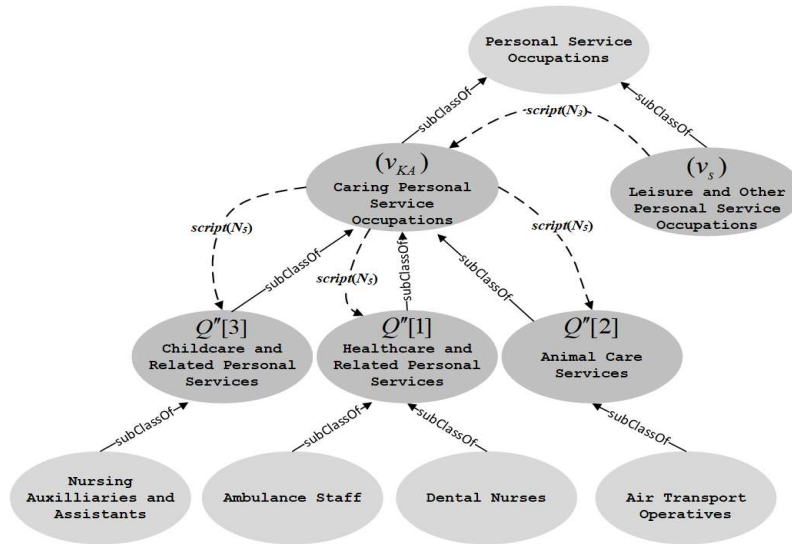


Figure 6.6. Extract from the Occupation class hierarchy in L4All showing exploration path of length 4 generated with first entity Leisure and Other Personal Service Occupations.

The exploration path for Leisure and Other Personal Service Occupations was generated using the closest knowledge anchor Caring Personal Service Occupations and narrative types N_3, N_5 given in Table 6.1. The first transition narrative is between the first entity Leisure and Other Personal Service Occupations and the closest knowledge anchor Caring Personal Service Occupations (using N_3 in Table 6.1). After that, the closest knowledge anchor Caring Personal Service Occupations is used to subsume new entities and to generate transition narratives in the exploration path using narrative type N_5 (subsume subclasses of Caring Personal Service Occupations – line 23 in Algorithm 6.2). Healthcare and Related Personal Services was subsumed first since it has the highest density (line 21 in Algorithm 6.2) compared to the other two subclasses of knowledge anchor: Childcare and Related Personal Services and Childcare and Related Personal Services. Transition narratives that were followed in generating the exploration path for Leisure and Other Personal Service Occupations are listed in Table 6.10.

Table 6.10. Transition narratives used for generating exploration path for Leisure and Other Personal Service Occupations

Transition Narratives for path of Leisure and Other Personal Service Occupations	Narrative Script
$\langle v_s, \text{script}(N_3), v_{KA} \rangle$	You may find it useful to know that ‘ <i>Leisure and Other Personal Service Occupations</i> ’ is similar to a familiar and well-known Occupation called ‘ <i>Caring Personal Service Occupation</i> ’. Let's explore ‘ <i>Caring Personal Service Occupation</i> ’.
$\langle v_{KA}, \text{script}(N_5), Q''[1] \rangle$	You may also find it useful to know that ‘ <i>Childcare and Related Personal Services</i> ’ belongs to ‘ <i>Caring Personal Service Occupations</i> ’. Let's explore ‘ <i>Childcare and Related Personal Services</i> ’.
$\langle v_{KA}, \text{script}(N_5), Q''[2] \rangle$	You may also find it useful to know that ‘ <i>Animal Care ^[SEP]Services</i> ’ belongs to ‘ <i>Caring Personal Service Occupations</i> ’. Let's explore ‘ <i>Animal Care ^[SEP]Services</i> ’.
$\langle v_{KA}, \text{script}(N_5), Q''[3] \rangle$	You may also find it useful to know that ‘ <i>Healthcare and Related Personal Services</i> ’ belongs to ‘ <i>Caring Personal Service Occupations</i> ’. Let's explore ‘ <i>Healthcare and Related Personal Services</i> ’.

Example 3. Exploration path Transport and Mobile Machine Drivers and Operatives

Figure 6.7 shows an example for generating exploration path for the instrument Transport and Mobile Machine Drivers and Operatives (first entity in exploration path) using closest knowledge anchor Transport Drivers and Operatives.

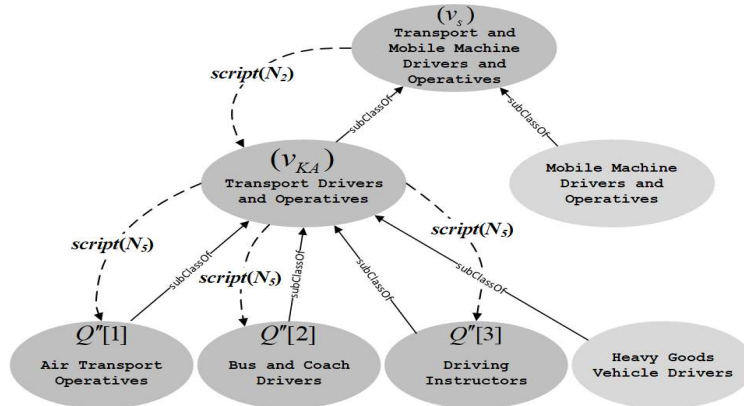


Figure 6.7. Extract from the Occupation class hierarchy in L4All showing exploration path of length 4 generated with first entity Transport and Mobile Machine Drivers and Operatives.

The exploration path for Transport and Mobile Machine Drivers and Operatives (has length of 4) was generated using the closest knowledge anchor Transport Drivers and Operatives and narrative types N_2 , N_5 given in Table 6.1. The first transition narrative is made between the first entity Transport and Mobile Machine Drivers and Operatives and the closest knowledge anchor Transport Drivers and Operatives (using N_2 in Table 6.1). After that, the closest knowledge anchor Transport Drivers and Operatives is used to subsume new entities and to generate transition narratives in the exploration path using narrative type N_5 (subsume subclasses of Transport Drivers and Operatives – line 23 in Algorithm 6.2). Transition narratives that were followed in generating the exploration path for Transport and Mobile Machine Drivers and Operatives are listed in Table 6.11.

Table 6.11. Transition narratives used for generating exploration path for Transport and Mobile Machine Drivers and Operatives

Transition Narratives for path of Transport and Mobile Machine Drivers and Operatives	Narrative Script
$\langle v_s, \text{script}(N_2), v_{KA} \rangle$	You may find it useful to know that a familiar and well-known occupation called ‘ <i>Transport Drivers and Operatives</i> ’ belongs to the Occupation ‘ <i>Transport and Mobile Machine Drivers and Operatives</i> ’. Let’s explore ‘ <i>Transport Drivers and Operatives</i> ’.
$\langle v_{KA}, \text{script}(N_5), Q''[1] \rangle$	You may also find it useful to know that ‘ <i>Air Transport Operatives</i> ’ belongs to ‘ <i>Transport Drivers and Operatives</i> ’. Let’s explore ‘ <i>Air Transport Operatives</i> ’.
$\langle v_{KA}, \text{script}(N_5), Q''[2] \rangle$	You may also find it useful to know that ‘ <i>Bus and Coach Drivers</i> ’ belongs to ‘ <i>Transport Drivers and Operatives</i> ’. Let’s explore ‘ <i>Bus and Coach Drivers</i> ’.
$\langle v_{KA}, \text{script}(N_5), Q''[3] \rangle$	You may also find it useful to know that ‘ <i>Driving Instructors</i> ’ belongs to ‘ <i>Transport Drivers and Operatives</i> ’. Let’s explore ‘ <i>Driving Instructors</i> ’.

6.4 Further Improvements

In this Chapter, we presented a systematic approach for applying the subsumption theory for meaningful learning to generate exploration paths in a data graph. This include two parts: (i) *finding the closest knowledge anchor to the first entity of an exploration path* (Algorithm 6.1), and (ii) *using the chosen knowledge anchor to subsume new entities* (Algorithm 6.2). In this Section, we will focus our discussion on possible improvements of these algorithms.

6.4.1 Algorithm for Identifying the Closest Knowledge Anchor

Algorithm 6.1 presented in Section 6.2 is generic and can be applied over different application domains represented as data graphs. The algorithm was applied to identify the closest knowledge anchors to first entities in two different application domains represented as data graphs: MusicPinta (in the musical instrument domain) and L4All (in the career domain), respectively. Algorithm 6.1 applies a semantic similarity metric to identify the closest knowledge anchor to first entity of an exploration path. The metric considers the class hierarchy in the data graph to identify similarity between two entities based on their depth. This brings the limitation of applying this algorithm in data graphs with shallow class hierarchies (e.g. class hierarchy depth = 1 or 2). In such case, will be no common ancestors for first entity and knowledge anchors in the data graphs, and hence the semantic similarity value between the two entities will be zero. Furthermore, we noticed that two (or more) knowledge anchors in a data graph can have the same semantic similarity value with first entity of an exploration path. For example, there were two knowledge anchors (`Flute` and `Reeds`) with the same semantic similarity value with the first entity `Bansuri`. One possible way to address this is through post processing. Knowledge anchors can be filtered based on their density in the data graph and give preference to the anchor with highest density among other anchors with same semantic similarity values to the first entity. Moreover, the semantic similarity metric can identify closest knowledge anchors which are not superclass (i.e. N_1 in Table 6.1), subclass (i.e. N_2 in Table 6.1) nor sibling (i.e. N_3 in Table 6.1) to the first entity. In other words, the closest knowledge anchor can't be reached directly from the first entity. For example, although the anchors `Flute` and `Reeds` have the same semantic similarity value with the first entity `Bansuri`, however `Flute` is a superclass of `Bansuri` and can be reached directly the narrative type N_1 in Table 6.1, whereas `Reeds` can't be reached directly from `Bansuri`. One possible solution is to identify semantic similarity to knowledge anchors which can be reached directly from the first entity. Another possible solution is to add a general narrative type in table 6.1 that indicates that the first entity and closest knowledge anchor belong to the same domain. However, such narrative may not increase the

users' domain knowledge since it does reflect the type of relationship between the start entity and closest knowledge anchor.

6.4.2 Algorithm for Generating Exploration Paths Using the Closest Knowledge Anchor

Algorithms 6.2 presented in Section 6.2 is generic and can be applied over different application domains represented as data graphs. The algorithm is underpinned the subsumption theory for meaningful learning to generate exploration paths for knowledge expansion. Algorithms 6.2 was applied to generate exploration paths in two different application domains represented as data graphs: MusicPinta (in the musical instrument domain) and L4All (in the career domain) data graphs, respectively. The algorithm used the identified closest knowledge anchor to generate transition narratives used to generate exploration paths of length m (i.e. number of transition narratives in exploration path). The algorithm is dependent on the sub-classes of the selected knowledge anchor – it may not be possible to generate m transition narratives in the exploration path (where m is the required length of exploration path identified as an input of the algorithm). Our implementation uses only the knowledge anchor, and can result in paths whose length of less than m . Another way to address this would be to continue the path, using another knowledge anchor. It is also possible to use superordinate categories to the knowledge anchor to extend an exploration path. This will be suitable for cases when the user is gained knowledge at the subordinate level and is ready to generalise to a more abstract level. In such cases, a user model is required.

6.5 Summary

In this Chapter, we have uniquely utilised the subsumption theory for meaningful learning to generate exploration paths for knowledge expansion in data graphs. Our approach for generating exploration paths includes two parts: (i) *finding the closest knowledge anchor to the first entity of an exploration path* (Algorithm 6.1), and *using the chosen knowledge anchor to subsume new entities in the exploration path* (Algorithm 6.2). The formal framework presented in this Chapter, that adapt Ausubel's subsumption theory for meaningful learning to data graphs is a key contribution of our work. The developed algorithms are fairly generic and can be applied over different application domains represented as data graphs. We have applied the algorithms over two applications for data exploration, semantic browsing (in the musical instrument domain) and semantic search (in the career domain), using the data graphs from the two applications, *MusicPinta* and *L4All*, respectively. The next step in this research, is to evaluate the subsumption algorithm with participants against free-exploration. This will be the focus of the next Chapter.

Chapter 7

Evaluation of Exploration Paths

7.1 Introduction

In this research, we aim to propose a new exploration support mechanism underpinned by the subsumption theory for meaningful learning to generate exploration paths for knowledge expansion. This theory postulates that new knowledge is grasped by starting from familiar entities in the data graph which serve as knowledge anchors from where links to introduce new knowledge are made. A core algorithmic component for adapting the subsumption theory to generate exploration paths is the automatic identification of knowledge anchors in a data graph (KA_{DG}). We have developed and validated metrics for automatic identification of KA_{DG} .

Using KA_{DG} , we developed a subsumption algorithm for generating exploration paths for knowledge expansion in a data graph. The algorithm applies a semantic similarity metric to identify the closest knowledge anchor to the first entity of an exploration path. The closest knowledge anchor is used to subsume entities used to generate the exploration path. The subsumption algorithm in Chapter 6 was implemented in MusicPinta and L4All, and illustrated with examples.

This Chapter aims to evaluate the subsumption algorithm with participants. To evaluate the subsumption algorithm it is important to identify a suitable application context, which would allow examining the generated exploration paths with participants to assess the knowledge utility of the generated exploration paths. We will use MusicPinta to evaluate the generated exploration paths with participants.

On the one hand, MusicPinta provides a uni-focal interface which enables users to start their exploration from a single entity in the data graph (i.e. the MusicPinta interface enables the participants to select the first entity of an exploration path). On the other hand, MusicPinta data graph provides sufficient class hierarchy for traversing the user to different areas of the graph (see Section 3.4.1).

To evaluate the subsumption algorithm, we will conduct an experimental user study with users to examine the knowledge utility and users' exploration experience of the generated exploration paths. We will compare two conditions:

Experimental condition (EC): where users follow exploration paths generated using the subsumption algorithm;

Control condition (CC): where users carry out free exploration and they are free to select entities to visit.

Comparing both conditions a controlled task-driven experimental user study will be conducted with participants to examine the following hypotheses:

- **H1.** *Users who follow EC expand their domain knowledge.*
- **H2.** *The expansion in the users' knowledge when following EC is higher than when following CC.*
- **H3.** *The usability when EC is followed is higher than when CC is followed.*

Next in this Chapter we will describe the steps we follow to design exploration task for users (Section 7.2). In Section 7.3 we will describe the experimental setup, and in Section 7.4 we will describe the results from the experimental user study examining how the exploration paths generated using the subsumption algorithm affected on knowledge utility and exploration experience on users' exploration. Section 7.5 will discuss the findings from the evaluation, and Section 7.6 will summarise the Chapter.

7.2 Exploration Task Design

Designing exploration tasks for users is considered an important requirement for evaluating data exploration approaches [164]. Exploration tasks can be characterised as learning or investigative oriented tasks, thus distinguishing them from lookup-oriented tasks [4]. A typical exploration task has to be generic (i.e. the scope of the task is broad and the user don't have specific information needs), realistic (i.e. real-life task that set in a familiar situation), discovery-oriented (i.e. users travel beyond what they know), open-ended (i.e. requires a significant amount of exploration, where open-endedness relates to uncertainty over the information available, or incomplete information on the nature of the search task), and set in an unfamiliar domain for the user [2, 164, 165].

A task taxonomy for graph visualization and exploration has been proposed in [119]. The taxonomy tasks are categorised into four groups: topology-based tasks, attribute-based tasks, browsing tasks, and overview task. Each task has general descriptions and example scenarios, as the following:

Topology-based tasks: include (i) adjacency tasks (e.g. finding a set of entities directly linked to an entity; identify how many entities are linked to a particular entity; or which entity has the maximum number of adjacent entities.); (ii) accessibility tasks (e.g. find the set of

entities accessible from a particular entity); and (iii) connectivity tasks (e.g. find the shortest path between two entities).

Attribute-based tasks: use the previous topology-based tasks with additional filters relates to entities (e.g. find the entities having a specific attribute value) or related to links (e.g. given an entity, find the entities connected only by certain link types).

Browsing tasks: include follow a given path (e.g. a user is given a set of sequential entities in the data graph to explore), and revisit entities (e.g. return to a previously visited entities in the data graph).

Overview task: is a compound exploratory task to get estimated values quickly (e.g. ask a user to estimate the size of the data graph). Furthermore, overview tasks may include asking the user to identify some patterns in the graph (i.e. types of entities are connected together).

Among the four task categories outlined above, the *topology-based tasks* and *browsing tasks* will be adopted in the experimental user study for evaluating exploration paths. *Topology-based tasks* will be used since the study participants will be given specific entities to explore (i.e. given entities represent the first entities of exploration paths), and then they will be asked questions about how these are associated with other entities in the data graph. These questions correspond to the three cognitive processes from Bloom's taxonomy [27] (as described in Section 2.7.1): *remember* (i.e. finding entities in the data graph that are related to the a given entity), *categorise* (i.e. finding entities in the data graph that the given entity belongs to) and *compare* (i.e. finding entities in the data graph that are similar to the given entity). *Browsing tasks* will be used since the participants will be given exploration paths (i.e. *EC* experimental condition) and will be asked to follow these paths. Furthermore, the semantic data browser (MusicPinta) which will be used in the experimental user study supports: *topology-based tasks* (i.e. show connections between entities in the graph) and *browsing tasks* (i.e. enable the user to follow an exploration path represented as a set of entities linked via edge labels). However, *attribute-based tasks* and *overview task* will not be used in the experimental user study since the metric for measuring knowledge utility of exploration path (described in Section 2.7.1) considers how entities are connected in the data graph rather than identifying specific attributes relates to a given entity in the graph, or providing estimations about the graph as a whole or identifying specific patterns, respectively. Furthermore, MusicPinta does not provide textual information nor visual representation about the overall data graph.

The authors in [164, 166] applied a two-step approach for designing data exploration tasks for participants: (i)

- Designing a task template that places the participant in a familiar situation which involves exploring multiple entities in an unfamiliar domain (e.g. a researcher at university wants to write a paper (familiar situation) about new topic.
- Identifying unfamiliar candidate entities (e.g. find new research topic) in the domain that could be plugged into the task template.

The main idea of using a task template is to put the users in a familiar situation where they will be asked to find some entities. The study in [164] involved university participants, and hence a familiar situation was writing a paper for a class. Accordingly, the following task template was suggested [164]:

“Imagine that you are taking a class called _____. For this class, you need to write a paper on the topic _____. Use the catalogue to find two possible topics for your paper. Find three books for each topic.”

Using the above template, a task scenario was designed [164], and involved asking participants to find items – to which the specific topics could be plugged into:

*“Imagine you are taking a class titled “Great Britain and its Colonies in the Twentieth Century”. For this class you need to write a research paper on some aspect of the relationship between Great Britain and its Colonies in the Twentieth Century but you have yet to decide on one. Use the catalogue to **find two possible topics** for your paper. Then use the catalogue to **find three books for each topic** so that you might make a decision as to which topic to write about”.*

In this work, we follow similar approach to the one suggested in [164], utilising the two steps described above, as follows:

Designing the task template. We aim to design a generic task template that encourages layman users to seek knowledge in a domain unfamiliar to them. Therefore, we designed the task template in the context of a general knowledge quiz show where users need to acquire as much knowledge as they can. As discussed in Section 6.3, we identified the musical instrument domain (MusicPinta data graph) as our application context for implementing the subsumption algorithms to generate exploration paths for knowledge expansion. Accordingly, we designed the exploration task template in the musical instrument domain using MusicPinta to evaluate the generated exploration paths. Inspired by the task templates in [164], the task template presented in Table 7.1 was designed to suit the musical instrument domain.

Table 7.1 Task template used in the experimental user study

Task template
<i>“Imagine that you are a member of a team which will take part in a general knowledge quiz show. You have been asked to explore two musical instruments for 20 minutes in order to prepare a short presentation to describe to your team what you have learned about these instruments”.</i>

As can be seen in the task template, a user will be asked to explore two musical instruments (i.e. unfamiliar entities) and prepare a presentation for his/her team in a knowledge quiz show (i.e. familiar situation).

Identify unfamiliar entities in the domain of the user study. The second step in designing the task template is to identify unfamiliar topics (in our case, unfamiliar musical instruments) to be bugged in the task template in Table 7.1. For this we ran a questionnaire with users to identify the unfamiliar entities in the `String Instrument` and `Wind Instrument` class hierarchies in the MusicPinta data graph. These two class hierarchies have the richest class representation in terms of the number of classes and the hierarchy depth as discussed in Section 3.4.1, and have the highest number of knowledge anchors (9 anchors in the `String Instrument` class hierarchy and 10 anchors in the `Wind Instrument` class hierarchy – out of 24 anchors in MusicPinta data graph).

