Using Limited Common Sense Knowledge to Guide Knowledge Acquisition for Information Agents

Xiaoying (Sharon) Gao and Leon Sterling

Intelligent Agent Laboratory
Department of Computer Science
The University of Melbourne
Parkville 3052, Australia
{xga, leon}@cs.mu.oz.au
Phone: (03) 9344 9101
FAX: (03) 9348 1184

Abstract

Acquiring knowledge from the Internet to improve performance is a very challenging task for information agents. This research aims to use common sense knowledge to guide the learning. A limited common sense knowledge base is handcrafted into an agent using extended Definite Clause Grammar (DCG) rules. A Classified Advertisement Search Agent (CASA) reading advertisements in HTML has been built and has been tested on real estate advertisements. CASA learns new knowledge about suburbs automatically, and acquires statistical information about prices that can be used for making suggestions to modify user queries, and to fill in default parameters.

Keywords: Knowledge acquisition, common sense knowledge, information agents, Web mining

1 Introduction

Information agents, which are also called Internet agents and/or Web agents, are intelligent software programs that autonomously navigate the Internet to gather information. As an environment, the Internet consists of a huge amount of data, information and knowledge. It is a challenge to enable the agents not only to search for information, but also to acquire knowledge from the Internet during searches. Ideally, the agents will learn from their experience and improve their performance.

Knowledge acquisition from the Internet is difficult. Most information on the Internet is written in natural language with a human reader in mind. A lot of common sense knowledge is needed to understand the information and acquire knowledge from it. Common sense knowledge is the knowledge that everybody knows, and is regarded as the foundation of natural language understanding and machine learning [Lenat and Guha 1990]. Our research is predicated on the belief that common sense knowledge plays an important role for guiding Internet knowledge acquisition.

This paper outlines the construction of a limited common sense knowledge base and a learnable agent: the Classified Ad Search Agent (CASA). The paper is organized as follows. Section 2 gives an overview of our particular research domain, online classified ads, and specifies the tasks. Section 3 discusses common sense knowledge for the domain and the tasks. In the next three sections, knowledge representation, the parsing and learning algorithms and agent architecture are covered. Our results are presented in Section 7, followed by a summary of related work and conclusions.

2 Task Description

We have chosen search through online classified ads as the domain for our research. The type of query we would like an information agent to answer is “I want to rent a flat which is near the city and cheap”.

Answering this query is not obviously handled by keyword search. To answer the query, the classified ads search agent needs knowledge, such as, which places are near the city, and the market prices for renting a flat. Similarly, to suggest a car or bicycle to buy, the agent would need brand information along with a range of properties of the vehicles and their normal prices. Investigating classified ads requires us to deal with everyday life concepts, such as jobs, cars, houses, which are not very well defined compared with mathematical or scientific domains.

The information presented in classified ads is in a semi-structured form intermediate between natural language and databases. On the one hand, it reduces the complexity of natural language understanding. On the other hand, it must cope with diversity and ambiguity in contrast to data mining from databases.

The size of the knowledge the classified ad agent needs is huge. Currently, most of the knowledge needs to be handcrafted. The agent’s task is to learn part of the knowledge from the Internet when it reads through the ads. This paper focuses on two main tasks: one is to identify some knowledge units from the text, for example, to identify car brand names from car ads and to identify suburb names from real estate ads. The second task is to acquire statistical information on some knowledge units, particularly on price.

3 Limited Common Sense Knowledge

This section discusses the common sense knowledge needed for the domain and tasks. The nature of the knowledge is the major influence on our choice of knowledge representation and design of the agent, as discussed in Sections 4 and 6.

We particularize the two learning tasks for searching real estate ads. A knowledge unit identification task is to find all the suburb names and build a suburb name database. A knowledge unit statistical information learning task is to calculate the average rent price for renting a certain size property. The basic common sense knowledge is that the suburb name and the rent price can be found in the real estate ad itself. In order to guide the learning, the main common sense knowledge needed is how to recognize an individual real estate ad and identify the suburb name, price, size, etc., from the ad.

The knowledge to identify real estate ads:

- One ad usually comes in one paragraph.
- One ad is usually more than 20 and less than 800 characters long.
- For a property, the location (suburb name usually), price, size and type are more important than the other features.
- Every real estate ad usually contains the four important parts above and some other extra information. One ad usually starts with a suburb name, followed by price, size, type in any order and ends with the contact address or phone number.
- One property can only have one location, one price, one size and one type. If two of them are found then they belong to two different properties.
- If more than one property is advertised in a single ad, the different properties usually come on different lines.

