Property Induction using Structure-based Property Propagation

Hiroki Shimada

Master of Science
Artificial Intelligence
School of Informatics
University of Edinburgh
2016

Abstract

Property induction is one representative area of inductive reasoning in which one is supposed to infer the properties of unseen concepts based on knowledge about the properties of other instances. We aim to propose a methodology for property induction that can be applied to general problems from a knowledge base that does not include general rules. Our assumption is that when knowledge is represented as resource description framework (RDF) graphs, features that are important for inductive reasoning appear in the graph structure. Based on this assumption, we propose ”Structure-based Property Propagation” which predicts blank properties by propagating known properties through the graph arcs and vertices not depending on previous knowledge concerning their label.

Acknowledgements

First of all, I would like to thank my supervisor, Prof. Stuart Anderson, for his valuable guidance and insight throughout the duration of my dissertation. He frequently made time and had a meeting for me despite his busy schedule, and always provided me with helpful advices and feedbacks. Especially, he helped me a lot in polishing the paper by reading through and identifying possible improvements. Without him, I could not have performeed my work to such extent.

I was able to study at the University of Edinburgh through the generous financial support from Japan Student Services Organization (JASSO), which I also owe gratitude. Finally, thanks to my brother and my parents for their financial, mental and overall supports during this one long year.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Hiroki Shimada*)

Table of Contents

Chapter 1

Introduction

Inductive reasoning is defined as a reasoning process that infers general rules by observing and analyzing specific instances (Polya, 1945 cited in [1]). In this area, a great deal of research has been conducted especially in the property induction field, which is a sub-problem of the inductive reasoning. Property induction is an induction problem that involves inferring " blank" (i.e., unfamiliar, or novel) properties (e.g., crows have property A) based on known or familiar properties (e.g., sparrows have property A. Crows and sparrows are birds, hence similar). We do not assume that we have general rule (e.g., all birds have property A). If there is a general rule in the knowledge base, then the reasoning is to be deductive, not inductive. Therefore finding hidden general rules is the core problem of inductive reasoning.

In this paper, we propose a novel approach to the property induction. In our approach we use Resource Description Framework (RDF) to describe background knowledge and express the premise and conclusion of the induction, which allows more general expression. Normally, more complex representation causes less efficiency (i.e., ability to reason over the ontology). With reference to ontology though, we attempt to overcome this difficulty by focusing only on the structural features of the graphs.

1.1 Motivation

Many studies [2, 3, 4] have suggested that background knowledge is of considerable importance in making inductive inferences, and the range of application is limited by the expressive power of representation it uses to describe back ground knowledge. Previous works have used features[14], taxonomic categories[12], low dimensional model representing similarities between each concepts, and causal graphs as a repre-

sentations of background knowledge. These representations, however, fail to express complex concepts and knowledge because they lack expressivity. For instance, taxonomic categories can express the relation " crows are birds", but it cannot describe detailed features such as " crows eat mice". Although the feature-based induction model (FBIM) [14] is capable of dealing with such features by encoding them in binary form (where 1 means true and 0 means false), the expression does not contain any further information about mice. In the FBIM, " crows eat mice" and " crows eat squirrels" are treated as different features, although they are similar in that both mice and squirrels are small mammals (the drawbacks of previous methods are explained in Chapter 2). That is the motivation for using RDF for inductive reasoning. It allow us to express subtle similarities even between features because concepts are described as graphs and the similarities appears in their structural features. By using RDF, we can deal with richer contexts and knowledge, broadening the range of application and enabling more general induction.

From a more macro viewpoint, this research contributes to realizing a part of the human reasoning function. Unlike deductive reasoning, inductive reasoning contains ambiguity and entails creativity, hence comprising a challenging task. As often pointed out, inductive reasoning corresponds with much of what people do in everyday life [2], but as yet there is a lack of clarity concerning its mechanisms and related methodology. This research may contribute to enabling us to make inductive inference in more general representations, rather than depending on certain encoding such as features or categories. Therefore our research can be potentially applied to the following areas:

- Question Answering Systems : Even when the property queries is not in the database, the system can answer the questions by making inferences.
- Conversational agents : An agent can extrapolate new information from a conversation with a user, and make use of it in a following conversation.
- Information retrieval : A system can suggest possible candidates for results when it fails to find information regarding queries using inductive inference.
- Recommendation engines : The relation" A like B" can also be encoded by RDF triples, which allows us to infer the preference of an individual by incorporating relevant information of the individual.

The reason why we choose RDF as the representation is that RDF triples are open to use via services such as DBpedia, and they are relatively simple. Conceptual graphs

[5] are more expressive than RDF, for example, but they are much more complicated in that they entail multiple types of nodes and the graphs take different forms. RDF always takes the form of triples, and we can query using SPARQL, which is a great advantage in processing the graph structure.

1.2 Objectives

Simply put, this research aims to propose a way of integrating various type of background knowledge in inductive reasoning. In other words, our research objective is to propose the methodology for property induction that can be applied to general problems using a knowledge base that does not include general rules. This thesis proposes an approach that associates property induction with an RDF graph structure. In describing the knowledge base by including concepts and their relations in graph representation, hidden relationships may emerge in the structural properties such as the connectivity of each vertex and mutual part shared by multiple subgraphs. In this paper we use the term structure-based approach to refer to methods which use these properties.

We implement the inference system using the proposed methodology, conducting experiments with actual RDF data on DBpedia in order to verify whether the proposed methodology works for data in the real world.

1.3 Completion Criteria

We focus on how suitable our method is for practical application in terms of its speed, generality, versatility, and performance. Putting these factors together, we compare and evaluateaspects, summarized as follows:

1. Correctness : Does the method actually correct relations that are consistent with all other data in the knowledge base? Does it not infer incorrect properties that contradict existing instances?

2. Performance : Is the algorithm reasonable in terms of time and space complexity?

3. Applicability : Do the results found work for datasets in the real world? Is the method sufficiently general to be applicable for various types of practical problems?

Therefore, the completion criteria are specified according to these three points, that is, to identify the advantages of the proposed graphical structure-based approach over past methods (introduced in the next chapter) in terms of these three criteria.

1.4 Structure of the Dissertation

In what follows, we explore previous efforts and a new methodology for property induction, and how it works for real problems. Chapter 2 introduces background knowledge on this topic, including RDF, knowledge representation, and inductive reasoning. We also look at previous works on property induction, focusing on the similarity coverage model (SCM) and the feature-based induction model (FBIM). Chapter 3 provides a detailed description of our methodology. Chapter 4 introduces a complete picture of the experiment and the evaluation scheme. The evaluation results for each dataset are also provided. In Chapter 5, we analyze and discuss the results presented in the evaluation section. Finally, we summarize the contributions of our research and present some problems that cannot be addressed as potential future work in Chapter 6.

Chapter 2

Background and Related Works

2.1 Knowledge Representation

2.1.1 Knowledge Representation

Knowledge representation is formally defined as the field of Artificial Intelligence regarding how knowledge should be automatically manipulated and represented [6]. Several roles of knowledge representation are presented in [7], in this paper we take the position that knowledge representation is a medium for pragmatically efficient computation. Namely, knowledge representation is an environment for computers to think as humans do. In the context of inductive reasoning, the following properties are desired in the representation of background knowledge.

- It is possible to express any form of knowledge such as objects(e.g.*car, dog*), and facts (e.g. *Every trailer track has 18 wheels.*, *Elephants love onions*).
- Similarities and differences between concepts can be represented in a quantitative form.
- The form of representation is consistent so that conclusions and premise of reasoning can be described in the same form.
- The representation allows a certain level of deductive inference.

Though the expressivity (first property) is limited, RDF possesses the rest of characteristics, which will be the great advantages in making inductive reasoning. Further details about RDF is explained in next section.

In this research we assume Closed World Assumption (CWA). CWA is the assumption that what is not known is regarded as false, where the absence of information is interpreted as negative information. In other words, the system assumes information in the dataset (knowledge base) is all the information in the world[21]. Hence, if the system has limited information, it may infer something which does not hold in the real world. However. this is not inaccurate, as long as the results are consistent with other data in the knowledge base. For example, if the statement " Japanese language is difficult." is all information in the knowledge base, this implies that the Japanese language is generally difficult whatever the circumstances. In order to disambiguate, it is necessary to be more specific by adding more information such as " Writing in Japanese language is difficult for non-Japanese natives.".

It is reasonable to assume that knowledge representation is a representation of the world, and one knowledge base (generally, a set of representations) corresponds to the one world. That means if and only if knowledge base contains all the information in the real world, the system is expected to infer things which is true in the real world.

2.1.2 RDF

Resource Description Framework (RDF) is a data model that provides integrated and uniform access to information resources and services. RDF is often used for representing metadata about other web resources [8]. RDF takes the form of a triple, and it includes the information about subject, predicate, and object. For example, one RDF triple looks like `(Subject, Predicate, Object)`. In RDF, concepts and properties are represented as Uniform Resource Identifier (URI) or Literal. Thus, typical RDF structure takes the forms shown below in Figure2.1. Since showing predicates indi-

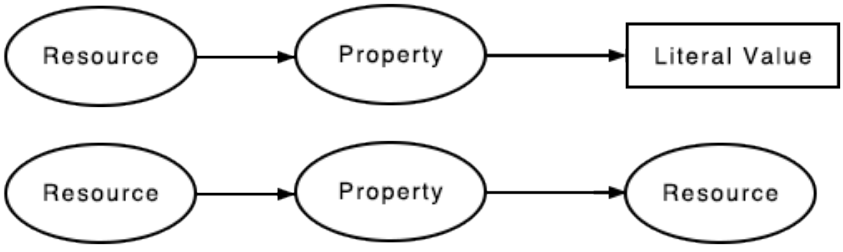

Figure 2.1: An RDF Triple

cates structural characteristics of the data, in this paper we sometimes describe pred-

icates as nodes, although they are typically represented as labels of arrows in other literatures.

When it comes to a set of multiple RDF triples, we can have a graph structure by merging the node referring to same URI. An example of an RDF graph is shown in Figure2.2 cited from [9].

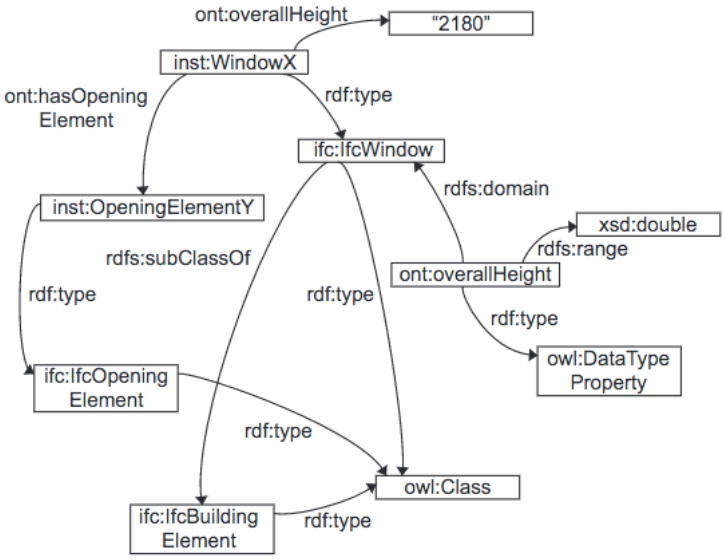

Figure 2.2: An RDF graph, obtained from ref. [9]

RDF triples are highly suitable for property reasoning because in RDF relationships between concepts are regularly formed (e.g. all relationships are represented as one for one) and thus it is easy to retrieve property information.

This feature enables RDF to make simple deductive reasoning. For example, if there is a relation `(A, rdfs:subClassOf, B)` and `(x, rdf:type, A)`, then we can derive `(x, rdf:type, B)`. That means Class A is a subclass of Class B and x is an instance of Class A, then x is also an instance of Class B. Besides this rule, there are several inference rules that allow deriving new information from existing knowledge.

Querying is also a key feature of RDF, which is a big advantage of RDF over other representation such as Conceptual Graph. Though there are several querying systems for RDF triples, we use SPARQL query. SPARQL contains capabilities for querying required and optional graph patterns along with their conjunctions and disjunctions [10]. It uses graph pattern matching to find the results, and it returns all the matching instances in the dataset. Following listing shows the sample query that retrieve the type

of the super class of ifc:ifcBuildingElement (prefix is omitted) against the dataset in Figure 2.2. The result of this query is expected to be owl:Class.

```
SELECT ?type
WHERE {
    ifc:ifcBuildingElement    rdfs:subClassOf ?superClass .
    ?superClass   rdf:type ?type .
}
```

Listing 2.1: SPARQL query

2.2 Inductive Reasoning

As is briefly explained in the introduction, inductive reasoning is a reasoning process to make an inference about general rules from knowledge or data of individual instances. Property induction is a subclass of the inductive reasoning problem, which has been explored by a number of researches. The main task in this area is to predict the property of novel category using other property and category knowledge.

One major model for this task is similarity based induction models. The model relies on taxonomic similarity and hierarchical categories[2], and it typically assumes two category: premise(base) and conclusion(target) categories. The information about the premise category is known, whereas no information is supplied for the properties of the conclusion category. Sometimes the properties of conclusion category are called *blank* properties, because they are unknown and are inferred from information of known categories. Similarity based induction models are based on the assumption that we are more likely to extend a property from a base premise to a target category to the extent that the target category is similar to the base category [11]. In this section we introduce two previous similarity-based approaches known as Similarity Coverage Model (SCM) and Feature Based Induction Model(FBIM).

As well as similarity based models, Bayesian Models are also common approaches to the property induction. Bayesian analysis assumes that an inductive argument is considered as estimating the range of the novel property [4]. That means the inductive problem is to determine which of familiar distributions the blank property most closely resembles [2]. We mainly explain Heit's model[4], because his model contains core ideas of the Bayesian induction models.