We extracted class entities at the bottom quartile of the two class hierarchies (note that the depth of the two class hierarchies is 7 – see Table 3.1, and entities of depth 6 or 7 are considered to be at the bottom quartile of the data graph). This is based on earlier Cognitive science studies acknowledging that layman users are not familiar with specific objects in a domain [167]. Overall 61 class entities from the `String Instrument` and `Wind Instrument` class hierarchies were used in the survey. The selected classes were randomised and distributed among twelve participants who are not experts in the musical instruments (the participants have limited knowledge about musical instruments and may have seen the instrument, and none of the participants had played on a musical instrument) using a survey which can be found in Appendix D.1. Each participant was asked to identify his/her familiarity with the musical instruments by selecting one of the following options.

- **High** (You have good knowledge and have played on the instrument).
- **Medium** (You have some knowledge and have listened to the instrument).
- **Low** (You have limited knowledge and have seen the instrument).
- **None** of the above.

To identify unfamiliar instruments from each class hierarchy, we chosen the entities which most users were not familiar with (i.e. users were not familiar at all with the instruments names). These entities were the musical instrument ‘Biwa’ (class hierarchy: `String Instrument`, origin: Japanese) and ‘Bansuri’ (class hierarchy: `Wind Instrument`, origin: Indian). These musical instrument were used in the task template in Table 7.1, and presented to the participants, as will be described in the next Section.

7.3 Experimental Setup

The goal of the experimental user study is to examine the knowledge utility and users' exploration experience of the generated exploration paths, in two conditions:

- **Experimental condition (EC):** where users follow exploration paths generated using the subsumption algorithm.
- **Control condition (CC):** where users carry out free exploration and they are free to select entities to visit.

We evaluate our algorithms for generating exploration paths by adopting a task-based approach (as discussed in Section 7.2). We approximate the knowledge utility of a path by adapting methods from Education to assess the users' conceptual knowledge, comparing the knowledge utility of generated exploration paths versus free exploration paths. We use free exploration as our evaluation base-line since users in normal life freely explore new domains to investigate a new topic. Furthermore, this is similar to other exploratory search evaluation approaches where users are asked to conduct specific tasks. For example, study participants in [64] were asked to freely explore a system (called Aeemo) to identify specific facts about a given topic. The same approach was used to evaluate FACETS [12] (system for exploring large graphs) where participants were free to choose entities while performing exploration tasks.

Comparing both conditions, a controlled task-driven user study will be conducted with participants to examine the following hypotheses:

- **H1.** Users who follow *EC* expand their domain knowledge.
- **H2.** The expansion in the users' knowledge when following *EC* is higher than when following *CC*.
- **H3.** The usability when *EC* is followed is higher than when *CC* is followed.

Participants. 32 participants consisting of university students and professionals (24 students and 8 professionals⁵⁸) were recruited on a voluntary basis (a compensation of £5 Amazon voucher was offered). The participants varied in age 18–45 (mean age is 30), and cultural background (1 Austria, 9 British, 1 Chinese, 3 Greek, 1 Italian, 5 Jordanian, 1 Libyan, 2 Malaysian, 6 Nigerian, 1 Polish, 1 Romanian and 1 Saudi).

Method. Four online surveys⁵⁹ were run (see example in Appendix D3). Every survey had 8 participants. Each participant was allocated one survey. Figure 7.1 shows the overall structure of the user study. Each participant explored both musical instruments (i.e. Biwa, Bansuri),

⁵⁸ University lecturers and private Sector employees (Banking and Airlines).

⁵⁹ The study was conducted with Qualtrics (www.qualtrics.com).

where each of the instrument was allocated to an exploration strategy (*EC* or *CC*). The order of *EC* and *CC* was randomised to counter balance the impact on the results. Every participant session was *conducted separately* and observed by the author. All participants were asked to provide feedback before, during, and after the interaction with MusicPinta.

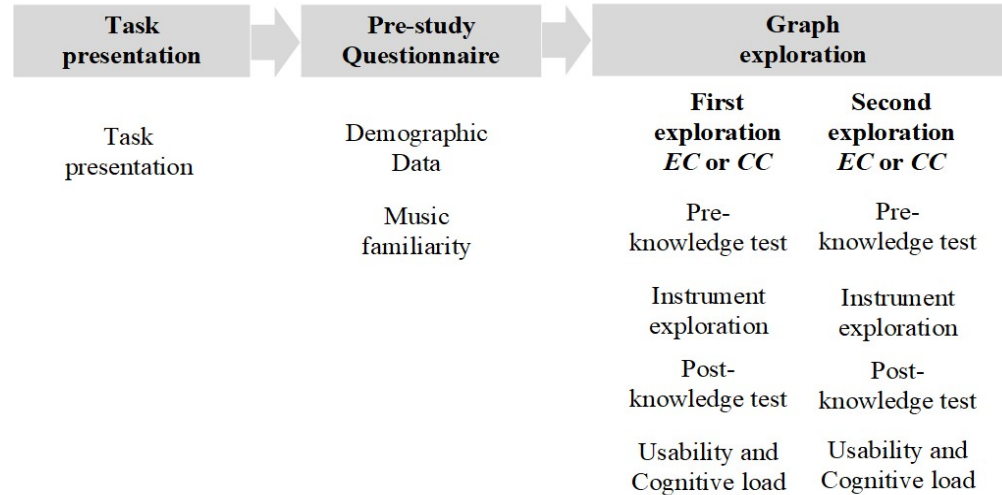


Figure 7.1. Structure of the user study to examine *EC* against *CC* in terms of knowledge utility, usability and cognitive load.

Task presentation [1 min] – utilise the task template (as described in Section 7.2) to present the data exploration task for the users at the beginning of their exploration session. The musical instruments *Biwa* and *Bansuri* have been is plugged into the task template presented in Table 7.1.

Pre-study questionnaire [2 min] - collect information about the participants’ profiles, and their familiarity with the music domain, focusing on the two musical instrument class hierarchies which would be explored – *String Instrument* and *Wind Instrument*. The participants’ familiarity with the two class hierarchies varied from low to medium (63% and 78% of the participants had low familiarity with *String Instrument* and *Wind Instrument*, respectively). Participants familiarity with the two class hierarchy can be found in Appendix D.2.

Graph exploration [20 min] – each user explored the two exploration strategies (*EC* and *CC*), where each exploration strategy corresponds to one of the two unfamiliar musical instruments identified (*Biwa* or *Bansuri*). Examples of the exploration paths generated for the instruments *Biwa* and *Bansuri* are described in Section 6.3 (See Figures 6.1 and 6.3). The transition narratives used in the exploration path for *Biwa* and *Bansuri* are listed in Tables 6.5 and 6.7, respectively in Section 6.3.

Figures 7.2 and 7.3 show examples of what the participants have seen while following the exploration paths for Biwa (transition narrative $\langle v_s, script(N_1), v_{KA} \rangle$ in Table 6.5) and Bansuri (transition narrative $\langle v_{KA}, script(N_4), Q'[2] \rangle$ in Table 6.7), respectively. The complete examples of transition narratives for Biwa and Bansuri are in Appendix D.3.

You may find it useful to know that 'Biwa' belongs to a familiar and well-known instrument called 'Lute'. Let's explore 'Lute'

Read all the following content of 'Description', 'Features' and 'Relevant information' of 'Lute'

MusicPinta
Social . Semantic . Music

Logout
dicode


Home Semantic Search Contribute Help

Home > Semantic Search > Lute

Lute

Description Features Relevant Information Reviews Link History

Description is extracted from Wikipedia when available.

 Lute can refer generally to any plucked string instrument with a neck (either fretted or unfretted) and a deep round back, or more specifically to an instrument from the family of European lutes. The European lute and the modern Near-Eastern oud both descend from a common ancestor via diverging evolutionary paths.

Description Features Relevant Information Reviews Link History

Lute is:

[Instrument](#)

Lute belongs to:

[Arabic words and phrases](#) [Baroque instruments](#) [Bouzouki](#) [Composite chordophones](#) [Early musical instruments](#) [Instrument](#) [Lutes](#) [Necked bowl lutes](#) [Plucked string instruments](#) [String instruments](#)

Description Features Relevant Information Reviews Link History

The following are **Lute**:

No items found.

The following share features with **Lute**:

[A two-stringed lute with a circular flat body re...](#) [Balalaika](#) [Bandura](#) [Banjitar](#) [Banjo](#) [Biwa](#) [Cittern](#) [Cumbus](#) [Dan ty ba](#) [Electric sitar](#) [Mandola](#) [Mandolin](#) [Moon lute](#) [Oud](#) [Pipa](#) [Rebab](#) [Sanshin](#) [Sanxi](#) [Sarod](#) [Shamisen](#) [Sitar](#) [Tambura](#) [Tamburitza](#) [Theorbo](#) [Xalam \(khalam\)](#) [Zhongruan](#)

Figure 7.2. Example of a transition narrative shown to participants from the exploration path of Biwa

You may also find it useful to know that 'Transverse Flute' belongs to 'Fipple Flutes', and 'Bansuri' belongs to 'Transverse Flute'. Let's explore 'Transverse Flute'.

Read all the following content of '[Description](#)', '[Features](#)' and '[Relevant information](#)' of 'Transverse Flute'

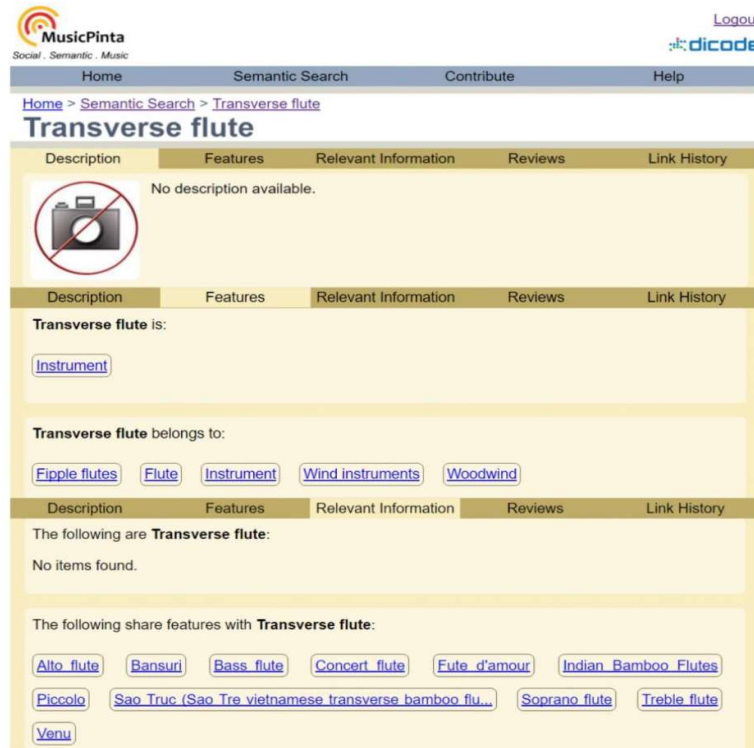


Figure 7.3. Example of a transition narrative shown to participants from the exploration path of Bansuri

We analysed *knowledge utility* and *user exploration experience* by usability aspects, associated with the user's exploration settings under the experimental condition (*EC*) and the control condition (*CC*), as the following:

- **Approximating knowledge utility.** To compare the *knowledge utility* of the exploration paths resulting from each strategy (*EC* and *CC*), the user knowledge was approximated before and after each exploration using the three questions that correspond to Bloom's cognitive processes of *remember*, *categorise* and *compare*. The difference between the participants' answers before and after exploration of each strategy indicated the knowledge utility (i.e. knowledge expansion) of exploration of that strategy (the process for approximating knowledge utility of an exploration path is described in Section 2.7.1). The knowledge utility values of participants' exploration for the two exploration strategies (*EC* and *CC*) will be presented in the study results in Section 7.4.1.
- **User's exploration experience.** After each exploration, the participants were asked to fill a questionnaire about their exploration experience and the cognitive load they have experienced (based on a modified version of the NASA-TLX questionnaire [161]).

Furthermore, the participants were asked to think aloud; notes of all comments were kept. The participants' exploration experience will be presented in Section 7.4.2.

Tables 7.2 and 7.3 show examples of the entities that were freely visited in the control condition *CC* (i.e. freely visited entities by the participants), and entities in the experimental condition (i.e. generated exploration paths that were followed by the participants) for the musical instruments *Biwa* and *Bansuri*.

Table 7.2. Examples of entities the participants have visited during their free exploration (*CC*) of *Biwa*, compared to generated exploration paths (*EC*).

<i>EC</i>	Example 1 <i>CC</i>	Example 2 <i>CC</i>	Example 3 <i>CC</i>
Biwa	Biwa	Biwa	Biwa
Lute	Bouzouki	String Instrument	Japanese Musical Instrument
Oud	Xalam	Guitar	Lute
Tambura	Banjitar	Acoustic Guitar	Moon Lute
Pipa	Plucked String Instrument	Classical Guitar	Bouzouki

Table 7.3 shows examples of the entities that were freely visited in the control condition *CC* for *Bansuri*.

Table 7.3. Examples of entities the participants have visited during their free exploration (*CC*) of *Bansuri*, compared to generated exploration paths (*EC*).

<i>EC</i>	Example 1 <i>CC</i>	Example 2 <i>CC</i>	Example 3 <i>CC</i>
Bansuri	Bansuri	Bansuri	Bansuri
Flute	Transverse Flute	Fipple Flute	Bamboo Musical Instrument
Fipple Flute	Saw Truck	Contrabass Recorder	Side-blown Flute
Transverse Flute	Fipple Flute	Recorder	Concert Flute
Indian Bamboo Flute	Flute D'amour	Great bass recorder	Fipple Flute

Both, *EC* and *CC* had the same length (*EC* had four transition narratives, and *CC* had four edge labels)⁶⁰.

⁶⁰ The length of the exploration path of four edges (5 entities) is based on Miller's Law [157], which indicates number of objects that an average human can hold in working memory is 7 ± 2 objects.

7.4 Results

In the following, we will present the results of evaluating the subsumption algorithm for generating exploration paths.

7.4.1 Measuring of Knowledge Utility

The users' knowledge was measured before and after each exploration using the three questions of the schema activation test (described in Section 2.7.1) related to the entities *Biwa* and *Bansuri* (Appendix D.4). Before exploration, none of the users were able to articulate any item linked to the two musical instruments (*Biwa* and *Bansuri*) using the three cognitive processes. The *knowledge utility* for the three cognitive process before and after exploration of *EC* and *CC* is shown in Figure 7.4.

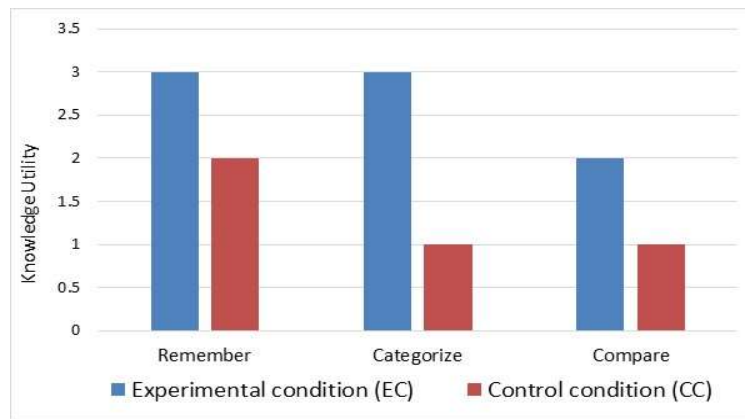


Figure 7.4. Knowledge utility of the two strategies (*EC* and *CC*) of the user cognitive processes (*median* of the knowledge utility of exploration for all users).

The knowledge utility of the exploration under experimental condition (*EC*) in the three cognitive processes was higher than the effect of free exploration under control condition (*CC*); and this difference is significant (See Table 7.4). The results showed that all participants were able to *remember* and *categorise* entities with *EC* (only 5 participants couldn't *compare* new entities). Whereas not all participant could *remember*, *categorise* or *compare* new entities after they have finished their exploration with *CC* (there were 2 participants that could not *remember* or *categorise* new entities and 13 participants could not compare between entities).

Table 7.4. Statistically significant differences of the values in Figure 7.4 (Mann-Whitney, 1-tail, $N_a=N_b=32$)

Difference in Knowledge Utility between <i>P</i> and <i>F</i>	Cognitive Process	Z-value	p
<i>EC > CC</i>	Remember	3.6	P<0.01
	Categorise	5.1	P<0.0001
	Compare	2.7	P<0.01

Notably, for the cognitive process *categorise* the bigger effect on exploration of the subsumption exploration strategy over the free exploration strategy is highly significant ($p < 0.0001$). The difference between median values for *categorise* cognitive process under *EC* and *CC* was higher than the *remember* and *compare* cognitive processes.

Furthermore, we examined the knowledge utility for the three cognitive processes for each instrument in its corresponding class hierarchies, as shown in Figure 7.5.

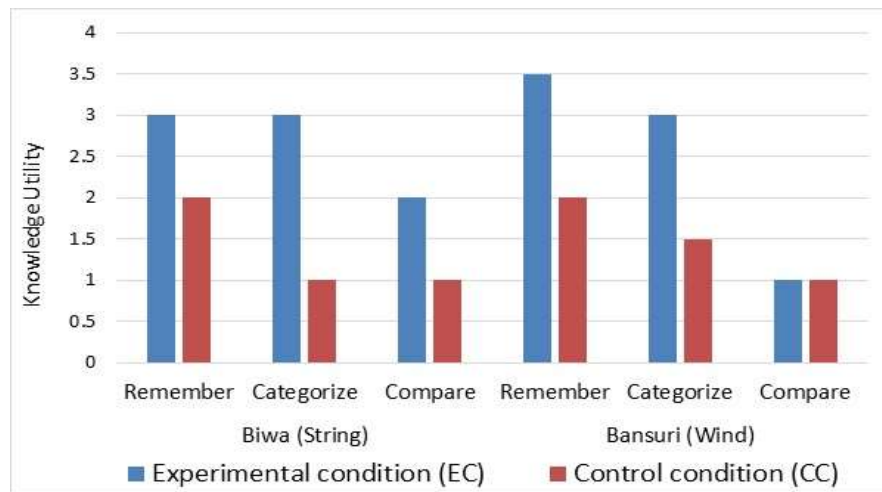


Figure 7.5. Knowledge utility of the two strategies (*EC* and *CC*) of the user cognitive processes (*median* of the knowledge utility of exploration for all users).

The knowledge utility of the exploration under experimental condition (*EC*) in the three cognitive processes was higher than the effect of free exploration under control condition (*CC*) for *Biwa* and was higher in the cognitive processes *compare* and *categorise* for *Bansuri*; and this difference in *EC* and *CC* is significant except for the cognitive process *compare* for instrument *Bansuri*, as shown in Table 7.5. The difference between *EC* and *CC* for the cognitive process *compare* for *Bansuri* was not significant as both *EC* and *CC* had the same median values as shown in Figure 7.5. Furthermore, by inspecting the participants' answers for the cognitive process *compare*, it was noticed that more than half of the participants (9 out of 16 participants for *EC*; 13 out of 16 participants for *CC*) had 0 or 1 values as their knowledge expansion in the cognitive process *compare* (0 value: participants didn't learn similar entities to *Bansuri*; 1 value: participants learned one entity similar to *Bansuri*). Since more than half of the results in the two lists (*EC* and *CC*) are 0 or 1, this decreases the difference between *EC* and *CC* (i.e. decrease the chance that a randomly selected value from *EC* will be higher than a randomly selected value from *CC*). Notably, for the cognitive process *categorise* the bigger effect on the exploration of *EC* over *CC* is highly significant ($p < 0.001$) for *Biwa* and *Bansuri*.