The knowledge to define a suburb name:

- A word usually appears in all capital letters.
- A suburb name is more than 3 and less than 20 characters long.
- Usually the ads are indexed by the suburb and most often the suburb is the first word of an ad.
- A suburb name in an ad is often followed by price and size in any order.
- There is at least one suburb in a single ad.
- Suburb names usually are not common words, such as HOUSES, TO LET, etc.
- Suburb names usually do not contain abbreviations, such as ST, RD, OSP, BIR, etc.

The knowledge to identify price:
• A price starts with a $ (in Australia) and is followed by a number, usually an integer no more than 4 digits (for renting), also usually followed by a time unit.
• One time unit is \textit{per week} and another is \textit{per month}. The translation between the two is based on that there are 7 days a week and 30 days a month.
• If the time unit is omitted, the default is \textit{per week}.

Similarly, the agent needs common sense to identify size and type, etc.

It can be seen that the main body of the common sense knowledge is about the recognition of knowledge units (suburb, price, size) and their environment (real estate ad) from an unrestricted text. For example, in the domain of real estate ads, the knowledge mainly answers questions such as what real estate ads look like, how a price is presented and what makes a word a suburb name.

4 Knowledge Representation

Our research approach is knowledge-based. Common sense knowledge is handcrafted, using rules and facts, into a common sense knowledge base. This knowledge is used to parse the information on the Internet and to learn new knowledge from the information. The knowledge representation formalism we have used is extended definite clause grammar rules (DCGs) [Sterling and Shapiro 1994].

4.1 Knowledge about text

Information agents process text. We have found it useful to distinguish the format of the text from its content and to represent the text in three levels. Accordingly, we have developed knowledge representation techniques to describe the three levels using both format and content.

4.1.1 Two components of text: format and content

Every piece of text can be divided into two parts: format and content. Format is the visual presentation of the words and the length of the words. The visual presentation of the words is usually expressed with HTML tags, character template, punctuation and special characters. Content is the symbols (words) representing word meaning, word order, and relations between the words. An example of the format and content of text “<B> Hello World! </B>” is given below.

\begin{verbatim}
Text
   "<B> Hello World! </B>"

Format
   HTML tags
      "<B>"   "</B>"
   Character template
      "clll clllll"
      (‘c’ represents capital letter; ‘l’ represents lower case letter; ‘n’ represents number)
   Punctuation
      "!"
   Length
      10

Content
   "hello world"
\end{verbatim}

4.1.2 Three levels of text: word, knowledge unit and context
Our research identifies three levels of text: word, knowledge unit and context levels. The word level, the lowest level, consists of the actual symbols that make up the text. The middle level consists of knowledge units, which are the smallest units that have independent meanings and usually are made up of words or phrases in some order. For example, “$200 per week” is a knowledge unit called price, while “$” “200”, “per”, “week” are individual words. The context level is the highest level and represents a larger independent meaning. For example, “Parkville, 2 br flat, $200 pw, gas cooking, …ph: 9329 8737” is a context called real estate ads. One example of the three levels of real estate ads is shown in Figure 1.

![Figure 1: The three levels of real estate ads: word, knowledge unit and context](image)

To represent a word, its symbol and character template are sufficient. To represent a knowledge unit is much more complex. A knowledge unit is linked to a group of words in certain order and format. The content of the knowledge unit can be represented by a group of words and their synonyms, abbreviations and similar words in a certain order. Sometimes, it is defined by the words that should be included and the words that should not be included. The format of the knowledge unit can be represented using a format template. For example, a template with a beginning format, an end format, and a maximum and minimum length. The content of context is usually defined by a collection of knowledge units and its format is similar to the format of a knowledge unit. The main common sense knowledge presented in Section 3 is the description of knowledge units and contexts.