2.2.1 Similarity Coverage Model

Osherson et al.[12] examined the condition under which the property of the conclusion category is attributable to that of the premise category. The SCM assumes that the likelihood of a blank property being generalized from a set of premises to a conclusion is determined by two factors: (a) the degree to which the premise categories are similar to the conclusion category and (b) the degree to which the premise categories are similar to members of the lowest level category, which includes both the premise and the conclusion categories [12]. Factor (a) is called similarity, and second (b) is called coverage. They also present 13 qualitative phenomena that tend to hold in category induction models.

Osherson et al. define an argument as a finite list of sentences, where the sentence is called the conclusion, and the rest are called its premises. We can denote an argument as $P_1, \cdots, P_n/C$, where P_i means a premise, C is a conclusion. Argument A is stronger than Argument B if the premise in A is more likely to be projected to its conclusion than that of B.

$$\frac{\text{Monkeys have BCC in their blood.}}{\text{Antelope have BCC in their blood.}} \tag{2.1}$$

$$\frac{\text{Elephants have BCC in their blood.}}{\text{Antelope have BCC in their blood.}} \tag{2.2}$$

For example, the argument (2.1) is stronger than the argument (2.4), in terms of similarity between premise and conclusion. (The category Antelope is more similar to Monkeys than to Elephants.) The sentences above the bar is the premise, and the statement below the bar is the conclusion.

$$\frac{\begin{array}{c}\text{Monkeys have BCC in their blood.}\\ \text{Elephants have BCC in their blood.}\end{array}}{\text{Antelope have BCC in their blood.}} \tag{2.3}$$

Argument (2.4) is even stronger than the argument (2.1). SCM achieves this by considering“ coverage”. As explained above, coverage is the degree to which the premise categories are similar to members of the lowest level category that includes both premise and conclusion categories. In this case, the lowest level category which includes both categories is a mammal, because monkeys, elephants, and antelopes are all included in that category, and all other categories that includes them are bigger(higher) than the mammals. The degree of the category mammals that is covered by the set {Elephants, Monkeys} is clearly larger than that of the set {Monkeys}.

Osherson et al. provides a quantitative test of the model. In their experiment, the argument strengths of properties of mammals are predicted by their model and the results are tested against ratings of argument strength provided by subjects (80 University of Michigan undergraduates). Correlation coefficient between predicted and observed confirmation score is used for evaluation, the average of correlations was approximately 0.87 or better, which let them to conclude that SCM reflects genuine psychological processes of induction, combined with other quantitative evidences presented in their paper.

2.2.2 Feature Based Induction Model

Sloman [14] proposes an alternative, feature based, model of the inductive task that SCM tries to solve. Feature Based Induction Model(FBIM) also emphasizes the similarity of premise and conclusion categories as a basis for induction [2], but instead of using hierarchical category structure, it depends on connection strength between features of the premise and the conclusion. FBIM assumes that argument strength is determined by the proportion of the conclusion category's feature that are shared by the premise categories. All categories are represented as vectors of numerical values over a set of features, and thus it does not need stable category structures unlike in SCM.

Suppose we infer if Falcons have predicate X given the information that Robins have property X.

$$\frac{\text{Robins have X.}}{\text{Falcons have X.}} \tag{2.4}$$

FBIM can decide the argument strength by considering the feature of each category. Since X is a blank predicate, at the beginning the unit representing predicate X is not connected to any featural units. The premise is encoded by connecting feature of Robins to the predicate unit (Figure 2.3). The weight is changed in accordance with following encoding rule:

$$w_i(P_0,P) = w_i(P_0) + \lambda_i[1 - a_x(P/P_0)]a_i(P) \tag{2.5}$$

Where w_i is weight corresponding to the i th feature, $a_x(C/P_1,\cdots,P_j)$ is an argument strength whose conclusion is C and premises are $P_1,\cdots,P_j$, and $\lambda_i = 1 - w_i(P_0)$ is a scalar coefficient which keep each weight between 0 and 1. In this example small number of binary feature is used to represent the categories.

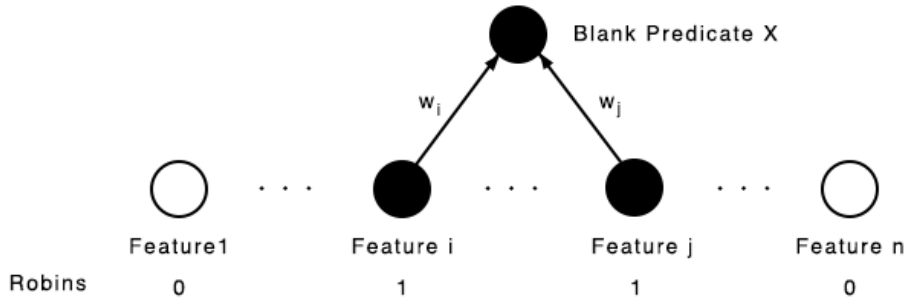

Figure 2.3: Encoding premise [14]

The conclusion is judged by determining the degree to which the predicate X is activated upon the features of conclusion category (Figure 2.4). In this example the value of predicate unit X of Falcons is 0.5 as shown in the figure, but generally it is decided by the following activation rule:

$$a_x(\boldsymbol{I}/P_1,\cdots,P_j) = \frac{\boldsymbol{W}(P_1,\cdots,P_j)\cdot\boldsymbol{A}(\boldsymbol{I})}{|\boldsymbol{A}(\boldsymbol{I})|^2} \tag{2.6}$$

Where $\boldsymbol{W} = [w_1,\cdots,w_n]$ is weight vector, P_i means premise, $\boldsymbol{I}$ denotes input vector, and $\boldsymbol{A}(Y) = [a_1(Y),\cdots,a_n(Y)]$ is activation vector (Y is a category). When $\boldsymbol{I}$ is a conclusion category, $a_x(\boldsymbol{I}/P_1,\cdots,P_j)$ is a model of argument strength.

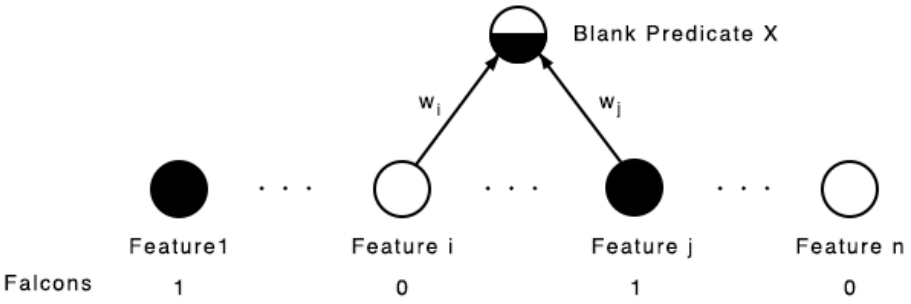

Figure 2.4: Testing concluson [14]

Sloman claims that though category-based method is not inconsistent with feature-based view, category-based approach fails to capture all the subtleties of induction. SCM also has a problem in deciding the category of new data, for example, the lowest-level category including ducks and geese can be “ birds”, “ water-birds”, and “ web-footed birds”. Since FBIM attributes both similarity and coverage to a single factor features, it needs only a set of features. Therefore, FBIM achieves great flexibility.

2.2.3 Bayesian Models

Heit[4] tries to apply an account from Bayesian statistics to the task of evaluating inductive arguments. It is helpful to present how Bayesian model works in the context of a simple inference example about animals. Suppose we make an inductive argument regarding just two categories of animals: horses and cows. There are four types of properties: properties that are true of both horses and cows, properties that are true of either one amimal, and properties that are true of neither cows nor horses. We name these types $H_1 \cdots H_4$ respectively. Let us say we are supposed to predict the probability that horses have Property P given prior probability $P(H_1) \cdots P(H_4)$. If we observe the fact " Cows have Property P", how is the prior probability $P(H_i)$ updated? Bayes's Theorem (2.7) gives the update rule of the probability that Property P belongs to type H_i given premise D.

$$P(H_i|D) = \frac{P(H_i)P(D|H_i)}{\sum_{j=1}^{n} P(H_j)P(D|H_j)} \tag{2.7}$$

D here means the dataset or the premise " Cows have Property P". Table 2.1[4] shows an example of how $P(H_i)$ is updated after observing the premise D.

Hypothesis	(Cow,Horse)	$P(H_i)$	$P(D\|H_i)$	$P(H_i\|D)$
H_1	(T,T)	0.70	1.0	0.93
H_2	(T,F)	0.05	1.0	0.07
H_3	(F,T)	0.05	0	0
H_4	(F,F)	0.20	0	0

Table 2.1: An Example of Bayesian Induction

Heit's model, however, does not provide details about how prior probability are computed[2]. On this point later researches such as the structured statistical approach (Kemps et al.[3]) try to give a mechanism to derive prior probabilities from background knowledge. They suggest the method to derive prior distribution from two background knowledge: knowledge about relationships between the categories and knowledge about how the property of interest depends on these relationships.

Kemps et al. also extend Heit's Bayesian model by introducing feature vector. Following example is provided in their paper[3]. Suppose there is a finite domain composed of n categories. In this example, a biological domain with four categories: cheetahs, lions, gorillas, and monkeys is used. Feature(property) is represented as

n-dimensional vector f that assigns 1 to the categories which have the feature and 0 to the categories which do not. Suppose the label vector $l_X = 1,0$ where $X = \{cheetah, monkey\}$. This indicates monkeys do not have this feature, but cheetahs do. Our task is to infer full feature vector f_N from l_X, where $N = \{cheetahs, lions, gorillas, monkeys\}$. Bayes's rule tells us a prior distribution $p(f_N)$ is updated to a posterior distribution $p(f_N|l_X)$ on the feature vector f_N.

$$p(f_N|l_X) = \frac{p(l_X|f_N)p(f_N)}{\sum_f p(l_X|f)p(f)}$$

,where the sum $\sum_f$ denotes the summation over all possible feature vectors f.

2.3 Property Propagation

Property propagation is a method to derive properties from the relevant objects of the target, which has been used in the information retrieval field. The deductive reasoning of the type using rdfs:subClassOf property introduced in section 2.1.2 is also the property propagation [36]. The property information of superclass is propagated through the arc to the subclass.

Tzompanaki et al.[37] proposed reasoning methodology based on property propagation using CIDOC-CRM and CRMdig based repositories. CIDOC-CRM is an RDF-like extensible semantic framework which is intended to promote a shared understanding of cultural heritage information,and CRMdig is its extension. Tzompanaki et al. introduced two inference rules for reasoning on provenance information of an object.

1. The property of an object is the aggregation of the explicitly defined property in the object itself and the respective properties of all its subparts.

2. Physical objects may share properties with their digital representations and their derivatives.

First and second rule describes *part-of chain* and *derivative chain* respectively.

Figure 2.5 illustrates how the part-of chain works. The Statue of Queen Victoria in Figure 2.5 does not have material information in the dataset, but its subparts have the information. The material information of subparts can be propagated to the target object (Statue of Queen Victoria) because of the *composed of* property. With the aid of the property propagation, the query " Find all statues made of Bronze" would return the Statue of Queen Victoria as an answer. Similarly, the result set of the query " Find

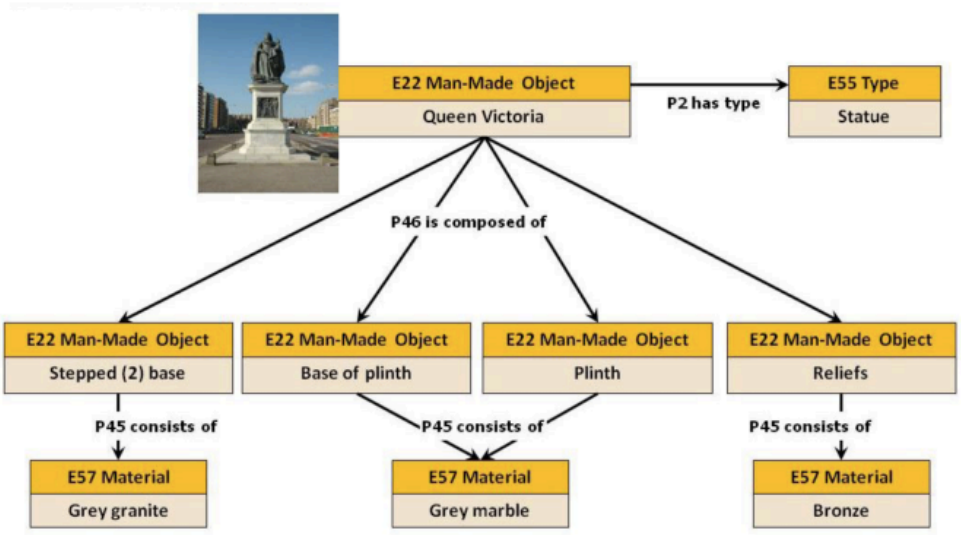

Figure 2.5: An example of part-of chain [37]

the material of the Queen Victoria Monument ” would be Grey granite, Grey marble, Bronze.