Table 7.5. Statistically significant differences of the values in Figure 7.5 (Mann-Whitney, 1-tail, $N_a=N_b=16$)

Difference in Knowledge Utility between <i>EC</i> and <i>CC</i>	Instrument (<i>class Hierarchy</i>)	Cognitive Process	Z-value	p
<i>EC > CC</i>	Biwa (String Instrument)	Remember	1.658	P<0.05
		Categorise	3.373	P<0.001
		Compare	2.449	P<0.05
<i>EC > CC</i>	Bansuri (Wind Instrument)	Remember	3.467	P<0.001
		Categorise	3.900	P<0.001
		Compare	1.280	P<0.50

To further inspect what caused the low knowledge utility for the cognitive process *compare* for instrument *Bansuri*, we looked into the participants' familiarity with the *Wind Instrument class hierarchy* (the class hierarchy that *Bansuri* belongs to) and noticed that 78% of the participants had low familiarity (i.e. participants have limited knowledge and they may have seen some instruments) with *Wind Instrument*, whereas 65% of the participants had low familiarity with the *String Instrument class hierarchy*.

One could argue that being more familiar with the *String Instrument class hierarchy* than the *Wind Instrument class hierarchy*, participants knew more entities to compare with. Furthermore, entities in the *String Instrument class hierarchy* are associated with more *DBpedia categories* compared to entities in the *Wind Instrument class hierarchy* (*String Instrument* has 255 and *Wind Instrument* has 161 *DBpedia categories*). Most of these categories are grouping musical instruments based on their cultural origin (e.g. *Chinese Musical Instruments*, *Japanese Musical instrument*, *Indian Musical Instruments*, *Greek Musical Instruments*), which helped the participants to associate entities from their cultures. This indicates important considerations of the data graph and the user familiarity for knowledge expansion.

7.4.2 User Exploration Experience

After each exploration strategy, the participants' feedback on the exploration experience during the path was collected including exploration usability and exploration complexity (adapted from *NASA-TLX*). Figures 7.6 and 7.7 give a summary of the users' feedback.

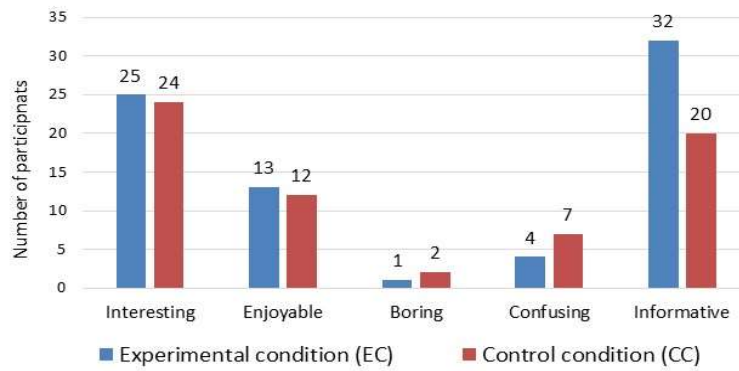


Figure 7.6. Users’ exploration experience of the two exploration strategies (*EC* and *CC*)

Values show number of participants (out of total of 32 participants) characterised *EC* and *CC* with the corresponding characteristics.

The exploration experience with *EC* was the most informative (all 32 participants identified their exploration experience with the exploration paths under *EC* as informative, whereas 20 participants indicated their exploration with *CC* as informative). The participants also found their exploration under *EC* to be slightly more interesting and enjoyable than *CC*. Furthermore, the participants found the exploration paths under *EC* to be the least boring and least confusing – only 3% (one participant) and 16% (four participants) of the participants founded their exploration with *EC* to be boring or confusing, respectively. For instance, the participant (User 22 in Appendix D.2) who indicated his exploration experience with *EC* as boring did this because he was not able to freely explore through entities in the graph (his feedback was “*Narratives in paths allow me to explore entities in a hierarchical fashion, and I would like to freely explore other types of relationships*”). Another participant (User 18 in Appendix D.2) indicated his exploration experience with *EC* to be confusing (his feedback was “*I saw the same instruments several times during my exploration*”).

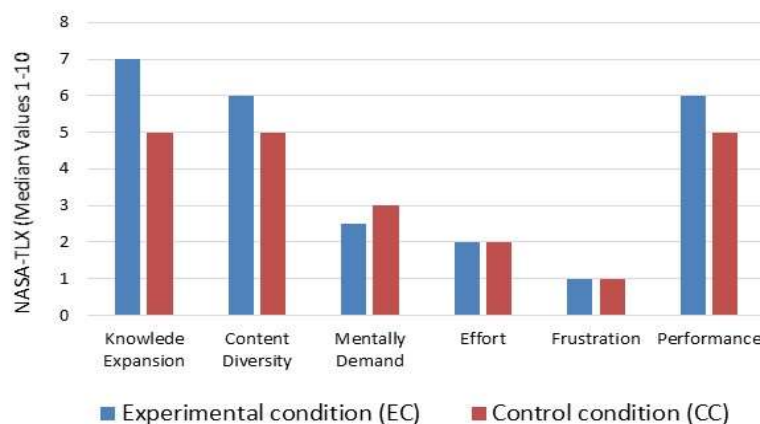


Figure 7.7. Users' subjective perception of the two exploration strategies (*EC* and *CC*), based on an adapted NASA-TLX questionnaire [160] (*median* values for users in range 1-10).

Questions of subjective tasks that were asked to the user based on an adapted NASA-TLX, are shown in Table 7.6.

Table 7.6. Questions of NASA-TLX questionnaire used to measure the users' perception of subjective process

Subjective process	Question text
Knowledge expansion	<i>How much the exploration expanded your knowledge?</i>
Content Diversity	<i>How diverse was the content you have explored?</i>
Mentally Demand	<i>How mentally demanding was this exploration?</i>
Effort	<i>How hard did you have to work in this exploration?</i>
Performance	<i>How successful do you think you were in this exploration?</i>

The effect of the exploration path under *EC* on the knowledge expansion and performance subjective processes was higher than the effect of the free exploration strategy; and this difference was significant, as shown in Table 7.7.

Table 7.7. Statistically significant differences of users' subjective perception of cognitive process (Mann-Whitney, 1-tail, $N_a=N_b=32$)

Difference in the users' experience	Subjective Process	Z value	p
<i>EC > CC</i>	Knowledge Expansion	3.98	$P<0.0001$
	Content Diversity	0.32	$P<0.50$
	Mentally Demanding	0.14	$P<0.50$
	Effort	0.14	$P<0.50$
	Performance	2.15	$P<0.05$

Notably, for the subjective process *knowledge expansion* the bigger effect on exploration of *EC* over *CC* is highly significant ($p<0.0001$).

7.5 Discussion

In this Section, we will focus our discussion on three parts: (i) the study hypothesis that we set out for the evaluation (ii) reflection on exploration, and (iii) applicability and limitations of evaluation.

7.5.1 Study Hypothesis

Overall, the evaluation results have supported our hypothesis as following.

H1. *Users who follow experimental condition (EC) expand their domain knowledge.* It was found that users who followed the exploration paths under *EC*, i.e. paths which were generated by the subsumption algorithm (presented in Section 6.3), would increase their knowledge. All participants in our study have indicated their exploration to be informative

with *EC*. This indicates that the exploration paths generated suit different users from different cultural background, and with different familiarities and needs in the exploration domain.

H2. *The expansion in the users' knowledge when following EC is higher than when following CC.* The results showed that participants who followed exploration paths *under EC* were able to *remember*, *categorise* and *compare* more entities compared to trajectories of free exploration in *CC*. All participants (100%) have indicated their exploration to be informative with *EC*, whereas 62% of the participants founded their free exploration trajectories to be informative. The results also showed that the cognitive process *categorise* had the bigger effect on expanding the participants' knowledge exploration of the subsumption strategy over free exploration. This was caused because of two things:

- The subsumption hierarchical relationship (`rdfs:subClassOf`) was used to create the narrative scripts between entities of an exploration path under *EC*, which helped the users to categorise new entities at different levels of abstraction at their cognitive structures.
- The subsumption process used to generate an exploration path, uses knowledge anchors to subsume and learn new sub-categories similar to the way a human mind works while learning a new concept.

H3. *Usability when EC strategy is followed is higher compare to when CC is followed.* The results showed that participants found exploration paths under *EC* to be more enjoyable and less confusing. Furthermore, participants felt that their overall performance using the exploration paths *under EC* was higher than free exploration in *CC*. However, one participant founded his experience with hierarchical narrative scrips to be boring. The participant suggested to diversify the types of narratives used between entities in the exploration path. In other words, to use other relationships, other than the subsumption relationship to produce interesting information about the entity (e.g. add information about the musical performances or events where a musical instrument has been played).

7.5.2 Reflection on Exploration

We will provide reflection on exploration paths under *EC* in terms knowledge expansion and exploration experience.

Knowledge expansion. The results showed that participants who followed the exploration path for *Biwa* were able to *compare* more entities than patriciates who followed exploration path for *Bansuri*. (median values of knowledge expansion: *Biwa* = 2; *Bansuri* = 1). By inspecting the exploration paths (Figures 6.1 and 6.3) and corresponding transition narratives (Tables 6.5, 6.7) for *Biwa* and *Bansuri*, respectively, we noticed that the subsumption class

hierarchy for *Biwa* (depth = 2) was shallower than *Bansuri* (depth = 4). Consequently, most transition narratives (3 transition narrative out of 4) used in constructing the exploration path for *Biwa* subsumed direct subclasses of closest knowledge anchor *Lute* (i.e. broadening users exploration). Whereas all transition narratives (4 transition narratives) used in the exploration path for *Bansuri* focused on deepening the exploration (i.e. each transition narrative subsumed higher depth entities). Furthermore, although the exploration path for *Bansuri* focused on deepening the exploration, however participants were able to *categorise* same number entities when explored *Biwa*. This suggests that an appropriate exploration strategy has to broaden the exploration first, then deepening it, which supports Shneiderman's hypothesis [84] (*overview first, zoom and filter*).

Exploration Experience. The results showed that participants who explored *Bansuri* were more confused (3 participants) and bored (1 participant) than participants who explored *Biwa* (one participant found it confusing). Participants indicated their exploration with *Bansuri* to be confusing since they were seeing similar same entities in each transition (i.e. subsumed entities in the exploration path share similar subclasses, and these shared subclasses were seen in each transition narrative). For example (See Figure 6.3), the 2nd (*Fipple Flutes*), 3rd (*Transverse Flute*) and 4th (*Indian Bamboo Flutes*) subsumed entities in the exploration paths for *Bansuri* shared the same subclasses *Bansuri* and *Venu*. This shows that users' exploration experience is affected by the quality of the ontology in terms of the richness of the class entities and the hierarchy depth (MusicPinta ontology has 364 class entities and depth of 7 – characteristics of MusicPinta data graphs are described in Section 3.4.1). Furthermore, having linear exploration paths in the data graphs from abstract entities to more specific entities in the graph linked via the subsumption relationship `rdfs:subClassOf` will have similar narrative scripts between entities in the path, and users will be exploring the entities at the bottom of the ontology at each conjunction (similar to the case of *Bansuri* described above). Which may have negative impact on the users' exploration experience and cognitive load.

7.5.3 Applicability of Evaluation Approach

Approximating Knowledge Utility. To approximate knowledge utility of an exploration path, the user knowledge is assessed before and after exploration using Bloom's cognitive processes of *remember*, *categorise* and *compare*. These were extracted from the first two cognitive categories in Bloom's taxonomy (namely *remember* and *understand*) which are directly related to exploration activities. In other evaluation contexts, more complex cognitive categories can be applied such as the cognitive category *analyse*. This category includes

several cognitive processes, such as *differentiate* (e.g. differentiate between two entities in the data graph) and *arrange* (e.g. arrange entities in the data graph from abstract to specific), which can be related to exploratory search tasks.

Exploration Task Design. We evaluated the subsumption algorithm for generating exploration paths against free exploration by adopting a task-based approach. It involved two steps: (i) designing a task template and (ii) identifying unfamiliar entities in the domain to be plugged into the task template. The task template presented is in the context of a general knowledge quiz that encourages users to seek knowledge in a given domain represented as a data graph. The template can easily be adapted for a range of data graph exploration tasks. This will require identifying unfamiliar entities to include in the template. We did this based on a small survey with participants to identify domain entities (at the bottom quartile of the data graph) which are likely to be unfamiliar to layman user. At a larger scale, crowdsourcing can be used to identify unfamiliar entities in the data graph.

7.6 Summary

In this Chapter, we evaluated the exploration paths generated from the subsumption algorithm in a controlled user study to examine whether the exploration paths increase the users' knowledge as compared to free exploration. A data exploration task was designed to evaluate the exploration paths. The task design included two steps: designing a task template, and identifying unfamiliar entities in the domain which can be plugged into the task template. We designed the task template in the context of a general knowledge quiz to encourage participants to learn as much as they can. A small-scale survey was conducted in the musical instrument domain (MusicPinta) to identify two unfamiliar instruments used in the task template.

The findings from the evaluation showed that exploration paths using knowledge anchors and subsumption lead to significantly higher increase in the users' knowledge. The evaluation approach presented in this Chapter validates our approach for generating exploration paths for knowledge expansion over data graphs, which enables its adoption over different domains and application contexts. The evaluation approach presented in this Chapter contributes to adopting the subsumption algorithm to develop usable semantic data graph exploration applications.

Chapter 8

Conclusion

8.1 Synopsis

This research dealt with the problem of supporting user exploration over data graphs. The ultimate goal is to develop an automatic approach for generating exploration paths for knowledge expansion in data graphs.

Towards this goal, in Chapter 1 two research questions were formulated: *RQ1. How to develop automatic ways to identify knowledge anchors in a data graph?*, and *RQ2. How to use knowledge anchors to generate exploration paths to facilitate domain knowledge expansion in a data graph?*

In Chapter 2, background information was provided to better understand the research context. Related work was discussed and key limitations of state of the art approaches were identified. Three main research fields were investigated: (i) *exploration of data graphs* – however these approaches don't take into account the learning effect of exploration paths over data graph, (ii) *identifying key entities in data graphs* – however, these approaches lacks applying the Cognitive science notion of BLO in the context of a data graph to identify familiar entities in the data graph, and (iii) *generating exploration paths in data graphs* – however, these approaches don't adapt the subsumption theory for meaningful learning to generate exploration paths for knowledge expansion in data graphs where familiar entities are used as knowledge anchors to introduce and learn new knowledge.

In Chapter 3, the overall research methodology was described. An exploratory user study that investigates three initial exploration strategies devised based on the structure of the data graph and user familiarity with the domain, was presented. The observations from the study directed us to investigate Ausubel's subsumption theory for meaningful learning [21] and adapt it as our underpinning model for generating exploration paths to facilitate knowledge expansion in a data graph. A core algorithmic component to adopt this theory for generating exploration paths in data graphs is the automatic identification of knowledge anchors.

In Chapter 4, we uniquely adopted Rosch's definitions of BLO and cue validity were in the context of data graphs and developed several metrics and the corresponding algorithm for identifying KA_{DG} . The developed KA_{DG} algorithms were applied over two applications for data exploration, semantic browsing (in the musical instrument domain) and semantic search (in the career domain), using the data graphs from the two applications, MusicPinta and L4All, respectively. The outputs of KA_{DG} algorithms in MusicPinta and L4All data graphs were identified. Discussion about the algorithms output and possible applications of the KA_{DG} algorithms was provided.

Chapter 5 presented two experimental user studies for evaluating the KA_{DG} algorithms over MusicPinta and L4All data graphs. We adapted Cognitive Science experimental approaches for deriving the BLO in domain taxonomies, and defined an algorithm for identifying human BLO_{DG} used as benchmark for evaluating the KA_{DG} algorithms. The performance of the KA_{DG} algorithms was examined by comparing the KA_{DG} with the derived human BLO_{DG} , and hybridization heuristics for improving the performance of the algorithms were suggested.

In Chapter 6, we adopted the subsumption theory for meaningful learning to generate exploration paths for knowledge expansion in data graphs using knowledge anchors. Two algorithms were presented: (i) an algorithm which uses semantic similarity to identify the closest knowledge anchor to first entity of an exploration path, and (ii) a subsumption algorithm for generating transition narrative used to construct an exploration path in the data graph. Narrative scripts were used between entities in the generated path to help the user to learn meaningful relationships between a familiar entity (i.e. a knowledge anchor) and a new entity (i.e. subsumed entity). Both algorithms were formally described and applied over MusicPinta and L4All data graphs using the identified KA_{DG} .

A task-driven experimental user study was conducted in Chapter 7 to evaluate the exploration paths generated from the subsumption algorithm as compared to free exploration where the participants freely visited entities in the graph. To conduct the experimental user study, a data graph exploration task was designed following two steps: (i) designing a task template in the context of a general knowledge quiz to encourage participants to learn as much as they can, and (ii) identifying unfamiliar entities in the domain which can be plugged into the task template. The findings from the evaluation showed that the generated exploration paths using the subsumption algorithm lead to significantly higher increase in the users' knowledge compared to free exploration, which enables its adoption over different domains and contexts.

This Chapter describes the main contributions and outlines directions for future work

8.2 Contributions

Contributions by this research are the results from the attempt to address the two research questions introduced in Chapter 1. These include:

Computational methods for identifying knowledge anchors in data graphs. We addressed *RQ1* by utilizing Rosch’s definitions of BLO and cue validity [22] to develop metrics and corresponding algorithms for identifying KA_{DG} . We adapted metrics from FCA to develop distinctiveness metrics for identifying differentiated categories whose members don’t share attributes with other categories in the data graph. We also applied set-based similarity metrics to develop homogeneity metrics whose members share many attributes together (as discussed in Section 4.2). The formal description of the algorithms provides a generic solution for identifying familiar entities over data graphs, which make such entities useful in different ways. KA_{DG} enables operationalising the subsumption theory for meaningful learning [21] to generate exploration paths for knowledge expansion. In this theory, the KA_{DG} represent familiar entities from where links to introduce knowledge can be made. Furthermore, our approach for identifying KA_{DG} can be applied to ontology summarisation where KA_{DG} allow capturing a layman user’s view of the domain. KA_{DG} can be applied to solve the key problem of ‘cold start’ in *personalization and adaptation*. One of the popular choices for addressing the cold start problem is a dialogue system with the user. The data graphs can provide a large knowledge pool to implement such probing dialogue, however, one needs to select entities from the vast amount of possibilities for probing to avoid too long interactions with the user. This has been the focus of our work in [168] where we proposed an approach to detect user domain familiarity by exploiting KA_{DG} for probing interactions over data graph concepts.

The KA_{DG} algorithms were implemented in two data graphs (MusicPinta and L4All) from two domains (musical instruments and career), respectively. The implementation showed the importance of identifying edge labels used to indicate memberships of category entities in a data graph (i.e. the edge labels used to identify members of a category entity as correspond to line 2 in Algorithms 4.1, 4.2). This is important as it will affect the metrics values allocated for each category entity. Hence, inspection of the data graph before running the KA_{DG} algorithms to identify which edge labels will be used to indicate membership of category entities, is important (as discussed in Section 4.4). Furthermore, we noticed that the three homogeneity metrics (*Common Neighbours*, *Jaccard* and *Cosine*) gave the same lists of KA_{DG} . This indicates that homogeneity metrics can have similar performance, and one metric can be for future applications.

In order to evaluate the KA_{DG} metrics, we compared the outputs of the KA_{DG} metrics over the MusicPinta and L4All versus a benchmarking set of human BLO_{DG} from the

categories in the data graph, as identified by humans. An important step in the evaluation is to identify a suitable cut-off point for the KA_{DG} metrics. In the current implementation, this was done through experimentation – the best performance was by using the 60th percentile. The same percentile was identified as best for the career domain (see [87]). We expect that when applied to a range of data graphs, the 60th percentile will give reasonable performance (the best cut-off point for a specific data graph would require experimentation comparing different percentiles). Furthermore, The analysis indicated that hybridisation of the metrics notably improved performance. Appropriate hybridisation heuristics for the upper level of the data graph is to combine the KA_{DB} metrics using majority voting. The hybridisation heuristics for the bottom level of the hierarchy are dependent on the domain-specific relationships in the data graph. Hence, to derive appropriate hybridisation heuristics that give good performance for categories at the bottom level, further experimentation will be required. This will include comparing the KA_{DB} derived using the various domain-specific relationships against human BLO_{DB} .