4.2 Knowledge representation: Extended DCG rules

DCG rules [Sterling and Shapiro 1994] are grammar rules that can be understood by Prolog. They are in a special format and a Prolog interpreter translates them to Prolog clauses automatically. Traditionally, they are used to write grammar rules for natural language processing. They can have variables, and normal Prolog clauses can be integrated, so they are very expressive. Here is one example of representing the knowledge unit price in DCG rules:

```prolog
price(X) --> [$], number(X1), timeunit(N), {X is X1*N}.
number(X) --> [X], {integer(X)}.
timeunit(1) --> [perweek]; [pw]; [pwk].
timeunit(7/30) --> [permonth]; [pcm]; [p,c,m].
timeunit(1) --> [].
/* There are three components in a price: $, an integer number and a unit. There are two kinds of units: per week and per month. The translation rate between the two is 1:7/30. The default time unit is per week. */
```
Standard DCG rules have limitations. DCG rules do not separate format from content. They are quite good at representing content but it is not easy to adapt them to represent format. For example, it is hard to represent that the suburb is a word made up of all capital letters. To solve this problem, a set of predicates such as \texttt{character\_template/1, \ tags/1, begin\_with/1, begin\_after/1, end\_with/1, end\_before/1, max\_length/1, min\_length/1} are created to represent format. Using these predicates, we are able to add format constraints to the knowledge base:

\begin{verbatim}
format(paragraph, end_with([tags("<p"), tags("<hr"), tags("<p")])). /* A paragraph ends with one of the paragraph tags or line tag */

format(suburb, begin_with([character_template("ccc")])).
format(suburb, end_before([character_template("*l"), tags("<")])).
format(suburb, max_length(20)). /* A suburb name is made up of capital letters. A suburb name has less than 20 characters */
\end{verbatim}

Standard DCG rules are usually sufficient to represent the content of knowledge units and words where knowledge units are non-terminals and the words are terminals. However, it is not easy to represent complex content such as the content of contexts. One main reason is that they only represent ordered and continuous structure. If the content of \textit{real estate ads} is represented using

\begin{verbatim}
real_estate_ads --> suburb(L), price(P), size(S), type(T).
\end{verbatim}

The four knowledge units have to appear continuously in this exact order. Most ads would fail the parsing because the four knowledge units may come in any order and a wide variety of extra information may appear among them. The standard DCG rules are too restricted to represent free order structure with discontinues constituents. To solve this problem, a set of predicates such as \texttt{include\_freeorder/1, include\_anyof/1, exclude/1} are created, which can handle free order structures with discontinuous constituents as variables represented in lists. Then some complex content can be represented as:

\begin{verbatim}
content(real_estate_ad, include\_freeorder([suburb, price, size, type])). /*A real estate ad consists of suburb, price, size and type. */

content(suburb, exclude([common\_words, abbreviations])). /*A suburb usually do not contain common words or abbreviations */
\end{verbatim}

In addition, a special DCG rule

\begin{verbatim}
unrecognized(Unknown) --> [Unknown].
\end{verbatim}

is added to the end of the common sense knowledge base. It will parse any word successfully and so it can be used to skip any extra information. This is particularly useful for parsing discontinuous constituents.

To sum up, the main part of the knowledge, especially the content of knowledge units and words, is represented in the DCG rules. On top of this there is a set of facts representing format constraints and complex content.

5 Parsing and Learning Algorithms

For our research, text parsing is the foundation of knowledge acquisition and information search. The text has to be parsed into structured data and then the data can be used for further learning and search.

5.1 The parsing algorithm

The parsing consists of two main parts: format parsing and content parsing. Format parsing reads through the text and is only interested in format symbols such as tags. It locates a part of the text using format constraints and is usually used to cut the fragment for further content parsing. It is the content parser that analyses the text, identifies the knowledge units and contexts, and saves them internally.

5.1.1 Format parsing
Format parsing has four steps:

a) The text is read into a text string.
b) The text string is parsed character by character to a string consisting only of format symbols. The text string and the format string have the same length. This parsing is only used when parsing detailed character templates such as the suburb character template. Here is one example of a text string and its format string:

The Text String:
FLOWERDALE $90 pw 1 BR cottage, wood htg, <BR>acre (steep)<BR>WHITTLESEA $105 pw 1 BR unit, lge kitchen/lnge<BR>9716 2302<BR><P ALIGN=RIGHT><A><HR SIZE=1></P>

Its Format String:
cccccccccccc $nn ll n cc llllllll, lll lllll, <BR>llll llllll<BR>cccccccccccc $nnn ll n cc lllll, lll llllllll/lillll<BR>nnnn nnnn<BR><P ALIGN=RIGHT><A><HR SIZE=1></P>

c) Set the begin and end pointer by checking the format constraints on either the format string (for character template parsing only) or the text string.