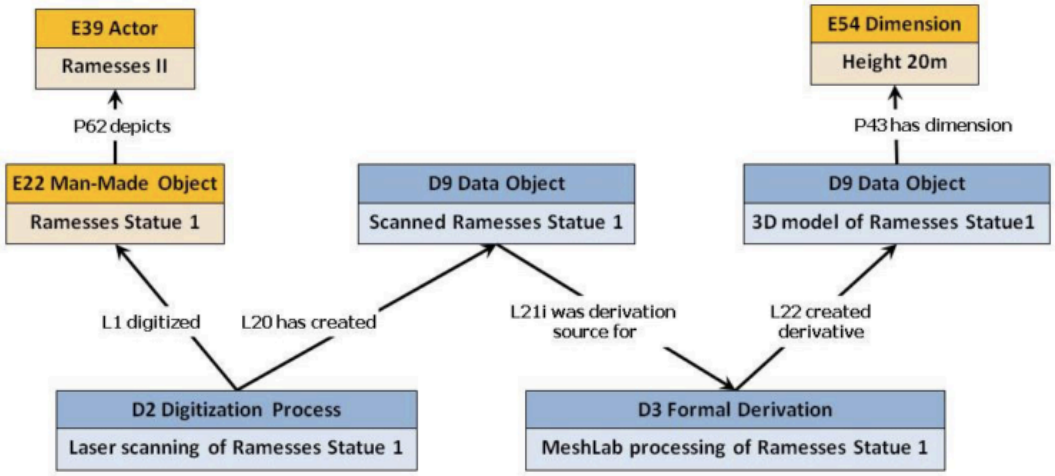

Figure 2.6: An example of derivative chain [37]

Figure 2.6 presents an example of rule 2 application. CRMdig contains information about digital objects such as the information of 3D-scanned modeling of real objects. Even though explicit declaration of the property of physical object is not available, they can be propagated from that of digital objects. In the example in Figure 2.6, any size information of the object“ Ramesses Statue 1 ”is not explicitly defined. However, their reasoning approach can answer the query “ Find the size of the Ramesses Statue

1 Object ”, because it retrieve the size information of “ 3D model of Ramesses Statue 1 ” assuming it is also applicable to the physical object.

Chapter 3

Methodology

3.1 Problem Definition

In this research we will focus on a property induction where categories are not necessarily assumed. Here we define the property induction problem with RDF data structure.

In previous researches [14, 12] introduced in Chapter 2, the premise and conclusion category are assumed. The premises are encoded as a set of multiple statements, while the conclusion is a sentence regarding conclusion category. In our formalization, both premise and conclusion are represented as a set because multiple statements can be inferred from the premises. We do not assume premise categories each time making an inference, instead, we regard the whole dataset $\boldsymbol{D}$ as premise information that the inference is based on. Precisely, $\boldsymbol{D}$ is a set of RDF triples, that is,

$$\boldsymbol{D} = \{\boldsymbol{t}_1, \cdots, \boldsymbol{t}_n\} \tag{3.1}$$

$$\boldsymbol{t}_i = (s, p, o), \tag{3.2}$$

Where s, p, o denote subjects, predicates, objects respectively. Subjects s in dataset(premises) are also called premise concepts. We call things described as URI *resource*, that is, s, p, and o are elements of the resource set $\mathcal{R}$, As a subset of $\mathcal{R}$, there are the concept set $\mathcal{C}$ and the predicate set $\mathcal{P}$. Both sets are the subsest of $\mathcal{R}$, and $s, o \in \mathcal{C}, p \in \mathcal{P}$.

Then, property induction for the subject s is defined as a problem to find a function that returns properties that should hold for s given premises $\boldsymbol{D}$.

$$f_{\boldsymbol{D}}(s) = \{(s, p, o) \mid (s, p, o) \text{ is inferred to hold in } \boldsymbol{D}\} \tag{3.3}$$

The right hand side can be thought of as a conclusion set, and we call subject s conclusion concept or target concept, which corresponds to conclusion category in previous

works. Note that we use Closed World Assumption (see Section 2.1.1), that is, f infers the fact that is likely to hold in the dataset $\boldsymbol{D}$, not the real world. Reasoning function f can also take a pair of a subject s and a predicate p as the argument as formula (3.5), and the subscript denotes the dataset it assumes.

$$f_{\boldsymbol{D}}((s,p)) = \{(s,p,o) \mid (s,p,o) \text{ is inferred to hold in } \boldsymbol{D}\} \tag{3.4}$$

This specifies both the subject and the predicate, and the system is supposed to infer the object that satisfies.

This definition is as expressive as the formalism used in [14, 12]. For instance, the inference (2.3) in Chapter 2 can be expressed as follows using our definition.

$$\boldsymbol{D} = \{(\text{Monkey}, \text{haveInBlood}, \text{BCC}), (\text{Elephant}, \text{haveInBlood}, \text{BCC})\} \cup \boldsymbol{D'} \tag{3.5}$$

$$\{(\text{Antelope}, \text{haveInBlood}, \text{BCC})\} \in f_{\boldsymbol{D}}(Antelope) \tag{3.6}$$

$\boldsymbol{D'}$ denotes the set of triples which represents other background knowledge implicitly assumed in the example. Note that Monkey, Elephant, BCC, Antelope and haveInBlood are all URI corresponding to each concept and property.

3.2 Assumption

Our methodology is based on the hypothesis that important features such as similarity, communality, inclusion relation, antonymy, and other forms of features can appear in a graph structure where knowledge is represented in the form of graphs. Since these features are closely linked to inductive reasoning, graph structure based approach is suitable for automated property induction. Regarding the features appeared in the graph structure and the reasoning which uses RDF graphs, we have the following four assumptions:

1. The more properties in common two concepts share, the more likely the properties are projected from one to the other.

2. First hypothesis can be true for both inbound and outbound properties.

3. Even if two concepts do not share common properties, as long as the two subjects have common concept that shares its properties with the two subjects, the two objects can be regarded as similar to some extent.

4. Even if two concepts do not share common properties, as long as the two subjects have properties whose objects share common properties, the two concepts can be considered as similar on one level or another.

Assumption1 The first assumption is illustrated in the Figure 3.1 and 3.2 [31]. Suppose we know the means of Subject A and B is flying, and the type of Subject B is Bird. Solid arrows in Figure 3.1 indicate the RDF predicate in the dataset. If these are all the knowledge in the dataset, it is possible to infer that Subject A also has a property " its type is Bird" because Subject A and B share the same property, though the inference is uncertain. In the next place, assume the data " The type of both Subject A and B is

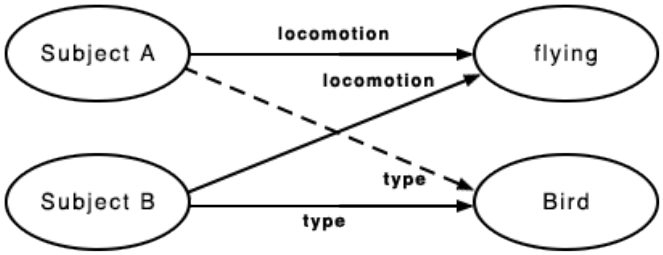

Figure 3.1: Assumption1 - Uncertain inference

Bird" is added to knowledge base. It is reasonable to think that the property " The type of Subject A is Bird" is more likely to be inferred than the case illustrated in Figure 3.1. This is because Subject A and B share additional properties in common, and that makes the two concepts more similar. As with previous approaches, we have adopted the assumption that we are more likely to extend a property from a base premise to a target premise to the extent that the target category is similar to the base category [11]. To put it plainly, similar concepts are more likely to have similar property. (Dashed arrows represent inferred properties.)

Assumption2 Suppose we have 4 RDF triples shown in Figure 3.3. First assumption tells us Subject A and B are similar because they share two properties " their locomotion is flying (they move by flying)" and " their locomotion is walking (they also moveby walking)". Second assumption claims that " walking" and " flying" are also similar, hence properties of one concept might be able to be projected onto the other. In Figure 3.3, for Subject A, the property locomotion-flying is an outbound property, but it is an inbound property for the node flying. This assumption is obvious because any property can be rewritten to point in the opposite direction (e.g. A statement " A eats B" can be written as " B is eaten by A".). Note that same the relationship between

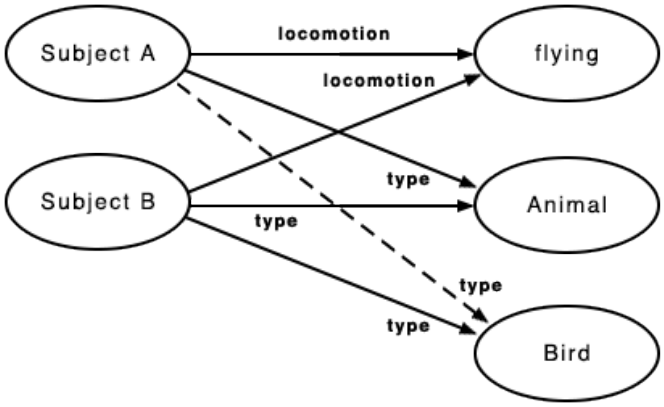

Figure 3.2: Assumption1 - More reliable inference

subjects does not necessarily hold in objects in this example. For example in Figure 3.3, “ flying” and “ Animal” do not have properties in common becaus the predicates “ type” and “ locomotion” are different.

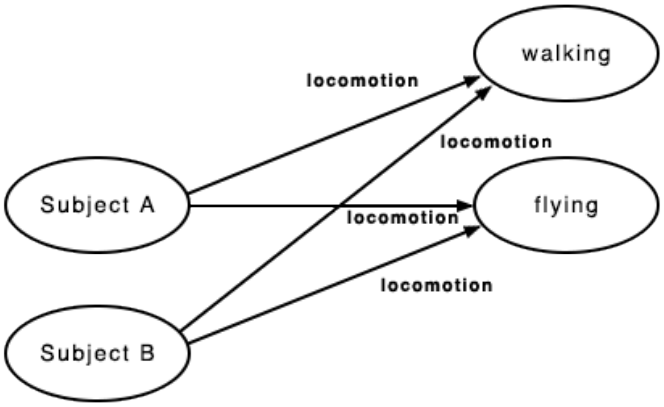

Figure 3.3: Assumption2

Assumption3 Putting the third assumption simply, if concepts A and B are similar, and concepts B and C are also similar, then we can conclude that concepts A and C are also similar. This assumption illustrates the transitivity of the similarity. In Figure 3.4, although Subject A and Subject C have no common properties, they have Subject B which shares its properties with Subject A and C. Because of interposition of Subject B, the properties of Subject A can be projected to Subject C (the arrows are omitted for the sake of simplicity). This assumption is also trivial because from the first assumption we can infer Subject A is likely to have the properties that Subject

B has, and the same thing holds between B and C. Of course, this inference is not as certain as the case where Subject A and C have direct relationship. In this sense properties can be " propagated" through the concepts in the dataset, and the degree of strength of propagation depends on the similarity between the concerned objects. We use the term *n-th order propagation* to refer to the propagation of property through $n-1$ concepts. Propagation of the property from Subject A to C in Figure 3.4 is a second order propagation.

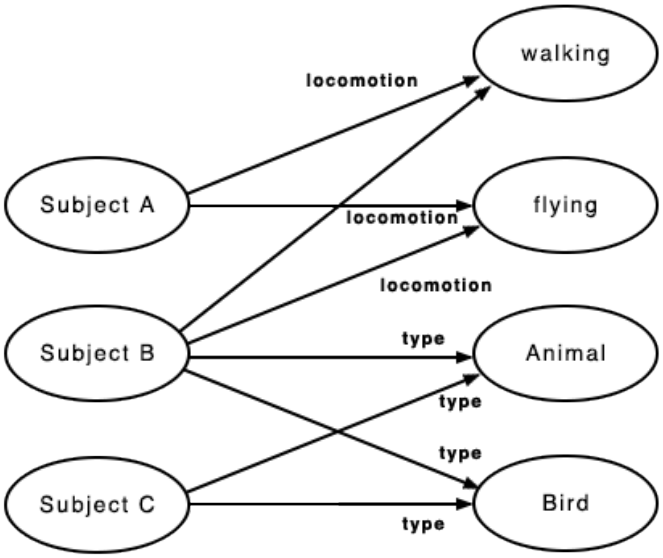

Figure 3.4: Assumption3

Assumption4 In the Figure 3.5, Subject A and Subject B has no properties in common, but both are instances of Bird class. In this case even though a direct relationship cannot be found, they should be considered as similar. Therefore, the property of one concept can be propagated to the other. In this example, Subject B is inferred to have the property `(locomotion, flying)` , because the property is held by the Subject A. As is the case with Figure 3.4 in the third assumption, this is also a second order propagation. While the path expands in breadth direction in the third assumption, the assumption 4 expands the path in depth.

None of our assumptions use label information such as " type" and " Bird" unlike Tzompanaki et al's method introduced in Section 2.3, where they used the relations of graph connections. We call this kind of approach structure-based approach. As illustrated in the examples, graph structure gives us a great deal of clues to inductive reasoning.. It is expected that observing these structural features in a quantitative way

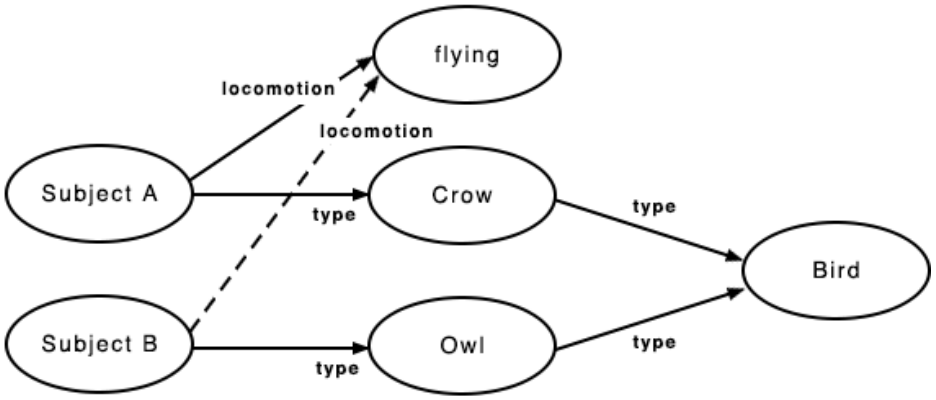

Figure 3.5: Assumption4

makes complicated property induction much easier. Though inferences in these examples are uncertain because there are not enough data in the knowledge base, the inferences can become more reliable as more data is available.