Computational methods for generating exploration paths for knowledge expansion. We addressed *RQ2* by adapting the subsumption theory of meaningful learning [21] to generate exploration paths for knowledge expansion using the knowledge anchors. Adopting this theory involves identifying the most appropriate knowledge in the data graph to start generating the exploration path, and then iteratively subsume entities to introduce new knowledge while generating an exploration path. For this, we formally described two algorithms:

- Algorithm for identifying the closest knowledge anchor to the first entity of an exploration path. The algorithm applies a semantic similarity metric to identify the closest knowledge anchor to first entity of an exploration path (i.e. knowledge anchor with highest semantic similarity value with the first entity). The metric considers the class hierarchy in the data graph to identify similarity between two entities based on their depth in the data graph. Therefore, applying this metric in data graphs with shallow class hierarchies (e.g. class hierarchy depth = 1 or 2) may not give reasonable output, as there might be no common ancestors for first entity and knowledge anchors in the data graphs (i.e. the semantic similarity value between the two entities will be zero).
- Algorithm for generating exploration path as a set of transition narratives using the closest knowledge anchor. The closest knowledge anchor can have many subclass entities that exist at different levels of abstractions in the data graph. Hence, this algorithm identifies which subclasses to subsume and in what order for generating an exploration path. Furthermore, the algorithm uses narrative scripts between the entities in the exploration path. Providing meaningful narrative scripts between entities can help

layman users to create meaningful relationships between familiar entities and the new subsumed entities.

Both algorithms are generic and can be applied over different domains represented as data graphs. The algorithms were applied over two data graphs (MusicPinta and L4All). A controlled task-driven user study over MusicPinta was conducted to examine the knowledge utility and users' exploration experience using two conditions (i) *Experimental condition (EC)*: where users follow exploration paths generated using the subsumption algorithm, and (ii) *Control condition (CC)*: where users carry out free exploration and they are free to select entities to visit. All participants in the study have indicated that their exploration in the experimental condition was informative (in other words, they felt that while following the path they were able to find useful information); whereas 62% of the participants founded their free exploration trajectories to be informative. Furthermore, The expansion of the users' knowledge when following the generated exploration paths was higher than when following free exploration - the participants were able to *remember*, *categorize* and *compare* significantly more entities. The results also showed that the cognitive process *categorise* had the bigger effect on expanding the participants' knowledge. This was caused because: (i) the subsumption hierarchical relationship (`rdfs:subclassOf`) was used to create the narrative scripts between entities of the generated exploration paths, which helped the users to categorise new entities at different levels of abstraction at their cognitive structures; and (ii) the subsumption process uses knowledge anchors to subsume and learn new sub-categories similar to the way a human mind works while learning a new concept. The results showed that participants found the generated exploration paths to be more enjoyable and less confusing than free exploration paths, and their assessment of performance was higher. One participant founded his experience with hierarchical narrative scrips to be boring; and suggested that the system should diversify the types of narratives used between entities in the exploration path. For example, to use other relationships, other than the subsumption relationship, for generating transition narratives.

Instruments for evaluation of data graph exploration. Three instruments were developed:

- (i) **Approximating knowledge utility.** We proposed a measure in Section 2.7.1 for approximating knowledge utility of exploration paths. It approximates knowledge utility using three questions correspond to three cognitive processes: *remember*, *categorise* and *compare*, adapted from Bloom's taxonomy [27]. The proposed measure was applied in two experimental user studies where participants followed exploration paths under different settings. In the study reported in Section 3.5, knowledge utility was approximated for three exploration strategies (Familiarity, Unfamiliarity and Density strategies). The proposed measure was also applied in the experimental user study in

Chapter 7 to approximate knowledge utility of free exploration and exploration paths generated using the subsumption algorithm. This shows the applicability of the proposed measure for measuring knowledge utility. Furthermore, the proposed measure can be modified to include additional questions that correspond to other cognitive processes from Bloom's taxonomy. The current cognitive processes are extracted from the first two cognitive categories in Bloom's taxonomy, namely *remember* and *understand* categories which are directly related to exploration activities. More complex cognitive categories can be applied such as the cognitive category *analyse*. This category includes several cognitive processes that can be applied such as: *differentiate* (e.g. differentiate between two entities in the data graph), and *arrange* (e.g. arrange entities in the data graph from abstract to specific).

- (ii) **Algorithm to obtain human BLO_{DG} .** We formally described a generic algorithm for identifying human BLO_{DG} . The algorithm was applied in two application domains for data exploration, musical instrument and career which allowed us to *illustrate two ways of instantiating the algorithm for obtaining human BLO_{DG}* . On the one hand, MusicPinta describes concrete objects - musical instruments - that can have digital representations (e.g. image, audio, video). An image stimulus was used to represent musical instruments, and a free-naming task is used by showing image representations of graph entities and asking the users to quickly name the entities they see. On the other hand, L4All comprises of abstract career categories, such as *Occupation* and *Subject*, which have text representations (i.e. labels of entities in the data graph) but no clearly distinguishable images. In this case, a category verification task was used to obtain human BLO_{DG} by showing text representations of graph entities and asking the user to identify the matching entity given some answers. Offering two ways of instantiating the BLO_{DG} increases the generality of the algorithm to be applied over different domains represented as data graphs. The identified human BLO_{DG} can have different applications. In the current work, we use human BLO_{DG} as benchmarking set for evaluating the KA_{DG} algorithms. An important outcome from this evaluation showed the sensitivity of the human BLO_{DG} algorithm to the quality of the ontology. This points at another possible application of human BLO_{DG} —peculiarities in the output can indicate deficiencies of the ontology which provide insights for ontology re-engineering.
- (iii) **User-driven exploration task.** We followed a two-step approach to design suitable data exploration task for users. The two steps involved designing a task template and then identifying unfamiliar entities in the domain which can plugged into the task template. The task template was designed in the context of a general knowledge quiz to encourage users to seek knowledge in a given domain represented as a data graph. To identify unfamiliar entities, we conducted a small-scale survey with participants in MusicPinta

data graph asking them about their familiarity with entities in domain. Overall 61 entities were selected from the bottom quartile of the MusicPinta data graph and presented to participants to indicate familiarity with the entities. However, following such approach for identifying unfamiliar entities can be demanding especially if the data graphs has 1000s of entities. In such case, crowdsourcing can be an optimal solution for identifying unfamiliar entities in the domain.

8.3 Future Work

Based on the identified limitations of the algorithms and evaluation presented in this research, this Section discusses immediate and future work.

8.3.1 Immediate

The immediate extensions of this work concern mainly technical improvements on the algorithms. In particular, we will focus on improving the algorithm for identifying the closest knowledge anchor to first entity of an exploration path (Algorithm 6.1). It was shown that two (or more) knowledge anchors in a data graph can have the same semantic similarity value with the first entity. For example, there were two knowledge anchors (`Flute` and `Reeds`) with the same semantic similarity value with the first entity `Bansuri`. One possible way to address this is to filter the knowledge anchors based on their density and give preference to the densest anchor as it will include many subclass members to subsume while generating the exploration path.

Another possible immediate future work is to enhance the subsumption algorithm (Algorithm 6.2). The algorithm is dependent on the sub-classes of the selected knowledge anchor – it may not be possible to generate m transition narratives in the exploration path (where m is the required length of exploration path identified as an input of the algorithm). Our implementation uses only the knowledge anchor, and can result in paths whose length of less than m . Another way to address this would be to continue the path, using another knowledge anchor. It is also possible to use superordinate categories to the knowledge anchor to extend an exploration path. This will be suitable for cases when the user is gained knowledge at the subordinate level and is ready to generalise to a more abstract level. In such cases, a user model will be required.

8.3.2 Long-term

In the long term, we will apply and evaluate the developed algorithms in another domain. Interesting domains include: `movie`, `animal` and `sport`. However, an important element that has to be taken into account in selecting a domain is the availability of application user

interface which allows users to initiate an exploration path from a single entity in the data graph, and then to follow transition narratives of exploration paths. Furthermore, while conducting experimental user studies for evaluating the algorithms in the new domains, we will identify interesting entities in the data graph by asking the participants to explicitly indicate interesting entities for them and to provide reasonable justification why these entities are interesting. This will help us to tune the KA_{DG} algorithms to identify familiar, yet interesting KA_{DG} .

Furthermore, in the long run we will extend the application of our work. An interesting application for the human BLO_{DG} algorithm is ontology validation. The human BLO_{DG} algorithm's output can indicate deficiencies of the ontology which can provide insights for re-engineering the ontology. For this, the human BLO_{DG} can be applied in crowdsourcing application to identify familiar entities in a domain as identified from the crowd used to validate the underpinning domain ontology. Another interesting application is to combine KA_{DG} with semantic search queries to enhance interfaces for users search. We have conducted jointly work in [169] towards this future direction. The work proposed a hybrid approach that has been applied to interacting with L4All to explore future career options. The future work will focus on evaluating the proposed approach with groups of students and practitioners, and also to investigate other ways of hybridising flexible queries and KA_{DG} for filtering or ranking query results.

List of References

1. Palagi, E., Gandon, F., Giboin, A., Troncy, R., Antipolis, I.S., Gandon, F., Antipolis, I.S., Giboin, A., Antipolis, I.S.: A survey of definitions and models of exploratory search. In: ACM Workshop on Exploratory Search and Interactive Data Analytics - ESIDA '17. pp. 3–8 (2017).
2. Roth, R.W.W. and R.A.: Exploratory Search Beyond the Query–Response Paradigm. Morgan & Claypool (2009).
3. Belkin, N.: Anomalous States of Knowledge as the Basis of Information Retrieval. *Can. J. Inf. Sci.* 5, 133–143 (1980).
4. Marchionini, G.: Exploratory search: from finding to understanding. In: Communications of the ACM. p. 41 (2006).
5. Berners-lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Lerer, A., Sheets, D.: Tabulator : Exploring and Analyzing linked data on the Semantic Web. In: Rutledge, L., M C Schraefel, Bernstein, A., and Degler, D. (eds.) 3rd International Semantic Web User Interaction Workshop. Citeseer (2006).
6. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellamnn, S.: DBpedia - A cystallization point for the Web of Data. *Web Semant. Sci. Serv. Agents World Wide Web.* 7, 154–165 (2009).
7. Cheng, G., Zhang, Y., Qu, Y.: Explass: Exploring Associations between Entities via Top-K Ontological Patterns and Facets. In: ISWC '13. pp. 422–437 (2014).
8. Qu, G.C. and Y.: Searching linked objects with falcons: Approach, implementation and evaluation. *Int. J. Semant. Web Inf. Syst.* 5, 49–70 (2009).
9. Zheng, L., Qu, Y., Jiang, J., Cheng, G.: Facilitating entity navigation through top-K link patterns. In: In Proceedings of International Semantic Web Conference. pp. 163–179 (2015).
10. Sah, M., Wade, V.: Personalized concept-based search on the Linked Open Data. *Web Semant. Sci. Serv. Agents World Wide Web.* 36, 32–57 (2016).
11. Zheng, L., Qu, Y., Cheng, G.: Leveraging link pattern for entity-centric exploration over Linked Data. In: Proceedings of WWW. World Wide Web (2017).
12. Pienta, R., Tech, G., Vreeken, J., Tech, G., Abello, J.: FACETS : Adaptive Local Exploration of Large Graphs. In: IEEE SDM'17 (2017).
13. Dietze, S., Kaldoudi, E.: Socio-semantic Integration of Educational Resources - the Case of the mEducator Project. *Univers. Comput. Sci.* 19, 1543–1569 (2013).
14. Vakkari, P.: Searching as learning: A systematization based on literature. *J. Inf. Sci.* 42, 7–18 (2016).
15. Gwizdka, J., Hansen, P., Hauff, C., He, J., Kando, N.: Search As Learning (SAL) Workshop 2016. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 1249–1250. ACM, New York, NY, USA (2016).
16. Koesten, L., Kacprzak, E., Tennison, J.: Learning When Searching for Web Data. In: Sal@SigirSearch as Learning-SAL@SIGIR. pp. 2–3 (2016).
17. Dimitrova, V., Lau, L., Thakker, D., Yang-Turner, F., Despotakis, D.: Exploring exploratory

- search: a user study with linked semantic data. Proc. 2nd Int. Work. Intell. Explor. Semant. Data - IESD '13. 1–8 (2013).
18. Thakker, D., Dimitrova, V., Lau, L., Yang-Turner, F., Despotakis, D.: Assisting user browsing over linked data: Requirements elicitation with a user study. In: ICWE'13 International conference on Web Engineering. pp. 376–383 (2013).
 19. Maccatrozzo, V.: Burst the filter bubble: using semantic web to enable serendipity. In: ISWC '11 (2012).
 20. Al-tawil, M., Thakker, D., Dimitrova, V.: Nudging to Expand User ' s Domain Knowledge while Exploring Linked Data. In: IESD@ISWC (2014).
 21. Ausubel, D.P.: A subsumption theory of meaningful verbal learning and retention. *J. Gen. Psychol.* 66, 213–224 (1962).
 22. Rosch, E., Mervis, C.B., Gray, W.D., Johnson, D.M., P.Boyes-Braem: Basic Objects in Neutral categories. *Cogn. Psychol.* 8, 382–439 (1976).
 23. Belohlavek, R., Trnecka, M.: Basic level in formal concept analysis: Interesting concepts and psychological ramifications. *IJCAI Int. Jt. Conf. Artif. Intell.* 1233–1239 (2013).
 24. Liben-Nowell, D., Kleinberg, J.: The link prediction problem for social networks. In: Proceedings of the twelfth international conference on Information and knowledge management - CIKM '03. pp. 556–559 (2003).
 25. Ausubel, D.P.: The use of advance organizers in the learning and retention of meaningful verbal material. *J. Educ. Psychol.* 51, 267–272 (1960).
 26. Ausubel, D.P., Novak, J.D., Hanesian, H.: *Educational Psychology: A Cognitive View*. New York : Holt, Rinehart and Winston (1978).
 27. Krathwohl, D.R.: A Revision of Bloom's Taxonomy : An Overview. *Theory Pract.* 41, (2002).
 28. Lee, T.B., Hendler, J., Lassila, O., others: The semantic web. *Sci. Am.* 284, 34–43 (2001).
 29. Berners-lee, T., Cailliau, R.: The world-wide web. *Comput. Networks ISDN Syst.* 25, 454–459 (1992).
 30. Donato, D., Laura, L., Leonardi, S., Millozzi, S.: The Web as a graph. Proc. 2000 ACM SIGMOD-SIGACT-SIGART Symp. Princ. database Syst. (2000).
 31. Lehmborg, O., Meusel, R., Bizer, C.: Graph structure in the web. *Comput. Networks.* 33, 309–320 (2000).
 32. Bizer, C., Heath, T., Berners-Lee, T.: Linked data-the story so far. *Int. J. Semant. Web Inf. Syst.* (2009).
 33. Berners-Lee, T.: *Weaving the Web: The Past, Present and Future of the World Wide Web by its Inventor.* , London, Texere (2000).
 34. Heath, T., Bizer, C.: *Linked data: Evolving the Web into a global data space (1st edition)*. (2011).
 35. Manola, F., Miller, E., McBride, B.: RDF primer. *W3C Recomm.* 10, 1–107 (2004).
 36. Berners-lee, T.: *Linked Data - Design Issues*, <http://www.w3.org/DesignIssues/LinkedData.html>.
 37. Shadbolt, N., Hall, W., Berners-Lee, T.: The semantic web revisited. *IEEE Intell. Syst.* 21, 96–101 (2006).
 38. Gruber, T.R.: A Translation Approach to Portable Ontology Specifications. *Knowl. Acquis.* 5,

- 199–220 (1993).
39. Wang, H., Wu, T., Qi, G., Ruan, T.: Adoption of the Linked Data Best Practices in Different Topical Domains. *Int. Semant. Web Conf.* 8796, 293–308 (2014).
 40. W3C: RDF Vocabulary Description Language 1.0: RDF Schema, (2004).
 41. Deborah L. McGuinness, F. van H.: Owl web ontology language overview. *W3C Recomm.* 10.2004-03. 2004, 1–12 (2004).
 42. Recommendation, W.: SPARQL 1.1 Query Language, <http://www.w3.org/TR/2013/REC-sparql11-query-20130321/>.
 43. Feigenbaum, L.: SPARQL By Example - A tutorial, <https://www.w3.org/2009/Talks/0615-qbe/>.
 44. Faye, D.C., Cure, O., Blin, G.: A survey of RDF storage approaches. *ARIMA J.* 15, 11–35 (2012).
 45. Lehmann, J., Auer, S., Capadisli, S., Janowicz, K., Bizer, C., Heath, T., Hogan, A., Berners-Lee, T.: LDOW2017: 10th Workshop on Linked Data on the Web.
 46. Angles, R., Gutierrez, C.: Survey of graph database models. *ACM Comput. Surv.* 40, 1–39 (2008).
 47. Angles, R., Gutierrez, C.: Querying RDF Data from a Graph Database Perspective. In: Gómez-Pérez, A. and Euzenat, J. (eds.) *The Semantic Web: Research and Applications: Second European Semantic Web Conference, ESWC 2005, Heraklion, Crete, Greece, May 29--June 1, 2005. Proceedings.* pp. 346–360. Springer Berlin Heidelberg, Angles2005 (2005).
 48. Luo, Y., Picalausa, F., Fletcher, G.H.L., Hidders, J., Vansummeren, S.: Storing and indexing massive RDF datasets. In: *Semantic Search over the Web.* pp. 31–60. Springer Berlin Heidelberg (2012).
 49. Papailiou, N., Konstantinou, I., Tsoumakos, D., Karras, P., Koziris, N.: H2RDF+: High-performance distributed joins over large-scale RDF graphs. In: *Proceedings - 2013 IEEE International Conference on Big Data, Big Data 2013.* pp. 255–263 (2013).
 50. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In: Horrocks, I. and Hendler, J. (eds.) *The Semantic Web --- ISWC 2002: First International Semantic Web Conference Sardinia, Italy, June 9--12, 2002 Proceedings.* pp. 54–68. Springer Berlin Heidelberg, Berlin, Heidelberg (2002).
 51. Carroll, J.J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A., Wilkinson, K.: Jena: Implementing the Semantic Web Recommendations. In: *Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers.* pp. 74–83 (2004).
 52. Erling, O., Mikhailov, I.: Virtuoso: RDF Support in a Native RDBMS. In: de Virgilio, R., Giunchiglia, F., and Tanca, L. (eds.) *Semantic Web Information Management: A Model-Based Perspective.* pp. 501–519. Springer Berlin Heidelberg, Berlin, Heidelberg (2010).
 53. Harris, S., Lamb, N., Shadbolt, N.: 4store: The design and implementation of a clustered RDF store. In: *Proceedings of the 5th International Workshop on Scalable Semantic Web Knowledge Base Systems.* pp. 94–109 (2009).
 54. Husain, M.F., McGlothlin, J., Masud, M.M., Khan, L.R., Thuraisingham, B.: Heuristics-Based Query Processing for Large RDF Graphs Using Cloud Computing. *IEEE Trans. Knowl. Data Eng.* 23, (2011).