For example, to locate a paragraph, the begin pointer is set at the beginning of the text string, the end pointer is set at the first paragraph tag or line tag. To locate a suburb name, the begin pointer is set at the start of a “ccc” template and the end pointer is set at a “*l” template and the length in between must be shorter than 20 characters.

d) Use the begin and end pointers to read from the text string. The format string is only used to set the pointers. The output text is always read from the text string.

5.1.2 Content parsing

For the knowledge units and words represented in standard DCG rules, content parsing is a straightforward top down parsing. To parse the content when there are discontinuous constituents in a relatively free order is much more complex. We will only address the complex one in detail.

As mentioned in Section 4, free order structure with discontinuous constituents is represented in lists. The list is usually made up of knowledge units (e.g. [suburb, price, size, type]). If the list consists of other free order structures with discontinuous constituents, the algorithm is recursively called.

The parsing algorithm is as follows:

a) Get the list that represents the free order structure with discontinuous constituents. This is called the goal list in the following part.
b) Read the text on which the structure is going to be parsed to a word list that consists only of words without format symbols.
c) If the goal list is empty, the content is parsed successfully and stop.
d) Read one word from the word list.
e) Choose one knowledge unit from the goal list.
f) If the word marks the beginning of the knowledge unit, parse the rest of the word list using standard DCG rules of the knowledge unit.
5.2 Learning algorithms

The learning algorithms for two learning tasks are summarized.

5.2.1 Learning knowledge units:

As an example, the algorithm for learning suburb names is given in detail.

a) Format parsing to locate a paragraph.
b) Format parsing to find a possible suburb name.
c) Check whether it is in the current suburb database. If yes, go back to a) to locate another ad; If no, go to the next step.
d) Content parsing to check whether the possible suburb name is followed by the structure [price, size]. If no, it is ignored. If yes, the possible suburb name is represented in DCG rules
   
   suburb(suburb_name_learned) --> ["suburb_name_learned"].

   and inserted into the classified ads knowledge base.
e) Go back to a) and continue this process until the whole document is parsed.

5.2.2 Statistical analysis on price

The concept to be learned, such as the average price for certain size of property, is predefined and hard coded. So the main task is to identify the price and size pairs.

a) Format parsing to locate one paragraph;
b) Content parsing the structure [suburb, price, size, type] to locate an ad;
c) Find price and size pairs;
d) Calculate the average price for every size;

The learning depends heavily on parsing. The parsing is based on the common sense knowledge given in Section 3.

6 The Classified Ads Search Agent (CASA)

To test our ideas, we have built an online Classified Ads Search Agent (CASA). It is a knowledge-based, goal-oriented system. It searches for suitable ads according to a user’s query. The agent runs periodically and reports the search results to the user via email. Most importantly, as it navigates the Internet and reads through the ads, its knowledge base grows larger and its search performance is improved.

The agent mainly consists of three parts: the parser, the learning machine and the knowledge base. Its two processes, the search and the learning, are integrated. The agent’s architecture is shown in Figure 2.

The parser consists of one format parser and one content parser. The learning machine has two parts: the knowledge unit identifier and the knowledge unit statistical information calculator. There are two parts in the knowledge base. One is the common sense knowledge base that contains the representation of knowledge units and contexts. The other is the classified ads knowledge base in which some of the knowledge such as suburb names is automatically learned and some knowledge, which is currently hard to learn, such as the URLs of classified ads sites, is handcrafted. The knowledge is written in DCG rules and facts.
In the search process, the user’s query and the Internet document are parsed through the parser into structured data. If the query data and document data match then the document is given as the output. If they do not match, the new document is downloaded and another round of search is started.

In the learning process, the parsed document data are given as input to the learning machine. The parsing and learning is based on the knowledge provided by the knowledge base. The knowledge acquisition result, which is the knowledge learned, is translated either to DCG rules or to facts and inserted into the classified ads knowledge base.

![Figure 2 The agent architecture](image)

### 7 Knowledge Acquisition Results

Currently, CASA is only capable of searching real estate ads. The preliminary learning results are a suburb database and a price and size comparison table.