3.3 Structure-based Property Propagation

3.3.1 Overview

Let us recall our problem specification. Our objective is to find the following reasoning function f.

$$f_{\boldsymbol{D}}(s_Q) = \{(s_Q, p, o) \mid (s_Q, p, o) \text{ is inferred to hold in } \boldsymbol{D}\} \tag{3.7}$$

$$f_{\boldsymbol{D}}((s_Q, p_Q)) = \{(s_Q, p_Q, o) \mid (s_Q, p_Q, o) \text{ is inferred to hold in } \boldsymbol{D}\} \tag{3.8}$$

The first argument of f is queried resource. If only a subject is queried, it returns all the inferred triples which is supposed to hold in the dataset D. If the pair of a subject and a predicate is queried, the answer might be the set of inferred triples whose subject and predicate are queried one.

As an example, assume we have the dataset regarding the ecology of birds as shown in Figure 3.6. We denote a set of RDF triples in the figure as D. Prefixes of properties are omitted as appropriate. The figure shows the relationship between the diet of the birds and their habitats. Our task here is to infer the habitat of the Bird D from the properties they have. All the information about Bird D is that their type is Bird, and they eat minnows. Intuitively, it seems that their habitats may be the riverside rather than forests, because they eat fish. It is, however, difficult to quantify such intuition

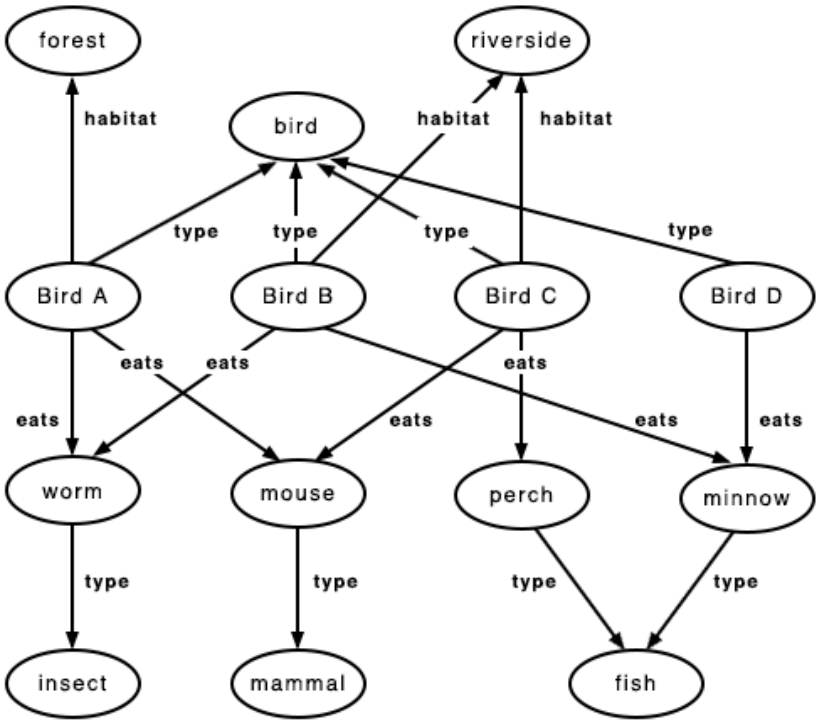

Figure 3.6: Example Dataset 1

without general rules such as " if something eats fish, its habitat may be the riverside.", and this is the main problem that inductive reasoning tries to solve.

Our methodology challenges this problem with structure-based property propagation. Though our method resembles the Tzompanaki's approaches [37] in that properties of other concepts propagate to the target object through edges, our approach only depends on the structure of the graph, which results in the generality and flexibility of the method. Assumption 1 in the previous section tells us we can project the properties of the concepts which share same properties of the target object. Since the concept Bird D has two properties `(eats, minnow)` and `(type, bird)`, it can inherit some properties from the nodes that also have the two properties. Figure 3.7 shows the subgraph composed of the nodes which have the property `(eats, minnow)` and their habitat property. There is a path in the figure that goes to Bird D from the node riverside through Bird B. The property `(habitat, riverside)` can be propagated to the Bird D through this path, because Bird D and Bird B are similar according to the Assumption 1.

We can assume the same property propagation for the property `(type, bird)`. In Figure 3.8, there are two paths from the riverside to the node Bird D, and one

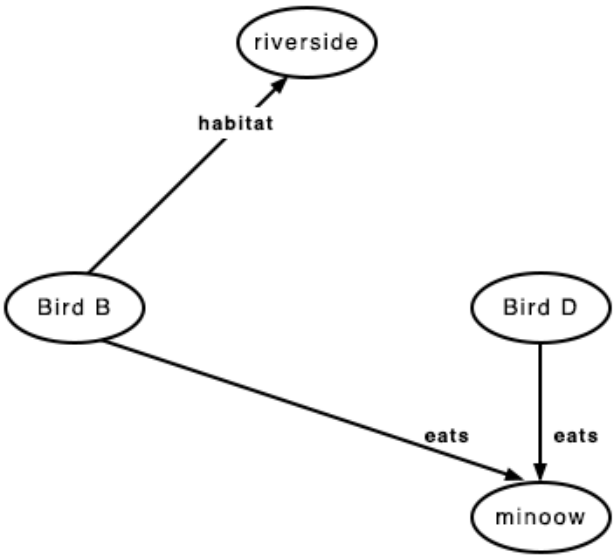

Figure 3.7: Subgraph1 - Propagation through the property (eats,minnow)

path going to the Bird D from the forest node. As with the previous case, the habitat properties of the relevant nodes are propagated in proportion to the number of paths. In this example, Bird D only has outbound properties, but same property propagation can be applied to inbound properties according to our second assumption.

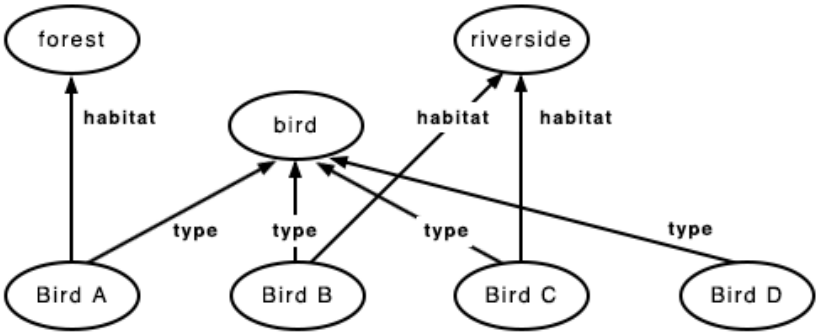

Figure 3.8: Subgraph2 - Propagation through the property (type,bird)

The simplest algorithm is to take the sum of the number of propagated properties, and choose the one with the highest votes as the most likely property. In this case there are 3 paths from the riverside to Bird D, and 1 path from forest to Bird D, hence the habitat of Bird D is estimated as the riverside. Assumption 1 is well reflected in this model, because the more properties two concepts share in common, the more paths between them appears, and then the properties of the object are more likely to

be propagated (In this example the property (habitat, riverside) of Bird B is counted twice instead of once). However, a downside is that this model treats every property equally. In fact, the property (type, bird) can be thought of as more typical than (eats, minnow) , because the property (type, bird) is held by more instances. Thus Bird B is better-characterized by the property (eats, minnow) than the other. Therefore we can introduce weights to discount more typical properties and emphasize properties which characterize the object better. We use weighted sums instead of mere summation of the number of properties. Without the weights, the propagated properties would be always the same as the majority of the concepts that the target object belongs to.

We can also consider assumption 3 in the previous section. For example, though Bird A and Bird D has no common properties except (type, bird) , they both share atypical properties with Bird B. Hence, using assumption 3, we can propagate Bird A's property (habitat, forest) to the Bird D in order to infer Bird D's habitat. Bird A and D in this case, however, are not considered as similar as the case where they have direct relationship, and thus the propagation strength is considered to be weaker than the propagation shown in the Figure 3.7. Propagation path from Bird A to Bird D is shown in the Figure 3.9.

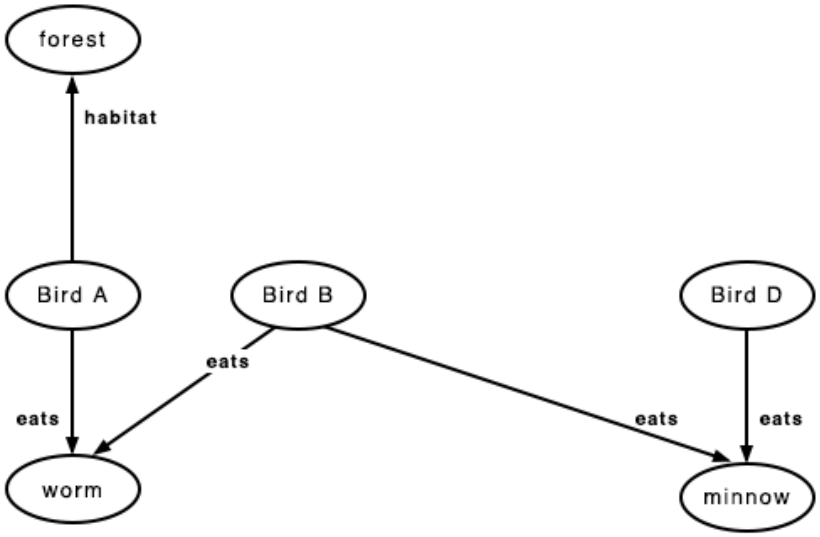

Figure 3.9: Subgraph3 - Propagation through

When considering assumption 4, we can make inferences in richer context. Bird C and Bird D have no shared properties other than (type, bird) , but they should be regarded as similar because they both eats fishes. In the simple model, the informa-

tion that they both eat fish is not considered to infer the value of the Bird D's habitat property. However, when we consider similarity between perches and minnows, the property of Bird C can be propagated to Bird D as shown in Figure 3.10. Similarly as the propagation which uses assumption 3, in this case the propagation strength is weaker than the first order propagation, because the connection between target objects is weaker than that of the case where they have direct relationship. Though it becomes computationally more intensive when taking higher order propagation into account, it can compensate the relations in the dataset especially when the dataset is not large enough.

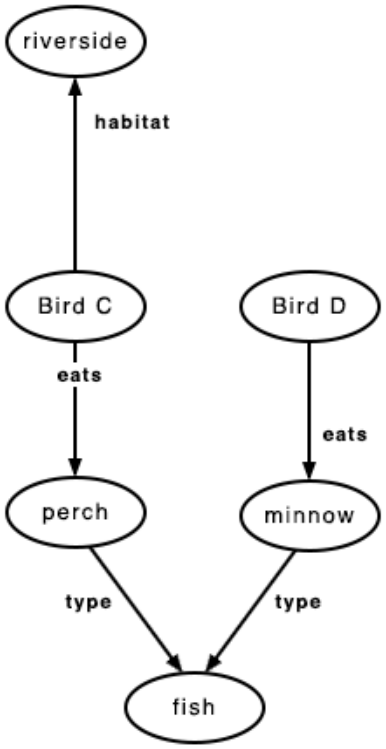

Figure 3.10: Subgraph4 - Propagation through

The example above illustrates how $f_{\boldsymbol{D}}((BirdD, habitat))$ is calculated, namely, it describes the case where the subject and the predicate are specified as query. The detailed explanation regarding mathematical formalization and other quantitative factors such as weights are provided in following sections.

3.3.2 Reasoning Function

Following formulas (3.9)-(3.11) describes the reasoning function $f_{\boldsymbol{D}}$ where the subject and the predicate are queried, which corresponds to the algorithm explained in Section 3.3.1. Every function symbol has $\boldsymbol{D}$ in their subscript. This means that every function implicitly assumes $\boldsymbol{D}$ is the knowledge base they are based on.

$$f_{\boldsymbol{D}}((s_Q, p_Q)) = \text{Activation}_{\boldsymbol{D}}(s_Q, p_Q) \tag{3.9}$$

$$\begin{aligned} Prop^n_{\boldsymbol{D}}(s_Q, p_Q, o) &= \sum_{(p_t,o_t)\in PO_{\boldsymbol{D}}(s_Q)} \left(\sum_{c\in CS^m_{\boldsymbol{D}}(s_Q,p_t,o_t)} Prop^{n-1}_{\boldsymbol{D}}(c, p_Q, o) \cdot w_{\boldsymbol{D}}(path) \right) \\ &+ \sum_{(s_t,p_t)\in PS_{\boldsymbol{D}}(s_Q)} \left(\sum_{c\in CO^m_{\boldsymbol{D}}(s_t,p_t,s_Q)} Prop^{n-1}_{\boldsymbol{D}}(c, p_Q, o) \cdot w_{\boldsymbol{D}}(path) \right) \end{aligned} \tag{3.10}$$

$$Prop^0_{\boldsymbol{D}}(s,p,o) = I_{\boldsymbol{D}}(s,p,o) = \begin{cases} 1 & \text{if } (s,p,o) \in \boldsymbol{D} \\ 0 & \text{if } (s,p,o) \notin \boldsymbol{D} \end{cases} \tag{3.11}$$

Propagation function $Prop^n_{\boldsymbol{D}}$ in (3.10) calculates how strongly properties of interest should propagate through the similar concepts to the queried concept. Activation function in (3.9) returns the set of triples that is likely to hold in the dataset using the result of propagation. Each function in the formulas is defined in the following subsections.