55. MA, Z., CAPRETZ, M.A.M., YAN, L.: Storing massive Resource Description Framework. *Knowl. Eng. Rev.* 31, 391–413 (2016).
56. Bönström, V., Hinze, A., Schweppe, H.: Storing RDF as a graph. In: *Proceedings of the First Conference on Latin American Web Congress*. pp. 27–36 (2003).
57. Baeza, P.B.: Querying graph databases. In: *Proceedings of the 32Nd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. pp. 175–188 (2013).
58. de Freitas, S., Harrison, I., Magoulas, G., Papamarkos, G., Poulouvassilis, A., Van Labeke, N., Mee, A., Oliver, M.: L4All, a Web-Service Based System for Lifelong Learners. *Learn. Grid Handb. Concepts, Technol. Appl.* 143–155 (2008).
59. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - A crystallization point for the Web of Data. *Web Semant. Sci. Serv. Agents World Wide Web.* 7, 154–165 (2009).
60. Becker, C., Bizer, C.: DBpedia mobile: A location-enabled linked data browser. In: *LDOW.*, Beijing, China (2008).
61. Alahmari, F., Thom, J.A., Magee, L., Wong, W.: Evaluating semantic browsers for consuming linked data. In: *Proceedings of the Twenty-* pp. 89–98 (2012).
62. Marie, N., Gandon, F.: Survey of linked data based exploration systems. In: *IESD@ISWC* (2014).
63. Brunetti, J.M., García, R., Auer, S.: From Overview to Facets and Pivoting for Interactive Exploration of Semantic Web Data. *Int. J. Semant. Web Inf. Syst.* 9, 1–20 (2013).
64. Nuzzolese, A.G., Presutti, V., Gangemi, A., Peroni, S., Ciancarini, P.: Aemoo: Linked Data exploration based on Knowledge Patterns. *Semant. Web.* 8, 87–112 (2017).
65. Troullinou, G., Kondylakis, H., Daskalaki, E., Plexousakis, D., Gandon, F., Sabou, M., Sack, H.: Ontology understanding without tears: The-summarization approach. *Semant. Web.* 8, 797–815 (2017).
66. Ferr'e, S., Hermann, A.: Semantic Search: Reconciling Expressive Querying and Exploratory Search. In: *10th International Semantic Web Conference*. pp. 177–192. Springer (2011).
67. Dadzie, A.S., Pietriga, E.: Visualisation of Linked Data - Reprise. *Semant. Web.* 8, 1–21 (2017).
68. Dadzie, A.S., Rowe, M.: Approaches to visualising Linked Data: A survey. *Semant. Web.* 2, 89–124 (2011).
69. Javed, W., Ghani, S., Elmqvist, N.: PolyZoom : Multiscale and Multifocus Exploration in 2D Visual Spaces. In: *CHI'12. ACM* (2012).
70. Zviedris, M., Barzdins, G.: Viziquer: A tool to explore and query SPARQL endpoints. In: *Proceedings of 8th ESWC*. pp. 441–445 (2011).
71. Huynh, D., Karger, D.: Parallax and companion: Set-based browsing for the data web. *WWW Conf. 2005–2008* (2009).
72. Popov, I.O., Schraefel, M.C., Hall, W., Shadbolt, N.: Connecting the Dots: A Multi-pivot Approach to Data Exploration. In: *ISWC'10*. pp. 553–568 (2011).
73. Pound, J., Mika, P., Zaragoza, H.: Ad-hoc object retrieval in the web of data. In: *Proceedings of the 19th international conference on World wide web - WWW '10*. p. 771 (2010).
74. Harth, A.: Visinav: Visual web data search and navigation. In: *DEXA*. pp. 214–228 (2009).

75. Sah, M., Wade, V.: Personalized Concept-Based Search and Exploration on the Web of Data Using Results Categorization. In: Proceedings of the Extended Semantic Web Conference. pp. 532–547 (2013).
76. Tzitzikas, Y., Manolis, N., Papadakos, P.: Faceted exploration of RDF/S datasets: a survey. *J. Intell. Inf. Syst.* 48, 329–364 (2017).
77. Koren, J., Zhang, Y., Liu, X.: Personalized interactive faceted search. *Proceeding 17th Int. Conf. World Wide Web - WWW '08.* 477–485 (2008).
78. Tummarello, G., Cyganiak, R., Catasta, M., Danielczyk, S., Delbru, R., Decker, S.: Sig.ma: Live views on the Web of Data. *Web Semant. Sci. Serv. Agents World Wide Web.* 8, 355–364 (2010).
79. Usbeck, R., Ngomo, A.C.N., Bühmann, L., Unger, C.: HAWK - hybrid question answering using linked data. In: Proceedings of the 12th European Semantic Web Conference, ESWC 2015. pp. 353–368. Springer International Publishing (2015).
80. Ding, L., Pan, R., Finin, T., Joshi, A., Peng, Y., Kolari, P.: Finding and ranking knowledge on the semantic web. In: Proceedings of the 4th International Semantic Web Conference. pp. 156–170 (2005).
81. Oren, E., Delbru, R., Catasta, M., Cyganiak, R., Stenzhorn, H., Tummarello, G.: Sindice.com: a document-oriented lookup index for open linked data. *Int. J. Metadata, Semant. Ontol.* 3, 37–52 (2008).
82. D'Aquin, M., Motta, E., Sabou, M., Angeletou, S., Gridinoc, L., Lopez, V., Guidi, D.: Toward a new generation of semantic web applications. *IEEE Intell. Syst.* 23, 20–28 (2008).
83. Marchionini, G., Shneiderman, B.: Finding Facts vs. Browsing Knowledge in Hypertext Systems. *Computer (Long. Beach. Calif.)* 21, 70–80 (1988).
84. Shneiderman, B.: The eyes have it: a task by data type taxonomy for information visualizations. In: Proceedings 1996 IEEE Symposium on Visual Languages. pp. 336--343. IEEE (1996).
85. Mazumdar, S., Petrelli, D., Elbedweihy, K., Lanfranchi, V., Ciravegna, F.: Affective graphs: The visual appeal of Linked Data. *Semant. Web.* 6, 277–312 (2015).
86. Valsecchi, A., Abrate, M., Bacciu, C., Tesconi, M., Marchetti, A.: Linked Data Maps: Providing a Visual Entry Point for the Exploration of Datasets. *IESD@ISWC.* 1472, (2015).
87. Smart, P.R., Russell, A., Braines, D., Kalfoglou, Y., Bao, J., Shadbolt, N.: A Visual Approach to Semantic Query Design Using a Web-Based Graphical Query Designer. In: Proceedings of the 16th International Conference on Knowledge Engineering and Knowledge Management. pp. 275–291 (2008).
88. Scheider, S., Degbelo, A., Lemmens, R., Van Elzakker, C., Zimmerhof, P., Kostic, N., Jones, J., Banhatti, G.: Exploratory querying of SPARQL endpoints in space and time. *Semant. Web.* 8, 65–86 (2017).
89. Heim, P., Ziegler, J., Lohmann, S.: GFacet: A browser for the web of data. In: IMC-SSW'08. pp. 49–58 (2008).
90. Brunk, S., Heim, P.: Tfacet: Hierarchical faceted exploration of semantic data using well-known interaction concepts. In: DCI@ INTERACT. pp. 31–36 (2011).
91. Stadler, C., Martin, M., Auer, S.: Exploring the web of spatial data with facete. *Proc. 23rd Int.*

- Conf. World Wide Web - WWW '14 Companion. 175–178 (2014).
92. Papadakos, P., Tzitzikas, Y.: Hippalus: Preference-enriched faceted exploration. In: EDBT@ICDT (2014).
 93. Wongsuphasawat, K., Moritz, D., Anand, A., Mackinlay, J., Howe, B., Heer, J.: Voyager: Exploratory Analysis via Faceted Browsing of Visualization Recommendations. *IEEE Trans. Vis. Comput. Graph.* 22, 649–658 (2016).
 94. Rossel, O.: Implementation of a “ search and browse ” scenario for the LinkedData. In: IESD@ISWC (2014).
 95. De Vocht, L., Dimou, A., Breuer, J., Van Compernelle, M., Verborgh, R., Mannens, E., Mechant, P., Van De Walle, R.: A visual exploration workflow as enabler for the exploitation of linked open data. In: IESD@ISWC (2014).
 96. Dojchinovski, M., Vitvar, T.: Personalised Access to Linked Data. In: *Knowledge Engineering and Knowledge Management*. pp. 121–136 (2014).
 97. Musto, C., Lops, P., Basile, P., de Gemmis, M., Semeraro, G.: Semantics-aware Graph-based Recommender Systems Exploiting Linked Open Data. In: *UMAP '16*. pp. 229–237 (2016).
 98. Bianchi, F., Palmonari, M., Cremaschi, M., Fersini, E.: Actively Learning to Rank Semantic Associations for Personalized Contextual Exploration of Knowledge Graphs. In: *Proceedings of the 14th International Conference, ESWC 2017*. pp. 120–135 (2017).
 99. Freeman, L.C.: Centrality in social networks conceptual clarification. *Soc. Networks.* 1, 215–239 (1978).
 100. Li, N., Motta, E.: Evaluations of user-driven ontology summarization. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*. 6317, 544–553 (2010).
 101. Li, N., Motta, E.: Evaluations of user-driven ontology summarization. *EKAW 17th Int. Conf. Knowl. Eng. Manag. by masses*. 6317, 544–553 (2011).
 102. Fu, B., Noy, N.F., Storey, M.: Eye Tracking the User Experience - An Evaluation of Ontology Visualization Techniques. *Semant. Web J.* 8, 23–41 (2017).
 103. Zhang, X., Cheng, G., Qu, Y.: Ontology summarization based on rdf sentence graph. In: *Proceedings of the 16th international conference on World Wide Web WWW 07*. ACM Press (2007).
 104. Wang, K., Wang, Z., Topor, R., Pan, J.Z., Antoniou, G.: Eliminating concepts and roles from ontologies in expressive descriptive logics. *Comput. Intell.* 30, 205–232 (2014).
 105. Troullinou, G., Kondylakis, H., Daskalaki, E., Plexousakis, D.: RDF Digest: Efficient Summarization of RDF / S KBs. In: Gandon F., Sabou M., Sack H., d’Amato C., Cudré-Mauroux P., Zimmermann A. (eds) *The Semantic Web. Latest Advances and New Domains. ESWC 2015. Lecture Notes in Computer Science*, vol 9088. Springer, Cham (2015).
 106. Wille, R.: Formal Concept Analysis as Mathematical Theory of Concepts and Concept Hierarchies. *Form. Concept Anal.* 1–33 (2005).
 107. Poelmans, J., Ignatov, D.I., Kuznetsov, S.O., Dedene, G.: Formal concept analysis in knowledge processing: A survey on applications. *Expert Syst. Appl.* 40, 6538–6560 (2013).
 108. Cimiano, P., Hotho, A., Staab, S.: Learning Concept Hierarchies from Text Corpora Using Formal

- Concept Analysis. *J. Artif. Int. Res.* 24, 305–339 (2005).
109. Cho, W.C., Richards, D.: Improvement of precision and recall for information retrieval in a narrow domain: Reuse of concepts by formal concept analysis. In: *Proceedings - IEEE/WIC/ACM International Conference on Web Intelligence, WI 2004*. pp. 370–376 (2004).
 110. Richards, D.: Ad-Hoc and Personal Ontologies: A Prototyping Approach to Ontology Engineering. In: *PKAW*. pp. 13–24 (2006).
 111. Sertkaya, B.: OntoComp: A Protégé Plugin for Completing OWL Ontologies. In: Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvönen, E., Mizoguchi, R., Oren, E., Sabou, M., and Simperl, E. (eds.) *The Semantic Web: Research and Applications: 6th European Semantic Web Conference, ESWC 2009 Heraklion, Crete, Greece, May 31--June 4, 2009 Proceedings*. pp. 898–902. Springer Berlin Heidelberg, Berlin, Heidelberg (2009).
 112. Monnin, P., Lezoche, M., Napoli, A., Coulet, A.: Using formal concept analysis for checking the structure of an ontology in LOD: The example of DBpedia. In: *23rd International Symposium on Methodologies for Intelligent Systems, ISMIS (2017)*.
 113. Peroni, S., Motta, E., Aquin, M.: Identifying key concepts in an ontology , through the integration of cognitive principles with statistical and topological measures. In: *ASWC '08 (2008)*.
 114. Cai, Y., Chen, W.H., Leung, H.F., Li, Q., Xie, H., Lau, R.Y.K., Min, H., Wang, F.L.: Context-aware ontologies generation with basic level concepts from collaborative tags. *Neurocomputing*. 208, 25–38 (2016).
 115. Belohlavek, R., Trnecka, M.: Basic level of concepts in formal concept analysis. *ICFCA'10*. 7278 LNAI, 28–44 (2012).
 116. Jones, G. V: Identifying Basic Categories. *Psychol. Bull.* 94, 423–428 (1983).
 117. Corter, James E.; Gluck, M.A.: *Explaining Basic Categories: Feature Predictability and Information*, (1992).
 118. Rosch, E., Lloyd, B.B.: *Cognition and Categorization*. *Lloydia Cincinnati*. pp, 27–48 (1978).
 119. Lee, B., Plaisant, C., Sims, C., Fekete, J., Lee, B., Plaisant, C., Sims, C., Fekete, J., Henry, N., Lee, B., Plaisant, C.: Task taxonomy for graph visualization. In: *BELIV '06*. pp. 1–5 (2006).
 120. Dörk, M., Henry Riche, N., Ramos, G., Dumais, S.: PivotPaths: Strolling through Faceted Information Spaces. *IEEE Trans. Vis. Comput. Graph.* 18, 2709–2718 (2012).
 121. Kleineremann, F., De Troyer, O., Creelle, C., Pellens, B.: Adding semantic annotations, navigation paths and tour guides to existing virtual environments. In: *Proceedings of the International Conference on Virtual Systems and Multimedia, VSMM 2007*. pp. 100–111 (2007).
 122. Bhandari, U., Sugiyama, K., Datta, A., Jindal, R.: Serendipitous recommendation for mobile apps using item-item similarity graph. In: *Information Retrieval Technology. AIRS 2013. Lecture Notes in Computer Science*. Springer, Berlin, Heidelberg (2013).
 123. Yang, F., Li, F.W.B., Lau, R.W.H.: A fine-grained outcome-based learning path model. *IEEE Trans. Syst. Man, Cybern. Syst.* 44, 235–245 (2014).
 124. Tam, V., Lam, E.Y., Fung, S.T.: A new framework of concept clustering and learning path optimization to develop the next-generation e-learning systems. *Comput. Educ.* 1, (2014).
 125. Muhammad, A., Zhou, Q., Beydoun, G., Xu, D., Shen, J.: Learning path adaptation in online

- learning systems. In: 2016 IEEE 20th International Conference on Computer Supported Cooperative Work in Design (CSCWD). pp. 421–426. IEEE (2016).
126. Chen, C.M.: Ontology-based concept map for planning a personalised learning path. *Br. J. Educ. Technol.* 40, 1028–1058 (2009).
 127. Li, Z., Papaemmanouil, O., Koutrika, G.: CourseNavigator: Interactive Learning Path Exploration. In: Proceedings of the 3rd International Workshop on Exploratory Search in Databases and the Web. pp. 6–11 (2016).
 128. Bonifati, A., Ciucanu, R., Lemay, A.: Learning Path Queries on Graph Databases. In: Proceedings of the 18th International Conference on Extending Database Technology (EDBT) (2015).
 129. Anyanwu, K., Sheth, A.: ρ -Queries Enabling Querying for Semantic Associations on the Semantic Web.pdf. In: Proceedings of the 12th International Conference on World Wide Web. pp. 690–699 (2003).
 130. Heim, P., Hellmann, S., Lehmann, J., Lohmann, S.: RelFinder : Revealing Relationships in RDF Knowledge Bases.
 131. García, R., Gil, R., Gimeno, J.M., Bakke, E., Karger, D.R.: BESDUI: A benchmark for end-user structured data user interfaces. In: ISWC '16th. pp. 65–79 (2016).
 132. Lamprecht, D., Strohmaier, M., Helic, D., Nyulas, C., Tudorache, T., Noy, N.F., Musen, M.A.: Using ontologies to model human navigation behavior in information networks: A study based on Wikipedia. *Semant. Web.* 6, 403–422 (2015).
 133. Wildemuth, B.M., Freund, L.: Assigning search tasks designed to elicit exploratory search behaviors. In: Proceedings of the Symposium on Human-Computer Interaction and Information Retrieval - HCIR '12. pp. 1–10. ACM Press, New York, New York, USA (2012).
 134. Hersh, W., Pentecost, J., Hickam, D.: A task-oriented approach to information retrieval evaluation. *Jasis.* 47, 50–56 (1996).
 135. Carr, S.C., Thompson, B.: The effects of prior knowledge and schema activation strategies on the inferential reading comprehension of children with and without learning disabilities. *Learn. Disabil. Q.* 19, 48–61 (1996).
 136. Egusa, Y., Saito, H., Takaku, M., Terai, H., Miwa, M., Kando, N.: Using a concept map to evaluate exploratory search - p175-egusa.pdf. In: IIX '10 Proceedings of the third symposium on Information interaction in context. pp. 175–184 (2010).
 137. Freund, L., Dodson, S., Kopak, R.: On measuring learning in search: A position paper. In: Search as Learning (SAL) workshop. pp. 1–2 (2016).
 138. Bloom, B.S., Englehard, M.D., Furst, E.J., Hill, W.H., Krathwohl, D.R.: Taxonomy of Educational Objectives: The Classification of Educational Goals: Handbook I Cognitive Domain. New York. 16, 207 (1956).
 139. McVee, M.B., Dunsmore, K., R, J.: Schema Theory Revisited. *Rev. Educ. Res.* 75, 531–566 (2005).
 140. Bartlett, F.: Thinking: An experimental and social study. , New York: Basic Books (1958).
 141. Schunk, D.H.: Learning theories : an educational perspective. Pearson Education (2012).
 142. Cohen, C.E., Ebbesen, E.B.: Observational goals and schema activation: A theoretical framework

- for behavior perception. *J. Exp. Soc. Psychol.* 15, 305–329 (1979).
143. RUMELHART, D.E.: Schemata and Cognitive Systems. *Handb. Soc. Cogn.* 1, (1984).
 144. Marshall, S.P.: *Schemas in Problem Solving*. Cambridge University Press (1995).
 145. Kalyuga, S.: Rapid Assessment of Learners' Knowledge in Adaptive Learning Environments. In: *Artificial Intelligence In Education*. pp. 167–174 (2003).
 146. Rumelhart, D.E., Norman, D.A.: Accretion, Tuning and Restructuring: Three Modes of Learning, (1976).
 147. Ausubel, D.P., Fitzgerald, D.: ANTECEDENT LEARNING VARIABLES IN SEQUENTIAL VERBAL LEARNING. (1962).
 148. Bruce Joyce, Marsha Weil, E.C.: *Models of Teaching*. (2004).
 149. Ausubel, D.P., Novak, J.D., Helen Hanesian: *Education Psychology: A cognitive View*. (1968).
 150. Ausubel, D.P.: Cognitive structure and the facilitation of meaningful verbal learning. *J. Teach. Educ.* 14, 217–222 (1963).
 151. Palmer, C.F., Jones, R.K., Hennessy, B.L., Unze, M.G., Pick, A.D.: How Is a Trumpet Known? The “Basic Object Level” Concept and Perception of Musical Instruments. *Am. J. Psychol.* 102, (1989).
 152. Joliceur, P., Gluck, M.A., Kosslyn, S.M.: Pictures and Naming: Making the Connection. *Cogn. Psychol.* 16, 243–275 (1984).
 153. Henry Kucera, W.N.F.: *Computational Analysis of Present-Day American English*. *Am. Doc.* (1968).
 154. Poulouvassilis, A., Selmer, P., Wood, P.T.: Flexible querying of lifelong learner metadata. *IEEE Trans. Learn. Technol.* 5, 117–129 (2012).
 155. Landherr, A., Friedl, B., Heidemann, J.: A Critical Review of Centrality Measures in Social Networks. *Bus. {&} Inf. Syst. Eng.* 2, 371–385 (2010).
 156. Landherr, A., Friedl, B., Heidemann, J.: A Critical Review of Centrality Measures in Social Networks. *Bus. {&} Inf. Syst. Eng.* 2, 371–385 (2010).
 157. Miller, G.A.: The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychol. Rev.* 63, (1956).
 158. Dahlbäck, N., Jönsson, A., Ahrenberg, L.: Wizard of Oz studies — why and how. *Knowledge-Based Syst.* 6, 258–266 (1993).
 159. Gupta, V., Hanges, P.J., Dorfman, P.: Cultural clusters: Methodology and findings. *J. World Bus.* 37, 11–15 (2002).
 160. Sandra G. HartLowell E. Staveland: Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. *Adv. Psychol.* 52, 139–183 (1988).
 161. Hart, S.G., Staveland, L.E.: Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. *Adv. Psychol.* 52, 139–183 (1988).
 162. Wille, R.: RESTRUCTURING LATTICE THEORY: AN APPROACH BASED ON HIERARCHIES OF CONCEPTS. In: Ferré, S. and Rudolph, S. (eds.) *Formal Concept Analysis: 7th International Conference, ICFCA 2009 Darmstadt, Germany, May 21-24, 2009 Proceedings*. pp. 314–339. Springer Berlin Heidelberg, Berlin, Heidelberg (2009).

