Computer and Information Science
www.ccsenet.org/journal.html
Vol. 2, No. 2
May 2009

A Novel Data Mining Platform Design with Dynamic Algorithm Base

HebiaoYang, Yukun Chen & Rengang Hou
School of Computer Science and Telecommunications Engineering, Jiangsu University
Zhenjiang, 212013, Jiangsu, China
Tel: 86-511-8897-2712 E-mail: yinrenxingzhe@163.com

Abstract

With the application domain of data mining technologies extends increasingly, some shortages of the commonly used commercial data mining tools, such as poor maintainability, less expansibility and high cost, are gradually exposed in practice. As such, these tools cannot flexibly and effectively meet the apt-to-change demands of users. This paper proposes a plugin style method for dynamic algorithm base design, discusses issues related to the joint between data set and mining algorithms and implements the platform prototype, providing a frame reference for constructing a wieldy, usable, extensible and low cost data mining platform.

Keywords: Data mining, Platform, Pattern, Dynamic algorithms

1. Introduction

Generally, data mining is the process of analyzing a large amount of accumulated data from different perspectives and summarizing it into useful information to serve production and life. Data mining technology in IBM and SPSS has made great progress that they have developed the enterprise Intelligent Miner and Clementine data mining and analyzing system respectively, in this case, data mining technology is so advancing forward quickly. These systems integrate lots of kinds of functions to mine and analyze data, however, their practical application is complicated and the redevelopment on top of them is difficult and high cost. Therefore, these deficiencies lead to great negative impacts on their marketing promotion in practice.

In view of these problems, a plugin style dynamic data mining platform design approach is proposed in this paper, which achieves good flexibility, portability and expansibility. In such a platform, developers only need to create different algorithm plugins in accordance with the consistent interfaces and add the related configuration information to extend the platform with the new algorithms. In this manner, between data mining system platform and data mining algorithms forms a loosely coupled relationship, greatly improving the expansibility and flexibility and meeting the needs of different sorts of applications and effective data mining works.

2. Data Mining Platform Model

A typical knowledge discovery process is wholly divided into three main steps that include data preprocessing, data mining and result analyzing, each of which includes many subtasks to do concrete jobs, the flow is modeled as the following diagram.

3. Data Mining Platform System Structure

3.1 Design of Platform Structure

On the top of and complying with the standard data mining model has positive impact on developing an effective and completed system platform. Figure2 shows the system structure of the data mining platform discussed in this paper. For the system structure, there are the very mappings to the three phrases in data mining process model, reflecting a complete knowledge discovery system process which includes modules as user interface, data preprocessing, data mining algorithm base and pattern visualizing. All of these modules have mutual relationships and coordination among them to help the users gain optimal knowledge and patterns using the effective data mining algorithms.

The most important work to realize a dynamic data mining platform is to make the core data mining algorithms dynamic. That is to say it is necessary and needed to design dynamic algorithm base because of the emergence of the new or improved data mining algorithms. In this case, the new or modified data mining methods should be able to be integrated into the data mining system platform to deal with the frequent application change requirements. Therefore, the approach of plugin style algorithm base design provides an option for designing a dynamic and expansible data

mining system.

3.2 Dynamic Algorithm Base Design and Implementation

In order to develop a dynamic and extensible data mining algorithm base, the method of plugin algorithm was adopted in this paper. Generally, a plugin indicates these utilities which extend the main system by complying with certain predefined standard interfaces. According to different requirements coming from different applications, developers can create their own algorithm plugins to implement various kinds of functions and extend some existent programs. The management task on algorithms in the platform is achieved by using hash table where algorithm name is taken as the hash key and algorithm object as the hash value. It is needed to register the related algorithm information like algorithm type, algorithm path, algorithm name and so on in the algorithm manager when to integrate the external data mining algorithms into the platform. The data mining algorithm class definition and hast table mapping information are showed as follows:

```
class DMAlgorithm{
private String dmAlgorithmName; private File dmAlgorithmFile; // data mining algorithm class path
private String dmAlgorithmType;
private String dmAlgorithmEntry; // data mining algorithm entry function name
private LinkedHashTable algorithmEntryParameters; // parameter list of the entry functions
}
```

The following figure 3 is the corresponding hash table mapping information.