3.3.2.1 Activation Function

Activation function in (3.9) decides which concept indicated in the subscript gives the most likely property. In other words, it activates the properties which are likely to hold according to the propagated results. It is supposed to return the set of triples which is judged to hold in the dataset with respect to the concept in the subscript. Activation function also works for the pair of the predicate and the object. For example, when only the subject is queried, reasoning function is represented as formula (3.12). In this way we can describe the reasoning function in the unified form.

$$f_{\boldsymbol{D}}(s_Q) = \text{Activation}_{\boldsymbol{D}}(s_Q) \tag{3.12}$$

Hereinafter, it is assumed that the subject and the predicate are specified for the reasoning function if not otherwise specified.

In the example in Section 3.3.1, the activation function works in the same way as argmax, because it chooses the object which is propagated more than any other. Here

$\underset{k}{\text{argmax}}$ function returns the set of top k arguments which give the k biggest values. The following formula (3.13) shows the argmax activation function where the subject and the predicate are specified.

$$\text{Activation}_{\boldsymbol{D}}(s_Q, p_Q) = \{\, (s_Q, p_Q, o_R) \mid o_R \in \underset{o \in Obj^p_{\boldsymbol{D}}(p_Q)}{\overset{k}{\text{argmax}}} [Prop^n_{\boldsymbol{D}}(s_Q, p_Q, o)] \,\} \quad (3.13)$$

Where $Obj^p_{\boldsymbol{D}}(p_Q)$ is a set of possible objects for property p_Q. For example, in the example in Figure, 3.6, $Obj^p_{\boldsymbol{D}}(\text{habitat}) = \{\,\text{forest}, \text{riverside}\,\}$.

$$Obj^p_{\boldsymbol{D}}(p_Q) = \{\, o \mid (s, p_Q, o) \in \boldsymbol{D} \,\} \quad (3.14)$$

Generally, the activation function is not merely the same as argmax. Choice of activation function is also one of the parameters which determines the performance of the model. As an example of the activate function, We can activate the property whose value of propagation function is greater than the threshold t.

$$\begin{aligned} &\text{Activation}_{\boldsymbol{D}}(s_Q, p_Q) \\ &\quad = \quad \left\{\, (s_Q, p_Q, o_R) \;\middle|\; o_R \in \{\, o \mid Prop^n_{\boldsymbol{D}}(s_Q, p_Q, o) \geq t \wedge o \in Obj^p_{\boldsymbol{D}}(p_Q) \,\} \,\right\} \end{aligned} \quad (3.15)$$

This type of the function is also useful when only the subject is queried because it is reasonable to assume the subject has multiple properties rather than only one.

Activation function can also be non-deterministic. We can think of the triple of interest T as the probability variable following the probability which is proportional to the value of propagation function. In this instance, the activation function becomes as follows:

$$\begin{aligned} &\text{Activation}_{\boldsymbol{D}}(s_Q, p_Q) \\ &\quad = \quad \left\{ T \;\middle|\; P(T = (s_Q, p_Q, o_R)) = \frac{Prop^n_{\boldsymbol{D}}(s_Q, p_Q, o_R)}{\sum_o Prop^n_{\boldsymbol{D}}(s_Q, p_Q, o)} \wedge o_R \in Obj^p_{\boldsymbol{D}}(p_Q) \right\} \end{aligned} \quad (3.16)$$

Where the domain of T is $\boldsymbol{D}$, that is, the set of the all triples in the knowledge base.

3.3.2.2 Propagation Function

Before introducing propagation function $Prop^n_{\boldsymbol{D}}$ in the formula (3.10), it is necessary to define set functions which is integrated in the computation of the propagation. $PO_{\boldsymbol{D}}$ and $PS_{\boldsymbol{D}}$ in (3.10) are the resource functions which takes a concept and returns a set of resource pairs. $PO_{\boldsymbol{D}}(c)$ is the set of predicate/object pairs which the concept c holds as

their subject. $PS_{\boldsymbol{D}}(c)$ is returns the set of subject/predicate pairs that is true if c is their object. They are defined as follows:

$$PO_{\boldsymbol{D}}(c) = \{(p,o) \mid (c,p,o) \in \boldsymbol{D}\} \tag{3.17}$$

$$PS_{\boldsymbol{D}}(c) = \{(s,p) \mid (s,p,c) \in \boldsymbol{D}\} \tag{3.18}$$

$CS^m_{\boldsymbol{D}}(s,p,o)$ and $CO^m_{\boldsymbol{D}}(s,p,o)$ finds similar concepts as the subject s and the object o respectively. Similarity between two concepts is determined by the properties they share. $CS^1_{\boldsymbol{D}}(s,p,o)$ returns the set of concepts that also have the property (p,o).Likewise, $CO^1_{\boldsymbol{D}}(s,p,o)$ gives the set of concepts that have the same relationship (s,p). The concept sets $CS^m_{\boldsymbol{D}}$, $CO^m_{\boldsymbol{D}}$ in the case $m = 1$ are defined as follows.

$$CS^1_{\boldsymbol{D}}(s,p,o) = \{c \mid (s,p,o) \in \boldsymbol{D} \wedge (c,p,o) \in \boldsymbol{D},\ c \neq s\} \tag{3.19}$$

$$CO^1_{\boldsymbol{D}}(s,p,o) = \{c \mid (s,p,o) \in \boldsymbol{D} \wedge (s,p,c) \in \boldsymbol{D},\ c \neq s\} \tag{3.20}$$

In the example dataset in Figure 3.6, for example,

$$CS^1_{\boldsymbol{D}}(\text{Bird A}, \text{eats}, \text{mouse}) = \{\text{Bird C}\} \tag{3.21}$$

$$CO^1_{\boldsymbol{D}}(\text{Bird A}, \text{eats}, \text{mouse}) = \{\text{worm}\} \tag{3.22}$$

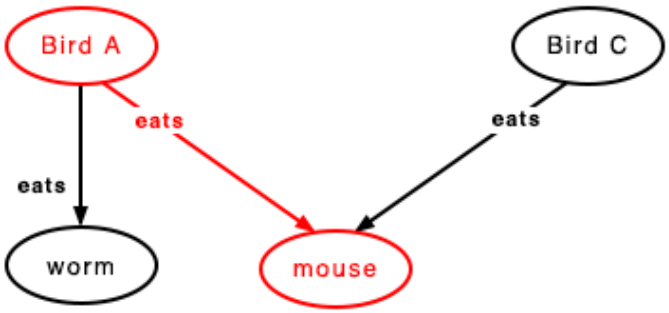

Figure 3.11: An example of concept sets 1

Figure 3.11 illustrates how $CS^1_{\boldsymbol{D}}$ and $CO^1_{\boldsymbol{D}}$ work using our example dataset. $CS^1_{\boldsymbol{D}}(\text{Bird A}, \text{eats}, \text{mouse})$ returns the set of the concepts which have structural connections to the subject of the triple (Bird A, eats, mouse) which is painted in red. Likewise, $CO^1_{\boldsymbol{D}}$ returns that of the object.

$m \in \mathbb{N}$ indicates the depth of the connection (see assumption 4 in Section 3.2). Larger m results in the larger set of concepts, because the result set may contain the

concept with weaker connection. Generally, concept sets are recursively defined as follows.

$$CS^m_{\boldsymbol{D}}(s,p,o) = \{c \mid (s,p,o) \in \boldsymbol{D} \wedge (c,p,o') \in \boldsymbol{D} \wedge o' \in CS^{m-1}_{\boldsymbol{D}}(o,p',o''),\ c \neq s\} \cup CS^{m-1}_{\boldsymbol{D}}(s,p,o) \quad (3.23)$$

$$CO^m_{\boldsymbol{D}}(s,p,o) = \{c \mid (s,p,o) \in \boldsymbol{D} \wedge (s',p,c) \in \boldsymbol{D} \wedge s' \in CO^{m-1}_{\boldsymbol{D}}(s'',p',s),\ c \neq s\} \cup CO^{m-1}_{\boldsymbol{D}}(s,p,o) \quad (3.24)$$

Similarly, in Figure 3.6, for instance,

$$CS^1_{\boldsymbol{D}}(\text{Bird D}, \text{eats}, \text{minnow}) = \{\text{Bird B}\} \quad (3.25)$$

$$CS^2_{\boldsymbol{D}}(\text{Bird D}, \text{eats}, \text{minnow}) = \{\text{Bird C}\} \cup \{\text{Bird B}\} = \{\text{Bird C}, \text{Bird B}\} \quad (3.26)$$

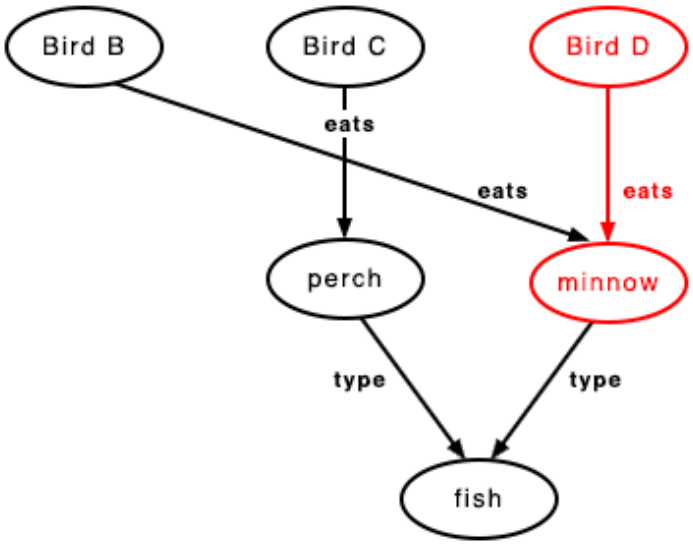

Figure 3.12: An example of concept sets 2

Figure 3.12 also illustrates how $CS^2_{\boldsymbol{D}}$ works differently from $CS^1_{\boldsymbol{D}}$. Both $CS^1_{\boldsymbol{D}}$ and $CS^2_{\boldsymbol{D}}$ return the set of concepts which connected to the subject of the triple painted in red, but the latter contains the concepts with up to the 2nd-order-depth connections.

The function $Prop^n_{\boldsymbol{D}}$ in the formula (3.10) is the propagation function, which passes the queried property to the queried subject. n is a non-negative integer, and increase in n corresponds to the increase in breadth of the path, while m corresponds to the depth (see Section 3.2 as for the degree of propagation). When $n = 1$, propagation function merely counts the number of properties with respect to each object multiplying

the weights, because $Prop^{o}_{\boldsymbol{D}}(s,p,o)$ just indicates the presence of the (s,p,o) in the dataset. When $n \geq 2$, it counts the propagated properties instead of the number of the direct properties. The weight $w_{\boldsymbol{D}}(path) \in [0,1]$ is multiplied the same time as n, which corresponds to the assumption that higher order propagation should be discounted because the relationship between the concepts is weaker than the first order propagation. As for the weight function, we will provide the description later in this chapter. The first term of the propagation function (3.10) corresponds to the propagation through outbound properties, and second indicates the inbound properties, which are introduced in the assumption 2 in the Section 3.2. According to the assumption 2, inbound properties can be treated in the same manner as outbound properties. Therefore, these two terms can merely summed up as the result of the propagation.

3.3.2.3 Weight Function

The weight function introduced in the last section takes the path between the concepts and returns the real number in $[0,1]$. The weight indicates how strongly the property in the path should be projected from one concept to the other. 0 means that one concept is not characterized by the property in the path at all, and 1 means the opposite.

The path is defined as the sequence of the edges and vertices which are all distinct and connected. Obviously, there can be multiple paths between two objects. For example in the Figure 3.7, there are two paths between Bird B and Bird D: one is the path [Bird B, $\overrightarrow{\text{type}}$,bird,$\overleftarrow{\text{type}}$,Bird D] and the other is [Bird B,$\overrightarrow{\text{eats}}$,minnow,$\overleftarrow{\text{eats}}$,Bird D]. We denote the path using square brackets as described. The arrows over the predicates describe the direction of the predicates. The arrow is supposed to point from subjects to objects.

One important function of the weight is to discount the influence of too typical property. In the Figure 3.7, there are only four subjects which have a property `(type,bird)` , but in the real dataset, there are many other relevant/irrelevant concepts. Suppose if there are 100 objects which have the property `(type,bird)` in the dataset and 80 of them have a property `(habitat,forest)` . In that case the Bird D's habitat is expected to be the forest, even though there are the data which suggest its habitat is the riverside. That means without weights, the inferred property is likely to be the same as that of the majority population. However, the knowledge that 80% of birds live in the forest should also be taken into account, because that is also necessary information to infer the property of the particular bird. Therefore the weight should be smaller when the property of interest has the larger number of members, but it should

be properly adjusted so that it would not be too small.

Second role of the weight function is to discount the effect of the distant relationship. Generally, the deeper(longer) the path becomes, the smaller the weight of the path should be. In the Figure 3.10, there is a path between Bird C and Bird D. The propagation strength through that path is weaker than the case where the two objects have direct relationship, and thus the weight of the such path should be discounted.