163. Wu, Z., Palmer, M.: Verbs semantics and lexical selection. Proc. 32nd Annu. Meet. Assoc. Comput. Linguist. -. 133–138 (1994).
164. Kules, B., Capra, R., Banta, M., Sierra, T.: What Do Exploratory Searchers Look at in a Faceted Search Interface? JCDL '09 Proc. 9th ACM/IEEE-CS Jt. Conf. Digit. Libr. . 313–322 (2009).
165. Nunes, T., Schwabe, D.: Frameworks of Information Exploration - Towards the Evaluation of Exploration Systems Conceptual View of an Exploration Framework. In: IESD@ISWC (2016).
166. Kules, B., Capra, R.: Creating exploratory tasks for a faceted search interface. In: 2nd Workshop on Human–Computer Interaction (2008).
167. Tanaka, J.W., Taylor, M.: Object categories and expertise: Is the basic-level in the eye of the beholder? Cogn. Psychol. 23, 457–482 (1991).
168. Al-tawil, M., Dimitrova, V., Thakker, D.: Using Basic Level Concepts in a Linked Data Graph to Detect User’s Domain Familiarity. In: UMAP. , Dublin, Ireland (2015).
169. Poulouvassilis, A., Al-Tawil, M., Frosini, R., Dimartino, M., Dimitrova, V.: Combining Flexible Queries and Knowledge Anchors to facilitate the exploration of Knowledge Graphs. In: IESD@ISWC (2016).
170. Russell, D.M., Stefik, M.J., Pirolli, P., Card, S.K.: The cost structure of sensemaking. Proc. SIGCHI Conf. Hum. factors Comput. Syst. - CHI '93. 269–276 (1993).

Appendix A

Exploratory User Study

A.1 Exploratory User Study Structure

A.1.1 Introduction about the Exploratory User Study

You are taking part in an experimental study using a system called **MusicPinta**. MusicPinta provides information about musical instruments, album recordings, artists, and reviews.

MusicPinta allows browsing through the following data sets:

- **MusicOntology** - provides main concepts and properties for describing music, albums, tracks, performances and arrangements.
- **MusicBrainz** - a community-maintained open source encyclopaedia of music information (e.g. artists and their releases and recordings).
- **Jamendo** - a community site for download music albums under creative commons licence.
- **Megatunes** – information about artists, tracks and album records.
- **Amazon reviews** – reviews for selected instruments.

You will be asked to **explore three musical instruments from different national cultures** using MusicPinta.

The study includes four steps:

Step 1. Pre-study questionnaire - 5 min

An outline of your profile and your familiarity in the music domain.

Step 2. Introduction to MusicPinta - 5 min

You will explore the information about an example instrument.

Step 3. Conduct your exploration - 30 min

You will be asked to explore three musical instruments from different national cultures using MusicPinta.

Step 4. Post-study Interview - 5 min

An overall impression about your exploration.

Your participation is anonymous; the data collected will be stored safely and used solely for research purposes.

Thank you very much for taking part in this study. As a small gesture of gratitude for your time after the completion of the study, you will receive a 10€ Amazon voucher.

A.1.2 Pre-study questionnaire

Are you a:

- Male Female

What is your age group?

- 18–24 25–34 35–44 45–54 55–64 Over 65

What is your nationality? _____

How often do you use the web for exploration (e.g. investigating new/unfamiliar topics that you want to increase your knowledge about **or** looking for places to visit **or** looking for items to buy).

- never
 daily - at least once a day
 weekly - at least once a week
 monthly - at least once a month
 occasionally – once in a while, not on a regular basis

Have you played any musical instrument(s)? Yes No

If yes, please list which one(s): _____

How often do you read online materials about music (e.g. instruments, events and performances)?

- never
 daily - at least once a day
 weekly - at least once a week
 monthly - at least once a month
 occasionally – once in a while, not on a regular basis

How often do you listen to music?

- never
 daily - at least once a day
 weekly - at least once a week
 monthly - at least once a month
 occasionally – once in a while, not on a regular basis

What is your familiarity level with String Musical Instruments?

- High (You have good knowledge and have played a string instrument(s)).
 Medium (You have some knowledge and have listened to string instruments).
 Low (You have limited knowledge and have seen some string instruments).
 Non of the above

What is your familiarity level with Wind Musical Instruments?

- High (You have good knowledge and have played a wind instrument(s)).
 Medium (You have some knowledge and have listened to wind instruments).
 Low (You have limited knowledge and have seen some wind instruments).
 Non of the above

What is your familiarity level with Percussion Musical Instruments?

- High (You have good knowledge and have played on a percussion instrument(s)).
 Medium (You have some knowledge and have listened to percussion instruments).
 Low (You have limited knowledge and have seen some percussion instruments).
 Non of the above
-

A.1.3 Introduction to MusicPinta (Exploration Example about 'Tabla')

To familiarise you with MusicPinta, you will explore the musical instrument '**Tabla**'

Perform the following steps: ** THINK OUT LOUD DURING YOUR EXPLORATION.

1. Login to MusicPinta as (user___) and your password is (musicpinta).
2. Click on MusicPinta's **Semantic Search** button (on top of the screen).
3. Enter the word '**Tabla**' in the search area, and click **Go** button (see figure 1).
4. **PLEASE WAIT** until you see the result page with information about 'Tabla' (see figure 2).

Figure 1: Semantic search area in MusicPinta

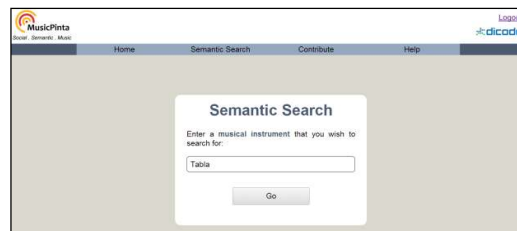


Figure 2: Result page of 'Tabla'



5. Notice the three section with information about 'Tabla':
 - a. **Description** of 'Tabla', extracted from MusicPinta's music knowledge resource.
 - b. **Features** of 'Tabla', provide categories from MusicPinta which 'Tabla' belongs to.
 - c. **Relevant information** to 'Tabla', provides terms and features from MusicPinta relevant to 'Tabla'.

**** We are NOT using the Reviews section in this study.**

6. Click on **Description**: See the page and read the description of 'Tabla' from Wikipedia.
7. Click on **Features**: See the page and read the categories that 'Tabla' belongs to. You may need to scroll down to see all links.
8. Click on **Relevant information**: See the page and read similar instrument that shares features with 'Tabla'. You may need to scroll down to see all links.

REMEMBER: At any point you can go back using the **BACK BUTTON** of your browser. Search tasks may take time. please be patient when you click on the selected links and wait until MusicPinta loads the corresponding information.

A.1.4 Conduct Your Exploration

Explore the musical instrument 'Oud' using the D-Strategy.

1. Pre-knowledge Test

- What comes in your mind when you hear the word 'Oud' ?
- What musical instrument categories does 'Oud' belong to?
- What musical instruments are similar to 'Oud'?

2. Instrument Exploration (e.g. explore the instrument 'Oud' using D-Strategy).

- Go to 'Semantic Search' in MusicPinta and type the word '**Oud**', and click 'Go'.
- Read all** content of 'Description', 'Features' and 'Relevant information'. List to the experimenter what you recognise.
- From features of '**Oud**', click on 'String Instruments', and repeat step (b).
- From relevant information of '**string instruments**' click on 'Guitar', and repeat step (b).
- From features of '**Guitar**' click on 'Plucked String Instruments', and repeat step (b).
- From 'Plucked String Instruments' click on 'Bouzouki', and repeat step (b).

3. Post-knowledge Test

- What comes in your mind when you hear the word 'Oud'?
- What musical instrument categories does 'Oud' belong to?
- What musical instruments are similar to 'Oud'?

Questions of exploration experience

- Which of the aspect(s) bellow relate to your exploration?

- Frustrating Interesting Enjoyable Boring Confusing Informative

- What unexpected and interesting things did you find about your journey?

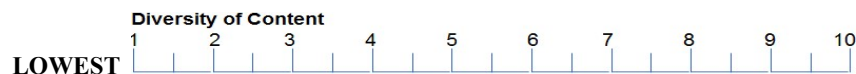
- Did the exploration expanded your knowledge?



HIGHEST

Justify your score: _____

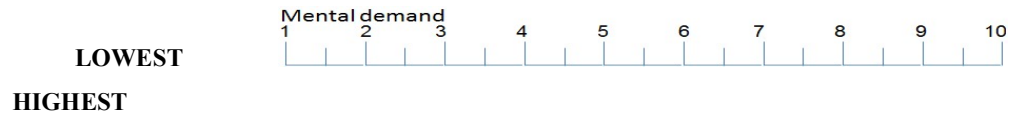
-How diverse was the content you have explored?



HIGHEST

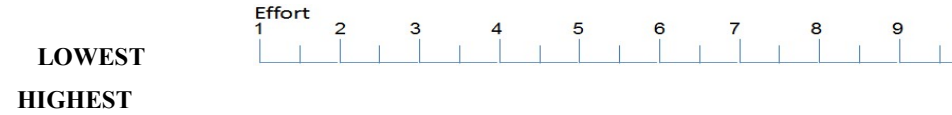
Justify your score: _____

- How mentally demanding was this exploration (e.g. how much mental and perceptual activity was required – thinking, deciding, calculating, remembering, looking, searching)?



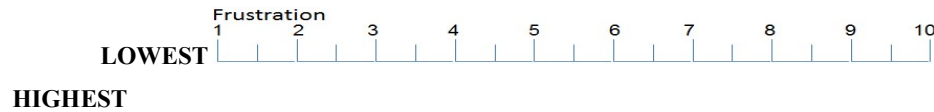
Justify your score: _____

- How hard did you have to work in this exploration (e.g. how much mentally and physically effort)?



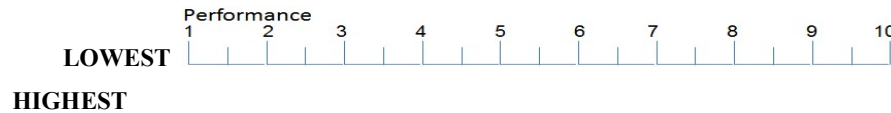
Justify your score: _____

- How discouraged, irritated, stressed and annoyed you were in this exploration?



Justify your score: _____

How successful do you think you were in this exploration (e.g. how confident you are you're your answers in the Post-exploration knowledge-questions?)



Justify your score: _____

A.1.5 Post-study Interview

Did you have any problems using MusicPinta for exploration?

- No problems
- Minor problems
- Major problems

Write any comments you have about this study

A.2 Participants' Information and Familiarity

User ID	Gender	Age	Nationality	Familiarity with Exploration	Instruments that plays on	How Much Read about Music	How Much Listen to Music	String Instrument Familiarity	Wind Instrument	Percussion Instrument
User 1	F	25-34	Jordanian	Weekly	NON	Occasionally	Weekly	Medium	NON	Medium
User 2	F	18-24	Polish	Weekly	Piano	Weekly	Daily	Low	Low	Low
User 3	M	25-34	Saudi	Daily	Guitar, Piano	Never	Weekly	Low	NON	Low
User 4	M	25-34	Saudi	Daily	Drum, Oud	Occasionally	Weekly	High	NON	High
User 5	F	18-24	Mexican	Weekly	NON	Occasionally	Weekly	Low	Low	Low
User 6	M	25-34	Malaysian	Daily	NON	Daily	Daily	Medium	Low	Low
User 7	M	25-34	Iranian	Daily	NON	Monthly	Weekly	Medium	Medium	Medium
User 8	M	25-34	Jordanian	Daily	Piano	Occasionally	Daily	Medium	Low	Low
User 9	M	35-44	Malaysian	Daily	Yes	Occasionally	Weekly	Medium	NON	Low
User 10	F	18-24	Chinese	Daily	Violin	Daily	Daily	Medium	Medium	Low
User 11	M	18-24	Indian	Daily	NON	Daily	Daily	Medium	Medium	Low
User 12	M	25-34	Greek	Daily	Piano, Accordion	Occasionally	Weekly	Medium	High	Medium

A.3 Knowledge Expansion Results

User ID	Exploration Strategy	Pre-Exploration			Post-Exploration			Recognition During Exploration				Post MINUS Pre EXPLORATION (Knowledge utility)			
		Remember	Categorise	Compare	Remember	Categorise	Compare	1st Click	2nd Click	3rd Click	4th Click	5th Click	Remember	Categorise	Compare
User 1	Dense	2	1	2	4	5	6	4	8	5	9	4	2	4	4
	Familiar	0	0	0	2	1	2	0	3	3	3	0	2	1	2
	Unfamiliar	0	0	0	2	1	1	0	3	0	0	1	2	1	1
User 2	Dense	0	0	0	3	2	3	1	32	1	17	6	3	2	3
	Familiar	0	0	0	3	1	1	1	25	3	2	2	3	1	1
	Unfamiliar	1	1	0	2	1	1	4	17	1	1	1	1	0	1
User 3	Dense	1	1	2	3	4	5	2	6	5	4	4	2	3	3
	Familiar	0	0	0	1	1	2	1	2	2	0	3	1	1	2
	Unfamiliar	0	0	0	1	1	1	2	2	3	3	1	1	1	1
User 4	Dense	2	1	1	6	3	5	5	10	4	6	3	4	2	4
	Familiar	0	0	0	2	1	0	0	3	2	1	2	2	1	0
	Unfamiliar	0	0	0	1	0	0	0	6	2	2	5	1	0	0
User 5	Dense	0	0	0	10	3	7	3	20	6	16	5	10	3	7
	Familiar	0	0	0	2	0	7	3	17	2	3	5	2	0	7
	Unfamiliar	2	1	2	7	1	6	2	16	1	10	3	5	0	4
User 6	Dense	0	0	0	5	1	7	3	27	7	18	6	5	1	7
	Familiar	0	0	0	1	1	1	3	11	2	3	2	1	1	1
	Unfamiliar	1	1	0	1	1	4	6	14	3	3	2	0	0	4
User 7	Dense	0	0	0	2	2	9	5	31	7	18	10	2	2	9
	Familiar	0	0	0	1	2	2	3	11	2	9	2	1	2	2
	Unfamiliar	1	1	1	1	1	5	2	12	3	3	3	0	0	4
User 8	Dense	1	1	3	3	2	5	3	17	5	9	6	2	1	2
	Familiar	0	0	0	1	1	3	5	5	2	6	2	1	1	3
	Unfamiliar	0	0	0	1	1	1	7	15	6	3	7	1	1	1
User 9	Dense	0	0	0	1	2	3	2	10	4	6	4	1	2	3
	Familiar	1	0	1	1	1	2	4	8	2	1	1	0	1	1
	Unfamiliar	0	0	0	0	1	3	4	7	1	1	2	0	1	3
User 10	Dense	0	0	0	1	2	5	3	27	6	10	6	1	2	5
	Familiar	0	0	0	1	2	1	5	6	3	6	2	1	2	1
	Unfamiliar	0	0	0	1	1	0	2	6	0	3	0	1	1	0
User 11	Dense	0	1	0	6	7	3	3	16	6	10	6	6	6	3
	Familiar	1	1	2	1	1	2	6	7	3	4	4	0	0	0
	Unfamiliar	0	0	0	3	3	2	2	5	0	1	1	3	3	2
User 12	Dense	1	1	2	1	3	4	3	9	3	6	3	0	2	2
	Familiar	0	0	0	1	1	1	1	7	1	4	2	1	1	1
	Unfamiliar	0	0	0	0	1	2	1	4	2	2	1	0	1	2

A.4 User Exploration Experience

User ID	Exploration Strategy	Exploration Experience	Knowledge Expansion	Content Diversity	Mentally Demand	Effort	Frustration	Performance
User 1	Dense	Interesting / Informative	7	6	3	3	3	7
	Familiar	Frustrating / Boring	3	4	3	7	7	3
	Unfamiliar	Frustrating / Boring	3	4	6	7	7	3
User 2	Dense	Interesting / Informative	7	8	2	2	3	7
	Familiar	Informative	8	10	2	1	2	6
	Unfamiliar	Informative	5	8	2	1	2	8
User 3	Dense	Interesting / Enjoyable/ Informative	7	5	8	2	1	10
	Familiar	Interesting / Enjoyable/ Informative	7	5	3	1	1	10
	Unfamiliar	Interesting / Confusing	3	2	9	5	5	8
User 4	Dense	Interesting / Enjoyable/ Informative	9	7	7	5	2	8
	Familiar	Informative	8	5	6	5	5	5
	Unfamiliar	Confusing	5	5	4	7	8	4
User 5	Dense	Interesting / Informative	7	8	3	3	2	8
	Familiar	Interesting / Informative	7	8	4	4	3	8
	Unfamiliar	Interesting / Enjoyable/ Informative	7	6	7	4	3	6
User 6	Dense	Enjoyable/ Informative	8	5	5	6	2	7
	Familiar	Interesting/ Informative	5	8	5	5	5	7
	Unfamiliar	Frustrating	6	8	5	6	8	6
User 7	Dense	Interesting / Enjoyable/ Informative	10	10	2	1	1	10
	Familiar	Interesting / Enjoyable/ Informative	10	8	3	2	3	9
	Unfamiliar	Interesting / Enjoyable/ Informative	8	10	1	1	3	10
User 8	Dense	Interesting / Informative	9	7	8	3	4	7
	Familiar	Boring	8	8	5	3	4	8
	Unfamiliar	Confusing	6	8	2	2	5	4
User 9	Dense	Interesting / Confusing Informative	6	6	7	7	8	6
	Familiar	Interesting / Enjoyable Informative	6	6	8	8	8	6
	Unfamiliar	Frustrating/ Confusing / Informative	8	7	6	7	7	5
User 10	Dense	Interesting / Enjoyable/ Informative	8	7	7	3	2	7
	Familiar	Enjoyable/ Informative	8	7	7	5	3	8
	Unfamiliar	Interesting / Informative	7	5	4	3	3	5
User 11	Dense	Enjoyable/ Informative	8	5	5	6	2	7
	Familiar	Interesting/ Informative	5	8	5	5	5	7
	Unfamiliar	Frustrating	6	8	5	6	8	6
User 12	Dense	Interesting / Enjoyable/ Informative	8	9	3	2	1	8
	Familiar	Interesting / Enjoyable/ Informative	7	8	4	4	3	7
	Unfamiliar	Frustrating/ Confusing	5	7	6	5	8	7

Transverse flute	0.484	0.000	0.000	0.000	0.280	0.634	0.389	0.000	0.055	0.000	0.081	0.000	0.000	1.000
Trombone	0.243	0.000	0.000	0.000	0.257	0.671	0.288	0.000	0.036	0.000	0.050	0.000	0.000	0.775
Trumpet	0.069	0.000	0.000	0.000	0.017	0.041	0.047	0.000	0.041	0.000	0.078	0.000	0.000	1.000
Tuba	0.430	0.000	0.000	0.531	0.367	0.684	0.499	0.000	0.117	0.000	0.105	0.000	0.000	0.531
Tuned percussion	0.308	0.000	0.000	0.400	0.294	0.634	0.414	0.000	0.096	0.000	0.084	0.000	0.288	0.400
Valved brass instruments	0.111	0.000	0.000	0.111	0.192	0.139	0.131	0.000	0.032	0.000	0.034	0.000	0.200	0.111
Violins	0.000	0.000	0.000	0.000	0.189	0.000	0.577	0.000	0.064	0.000	0.088	0.000	0.866	1.000
Washboard	0.000	0.000	0.000	0.000	0.120	0.000	0.333	0.000	0.039	0.000	0.057	0.000	0.750	1.000
Wind instruments	0.000	0.000	0.000	0.000	0.171	0.000	0.857	0.000	0.303	0.000	0.299	0.000	0.429	0.857
Wood block	0.129	0.012	0.000	0.000	0.052	0.058	0.103	0.000	0.502	0.000	0.646	0.000	0.034	0.612
Woodwind	0.081	0.007	0.000	0.000	0.019	0.071	0.133	0.000	0.536	0.000	0.702	0.046	0.016	1.000
Xalam (khalam)	0.015	0.000	0.000	0.000	0.007	0.021	0.043	0.000	1.000	0.000	0.897	0.003	0.003	0.096
Xylophone	0.118	0.000	0.000	0.090	0.431	0.156	0.188	0.000	0.385	0.000	0.344	0.170	0.302	0.076
Yangqin	0.594	0.000	0.000	0.255	3.426	1.002	0.814	0.000	7.253	0.000	5.122	0.403	1.639	0.197
Zurna	0.079	0.000	0.000	0.034	0.455	0.133	0.108	0.000	0.964	0.000	0.681	0.054	0.218	0.026
	0.196	0.000	0.000	0.000	0.199	0.107	0.224	0.000	1.000	0.000	0.715	0.018	0.071	0.089
	0.047	0.000	0.000	0.000	0.198	0.071	0.169	0.000	1.000	0.000	0.642	0.024	0.061	0.092
	0.003	0.000	0.000	0.000	0.071	0.010	0.028	0.000	0.755	0.000	0.376	0.002	0.009	0.007

Appendix C

KA_{DG} Metrics Output of Knowledge Anchors

C.1 Precision, Recall and F1 values of KA_{DG} Metrics in MusicPinta

Precision, recall and F1 values by comparing 60th, 70th, 80th and 90th percentiles of the KA_{DG} metrics normalised output values over hierarchical (*H*) and domain-specific (*D*) relationships, with BLO derived in MusicPinta.