#### 7.1 Suburb database

The learning results based on 4 sets of sample data are given in detail. Each set of data contains about 200 local real estate ads downloaded from 7 online classified ads web pages. The results on one set of data are shown in Table 1. The precision and recall is calculated by comparing the learned database with human manual learning results. For this particular domain and learning task, precision is more meaningful and more important than the recall because one suburb may appear many times in the same document and our goal is to get a suburb database with high accuracy rate. For the other three sets of data, only precision is calculated by comparing the learned database with a local suburb name list. The results are shown in Table 2.

It can be seen that the precision rate is over 85%. Manual work is still needed to edit the database to delete false suburb names. Analyzing the agent’s learning performance, one major problem is that the agent fails to recognize suburbs with abbreviation prefixes, such as "St. KILDA", "Mt. Eliza", "M.PONDS". Future research is needed to improve learning accuracy.

To exactly recognize the suburb is absolutely crucial in real estate ad search because one ad usually starts with a suburb name and ads for different properties are usually separated by their suburb names. After the suburb database is built, if one suburb in the database appears in one ad, it can be identified immediately. This improves search accuracy and efficiency.
Comparing a learned database with a static database, the advantages are obvious. The learned database is built dynamically, so it has no geographical limitations and can therefore be enlarged and adapted easily. In contrast, using a static suburb database, a program that works in Melbourne may not work in Sydney.

<table>
<thead>
<tr>
<th>Data Group</th>
<th>The number of real estate ads*</th>
<th>The number of learned suburbs (NResponse)</th>
<th>Recall (NCorrect/NKey)</th>
<th>Precision (NCorrect/NResponse)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>58</td>
<td>25</td>
<td>22/33 = 66%</td>
<td>22/25 = 88%</td>
</tr>
<tr>
<td>2</td>
<td>44</td>
<td>14</td>
<td>11/15 = 73%</td>
<td>11/14 = 79%</td>
</tr>
<tr>
<td>3</td>
<td>51</td>
<td>26</td>
<td>24/30 = 80%</td>
<td>24/26 = 92%</td>
</tr>
<tr>
<td>4</td>
<td>46</td>
<td>21</td>
<td>18/25 = 72%</td>
<td>18/21 = 86%</td>
</tr>
<tr>
<td>5</td>
<td>53</td>
<td>25</td>
<td>21/28 = 75%</td>
<td>21/25 = 84%</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>5</td>
<td>5/5 = 100%</td>
<td>5/5 = 100%</td>
</tr>
<tr>
<td>7</td>
<td>16</td>
<td>4</td>
<td>4/9 = 44%</td>
<td>4/4 = 100%</td>
</tr>
<tr>
<td>Total</td>
<td>277</td>
<td>120</td>
<td>105/145 = 73%</td>
<td>105/120 = 88%</td>
</tr>
</tbody>
</table>

*The number of real estate ads is given to show the size of the data.

Table 1 Suburb name learning results (one set of data)

<table>
<thead>
<tr>
<th>Data Set</th>
<th>The number of paragraphs*</th>
<th>The number of suburb learned (NResponse)</th>
<th>Precision (NCorrect/NResponse)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>224</td>
<td>113</td>
<td>97/113 = 86%</td>
</tr>
<tr>
<td>2</td>
<td>224</td>
<td>108</td>
<td>102/108 = 94%</td>
</tr>
<tr>
<td>3</td>
<td>213</td>
<td>93</td>
<td>82/93 = 88%</td>
</tr>
</tbody>
</table>

*The number of paragraphs is given to show the size of the data instead of the number of ads. There is at least one real estate ad in one paragraph.

Table 2 Suburb name learning results (3 sets of data)

7.2 The price and size comparison table

The price and size comparison table gives price ranges for 1 bedroom to 6 bedroom properties. After the system runs on the set of data used in table 1, the price table represented as facts in the knowledge base is as follows:

average_price(example(22), price(100.681816), size(1)).
average_price(example(63), price(141.0952), size(2)).
average_price(example(84), price(198.880951), size(3)).
average_price(example(7), price(202.857147), size(4)).
average_price(example(2), price(300.0), size(5)).
average_price(example(0), price(0), size(6)).