In conclusion, considering those two roles above, weight function is defined recursively as formula (3.27)-(3.30). In the formula, E and N means edges(predicates) and nodes(concepts) respectively.

$$w_{\boldsymbol{D}}([N_1, \overrightarrow{E_1}, path, \overleftarrow{E_2}, N_2]) = w_{\boldsymbol{D}}([N_1, \overrightarrow{E_1}, N_f]) \times w_{\boldsymbol{D}}(path) \times w_{\boldsymbol{D}}([N_2, \overrightarrow{E_2}, N_l]) \quad (3.27)$$

$$w_{\boldsymbol{D}}([N, \overrightarrow{E}, N']) = \frac{a}{f(|\{s \mid (s, E, N') \in \boldsymbol{D}\}|)} \quad (3.28)$$

$$w_{\boldsymbol{D}}([N]) = 1 \quad (3.29)$$

$$w_{\boldsymbol{D}}([N, \overrightarrow{E}, N']) = w_{\boldsymbol{D}}([N', \overleftarrow{E}, N]) \quad (3.30)$$

,where N_f and N_l denotes the first and the last node in the *path* respectively, and E without arrow in (3.28) means the label(the predicate) of the edge $\overrightarrow{E}$. We assume the labels of E_1 and E_2 are the same but their direction is opposite, because our assumption 1 and 4 says two subjects are not similar unless they share the same properties. Recursive computation reflects the effect of the depth discount, because multiplying weights results in smaller value. $\{s \mid (s, E, N') \in \boldsymbol{D}\}$ in (3.28) means the number of the subjects which holds the property (E, N'). Weight is inversely proportional to the f value of the number of members which has the property of interest. When $f(x) = x$, weight and the number of the members are merely in inverse proportion. a is a constant which ranges from 0 to 1.

Let us examine how the weight function works in our dataset shown in Figure 3.7. As shown in following formulas, the weight for typical and distant relationship is properly discounted as expected. In this example $a = \frac{1}{\sqrt{2}}$, and $f(x) = \sqrt{x}$.

$$\begin{aligned} & w_{\boldsymbol{D}}([\text{Bird B}, \overrightarrow{\text{type}}, \text{bird}, \overleftarrow{\text{type}}, \text{Bird D}]) \\ &= w_{\boldsymbol{D}}([\text{Bird B}, \overrightarrow{\text{type}}, \text{bird}]) \times w_{\boldsymbol{D}}([\text{bird}]) \times w_{\boldsymbol{D}}([\text{Bird D}, \overrightarrow{\text{type}}, \text{bird}]) \\ &= \frac{a}{\sqrt{4}} \times 1 \times \frac{a}{\sqrt{4}} = \frac{a^2}{4} = \frac{1}{8} \end{aligned} \quad (3.31)$$

$$\begin{aligned} & w_{\boldsymbol{D}}([\text{Bird B}, \overrightarrow{\text{eats}}, \text{minnow}, \overleftarrow{\text{eats}}, \text{Bird D}]) \\ &= w_{\boldsymbol{D}}([\text{Bird B}, \overrightarrow{\text{eats}}, \text{minnow}]) \times w_{\boldsymbol{D}}([\text{minnow}]) \times w_{\boldsymbol{D}}([\text{Bird D}, \overrightarrow{\text{eats}}, \text{minnow}]) \\ &= \frac{a}{\sqrt{2}} \times 1 \times \frac{a}{\sqrt{2}} = \frac{a^2}{2} = \frac{1}{4} \end{aligned} \quad (3.32)$$

$$
\begin{aligned}
& w_{\boldsymbol{D}}([\text{Bird C}, \overrightarrow{\text{eats}}, \text{perch}, \overrightarrow{\text{type}}, \text{fish}, \overleftarrow{\text{type}}, \text{minnow}, \overleftarrow{\text{eats}}, \text{Bird D}]) \\
&= w_{\boldsymbol{D}}([\text{Bird C}, \overrightarrow{\text{eats}}, \text{perch}]) \times w_{\boldsymbol{D}}([\text{perch}, \overrightarrow{\text{type}}, \text{fish}, \overleftarrow{\text{type}}, \text{minnow}]) \times w_{\boldsymbol{D}}([\text{Bird D}, \overrightarrow{\text{eats}}, \text{minnow}]) \\
&= w_{\boldsymbol{D}}([\text{Bird C}, \overrightarrow{\text{eats}}, \text{perch}]) \times w_{\boldsymbol{D}}([\text{perch}, \overrightarrow{\text{type}}, \text{fish}]) \times w_{\boldsymbol{D}}([[\text{fish}]]) \\
&\quad \times w_{\boldsymbol{D}}([\text{minnow}, \overrightarrow{\text{type}}, \text{fish}]) \times w_{\boldsymbol{D}}([\text{Bird D}, \overrightarrow{\text{eats}}, \text{minnow}]) \\
&= \frac{a}{\sqrt{1}} \times \frac{a}{\sqrt{1}} \times 1 \times \frac{a}{\sqrt{1}} \times \frac{a}{\sqrt{2}} = \frac{a^4}{\sqrt{2}} = \frac{1}{4\sqrt{2}}
\end{aligned} \tag{3.33}
$$

3.4 Extract General Rules

The first step of inductive reasoning is to find the properties or facts that is likely to hold in the dataset. As is stated in the definition of inductive reasoning (see Section 2.2), the next step of induction may be to find general rules from observing the instances and their relationships. In the example dataset (Figure 3.7) in Section 3.3.1, we predict the habitat of Bird D. By general rules we mean, for example, " if something eats fish, its habitat is likely to be the riverside" or these type of statement. Though property propagation tells us what the properties of the particular object are predicted to be, we might need another method to extract general rules out of the structure of the graph.

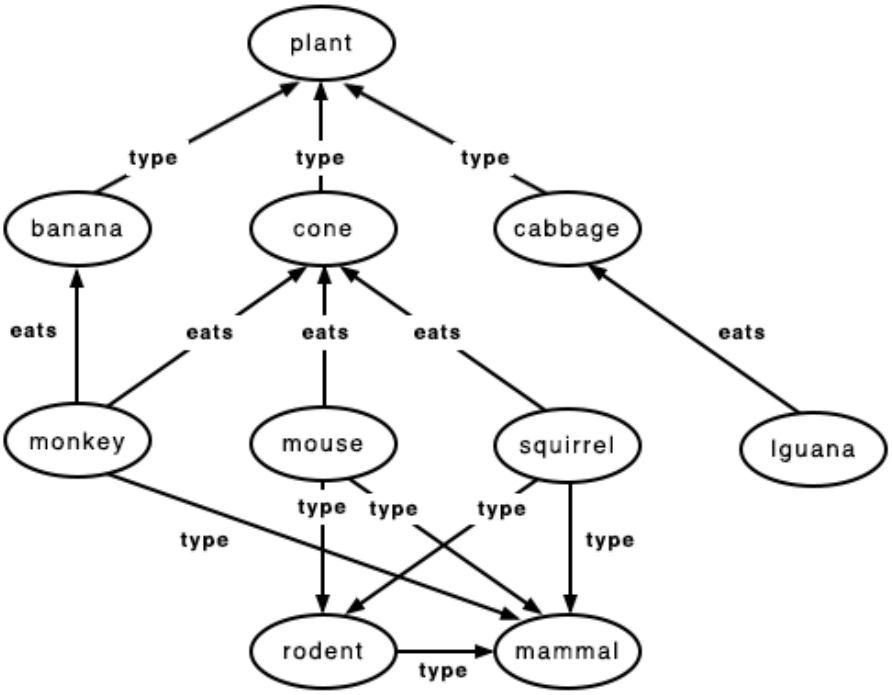

Figure 3.13: Example Dataset 2

In the dataset shown in Figure 3.13, it is easily recognized that all rodents are mammals, because mice and squirrels are all the concepts whose types are rodents, and both of them also have mammals as their type. This is appeared in the property connection of the concepts, that is, following relation holds:

$$\forall X \in \mathcal{C},\ [\text{rodent}, \overleftarrow{\text{type}}, X] \in \boldsymbol{P_D} \Rightarrow [X, \overrightarrow{\text{type}}, \text{mammal}] \in \boldsymbol{P_D} \tag{3.34}$$

,where $\boldsymbol{P_D}$ means the set of all paths existing in $\boldsymbol{D}$ In an opposite way, the reverse of (3.34) is not applied, which means all mammals are not rodents. In this way, we can extract the general rules by tacking the path starting from one property to the other property. Note that this can also be applied to longer path.

$$\forall X \in \mathcal{C},\ [\text{plant}, \overleftarrow{\text{type}}, \_, \overleftarrow{\text{eats}}, X] \in \boldsymbol{P_D} \Rightarrow [X, \overrightarrow{\text{type}}, \text{mammal}] \in \boldsymbol{P_D} \tag{3.35}$$

Where“ _” means blank concept, which represents any concept in the dataset. Formula (3.35) means “ all concepts that eats something whose type is plant are mammals”. This statement is not true in the dataset because the concept iguana eats plant, but it is not a mammal. The path may be of any length. Longer path can express the more complicated relationships, which might be one of the advantages of structure based approach.

In the real dataset, however, there are few rules which always hold. Even the obvious relationship such as“ all rodents are mammals” may not be always true in some datasets, because it is quite common that the data lacks some information (e.g., some information resources of rodents may not have the class property). Hence, we need a methodology to express to what extent the rule is applied to general cases. Probabilistic model may be one possible solution, namely, we can use conditional possibility to describe how likely the rule holds generally in the dataset. (i.e. the probability of the rule is 1.0 if the rule always holds.) Simply, the probability is calculated by counting the concepts that have the properties the paths indicate. Following formula (3.36) calculates the probability that the a concept which have the property described by the path $[N_1E_1, \cdots, E_n]$ may hold the property $[E_{n+1}, \cdots, E_{n+m}, N_{n+m}]$.

$$\begin{aligned} &P([E_{n+1}, \cdots, E_{n+m}, N_{n+m}] \mid [N_1E_1, \cdots, E_n]) \\ &\quad = \frac{\{X \mid [N_1E_1, \cdots, E_n, X, E_{n+1}, \cdots, E_{n+m}, N_{n+m}] \in \boldsymbol{P_D}\}}{|\{X \mid [N_1E_1, \cdots, E_n, X] \in \boldsymbol{P_D}\}|} \end{aligned} \tag{3.36}$$

For instance, in the Figure 3.13,

$$\{X \mid [\text{plant}, \overleftarrow{\text{type}}, \_, \overleftarrow{\text{eats}}, X]\} = \{\text{monkey}, \text{mouse}, \text{squirrel}, \text{iguana}\} \tag{3.37}$$

$$\{X \mid [\text{plant}, \overleftarrow{\text{type}}, \_, \overleftarrow{\text{eats}}, X, \overrightarrow{\text{type}}, \text{mammal}]\} = \{\text{monkey}, \text{mouse}, \text{squirrel}\} \tag{3.38}$$

Therefore,

$$P([\overrightarrow{\text{type}}, \text{mammal}] \mid [\text{plant}, \overleftarrow{\text{type}}, \_, \overleftarrow{\text{eats}}]) = \frac{3}{4} \tag{3.39}$$

This conditional probability expresses to what extent the property of interest is attributed to the property in the condition, though it does not necessarily imply a causal relationship between them. (3.39) means if something eats plants, the subject is likely to be mammals, but it is not always the case.

Structure-based property propagation is considered to be the method to gather general rules underlying the relations and give the certain result set according to the rules. That is because the paths in the property propagation is considered to be the same as the paths in the rule extraction, and more paths between the properties to infer result in stronger relations between the properties in both cases.

Chapter 4

Evaluation

4.1 Experiment and Dataset

We designed an experiment to see whether our methodology and inference system satisfies three criteria introduced in Section 1.3; correctness, performance, and applicability. In the experiment, some of the properties(triples) are removed from the dataset, and we let the system to make inferences on what eliminated properties used to be. We used actual foodweb dataset of Mondego River in Portugal which is obtained from Pajek datasets[38, 39]. The dataset contains 46 vertices and 400 arcs, which correspond to species and prey/predator relationships between the species. We added class information such as " Bivalvia", " Malacostraca", and " Bird" to each species. They are mainly from dbo:class property in DBPedia. Some vertices which cannot be categorized into one class such as " Zooplankton consumer" and " Macrofauna predators" are eliminated from the original dataset. Finally 35 species, 194 prey/predator relationships and 11 classes are used and transformed into RDF triples, since the original data format is not RDF triples.

In one trial, class information of randomly chosen several(3-5) species are removed, and they are inferred by the inference system. The trial is repeated several times and the numbers of correctly inferred triples and all inferred triples are counted. The reason we repeat the trial is that removing too much triples at once makes an inaccurate inference and this inference relies on the existing class information of other species. In summary, the task in the experiment is to predict the class of an animal or a plant from their food-web relations. This is truly a problem of property induction, because in the experiment the system finds underlying rules or results caused by them only from the instance information.