Percentile	Relationship types	Precision				Recall				F1			
		AV	CAC	CU	Jac	AV	CAC	CU	Jac	AV	CAC	CU	Jac
60 th	<i>H</i>	0.31	0.32	0.34	0.37	0.66	0.71	0.67	0.54	0.42	0.44	0.46	0.44
	<i>D</i>	0.31	0.29	0.31	0.32	0.54	0.50	0.54	0.36	0.39	0.36	0.39	0.35
70 th	<i>H</i>	0.32	0.26	0.35	0.37	0.50	0.45	0.58	0.54	0.39	0.33	0.44	0.44
	<i>D</i>	0.32	0.26	0.29	0.32	0.42	0.33	0.38	0.38	0.36	0.29	0.33	0.25
80 th	<i>H</i>	0.33	0.26	0.35	0.37	0.54	0.33	0.38	0.50	0.41	0.29	0.36	0.43
	<i>D</i>	0.29	0.19	0.19	0.29	0.25	0.16	0.16	0.25	0.26	0.17	0.17	0.26
90 th	<i>H</i>	0.33	0.27	0.20	0.41	0.16	0.21	0.13	0.29	0.22	0.24	0.15	0.34
	<i>D</i>	0.20	0.10	0.10	0.30	0.08	0.04	0.04	0.13	0.12	0.06	0.09	0.18

C.2 KA_{DG} Identified in MusicPinta

Metrics output of KA_{DG} identified over the hierarchical (H) and domain-specific (D) relationships in the MusicPinta data graph using the 60th percentile as a cut-off threshold point. Each musical instrument name with a value of 1 means that it has been identified as KA_{DG} by its corresponding metric.

Musical Instrument Name	AV(H)	CAC(H)	CU(H)	Jaccard(H)	AV(D)	CAC(D)	CU(D)	Jaccard(D)
Accordion	1	1	1	1	0	0	0	0
Acoustic guitar	0	0	0	0	1	1	0	0
Acoustic upright bass	0	0	0	0	0	0	0	0
Archaic and other bowed string-instruments	1	1	1	1	0	0	0	0
Baltic Psalteries	0	0	0	0	0	0	0	0
Bamboo Angklung	0	0	0	0	0	0	0	0
Banjo	0	0	0	0	0	0	0	0
Bass	1	1	1	1	1	1	1	0
Bass guitar	0	0	0	0	1	1	1	1
Basset horn	0	0	0	0	0	0	0	0
Bell	1	1	1	1	0	0	0	0
Bouzouki	1	1	1	1	1	1	1	1
Bowed string instruments	1	1	1	1	1	1	1	1
Brass	1	1	1	1	1	1	1	1
Button accordion	0	0	0	0	0	0	0	0
Castanets	1	1	1	1	0	0	0	0
Cello	0	0	0	0	0	0	0	0
Clarinet	1	1	1	1	1	1	1	1
Classical guitar	1	1	1	0	1	1	1	1
Clavichord	0	0	0	0	0	0	0	0
Concert harp	0	0	0	0	0	0	0	0
Contrabass recorder	0	0	0	0	0	0	0	0
Double reed	1	1	1	1	1	1	1	1
Drums	1	1	1	1	1	1	1	0
Drumset	0	0	0	0	0	0	0	0
Electric lap steel guitar	1	1	0	0	1	1	1	0
Electric piano	0	0	0	0	1	1	1	0
Electronic instruments	1	1	1	0	1	1	1	1
Erhu (Chinese Violin)	0	0	0	0	0	0	0	0
Fiddle	0	0	0	0	0	0	0	0
Fipple flutes	1	1	1	1	1	1	1	1
Flute	1	1	1	1	1	1	1	1
Folk harp	0	0	0	0	0	0	0	0
Frame drum	1	1	1	1	0	0	0	0
Gamelan	1	1	0	0	0	0	0	0

Gehu	0	0	0	0	0	0	0	0
Glockenspiel	0	0	0	0	0	0	0	0
Goblet drum	1	1	1	0	0	0	0	0
Gong	1	1	1	0	0	0	0	0
Gudok	0	0	0	0	0	0	0	0
Guitar	1	1	1	1	1	1	1	1
Hammered dulcimer	1	1	1	0	0	0	0	0
Harmonica	0	0	0	0	0	0	0	0
Harp	1	1	1	0	0	0	1	0
Huqin	1	1	1	1	0	0	0	0
Indian Bamboo Flutes	1	1	1	0	0	0	0	0
Keyed brass instruments	0	0	0	0	0	0	0	0
Koto	0	0	0	0	0	0	0	0
Lap steel guitar	1	1	1	0	1	1	1	0
Lute	1	1	1	1	1	1	1	1
Lyre	1	1	1	1	1	1	1	0
Mandola	0	0	0	0	1	1	1	0
Moon lute	0	0	0	0	0	0	0	0
Musical bow	0	0	0	0	0	0	0	0
Natural brass instruments	1	1	0	0	0	0	0	0
Oboe	0	0	0	0	1	1	1	0
Organ	1	1	1	0	1	1	0	1
Other flutes	1	1	1	0	0	0	1	0
Other instruments	0	0	0	0	0	0	1	0
Other string instruments	0	0	0	0	0	0	0	0
Oud	0	0	0	0	0	0	0	0
Pan flute	0	0	0	0	0	0	0	0
Percussion instruments	1	1	1	1	1	1	1	1
Pipa	0	0	0	0	0	0	0	0
Plucked string instruments	1	1	1	1	1	1	1	1
Psaltery	0	0	0	0	0	0	0	0
Rattle	0	0	1	0	0	0	0	0
Rebab	0	0	0	0	0	0	0	0
Recorder	0	1	1	0	0	0	0	0
Reed organ	1	1	0	0	0	1	0	0
Reeds	1	1	1	1	1	1	1	1
Resonator guitar	0	0	0	0	0	0	0	0
Sanh Tien (Coin-dappers)	0	0	0	0	0	0	0	0
Sanshin	0	0	0	0	0	0	0	0
Sanxin	0	0	0	0	0	0	0	0
Saxophone	0	0	0	0	1	1	1	1
Shawm	1	1	1	0	1	1	1	1
Sheng	0	0	0	0	0	0	0	0
Singular reed	1	1	1	1	1	1	1	1
Sitar	0	0	0	0	0	0	0	0
Slide brass instruments	1	1	1	0	1	1	0	0
Steel guitar	1	1	1	1	1	1	1	0

String instruments	1	1	1	1	1	1	1	1
Struck string instruments	1	1	1	1	1	1	1	1
Table steel guitar	0	0	0	0	0	0	0	0
Tambourine	0	0	0	0	0	0	0	0
Tambura	1	1	1	1	1	1	1	0
Tibetan water drum	0	0	0	0	0	0	0	0
Tin whistle	0	0	0	0	0	0	0	0
Tiple	0	0	0	0	0	0	0	0
Transverse flute	1	1	1	1	1	1	1	1
Trombone	0	0	0	0	0	0	0	0
Trumpet	0	0	0	0	0	0	0	0
Tuba	1	1	1	1	0	0	0	0
Tuned percussion	1	1	1	1	1	1	1	1
Valved brass instruments	1	1	1	1	1	1	1	1
Violin	1	1	1	1	1	1	1	1
Washboard	0	0	0	0	0	0	0	0
Wind instruments	1	1	1	1	1	1	1	1
Wood block	0	0	0	0	0	0	0	0
Woodwind	1	1	1	1	1	1	1	1
Xalam (khalam)	1	1	1	0	1	1	1	0
Xylophone	1	1	1	1	1	0	1	0
Yangqin	0	0	0	0	0	0	0	0
Zurna	1	1	1	1	1	1	1	1

C.3 Precision, Recall and F1 values of KA_{DG} Metrics in Occupation and Subject Class Hierarchies in L4All

Precision, recall and F1 values for comparing the 60th, 70th, 80th and 90th percentiles of the KA_{DG} metrics normalised output values over the hierarchical (H) and domain-specific (D) relationships with human BLO_{DG} derived, in the Occupation and Subject class hierarchy in L4All data graph.

Class Hierarchy	Percentile	Relationship type	Precision				Recall				F1			
			AV	CAC	CU	Jac	AV	CAC	CU	Jac	AV	CAC	CU	Jac
Occupation Class Hierarchy	60 th	H	0.59	0.59	0.74	0.74	0.92	0.92	0.45	0.45	0.72	0.72	0.56	0.56
		D	0.77	0.76	0.00	0.00	0.95	0.92	0.00	0.00	0.85	0.83	0.00	0.00
	70 th	H	0.55	0.53	0.74	0.74	0.68	0.66	0.45	0.45	0.61	0.59	0.56	0.56
		D	0.82	0.76	0.00	0.00	0.74	0.68	0.00	0.00	0.77	0.72	0.00	0.00
	80 th	H	0.61	0.53	0.74	0.74	0.53	0.50	0.45	0.45	0.65	0.51	0.56	0.56
		D	0.82	0.78	0.00	0.00	0.50	0.47	0.00	0.00	0.62	0.59	0.00	0.00
	90 th	H	0.75	0.41	0.69	0.67	0.32	0.34	0.24	0.32	0.44	0.30	0.36	0.43
		D	0.82	0.73	0.00	0.00	0.34	0.21	0.00	0.00	0.37	0.33	0.00	0.00
Subject Class Hierarchy	60 th	H	1	0.89	0.00	0.00	0.56	0.50	0.00	0.00	0.72	0.64	0.00	0.00
		D	1	1.00	0.00	0.00	0.56	0.56	0.00	0.00	0.72	0.72	0.00	0.00
	70 th	H	1	1.00	0.00	0.00	0.38	0.38	0.00	0.00	0.55	0.55	0.00	0.00
		D	1	1.00	0.00	0.00	0.38	0.38	0.00	0.00	0.55	0.55	0.00	0.00
	80 th	H	1	1.00	0.00	0.00	0.25	0.25	0.00	0.00	0.40	0.40	0.00	0.00
		D	1	1.00	0.00	0.00	0.25	0.25	0.00	0.00	0.40	0.40	0.00	0.00
	90 th	H	1	1.00	0.00	0.00	0.13	0.13	0.00	0.00	0.22	0.22	0.00	0.00
		D	1	1.00	0.00	0.00	0.13	0.13	0.00	0.00	0.22	0.22	0.00	0.00

C.4 KA_{DG} Identified in Occupation Class Hierarchy in L4All

Metrics output of KA_{DG} identified over the hierarchical (*H*) and domain-specific (*D*) relationships in the Occupation class hierarchy using the 60th percentile as a cut-off threshold point. Each Occupation name with a value of 1 means that it has been identified as KA_{DG} by its corresponding metric.

Occupation Name	AV(H)	CAC(H)	CU(H)	Jaccard(H)	AV(D)	CAC(D)	CU(D)	Jaccard(D)
Administrative and Secretarial Occupations	1	1	1	1	1	1	0	0
Administrative Occupations	1	1	1	1	1	1	0	0
Administrative Occupations: Communications	0	0	0	0	0	0	0	0
Administrative Occupations: Finance	0	0	0	0	0	0	0	0
Administrative Occupations: General	1	1	0	0	1	1	0	0
Administrative Occupations: Government and Related Organisations	1	1	0	0	1	1	0	0
Administrative Occupations: Records	1	1	0	0	1	1	0	0
Agricultural Trades	0	0	0	0	0	0	0	0
Animal Care Services	0	0	0	0	0	0	0	0
Architects, Town Planners and Surveyors	0	0	0	0	0	0	0	0
Artistic and Literary Occupations	0	0	0	0	0	0	0	0
Assemblers and Routine Operatives	1	1	0	0	1	0	0	0
Associate Professional and Technical Occupations	1	1	1	1	1	1	0	0
Building Trades	0	0	0	0	0	0	0	0
Business and Finance Associate Professionals	1	1	0	0	1	1	0	0
Business and Public Service Associate Professionals	1	1	1	1	1	1	0	0
Business and Public Service Professionals	1	1	1	1	1	1	0	0
Business and Statistical Professionals	0	0	0	0	0	0	0	0
Caring Personal Service Occupations	1	1	1	1	1	1	0	0
Childcare and Related Personal Services	0	0	0	0	0	0	0	0
Conservation Associate Professionals	0	0	0	0	0	0	0	0
Construction Operatives	0	0	0	0	1	1	0	0
Construction Trades	0	0	0	0	0	0	0	0
Corporate Managers	1	1	1	1	1	1	0	0
Corporate Managers and Senior Officials	0	0	0	0	0	0	0	0
Culture, Media and Sports Occupations	1	1	1	1	1	1	0	0
Customer Service Occupations	1	1	1	1	1	1	0	0
Customer Service Occupations and Related	1	1	0	0	1	1	0	0
Design Associate Professionals	1	1	0	0	1	1	0	0
Draughtspersons and Building Inspectors	1	1	0	0	1	1	0	0
Electrical Trades	0	0	0	0	0	0	0	0
Elementary Administration and Service Occupations	1	1	0	0	0	0	0	0

Elementary Administration Occupations	0	0	0	0	0	0	0	0
Elementary Agricultural Occupations	0	0	0	0	0	0	0	0
Elementary Cleaning Occupations	0	0	0	0	0	0	0	0
Elementary Construction Occupations	0	0	0	0	0	0	0	0
Elementary Goods Storage Occupations	0	0	0	0	0	0	0	0
Elementary Occupations	1	1	1	1	0	0	0	0
Elementary Personal Services Occupations	0	0	0	0	0	0	0	0
Elementary Process Plant Occupations	0	0	0	0	0	0	0	0
Elementary Sales Occupations	0	0	0	0	0	0	0	0
Elementary Security Occupations	0	0	0	0	0	0	0	0
Elementary Trades, Plant and Storage Related Occupations	1	1	0	0	0	0	0	0
Engineering Professionals	1	1	0	0	1	1	0	0
Financial Institution and Office Managers	0	0	0	0	0	0	0	0
Food Preparation Trades	0	0	0	0	0	0	0	0
Functional Managers	1	1	0	0	1	1	0	0
Hairdressers and Related Occupations	0	0	0	0	0	0	0	0
Health and Social Services Managers	0	0	0	0	0	0	0	0
Health and Social Welfare Associate Professionals	1	1	0	0	0	0	0	0
Health Associate Professionals	0	0	0	0	0	0	0	0
Health Professionals	1	1	0	0	0	0	0	0
Healthcare and Related Personal Services	1	1	0	0	1	1	0	0
Healthcare Professionals	0	0	0	0	0	0	0	0
Housekeeping Occupations	0	0	0	0	0	0	0	0
Information and Communication Technology Professionals	1	1	0	0	1	1	0	0
IT Service Delivery Occupations	1	1	0	0	1	1	0	0
Legal Associate Professionals	0	0	0	0	0	0	0	0
Legal Professionals	0	0	0	0	0	0	0	0
Leisure and Other Personal Service Occupations	1	1	0	0	0	0	0	0
Leisure and Travel Service Occupations	0	0	0	0	0	0	0	0
Librarians and Related Professionals	1	1	0	0	1	1	0	0
Managers and Proprietors in Agriculture and Services	1	1	0	0	0	0	0	0
Managers and Proprietors in Hospitality and Leisure Services	0	0	0	0	0	0	0	0
Managers and Proprietors in other service industries	0	0	0	0	0	0	0	0
Managers and Senior Officials	1	1	1	1	1	1	0	0
Managers in Distribution, Storage and Retailing	0	0	0	0	0	0	0	0
Managers in Farming, Horticulture, Forestry and Fishing	0	0	0	0	0	0	0	0
Media Associate Professionals	0	0	0	0	0	0	0	0
Metal Forming, Welding and Related Trades	0	0	0	0	0	0	0	0
Metal Machining, Fitting and Instrument Making Trades	0	0	0	0	0	0	0	0
Mobile Machine Drivers and Operatives	1	1	0	0	1	1	0	0
Personal Service Occupations	1	1	1	1	1	1	0	0

Personal Services Occupations N.E.C.	0	0	0	0	0	0	0	0
Plant and Machine Operatives	1	1	0	0	1	1	0	0
Printing Trades	0	0	0	0	0	0	0	0
Process, Plant and Machine Operatives	1	1	1	1	1	1	0	0
Process, Plant and Machine Operatives and Related	1	1	1	1	1	1	0	0
Process Operatives	1	1	0	0	1	1	0	0
Production Managers	0	0	0	0	0	0	0	0
Professional Occupations	1	1	1	1	1	1	0	0
Protective Service Associate Occupations	0	0	0	0	0	0	0	0
Protective Service Occupations	1	1	0	0	0	0	0	0
Protective Service Officers	0	0	0	0	0	0	0	0
Public Service and Other Associate Professionals	0	0	0	0	0	0	0	0
Public Service Professionals	0	0	0	0	0	0	0	0
Quality and Customer Care Managers	0	0	0	0	0	0	0	0
Research Professionals	1	1	0	0	1	1	0	0
Sales and Customer Service Occupations	1	1	1	1	1	1	0	0
Sales and Related Associate Professionals	0	0	0	0	0	0	0	0
Sales Assistants and Retail Cashiers	1	1	0	0	1	1	0	0
Sales Occupations	1	1	1	1	1	1	0	0
Sales Related Occupations	1	1	0	0	1	1	0	0
Science and Engineering Technicians	1	1	0	0	1	1	0	0
Science and Technology Associate Professionals	1	1	1	1	1	1	0	0
Science and Technology Professionals	1	1	1	1	1	1	0	0
Science Professionals	1	1	0	0	1	1	0	0
Secretarial and Related Occupations	1	1	1	1	1	1	0	0
Secretaries and Related	1	1	0	0	1	1	0	0
Skilled Agricultural Trades	1	1	0	0	0	0	0	0
Skilled Construction and Building Trades	1	1	0	0	0	0	0	0
Skilled Metal and Electrical Trades	1	1	0	0	0	0	0	0
Skilled Trades N.E.C.	0	0	0	0	0	0	0	0
Skilled Trades Occupations	1	1	1	1	0	0	0	0
Social Welfare Associate Professionals	0	0	0	0	0	0	0	0
Sports and Fitness Occupations	0	0	0	0	0	0	0	0
Teaching and Research Professionals	1	1	1	1	1	1	0	0
Teaching Professionals	1	1	0	0	1	1	0	0
Textiles, Printing and Other Skilled Trades	1	1	0	0	0	0	0	0
Textiles and Garments Trades	0	0	0	0	0	0	0	0
Therapists	0	0	0	0	0	0	0	0
Transport and Mobile Machine Drivers and Operatives	1	1	1	1	1	1	0	0
Transport Associate Professionals	0	0	0	0	0	0	0	0
Transport Drivers and Operatives	1	1	0	0	1	1	0	0
Vehicle Trades	0	0	0	0	0	0	0	0

Appendix D

User Study to Evaluate the Subsumption Algorithm

D.1 Survey for Identifying User Familiarity with Entities in MusciPinta

Could you please indicate how familiar you are in the following musical instruments?

Please select one of the following options:

- High (You have good knowledge and have played on the instrument).
- Medium (You have some knowledge and have listened to the instrument).
- Low (You have limited knowledge and have seen the instrument).
- None of the above.