/* The fact “average_price(example(X), price(P), size(S)).” means that:
   based on X examples, the average price for S bedroom properties is $P per week */

Currently, the price table is updated periodically as the agent runs. This ensures that the agent has the current information. But there are two problems: one problem is that the system always calculates the current average price based on old records and this may result in losing the currency of the data. The other problem is that the agent cannot distinguish the new ads from those already parsed because it is too expensive to keep records of all the ads parsed. So if the same ad is read, the agent will still count it. These two problems can easily be solved by separating the learning and the search process and treating learning as a training process. However, this conflicts with our main idea, which is to enable learning while searching. Future work will address the problem by periodic reinitialization and by limiting and controlling the
learning time period. Also future work needs to be done to check prices and reject abnormal prices. Statistical information on price, considering other knowledge units such as type and suburb, will also be explored.

The price table can be used to fill in default prices and to give suggestions for modifying the query. For example, if a user is looking for a 2 bedroom flat and does not know the normal price, the average price can be filled in as the default parameter. Another example is that a user may give a price that is too high or too low and the agent can find nothing after searching through some documents. Then the agent is able to modify the query by either increasing or decreasing the price according to the average price. These functions are important features of a “smart” information agent.

8 Related Work

Most learnable information agents are interface agents that learn from users’ feedback [Balabanovic and Shoham 1995] or users’ browsing behavior [Lieberman 1995] to detect users’ interests and help the browsing or search. In contrast, our research is concerned with acquiring knowledge from the documents on the Internet, which is a quite different problem.

Two similar agent systems are ILA (Internet Learning Agent) [Perkowitz and Etzioni ] and Shopbot [Perkowitz et al. 1997]. Both are agents that learn from Internet documents using a small amount of domain knowledge. But their learning methods and purposes are quite different from our work. ILA uses a context-free algorithm to translate information sources into its own internal concepts. Shopbot learns how to shop at online stores by making example queries and generalizing them based on the response. Our agent learns knowledge units and their statistical information based on common sense knowledge representing both text format and text content.

Regarding the application domain, two projects on online classified ads search are Ineeda at http://www.newsclassifieds.com.au/ineeda/, which is a project by Australian Newsclassifieds, and Cybergold at http://www.cybergold.com , which is an American classified ads service. Unfortunately, not many details are available on their web pages.

In terms of common sense knowledge, one large-scale project is CYC [Lenat 1997; Lenat 1995]. The project aims to provide a common sense knowledge base for all kinds of AI systems, especially for natural language processing (NLP), machine learning (ML) and machine translation (MT) systems [Lenat and Guha 1990]. Its recent work [Guha and Lenat 1994; Mayfield and Finin 1995] shows that common sense knowledge is essential for agent cooperation and agent communication. Some other large scale projects on common sense are WordNet [Miller 1995; Miller et al. 1993], EDR [Yokoi 1995], and ThoughtTreasure [Mueller 1996]. Instead of building a large knowledge base, our research is trying to use only very small amounts of common sense knowledge to guide the learning of information agents.

Our research has benefited from work in different contexts. Using DCGs to represent biological descriptions can be found in [Taylor 1995]. One of the conclusions is that Prolog is very suitable for text representation and matching. Prior research in our lab was influential, especially CiFi [Loke et al. 1996], which is a rule-based agent that searches for citations on the Internet.

9 Conclusions and Future Work

The main idea of our research is coding common sense knowledge to enable information agents to learn from their environment. This research summarizes the limited common sense knowledge useful for learning. The knowledge is represented in DCG rules and facts and integrated into the agent. An example agent in the real estate ads domain is able to learn a suburb database and an average price table from online ads based on very limited common sense knowledge. This research shows that common sense knowledge plays an important role in guiding knowledge discovery for information agents.
This work combines information search and knowledge acquisition in an agent system. The agent built not only searches for information but also acquires knowledge from the Internet and uses this knowledge to improve its performance. The knowledge learned from the Internet is concurrent and dynamic. This is particularly important for information agents, because no human handcrafting could match the rapid growth and changes of the Internet. Knowledge acquisition ability gives our information agent a powerful feature that distinguishes it from other knowledge-based information agents.

The experiment also shows that even though the Internet is unrestricted and dynamic, it still has the potential to be used in knowledge acquisition. Both agent-based approaches and knowledge-based approaches play important roles here.

Future work will be done to adapt our agent CASA to a larger domain and to test the knowledge representation and learning algorithm for broader learning tasks. Learning strategies for acquiring and amending part of the common sense knowledge from the Internet will be explored. This will reduce the handcrafting efforts and is an important step towards generalizing the knowledge acquisition method and adapting it to other domains.

References