Here we can introduce precision and recall to evaluate the performance of the approach. Let N_{rem} be the total number of the removed properties, and N_{inf} denote the number of inferred properties, and N_{cor} be the number of correctly inferred properties. Then we can represent precision and recall as follows:

$$Precision = \frac{N_{cor}}{N_{inf}}, \quad Recall = \frac{N_{cor}}{N_{rem}}$$

Note that N_{inf} can be bigger than N_{rem} depending on the configuration of the activation function. Here high recall indicates that the system is more complete, and high precision indicates the soundness of the system. As is the case with many other circumstances, precision and recall here is trade-off, therefore it is reasonable to use the following F measure [40] F to evaluate the system's performance.

$$F = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

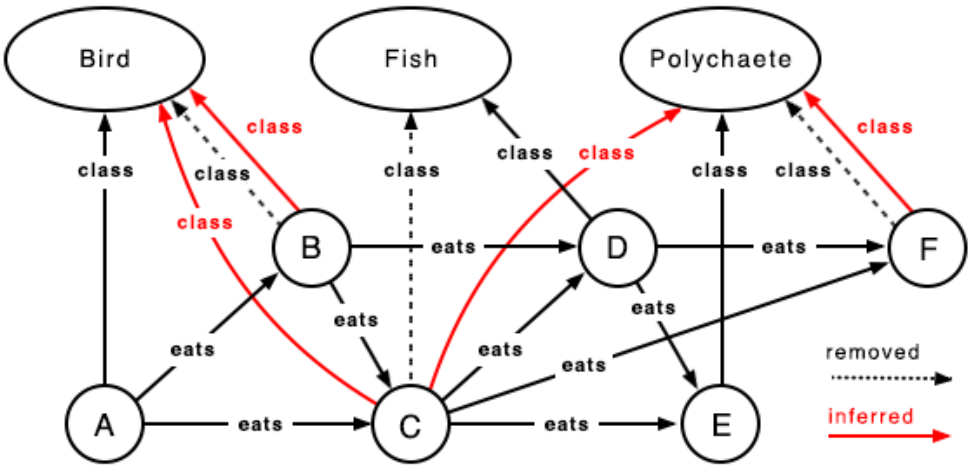

Figure 4.1: An example of precision and recall

Figure 4.1 shows an example of our experiment. The nodes A,B,C,D,E and F represent each species, and all of them belong to one class. Dashed arrows correspond to the removed properties, and red arrows denote the inferred properties. In the experiment the system is supposed to infer correct classes from the network of " eats" properties. In the example shown in Figure 4.1, four inferences are made, and out of three removed class properties, two are correctly inferred. Therefore, precision and recall in this case are as follows:

$$Precision = \frac{2}{4}, \quad Recall = \frac{2}{3}$$

4.2 Experimental Results

In this experiment, in order to find the best condition for our methodology we swept the parameters introduced in Chapter 3. There are following 5 factors and parameters which are open to change: activation function (3.12), propagation breadth n (3.10), propagation depth m (3.10), a in weight function (3.28), and f in weight function (3.28). As we change one of those parameters, other parameters shall be fixed to the default value. The table 4.1 shows the basic setting of parameters that other parameters are fixed to. In each experiment, 20 trials are conducted and the average recall / precision / f value for 20 trials are reported.

Activation	m	n	a	$f(x)$
$\underset{2}{\text{argmax}}$	1	1	$\frac{1}{\sqrt{2}}$	$\sqrt{x}$

Table 4.1: Basic parameter settings

Activation Function As for activation function, we used two activation functions introduced in Section 3.3.2.1: argmax (3.13) and threshold model (3.15). Let us recall the definitions of the two activation functions here.

$$\text{Activation}_{\boldsymbol{D}}(s_Q, p_Q) = \{\, (s_Q, p_Q, o_R) \mid o_R \in \underset{o \in Obj^p_{\boldsymbol{D}}(p_Q)}{\overset{k}{\text{argmax}}} [Prop^n_{\boldsymbol{D}}(s_Q, p_Q, o)] \,\} \quad (3.13)$$

$$\text{Activation}_{\boldsymbol{D}}(s_Q, p_Q) = \left\{ (s_Q, p_Q, o_R) \;\middle|\; o_R \in \{\, o \mid Prop^n_{\boldsymbol{D}}(s_Q, p_Q, o) \geq t \wedge o \in Obj^p_{\boldsymbol{D}}(p_Q) \,\} \right\} \quad (3.15)$$

The constant k in 3.13 is free parameter which decides how many properties to choose from the top of the list. Figure 4.2 shows the effect of k on recall, precision, and F measure.. Threshold t can be dynamically decided according to the maximum value of the propagation value as follows:

$$t = \max_o [Prop^n_{\boldsymbol{D}}(s_Q, p, o)] * s$$

Where $s \in [0,1]$ is a constant which decides latitude to activate the property. When $s = 1$, it works in the same way as $argmax^1$. The graph in Figure 4.3 indicates how s affects the results. Threshold t can also be taken constant value, which is also shown in the third graph in the Figure 4.4.

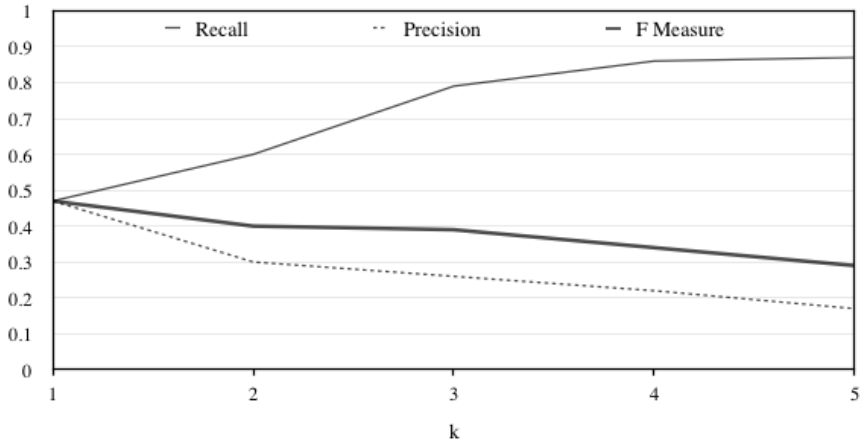

Figure 4.2: Argmax model with the parameter k

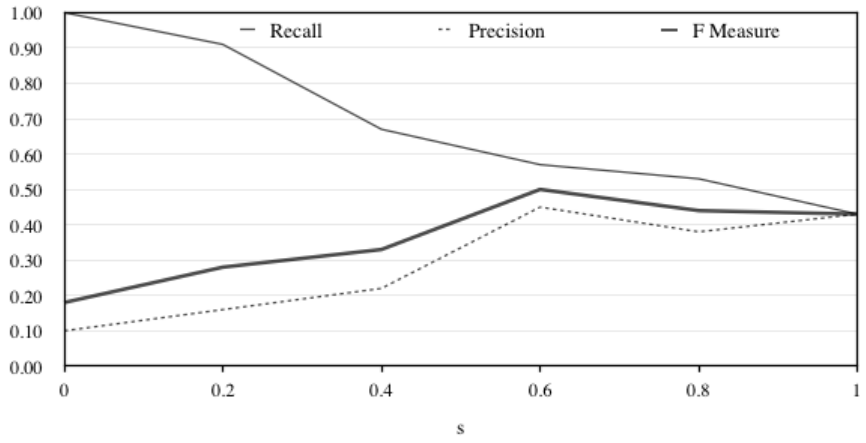

Figure 4.3: Threshold model with the parameter s

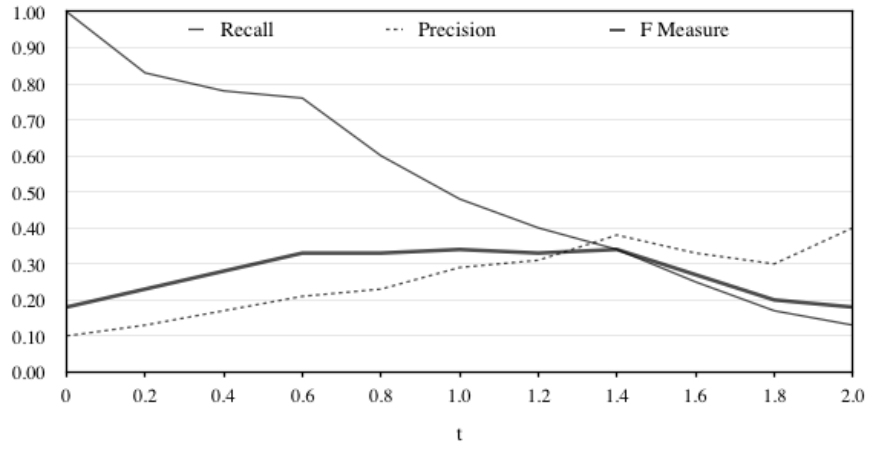

Figure 4.4: Threshold model with the parameter t

Since argmax function always activates the fixed number of properties, recall is always k times larger than precision (i.e. it infers k properties per one removed property). Threshold function with parameter $s = 0.6$ worked better than any other activation functions in terms of F measure. That is because this function works as follows: if propagation values of the top two or three properties are close to some extent, it activates all of them, and if there is enough difference between them, it activates only top-one property. That means the function can choose one property if it has enough " confidence", and it chooses more than one properties if it has no enough confidence about that. Fixed threshold does not seem to work well because propagation values vary depending on the number of the properties it propagates.

f in weight function Weight function is a function of the number of members which connects to the node of interest, because it should discount the effect of typical properties. Formula (3.28) shows how the number of members is integrated through the function f.

$$w_{\boldsymbol{D}}([N, \overrightarrow{E}, N']) = \frac{a}{f(|\{s \mid (s,E,N') \in \boldsymbol{D}\}|)} \tag{3.28}$$

To check the effect of f function in the weight function, we assume it takes a form of $f(x) = x^r$ and swept the value of r. The smaller r is, the less subject to the number of members the weight would be. It is reasonable to assume this functional form because our main purpose here is to confirm the assumption that a weight of a path should be discounted according to the number of members (i.e., $|\{s \mid (s,E,N') \in \boldsymbol{D}\}|$). Figure 4.5 shows the relationship between its value and the performance. The graph indicates

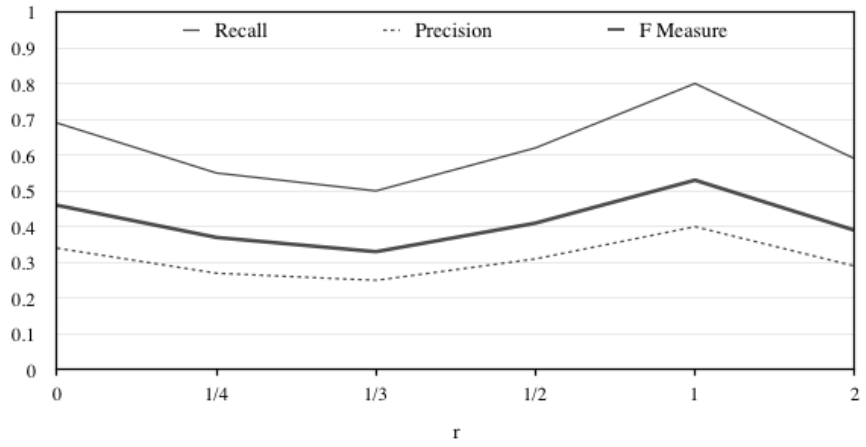

Figure 4.5: The results in terms of r where $f(x) = x^r$

that $f(x) = x$ made the best result, which implies that it is better to consider typicality of properties for property induction. However, since $f(x) = 1$ also makes relatively high performance, it is thought that f is not necessarily the crucial factor for the performance.

Propagation depth, breadth and *a* Since the constant a which appears in (3.28) matters only in the case of the higher order propagation, we made parameter sweeping on a together with the propagation breadth n and depth m. In theory, n and m can take any of natural number, but numbers larger than 2 cannot be used in practical application because it needs huge amount of computation. Hence, we tested only 1 and 2 for the value of n and m. Table 4.3 reports the F measure for the setting of n, m, a.

(n,m) \ a	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{\sqrt{2}}$	1
$(1,1)$	0.37	0.38	0.40	0.40
$(1,2)$	0.40	0.23	0.28	0.30
$(2,1)$	0.22	0.17	0.26	0.26

Table 4.2: F measure for the setting of n, m, a.

Contrary to expectations, propagation breadth n does not contribute to the performance for any value of a. That is probably because by considering the 2nd order realtionships, the number of properties to propagate explodes, and it results in making important features relatively less remarkable. Same thing can be applied to depth m, but it works under small a, because the propagation values of the 2nd propagations are properly discounted since it contains a^2 term in the weight function.

As for computation and execution time, Table 4.3 shows the average execution time of inference in each experiment (inference for 3-5 removed properties). As the table indicates, the higher-order propagation requires more intensive computation, and the propagation with higher-order breadth needs more computation than depth.

(n,m)	$(1,1)$	$(1,2)$	$(2,1)$
time(s)	0.92	41.0	139.5

Table 4.3: Average execution time

4.3 Overall Evaluation

Referring to the results of the experiments presented above, the parameter settings which yields the best performance are searched. In conclusion, the setting in the Table 4.4 turned out to be the best, which results in $Recall = 0.57$, $Precision = 0.54$, and $F = 0.55$.

Activation	m	n	a	$f(x)$
dynamic threshold model with $s = 0.9$	1	1	1.0	x

Table 4.4: Basic parameter settings

Now let us go back to the completion criteria presented in Section 1.3. In accord with the final result above, we made an overall evaluation in terms of the three criteria.

1. Correctness : Our inference system is able to infer more than half of the correct relationships under the proper parameter settings. Here correct relationships mean original relationships which actually hold in the dataset.

2. Performance : Although it works in relatively reasonable time (0.92 sec) for modestly sized dataset, there is still room for improvement regarding complexity and execution time especially for large dataset. Since the computational mechanism is naive, some improvements should be done on this point. Section provides some suggestions for it.

3. Applicability : Absolutely flexible and general. Our method can contain any data in the knowledge base as long as it can be represented as RDF triples. It can integrate different types of the data by transforming them to RDF. Validity for real dataset is verified by using actual dataset of foodchain in Mondego River [38, 39].

Chapter 5

Discussions and Analysis

5.1 Analysis of the Results

As a result, though our inference system finds important relationships in a dataset, the F measure remains at just over 50%. In this section we perform analysis of the results, experiments, possible improvements, and overall methodology.

First, it would appear that the size of the dataset and granularity of the category may affect performance. There are 11 classes, and each class only has 3.2 species on average; hence, if one node is removed, that means the system loses a large number of clues. Considering the properties removed from the dataset in the experiments, it might have been better to have retained more species for one class.

Although we only used the " eats " property to predict classes of species because not many data sources are available, it should be possible to use more varied data. For instance, if we could include information about habitats, locomotion and other knowledge, we might be able to improve performance by making affinities even clearer. As our methodology is particularly suitable for integrating the diverse data difficult to describe as a vector because of the versatility of the RDF format, we hope to consider these experiments in future works.

