MESSAGE RETRIEVAL AND CLASSIFICATION FROM CHAT ROOM SERVERS USING BAYESIAN NETWORKS

Debbie Zhang, Simeon Simoff and John Debenham
Faculty of Information Technology, University of Technology, Sydney

Abstract: Chat rooms and newsgroup on the internet is a valuable, and often free of charge, source of information. In this paper, a design of smart chat room bots that automatically retrieve and filter on line messages is proposed. The design is based on internet technology and Bayesian Networks. Technical details of connecting to and retrieving data from web based chat room servers are presented. A Naive Bayesian network classifier is implemented using frequency of the keywords that mostly appear in the selecting messages as input features. A prototype of such a message classification system has been implemented. It has been trialed on detecting investment related messages from four Australian chat room sites.

Key words: Information retrieval, Bayesian network, web mining

1. INTRODUCTION

It is widely understood that much intelligence can be gathered from sources which are publicly available from internet. In particular, this raises the need for some organizations to surveillance the information on the internet, particular informal sources such as chat rooms. As the amount of traffic through chat rooms each day is large and occurs in real time, the process of monitoring internet messages is expensive and in some sense time critical. To assist the monitoring process, a data classification process is employed. The chat room messages are classified into selected class and

non-selected class. Messages in the selected class require further processing. While it is time consuming for humans, the classification process does not require deep understanding of the text and thus can be efficiently performed by computers.

Software robots, so-called bots, which are smart software tools for retrieving useful data from internet web sites, can be applied to search and categorize chat room messages in an efficient way [1]. Bots have great potential in text mining, which find patterns in enormous amounts of data. Bots can save labor as they persist in a search, refining it as they go along. Intelligent bots that can make decisions based on past experiences are the perfect way to perform the methodical searches to uncover information from internet. This paper presents the methods of design and implementation of smart chat room bots using Bayesian Networks for message filtering. Technical details of connecting to and retrieving data from chat room web servers are described. A case study of using the developed chat room bots to classify the messages retrieved from four Australian chat room sites is conducted.

This paper presents the methods of design and implementation of smart chat room bots using Bayesian Networks for message filtering. Technical details of connecting to and retrieving data from chat room web servers are described. A case study of using the developed chat room bots to classify the messages retrieved from four Australian chat room sites is conducted.

2. DATA RETRIEVAL

2.1 Connecting to chat room web servers

The first step to develop a bot is to connect to the chat room web server that has been chosen. Java socket API is chosen to use to communicate to the web server since it is platform independent and comprehensive for network application development [2].

Java Sockets are a mechanism for communication over the Internet. Since web servers normally listen on TCP port 80, a socket with the IP address of the chat room can be constructed if the IP address of the chat room is know. Java encapsulates the concept of an ordinary TCP socket with the class Socket, and the concept of a server socket with the class ServerSocket. Data stream (document) that is input to or output from a socket is encapsulated in Java using the InputStream and OutputStream classes, respectively.

Documents on the Internet web server can also be retrieved by using Java URL class. The standard identifier for a document on the Internet is its Universal Resource Locator (URL). The URL object is constructed with the URL address of the chat room. Interacting with the URL requires that the connection is established with the Web server that is responsible for the document identified by the URL. The TCP socket for the connection is constructed by invoking the openConnection method on the URL object. This method also performs the name resolution necessary to determine the IP address of the Web server. This method returns an object of type URLConnection. The connection to the Web server is requested by calling connect on the URLConnection object. If the URL specifies the http protocol, then the URLConnection will actually be an object of the subclass HttpURLConnection which has additional methods specific to HTML documents.

2.2 HTML document analyzer

The documents retrieved from Web servers are HTML documents. The structure of the HTML document from each web site is different from each one. Therefore, retrieving messages from a chat room URL has to involve certain degree of customization. To minimize the customization level, a HTML document analyzer package is developed to allow bots to be built on top of it. Also, the difficulty of classifying a message increase dramatically if it contains HTML tags. The HTML document analyzer should provide the function to easily remove HTML tags in the document.

Sun provides a HTML parser package javax.swing.text.html.parser for parsing HTML documents. However, Swing's HTML parser fails to parse HTML documents into HTML objects. A similar approach presented by Somik Raha etc. was implemented to provide a fast real time parser for parsing HTML documents into HTML objects [3]. This HTML parser provides the following advantages:

- Handles formatted HTML as much as possible the way MS IE and Netscape do
- Dissects the document in an array of tag objects and string node objects
- Provides the value of a tag's attributes and comments, and
- Can also handle XML documents.

The HTML parser contains HTMLParser, HTMLReader, HTMLTag, HTMLTagScanner, HTMLTag and HTMLTagScanner six major classes.

The HTMLParser class is the access point to the package. It allows the user to register or remove the scanners for the type of tags. HTMLParser can be instantiated either with a string representing the URL to be parsed, or the html file on the hard disk. It can also be instantiated with a HTMLReader

that constructed by an inputStream class. This flexibility is important for supporting two types of web server connections discussed in section 2.1.

HTMLReader builds on java.io.Bu? eredReader, providing methods to read text from a character-input stream. It analysis the HTML document line by line using the registered HTML tag scanners and creates an array of HTML tag objects and string node objects.

HTMLTag is the parent interface for the specific tags, which represents a generic tag entity. Each type of tag needs to develop a corresponding HTMLxxxTag that contains the methods to set and get attributes of the tag.