In the dynamic data mining platform, what is used to load algorithms and to execute the entry functions of the algorithms is the method dmAlgorithm defined as the following code segments in which the core methods and objects are briefly described.

```
public void dmAlgorithmRun(String dmAlgorithmName, LinkedHashTable entryFunctionParameters){
DMAlgorithm dmAlgorithm= (DMAlgorithm)DMalgorithmManager.getValue(dmAlgorithmName); // get the algorithm instance with the designated data mining algorithm name
Object parasValue[]=new Object(entryFunctionParameters.size());
String dmAlgorithmDirectory=dmAlgorithm.getAlgorithmFile().getDirectory();
String dmAlgorithmFile=dmAlgorithm.getAlgorithmFile().getFile(); // get the path and name of the data mining algorithm class URL[]dmAlgorithmURL=new URL[]{newURL(dmAlgorithmDirectory)};
ClassLoader classLoader=new URLClassLoader(dmAlgorithmURL);
Class class= classLoader.loadClass(dmAlgorithmFile); // create the class loader and load the data mining class into JVM
Object dmAlgorithmObject= class.getConstructor().newInstance(); //create the constructor of the data mining algorithm and get the algorithm instance Method methodList[]=class.getDeclareMethods();// get all the methods' names in the data mining algorithm class
Method entryMethod;
entryMethod.invoke(dmAlgorithmObject,parasValue); // execute the data mining algorithm class
...
}
```

On the object oriented Java platform, the reflection mechanism is generally adopted to realize getting the fields and methods of the data mining algorithms. In this way, the class and object description information in JVM can be called and the developers are able to create flexible codes which are dynamically loaded at run time so as to update algorithms actively.

3.3 Dataset and Algorithms Contact Model

Since the dataset sources are various, it is necessary to supply data mining algorithms with dataset in a general way. Herein, the various data sources are abstracted into different data objects, the data mining algorithms deal with them through the unified call interface. In this case, the core problem centralizes designing a proper call interface between the data objects and data mining algorithms.

The centralized problem in data objects focuses on the call interface design and the information description of the data objects. For the design of data objects call interface, the abstract factory pattern in OOP is employed to model. Here, the

different data coming from different sources are taken as abstract data objects interface that provides methods to create data objects, and the concrete data object classes are used to create different data object instances according to different data. The figure 4 shows the completed data object creation procedure.

In this case, different data sources are abstracted into different data objects with related describing information and the unified call interface are provided for the data mining algorithms to call data objects, the smooth contact between dataset and data mining algorithms is achieved through the unified mechanism.

3.4 Miscellaneous Functions

Get the final results are mostly got through several iterations, so it is necessary to offer data mining algorithm configuration and data sources selection to adapt algorithm parameters and choose different algorithms for meeting different work modes. In addition, in order to make query and analysis convenient, the knowledge and pattern storage should be taken into account. Because of the implementation differences in various data mining models, a unified pattern presentation method is required. In recent years, the emergence of PMML which was built on the top of XML has provided a feasible way to express, explain and share the data mining results. Also, it even can be used to describe the complete data mining process.

4. Platform Prototype Implementation

To realize the dynamic data algorithm base using plugin idea requires the support of the two basic interface definitions as the platform extension interface and the plugin extension interface respectively. An algorithm plugin needs to call the platform extension realized by the platform, on the contrary, the system platform can also call and use the plugin extension interface realized by the algorithm plugin. Additionally, the primary involved components include the plugin management component, platform component and algorithm plugin component which mutually coordinate to work by the basic two interfaces. The figure 5 shows the platform prototype implementation profile.

The communication contents among these components are mainly include the algorithm plugins management and innovation, resources and data sharing like internal data usage and memory allocation. By the two basic extension interfaces, the data mining platform and algorithm plugins could communicate and achieve data and resource sharing.

5. Conclusion and Summary

This paper begins with analyzing the core parts of the standard data mining process model, designs a dynamic algorithm base on the top of the plugin principle which could better deal with weak points and problems encountered in redeveloping the commercial data mining tools, providing a good method to build a flexible and extensible data mining system platform. Yet, the data algorithms as the base and core of the system platform, their efficiencies directly determine the performance of the platform. Therefore, the emphasis of the future work will be paid to the improvement and modification of the data mining algorithms.

References

Chieh-Yuan, et al. (2005). A dynamic Web service based data mining process system. Computer and Information Technology, 2005. The Fifth International Conference on Sept. 2005 Page(s): 1033 -1039.

Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. (2002). *Design Patterns: Elements of Reusable Object-Oriented Software*. Beijing: China Machine Press 2002.

Liang Bao, Ping Chen. (2006). A Plug-in Based Solution of Heterologous Data Integration. *Computer Engineering,* 2006, (20).

Meilin Zhu, Chongjun Wang, et al. (2002). Discussing the Relationship between XML and Data Mining. *Computer Science,* 2002, (10).

Weining Qian, et al. (2002). A data mining system for very large databases. *Journal of Software*, 2002, (8).

Xingsen Li, et al. (2006). A Knowledge Management Platform for Optimization-based Data Mining. Data Mining Workshops, 2006. Sixth IEEE International Conference on Dec. 2006 Page(s):833-837.

Zhongzhi Shi, et al. (2007). MSMiner: a developing platform for OLAP. Decision Support Systems January 2007, 42(4) Page(s): 2016-2028.

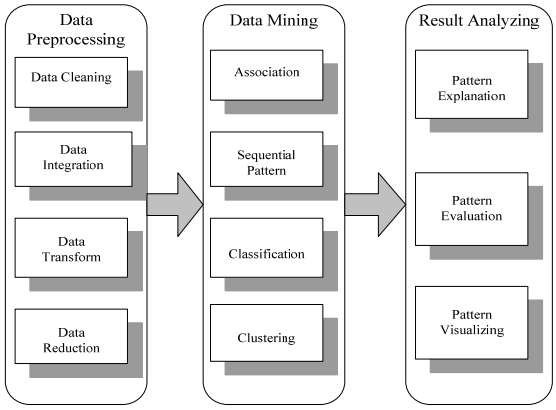

Figure 1. Data Mining Flow Model

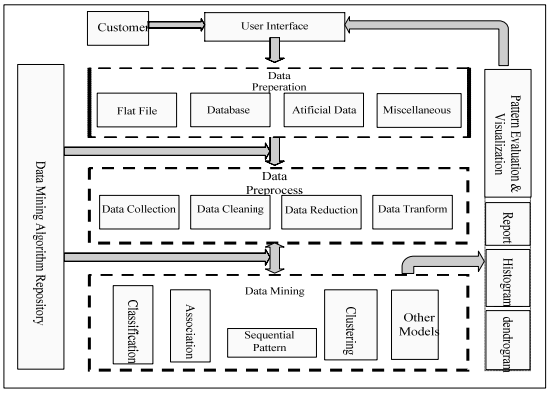

Figure 2. Data Mining Platform Structure

DMAlgorithmManager

Key	Value
AlgorithmName1	Algorithm1
AlgorithmName2	Algorithm2
...	...

DMAlgorithm

String	dmAlgorithmName
File	dmAlgorithmFile
String	dmAlgorithmType
String	dmAlgorithmEntry
String	algorithmEntryParameters

Figure 3. Hash Mapping of Algorithm Manager and DMAlgorithm

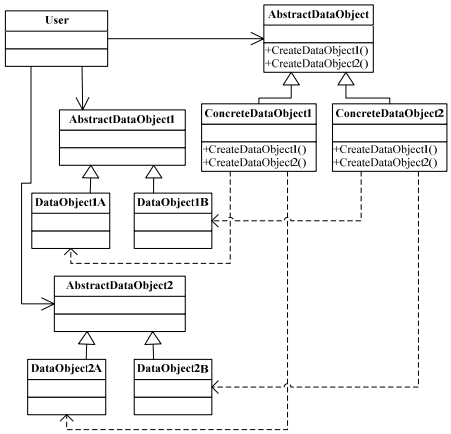

Figure 4. Data Object Build Model

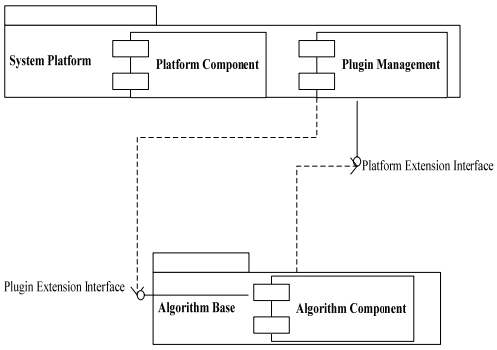

Figure 5. Dynamic Platform Realization