In addition, looking at the results for inference we found that the inference system tends to propose the same classes regardless of the kinds of species (e.g. it tends to propose the class Malacostraca more frequently than others). That means the prediction results are affected by the properties that the majority holds. In other words, the bigger the population the property in the dataset has, the higher its propagation value tends to be. Although our weight function is designed to mitigate this effect, since propagation value is summed up with respect to all similar concepts, the effect of ma-

jority may be more influential than weights if the number of concepts is large. Perhaps the performance would be better if some improvements were made in this area.

Contrary to our expectations, when $m = 2$, the performance is not improved. That may be because depth connections are not effectively used in the dataset which we used this time. It is not hard to imagine there is little relation between "species which eat animals which eats Animal X, " for example. In cases like this, depth and breadth may simply cause additional complexity and may not bring benefits depending on the predicates it contains. Datasets that make much use of depth connections are those with more hierarchical relations such as taxonomic trees. The results might be more interesting if we could conduct more experiments with other types of data.

Finally, we had the several experiments to find the best parameter settings. In practical use, one does not have to find the best parameters through experiments, because most of parameters such as f and a are not dependent on the dataset. However, activation function should be properly chosen according to the purpose and use, because whether precision (or recall) is more important than the other totally depends on the tasks.

5.2 Comparison

In accordance with what we have explored so far, we made qualitative comparisons with other methods for inductive reasoning and machine learning. This comparison can also clarify advantages of structure-based property propagation.

Comparison with the Feature Based Induction Model As explained in Section 2.2.2, FBIM expresses each category as a fixed-length vector whose element corresponds to a feature. This encoding is problematic especially for describing complex relationships between categories (concepts) such as the food web in our experiment, because describing the food web of N species in feature encoding, for example, requires the dimensionality to be at least $2N$ (what species to eat, and to be eaten by), which causes sparseness. Conversely, it is easy to express featural information in RDF format, because we can just add new triples to the dataset. In addition, if the domain of interest changes, FBIM needs to use different feature vector encoding because it uses a fixed-length vector; however, our approach can be used across different domains.

Comparison with Label-Based Property Propagation There are methods called

property propagation as introduced in Section 2.3. They use particular properties (such as“ composed of”) to find relevant concepts and infer the missing property. That means they rely on label information which is preliminarily specified by a human, and ignore all other labels which are not so specified. On the other hand, structure-based property propagation only focuses attention on structures and connections between vertices. Thus, it considers all relationships in a continuous manner rather than particular connections chosen in an arbitrary manner. Moreover, whereas the label-based approach has to change the label to use according to the domain (e.g. it uses the “ composed of” predicate for the dataset of buildings and constructions, but uses “ genus” or “ family” for the one about animals), our methodology can be generally applied to any domain without changes. Therefore, structure-based property propagation is considered to be a general form of label-based propagation.

Comparison with the Neural Network Although the neural network is a general and powerful approach for various problems in the real world, it also has the same problem with the encoding scheme as FBIM. In our approach, it is quite simple to add new knowledge into the dataset (e.g. we can easily add habitat information of animals to a food chain dataset by adding some more RDF triples to the dataset). However, in the case of neural networks, the dimensionality of input has to change if new concepts or new properties are added, and thus the whole network should be trained from the first. Encoding with vectors is not really suitable for dealing with complex relationships with multiple types of predicates. In terms of flexibility and expressivity, structure-based property propagation is more advantageous than a neural network. Additionally, the neural network typically fails to explain causal relationships between input and output, and it is difficult to know which factors contribute to a particular result or outcome. The proposed method, however, can explain correlativity by using conditional probability in the way explained in Section 3.4. On the other hand, the advantages of the neural network over the property propagation method are its computation and execution time. Although neural network takes time to train, inference finishes immediately as long as the training is properly done. Since our method has no training process, inference takes a while, especially for large datasets.

Comparison with Bayesian Model In the Bayesian induction model, there is a representation problem as stated above. The primitive Bayesian model by Heit[4] explained in Section 2.2.3 needs 2^N hypotheses if there are N species, which results in

very sparse probability distribution. Kemps and collegues' model [3] still needs vectors with at least the same dimensionality as the number of categories, and it does not solve the fundamental problem. Our methodology does not even need to use binary vectors representing features, let alone find prior distributions as most of the Bayesian models do.

Chapter 6

Conclusion

6.1 Summary of Contributions

In this research, we have explored the methodology for property induction which can integrate various type of background knowledge in reasoning. Based on the four assumptions made in Section 3.2, structured-bsaed property propagation method is desined in Chapter 3. We implemented inference system using the proposed method, and made experiments to confirm how it works. The results are evaluated and analyzed in both Chapter 4 and Chapter 5. In this section, we sum up two contributions that this research has made. Following list shows our main contributions.

1. We proposed to use RDF graphs to represent features that cannot be expressed by numerical values. RDF can express qualitative aspects of things while vectors are limited in expressivity. Many of the advantages explained in the comparison section come from this point. Approaches which use vector representations are inflexible and limited in representing complicated relations between concepts.

2. As the method of making inferences on RDF graphs, we designed structural property propagation. Since RDF is too expressive, it has been hard to make inductive inferences on the representation. Our method can convert qualitative attributes of RDF graphs to quantitative value, which enables us to make an inductive reasoning. It does not rely on previous knowledge about label information, while most of the previous works using RDF depends on label information. Structure based approach can be applied to any other graphical representation, not only RDF.

As a whole, we succeed in proposing the methodology to integrate various types of background knowledge in an inductive reasoning. This is our original research objective, and we implemented the inference system and proved its validity by designing experiments. If problems shown in the next section are solved, proposed method can achieve the level where we can use it practically. Then it allows more rich representation of the background knowledge, which cannot be achieved by vector representations. It will work effectively in the area of question answering, information retrieval, recommendation engine and many other applications.

As Sloman[14] suggests, inductive reasoning has something in common with human intuition. This means that thinking about inductive reasoning is necessary in understanding human intuition. We would be very happy if this research proves helpful for understanding the mechanism of human intuition and intelligence.

6.2 Future Works

There are several improvements that could have been made in our research if more time was available. In this section we suggest possible future works that would be necessary to apply our methodology to even more practical and complex problems.

First, since sufficient amount of experiments to confirm the effectiveness of our method and range of application were not be done, some more experiments with various daata should have been conducted. As written in Section 5.2, for esxample, we should have done experiments using datasets with more number of species for each class, or the one including additional predicate such as “ habitat” in addition to “ eats” relations. As for the method to calculate correlations between properties using the path given in the graph which is explained in section 3.4, we could not design an experiment to see how it works, hence supplementary experiments are desired as future works. In introduction, we introduced a recommendation engine as possible applications. It is desirable to have an experiment to infer the user’s preference based on their relationship related to their preferences in order to confirm that our inference method works for the real problems

As stated in Section 5.2, the limitation of our method would be the amount of computation and the working speed. It takes a long time in case of a large dataset because our method requires to scan all the RDF triples and propagate all the properties concerned when making inferences. It is possible to improve this problem of execution time if we can separate the learning phase from the inference phase by constructing

a network which would represent the structural similarity between the concepts in the dataset in advance. Alternatively, one can adopt a dynamic programming using a memory-based data structure which keeps a propagation value of each path. Therefore, improvement on this point might be a possible future work, because this is quite important especially for practical use.

RDF is inadequate to describe a more rich background knowledge and context. Although RDF is suitable for describing properties or attributes, it is not able to express complicated knowledge such as “ John is going to Boston by bus.”, in other words, it does not have an equal level of expressivity as sentences. A graphical representation has a stronger expressivity: Conceptual Graph, J.F.Sowa[5]. Conceptual Graph is as expressive as First Order Logic, which means it can describe almost all things that can be expressed by natural language. Although the methodology we propose here this time cannot be directly applied to Conceptual Graphs, structure-based approach can be applied to Conceptual Graphs because their effectiveness is verified in this research. Using Conceptual Graphs allows us to make broader and more general inferences, which might be one of the possible future works. Since the more expressive the representation becomes, the harder the inference on it would be, using Conceptual Graphs for inductive reasoning is more challenging but rewarding.

Bibliography

[1] L.Harvety, K.Koedinger, D.Klahr, et al. Solving inductive reasoning problems in mathematics: Not-so-trivial pursuit, Cognitive Science, 2000.

[2] B.K.Hayes, E.Heit, H.Swendson Inductive Reasoning, Wiley Interdisciplinary Reviews: Cognitive Science, 2010.

[3] C.Kemps, J.Tenenbaum, Structured statistical models of inductive reasoning, Psychological review, 2009.

[4] E.Heit, A bayesian analysis of some forms of cognition, Rational Models of Cognition, 1998.

[5] J.F.Sowa, Conceptual Graph, Handbook of Knowledge Representation, 2008.

[6] R.Brachman, H.Levesque, Knowledge Representation and Reasoning - The Morgan Kaufmann Series in Artificial Intelligence, Morgan Kaufmann Publishers, pp.17, 2004.

[7] R.Davis, H.Shrobe, P.Szolovits, What is a Knowledge Representation?, AI Magazine, 14(1):17-33, 1993.

[8] S.Decker, P.Mitra, S. Melnik, Framework for the semantic Web: an RDF tutorial , IEEE Internet Computing, 2000

[9] P.Pauwels, D.Van Deursen, R.Verstraeten, et al. A semantic checking environment for building performance checking , Automation in Construction, 2011.

[10] S.Harris, Garlik. SPARQL 1.1 Query Language, World Wide Web Consortium, 2013.

[11] D.L.Medin, J.D.Coley, G.Storms, et al. A relevance theory of induction, Psychonomic Bulletin & Review, 2003.

[12] Osherson, D.N.Smith, E.E.Wilkie, et al. Category-based induction, Psychological Review, Vol 97(2), 185-200, 1990,

[13] B.Rehder, R.Hastie. Category coherence and category-based property induction, Cognition, 113-153, 2004,

[14] S.A.Sloman, Feature-Based Induction, Cognitive Psychology, 1993.

[15] H.Zhu, J.Zhong, J.Li, et al. An Approach for Semantic Search by Matching RDF Graphs, AAAI, 2002.

[16] J.Hau, W.Lee, J.Darlington, et al. A Semantic Similarity Measure for SemanticWeb Services, Proc Workshop on Web Service Semantics, 2005.

[17] J.Poole, J.A.Campbell, A novel algorithm for matching conceptual and related graphs, Lecture Notes in Computer Science, 1995.

[18] R.Oldakowski, C.Bizer. SemMF: A Framework for Calculating Semantic Similarity of Objects Represented as RDF Graphs Radoslaw, ISWC, 2005.

[19] M.M.y.Gomez, A.L.Lopez, A.Gelbukh, Information Retrieval with Conceptual Graph Matching, Database and Expert Systems Applications, 2000.

[20] V.D.Blondel, A.Gajardo, M.Heyman, et al, A Measure of Similarity Between Graph Vertices: Applications to Synonym Extraction and Web Searching, SIAM Review, 2004.

[21] C.M.Keet, Closed World Assumption, Encyclopedia of Systems Biology, 2013.

[22] J.Lehmann, R.Isele, M.Jakob, DBpedia A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. Semantic Web Journal, Vol. 6 No. 2, pp 167195, 2015.

[23] J.F.Sowa, Knowledge Representation - Logical, Philosohical, and Computational Foundations, Brooks/Coke, pp. 476-488, 2000.

[24] J.F.Sowa, Conceptual graphs as a universal knowledge representation, Computers & Mathematics with Applications, 1992.

[25] J.F.Sowa, Conceptual structures, Information Processing in Mind and Machine. Addison-Wesley, 1993.

[26] S.Hensman, J.Dunnion, Automatically building conceptual graphs using VerbNet and WordNet, International symposium on Information and communication technologies, 2004.

[27] S.Hensman, Construction of Conceptual Graph representation of texts, 2004.

[28] L.Chang, Y.Yu Learning to Generate CGs from Domain Specific Sentences, Iccs 2001, 2001.

[29] S.Gelman, E.Markman Categories and induction in young children, Cognition, 1986.

[30] M.Stickel, A Prolog-like inference system for computing minimum-cost abductive explanations in natural-language interpretation, Annals of Mathematics and Artificial Intelligence, 1991.

[31] Masaaki Kanakubo, 不確実な推論, Sizuoka Institute of Science and Technology, Faculty of Comprehensive Informatics, Comptuer System Department and Intelligent Interaction Laboratory.
http://www.sist.ac.jp/ ~kanakubo/research/reasoning_kr/uncertain_logic.html

[32] D.Gentner, A.Markman, Structure Mapping in Analogy and Similarity, January 1997 American Psychologist. 1997.

[33] B.Falkenhainer, K.D.Forbus, The Structure-Mapping Engine*, AAAI-86 Proceedings, 1986.

[34] J.F.Sowa, A.K.Majumdar, Analogical Reasoning, AI Magazine, Harvard Law Review, 1993.

[35] D.Sperber, D.Wilson, Relevance: Communication and Cognition, 1986.

[36] H.Ghosh, S.Chaudhury, A.Gupta, et al. Knowledge Based Query Interpretation For Multimedia Retrieval In An Open Distributed Environment, Knowledge Based Computer Systems, 2000.

[37] K.Tzompanaki, M.Doerr, M.Theodoridou, et al. Reasoning based on property propagation on CIDOC-CRM and CRMdig based repositories, CRMEX 2013, 2013.

[38] V.Batagelj, A.Mrvar, Pajek datasets, Food-webs, 2006

[39] Patricio.J, (In Preparation) Master's Thesis. University of Coimbra, Coimbra, Portugal.

[40] J.Makhoul, F.Kubala, R.Schwartz, et al. Performance Measure for Information Extraction, Broadcast News Workshop '99 , 1999.