Musical Instrument Name	Please ✓ the corresponding box			
	High	Medium	Low	None
Nai				
Syrinx				
Indian Bamboo Flutes				
Hardingfele				
Contrabass clarinet				
Pedal steel guitar				
Subcontrabass recorder				
Harpsichord				
Table steel guitar				
Heckelphone				
Alto saxophone				
Diatonic accordion / Melodeon				
Tenor recorder				
Concert flute				
Shawm				
Basset clarinet				
Bandura				
Dan ty ba				
Low whistle				
Great bass recorder / C-bass recorder				
Alto recorder / Treble recorder				
Electric sitar				
Zhongruan				
Oboe				
Bass flute				
Tenor saxophone				

Musical Instrument Name	Please ✓ the corresponding box			
	High	Medium	High	None
Contrabass recorder				
Alto flute				
Biwa				
Electric lap steel guitar				
Acoustic guitar				
Kemenche				
Sanshin				
Soprano flute				
Sarod				
Banjo				
Venu				
Treble flute				
Bass recorder / F-bass recorder				
Mandolin				
Balalaika				
Sopranino recorder				
Cumbus				
Bansuri				
English horn				
Lap steel guitar				
German harp				
Irish harp				
Baritone saxophone				
Mandola				
Tamburitza				
Fute damour				
Alto clarinet				
Banjitar				
Basset horn				
Piccolo				
Garklein recorder				
Electric harp				
Soprano saxophone				
Bass clarinet				
Soprano clarinet				

D.2 Participants' Familiarity

User ID	Gender	Age	Nationality	String Instrument Familiarity	Wind Instrument Familiarity
User 1	F	25-34	Libyan	Medium	Low
User 2	F	35-44	Romanian	Low	Low
User 3	M	25-34	British	Medium	High
User 4	M	35-44	Nigerian	Low	Low
User 5	M	18-24	British	Medium	Medium
User 6	M	35-44	Jordanian	Low	Low
User 7	F	25-34	British	Low	Low
User 8	M	35-44	Greek	Medium	Medium
User 9	M	25-34	Austria	Low	Low
User 10	M	35-44	Jordanian	Low	Low
User 11	M	18-24	Nigerian	Medium	Medium
User 12	F	18-24	British	Low	Low
User 13	M	18-24	British	Low	Low
User 14	M	35-44	Saudi Arabia	Low	Low
User 15	F	25-34	Malaysian	Low	Low
User 16	F	25-34	Jordanian	Low	Low
User 17	F	25-34	British	Low	Low
User 18	M	25-34	British	Low	Low
User 19	M	18-24	Nigerian	Medium	Medium
User 20	M	25-34	Nigerian	Medium	Low
User 21	M	25-34	Nigerian	Low	Low
User 22	M	25-34	Italian	Medium	Low
User 23	F	25-34	China	Low	Low
User 24	M	25-34	British	Medium	Low
User 25	F	25-34	Greek	Low	Low
User 26	M	25-34	Jordanian	Low	Low
User 27	M	25-34	Jordanian	Low	Low
User 28	F	25-34	Polish	Medium	Medium
User 29	M	35-44	Nigerian	Low	Low
User 30	M	35-44	Malaysian	Low	Low
User 31	M	25-34	Greek	Medium	Low
User 32	M	25-34	British	Medium	Medium

D.3 Examples of Transition Narratives of Exploration Paths for Biwa and Bansuri

Exploration Path for Biwa

Read all the following content of 'Description', 'Features' and 'Relevant Information' of 'Biwa'

The screenshot shows the MusicPinta website interface for the 'Biwa' page. At the top, there is a navigation bar with 'Home', 'Semantic Search', 'Contribute', and 'Help'. Below this, the page title 'Biwa' is displayed. The main content area is divided into sections: 'Description', 'Features', 'Relevant Information', 'Reviews', and 'Link History'. The 'Description' section contains a text box with the following text: 'The biwa is a Japanese short-necked fretted lute, often used in narrative storytelling. The biwa is the chosen instrument of Benten, goddess of music, eloquence, poetry, and education in Japanese Shinto. It arrived in Japan in two forms. Since that time, the number of biwa has more than quadrupled. Guilds supporting biwa players, particularly the biwa hoshi, helped proliferate biwa musical development for hundreds of years.' Below the description, there is a list of features: 'Instrument'. The 'Relevant Information' section shows 'Biwa belongs to:' followed by a list of related terms: 'Bouzouki', 'Instrument', 'Japanese musical instruments', 'Lute', 'Necked bowl lutes', 'Oud', 'Plucked string instruments', and 'String instruments'. The 'Reviews' and 'Link History' sections are currently empty.

You may find it useful to know that 'Biwa' belongs to a familiar and well-known instrument called 'Lute'. Let's explore 'Lute'

Read all the following content of 'Description', 'Features' and 'Relevant information' of 'Lute'

The screenshot shows the MusicPinta website interface for the 'Lute' page. At the top, there is a navigation bar with 'Home', 'Semantic Search', 'Contribute', and 'Help'. Below this, the page title 'Lute' is displayed. The main content area is divided into sections: 'Description', 'Features', 'Relevant Information', 'Reviews', and 'Link History'. The 'Description' section contains a text box with the following text: 'Lute can refer generally to any plucked string instrument with a neck (either fretted or unfretted) and a deep round back, or more specifically to an instrument from the family of European lutes. The European lute and the modern Near-Eastern oud both descend from a common ancestor via diverging evolutionary paths.' Below the description, there is a list of features: 'Instrument'. The 'Relevant Information' section shows 'Lute belongs to:' followed by a list of related terms: 'Arabic words and phrases', 'Baroque instruments', 'Bouzouki', 'Composite chordophones', 'Early musical instruments', 'Instrument', 'Lutes', 'Necked bowl lutes', 'Plucked string instruments', and 'String instruments'. The 'Reviews' and 'Link History' sections are currently empty.

You may also find it useful to know that 'Oud' belongs to 'Lute', and 'Biwa' belongs to 'Oud'. Let's explore 'Oud'.

Read all the following content of 'Description', 'Features' and 'Relevant information' of 'Oud'

The screenshot shows the MusicPinta interface for the 'Oud' entry. At the top, there is a navigation bar with 'Home', 'Semantic Search', 'Contribute', and 'Help'. The page title is 'Oud'. Below the title, there are tabs for 'Description', 'Features', 'Relevant Information', 'Reviews', and 'Link History'. The 'Description' tab is active, showing a placeholder image (a camera icon with a red 'X') and the text: 'The Oud is a pear-shaped stringed instrument commonly used in North African and Middle Eastern music. The modern oud and the European lute both descend from a common ancestor via diverging paths. The oud is readily distinguished by its lack of frets and smaller neck.' Below this, there are sections for 'Oud is:' (Instrument), 'Oud belongs to:' (Arabic musical instruments, Bouzouki, Continuous pitch instruments, Early musical instruments, Greek musical instruments, Instrument, Lute, Necked bowl lutes, Plucked string instruments, String instruments, Turkish musical instruments), 'The following are Oud:' (No items found), and 'The following share features with Oud:' (Biwa, Cumbus).

You may also find it useful to know that 'Tambura' belongs to 'Lute'. Let's explore 'Tambura'.


Read all the following content of 'Description', 'Features' and 'Relevant information' of 'Tambura'

The screenshot shows the MusicPinta interface for the 'Tambura' entry. At the top, there is a navigation bar with 'Home', 'Semantic Search', 'Contribute', and 'Help'. The page title is 'Tambura'. Below the title, there are tabs for 'Description', 'Features', 'Relevant Information', 'Reviews', and 'Link History'. The 'Description' tab is active, showing a placeholder image (a camera icon with a red 'X') and the text: 'No description available.' Below this, there are sections for 'Tambura is:' (Instrument), 'Tambura belongs to:' (Bouzouki, Instrument, Lute, Plucked string instruments, String instruments), 'The following are Tambura:' (No items found), and 'The following share features with Tambura:' (Balalaika, Bandura, Mandola, Mandolin, Tamburitza).

You may also find it useful to know that '**Pipa**' belongs to '**Lute**'.
Let's explore '**Pipa**'.

Read all the following content of '**Description**', '**Features**' and '**Relevant information**' of '**Pipa**'

 **MusicPinta**
Social · Semantic · Music

[Logout](#)


[Home](#) [Semantic Search](#) [Contribute](#) [Help](#)

[Home](#) > [Semantic Search](#) > [Pipa](#)

Pipa

[Description](#) [Features](#) [Relevant Information](#) [Reviews](#) [Link History](#)

Description is extracted from Wikipedia when available.



The pipa is a four-stringed Chinese musical instrument, belonging to the plucked category of instruments. Sometimes called the Chinese lute, the instrument has a pear-shaped wooden body with a varying number of frets ranging from 12–26. Another Chinese four-string plucked lute is the liuqin, which looks like a smaller version of the pipa. The pipa is one of the most popular Chinese instruments and has been played for almost two thousand years in China.

[Description](#) [Features](#) [Relevant Information](#) [Reviews](#) [Link History](#)

Pipa is:

[Instrument](#)

Pipa belongs to:

[Bouzouki](#) [Chinese musical instruments](#) [Instrument](#) [Lute](#) [Necked bowl lutes](#) [Plucked string instruments](#) [String instruments](#)

[Description](#) [Features](#) [Relevant Information](#) [Reviews](#) [Link History](#)

The following are **Pipa**:

No items found.

The following share features with **Pipa**:

[Dan ty ba](#)

Exploration Path for Bansuri

Read all the following content of 'Description', 'Features' and 'Relevant information' of 'Bansuri'

The screenshot shows the MusicPinta website interface for the 'Bansuri' page. At the top, there is a navigation bar with 'Home', 'Semantic Search', 'Contribute', and 'Help'. Below this, the page title is 'Bansuri'. The main content area has tabs for 'Description', 'Features', 'Relevant Information', 'Reviews', and 'Link History'. The 'Description' tab is active, showing a text box with the text: 'Description is extracted from Wikipedia when available.' Below this is a photograph of a bansuri flute. To the right of the photo is a text block: 'The bansuri is a transverse flute of India made from a single hollow shaft of bamboo with six or seven finger holes. An ancient musical instrument associated with cowherds and the pastoral tradition, it is intimately linked to the love story of Krishna and Radha, and is depicted in Buddhist paintings from around 100 AD.' Below the description, there is a section 'Bansuri is:' with a link to 'Instrument'. Another section 'Bansuri belongs to:' lists various categories: 'Bamboo musical instruments', 'Bangladeshi musical instruments', 'Bansuri players', 'Fipple flutes', 'Flute', 'Hindustani musical instruments', 'Indian Bamboo Flutes', 'Indian musical instruments', 'Instrument', 'Nepalese musical instruments', 'Pakistani musical instruments', 'Side-blown flutes', 'Transverse flute', 'Wind instruments', and 'Woodwind'. At the bottom, there is a section 'The following are Bansuri:' with the text 'No items found.'

You may find it useful to know that 'Bansuri' belongs to a familiar and well-known instrument called 'Flute'. Let's explore 'Flute'

Read all the following content of 'Description', 'Features' and 'Relevant information' of 'Flute'.

The screenshot shows the MusicPinta website interface for the 'Flute' page. At the top, there is a navigation bar with 'Home', 'Semantic Search', 'Contribute', and 'Help'. Below this, the page title is 'Flute'. The main content area has tabs for 'Description', 'Features', 'Relevant Information', 'Reviews', and 'Link History'. The 'Description' tab is active, showing a text box with the text: 'No description available.' Below this is a photograph of a camera with a red 'no' symbol over it. Below the photo, there is a section 'Flute is:' with a link to 'Instrument'. Another section 'Flute belongs to:' lists categories: 'Instrument', 'Wind instruments', and 'Woodwind'. At the bottom, there is a section 'The following share features with Flute:' followed by a large list of related instruments: 'Alto flute', 'Alto recorder / Treble recorder', 'Bansuri', 'Bass flute', 'Bass recorder / F-bass recorder', 'Boatswain's Pipe', 'Concert flute', 'Contrabass recorder', 'Fipple flutes', 'Fute d'amour', 'Garklein recorder', 'Great bass recorder / C-bass recorder', 'Indian Bamboo Flutes', 'Low whistle', 'Nai', 'Nose flute', 'Ocarina', 'Other flutes', 'Pan flute', 'Piccolo', 'Recorder', 'Sao Truc (Sao Tre vietnamese transverse bamboo flu...', 'Shakuhachi', 'Slide whistle', 'Soprano recorder', 'Soprano flute', 'Subcontrabass recorder', 'Syrinx', 'Tenor recorder', 'Tin whistle', 'Transverse flute', 'Treble flute', 'Venu', 'Vertical flute', and 'Willow flute'.

You may also find it useful to know that 'Fipple Flutes' belongs to 'Flute', and 'Bansuri' belongs to 'Fipple Flutes'. Let's explore 'Fipple Flutes'.

Read all the following content of 'Description', 'Features' and 'Relevant information' of 'Fipple Flutes'

MusicPinta
Social · Semantic · Music

Logout
dicode

Home Semantic Search Contribute Help

Home > Semantic Search > Fipple flutes

Fipple flutes

Description Features Relevant Information Reviews Link History

No description available.

Description Features Relevant Information Reviews Link History

Fipple flutes is:

[Instrument](#)

Fipple flutes belongs to:

[Flute](#) [Instrument](#) [Wind instruments](#) [Woodwind](#)

Description Features Relevant Information Reviews Link History

The following share features with **Fipple flutes**:

[Alto flute](#) [Alto recorder / Treble recorder](#) [Bansuri](#) [Bass flute](#) [Bass recorder / F-bass recorder](#) [Concert flute](#) [Contrabass recorder](#) [Fute d'amour](#) [Garklein recorder](#) [Great bass recorder / C-bass recorder](#) [Indian Bamboo Flutes](#) [Low whistle](#) [Piccolo](#) [Recorder](#) [Sao Truc \(Sao Tre vietnamese transverse bamboo flu...\)](#) [Shakuhachi](#) [Slide whistle](#) [Soprano recorder](#) [Soprano flute](#) [Subcontrabass recorder](#) [Tenor recorder](#) [Tin whistle](#) [Transverse flute](#) [Trebble flute](#) [Venu](#) [Vertical flute](#) [Willow flute](#)

You may also find it useful to know that 'Transverse Flute' belongs to 'Fipple Flutes', and 'Bansuri' belongs to 'Transverse Flute'. Let's explore 'Transverse Flute'.

Read all the following content of 'Description', 'Features' and 'Relevant information' of 'Transverse Flute'

MusicPinta
Social · Semantic · Music

Logout
dicode

Home Semantic Search Contribute Help

Home > Semantic Search > Transverse flute

Transverse flute

Description Features Relevant Information Reviews Link History

No description available.

Description Features Relevant Information Reviews Link History

Transverse flute is:

[Instrument](#)

Transverse flute belongs to:

[Fipple flutes](#) [Flute](#) [Instrument](#) [Wind instruments](#) [Woodwind](#)

Description Features Relevant Information Reviews Link History

The following are **Transverse flute**:

No items found.

The following share features with **Transverse flute**:

[Alto flute](#) [Bansuri](#) [Bass flute](#) [Concert flute](#) [Fute d'amour](#) [Indian Bamboo Flutes](#) [Piccolo](#) [Sao Truc \(Sao Tre vietnamese transverse bamboo flu...\)](#) [Soprano flute](#) [Trebble flute](#) [Venu](#)

You may also find it useful to know that 'Indian Bamboo Flutes' belongs to 'Transverse Flute' , and 'Bansuri' belongs to 'Indian Bamboo Flutes' . Let's explore 'Indian Bamboo Flutes'.

Read all the following content of 'Description', 'Features' and 'Relevant information' of '**Indian Bamboo Flutes**'

The screenshot shows the MusicPinta website interface. At the top left is the MusicPinta logo with the tagline 'Social · Semantic · Music'. At the top right are links for 'Logout' and 'dicode'. A navigation bar contains 'Home', 'Semantic Search', 'Contribute', and 'Help'. Below this is a breadcrumb trail: 'Home > Semantic Search > Indian Bamboo Flutes'. The main title is 'Indian Bamboo Flutes'. A tabbed interface shows 'Description' selected, with other tabs for 'Features', 'Relevant Information', 'Reviews', and 'Link History'. The description area contains a camera icon with a red 'X' over it and the text 'No description available.'. Below this, a section titled 'Indian Bamboo Flutes is:' contains a link for 'Instrument'. Another section titled 'Indian Bamboo Flutes belongs to:' contains links for 'Fipple flutes', 'Flute', 'Instrument', 'Transverse flute', 'Wind instruments', and 'Woodwind'. A third section titled 'The following are Indian Bamboo Flutes:' shows 'No items found.'. A final section titled 'The following share features with Indian Bamboo Flutes:' contains links for 'Bansuri' and 'Venu'.

D.4 Knowledge Utility

User ID	Exploration Strategy	Pre-Exploration			Post-Exploration		
		Remember	Categorize	Compare	Remember	Categorize	Compare
User 1	EC(Biwa)	0	0	0	5	3	7
User 2	EC(Biwa)	0	0	0	6	2	4
User 3	EC(Biwa)	0	0	0	4	4	1
User 4	EC(Biwa)	0	0	0	4	3	1
User 5	EC(Biwa)	0	0	0	2	3	0
User 6	EC(Biwa)	0	0	0	6	4	3
User 7	EC(Biwa)	0	0	0	2	3	3
User 8	EC(Biwa)	0	0	0	2	2	3
User 25	EC(Biwa)	0	0	0	3	3	1
User 26	EC(Biwa)	0	0	0	1	1	2
User 27	EC(Biwa)	0	0	0	2	2	2
User 28	EC(Biwa)	0	0	0	3	2	7
User 29	EC(Biwa)	0	0	0	1	1	1
User 30	EC(Biwa)	0	0	0	6	4	2
User 31	EC(Biwa)	0	0	0	2	2	4
User 32	EC(Biwa)	0	0	0	3	3	1
User 9	EC(Bansuri)	0	0	0	5	5	1
User 10	EC(Bansuri)	0	0	0	6	4	3
User 11	EC(Bansuri)	0	0	0	4	3	4
User 12	EC(Bansuri)	0	0	0	4	3	3
User 13	EC(Bansuri)	0	0	0	3	1	4
User 14	EC(Bansuri)	0	0	0	6	5	2
User 15	EC(Bansuri)	0	0	0	2	4	0
User 16	EC(Bansuri)	0	0	0	3	5	2
User 17	EC(Bansuri)	0	0	0	5	3	1
User 18	EC(Bansuri)	0	0	0	3	2	0
User 19	EC(Bansuri)	0	0	0	4	4	1
User 20	EC(Bansuri)	0	0	0	2	2	0
User 21	EC(Bansuri)	0	0	0	3	3	0
User 22	EC(Bansuri)	0	0	0	3	3	1
User 23	EC(Bansuri)	0	0	0	6	6	1
User 24	EC(Bansuri)	0	0	0	3	3	4
User 1	CC(Bansuri)	0	0	0	2	1	2
User 2	CC(Bansuri)	0	0	0	4	3	2
User 3	CC(Bansuri)	0	0	0	3	2	0
User 4	CC(Bansuri)	0	0	0	2	2	1
User 5	CC(Bansuri)	0	0	0	2	2	0
User 6	CC(Bansuri)	0	0	0	3	2	1
User 7	CC(Bansuri)	0	0	0	1	1	0
User 8	CC(Bansuri)	0	0	0	0	1	1
User 25	CC(Bansuri)	0	0	0	2	2	1

User ID	Exploration Strategy	Pre-Exploration			Post-Exploration		
		Remember	Categorize	Compare	Remember	Categorize	Compare
User 26	CC(Bansuri)	0	0	0	1	1	1
User 27	CC(Bansuri)	0	0	0	3	1	0
User 28	CC(Bansuri)	0	0	0	3	2	7
User 29	CC(Bansuri)	0	0	0	1	1	1
User 30	CC(Bansuri)	0	0	0	1	1	0
User 31	CC(Bansuri)	0	0	0	2	1	0
User 32	CC(Bansuri)	0	0	0	2	2	1
User 9	CC(Biwa)	0	0	0	2	1	0
User 10	CC(Biwa)	0	0	0	2	2	2
User 11	CC(Biwa)	0	0	0	2	2	4
User 12	CC(Biwa)	0	0	0	3	1	0
User 13	CC(Biwa)	0	0	0	4	1	3
User 14	CC(Biwa)	0	0	0	3	2	1
User 15	CC(Biwa)	0	0	0	1	1	1
User 16	CC(Biwa)	0	0	0	1	1	0
User 17	CC(Biwa)	0	0	0	4	2	1
User 18	CC(Biwa)	0	0	0	2	1	1
User 19	CC(Biwa)	0	0	0	3	1	1
User 20	CC(Biwa)	0	0	0	1	0	0
User 21	CC(Biwa)	0	0	0	3	1	0
User 22	CC(Biwa)	0	0	0	1	1	0
User 23	CC(Biwa)	0	0	0	2	1	0
User 24	CC(Biwa)	0	0	0	2	3	3