HTMLTagScanner is the abstract base class of specific tag scanner classes. It contains evaluate() method, which evaluates a tag, and scan() method that constructs the tag object if it can handle it. Again, each type of tag requires a HTMLxxxScanner to implement its own scan() and evaluate() methods.

3. DATA FILTERING

The aim of this project is to categorize the chat room messages according to user requirements. Naive Bayesian classifier based on probabilistic learning method which is widely used for email message filtering is employed to classify the messages [4],[5].

A Bayesian network is a directed acyclic graph that compactly represents a probability distribution. The nodes in the graph represent random variables that associate with a conditional probability table. Network directed links signify direct causal influences between connected nodes. The posterior probabilities of nodes are computed in Bayesian networks [6][7][8].

A Naive Bayesian network is a simple structure Bayesian network that has the class node as the parent node of all other nodes. Naive Bayesian networks have been used as an effective classifier for many years. It is easy to construct. Also, the classification process is very efficient. To avoid the computational difficulty of exploring a previously unknown network and the speed requirement of this project, a Naïve Bayesian network is chosen for the message filtering algorithm.

Bayesian networks extend the concept of deterministic modeling by taking into account uncertainties. Outputs and inputs are not stated as fixed variables but whenever possible as probability distributions. This is particular suitable in the case of classifying messages where precise relationship between the message classes and message features of that class is difficult to achieve. In considering the specific problem of classifying certain type of messages, a database of the keywords that mostly appear in the selecting type of messages can be used as the feature variables. Langley

et al.'s study shows Naive Bayesian network has surprisingly outperformed many sophisticated classifier when the features are not strongly correlated, which is the case in this application [9].

From Bayes' theorem and the theorem of total probability, the probability that a message m with vector $\vec{x} = \langle x_1, ..., x_n \rangle$ belongs to class c is:

$$P(C = c \mid \vec{X} = \vec{x}) = \frac{P(C = c) \cdot P(\vec{X} = \vec{x} \mid C = c)}{\sum_{k \in \{c, \sim c\}} P(C = k) \cdot P(\vec{X} = \vec{x} \mid C = k)} \quad (1)$$

As the Naïve Bayesian classifier assumes that $x_1, ..., x_n$ are conditionally independent given the class C, which equation (1) can be simplified to:

$$P(C = c \mid \vec{X} = \vec{x}) = \frac{P(C = c) \cdot \prod_{i=1}^{n} P(X_i = x_i \mid C = c)}{\sum_{k \in \{c, \sim c\}} P(C = k) \cdot \prod_{i=1}^{n} P(X_i = x_i \mid C = k)} \quad (2)$$

$P(X_i \mid C)$ and $P(C)$ can be estimate by the frequencies of the training data.

4. CASE STUDIES

A prototype that implements the proposed methods has been built. Building a bot based on the HTML document analyser package described in section 2.2 is simple and straight forward. Adding a new chat room site only involves a few lines of coding. Thus the development time has been reduced to minimum.

Case studies using the prototype to classify on-line chat room messages into investment related category and non-investment related category were conducted. Data used in the experiments are real-world data collected from four Australian popular chat room sites. Two hundred messages were collected from each site. One fourth of the messages were used for testing and the remainder were used to build the network.

A database that contains 47 keywords such as "share", "price", "buy" etc. that mostly appear in investment related messages are used as feature variables. The frequencies of keywords occurred in the investment related messages and non-investment related messages are counted. The posterior probabilities of the test data were calculated using equation 2. A threshold

value was chosen for the classifier. Preliminary results have proved that the above concept is feasible and effective in retrieving and classifying internet messages. It is also found the entire data retrieving and classification process very efficient and suitable for real-time implementation.

5. CONCLUSIONS

This paper presents the methods to design and implement smart chat room bots using Bayesian Networks for message filtering. To reduce the level of customization of source code for retrieving data from different web site, a HTML document analyser has been implemented. This approach greatly increases the development efficiency. A prototype of the proposed system has been built. Case studies using the prototype were conducted. Experiment results show the proposed methods are feasible and efficient.

REFERENCES:

1. www.botspot.com
2. Hughes, M., Shoffner, M., Hamner, D.: Java network programming: a complete guide to networking, streams, and distributed computing, Greenwich (1997).
3. http://htmlparser.sourceforge.net
4. Sahami, M., Dumais, S.,Heckerman, D., and Horvitz, E.: A Bayesian Approach to Filtering Junk E-mail. Proc. of the AAAI'98 Workshop on Learning for Text Categorization, Madison, Wisconsin (1998).
5. Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Sakkis, G., Spyropoulos, C. and Stamatopoulos, P.: Learning to filter spam e-mail: A comparison of a naive Bayesian and a memory-based approach. Workshop on Machine Learning and Textual Information Access, 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (2000) pp.1-13.
6. Cowell, R., Dawid, A., Lauritzen, S., and Spiegelhalter, D.: Probabilistic Networks and Expert Systems, Springer (1999).
7. Pearl J.: Probabilistic Reasoning in Intelligent System, Morgen Kaufmann (1988).
8. Jordan, M.: Learning in Graphical Models, MIT (1999).
9. Langley, P., Iba, W., and Thompson, K.: An Analysis of Bayesian Classifiers. Proceedings of AAAI-92 (1992) pp. 223-228.