Volume 3, Issue 10, October 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Advanced Patterns and Techniques in Knowledge Discovery through Data Mining

Tiruveedula Gopi Krishna*
Lecturer, Computer Science Dept., Faculty of Science, Hoon, Al-Jufra, Sirt University, Libya.

Dr.Mohamed Abdeldaiem Abdelhadi.
Hod, Department of Computer Science, Hoon, Al-Jufra, Sirt University, Libya.

M.Madhusudhana Subramanyam
Ex. Lecturer, University of Dammam Dammam, Saudi Arabia.

Abstract-- The focus of this paper discussion was on mining large, massive, patterns, techniques and some new trends. Knowledge Discovery is a concept that describes the process of automatically searching large volumes of data for patterns that can be considered knowledge about the data. The most well-known application of knowledge Discovery is data mining also known as Knowledge Discovery in Databases (KDD). Another promising application of knowledge discovery in the area of software modernization which involves understanding existing software artifacts. This process is related to a concept of reverse engineering. Usually the knowledge obtained from existing software is presented in the form of models to which specific queries can be made when necessary. Undoubtedly, research in data mining will continue and even increase over coming decades. In this article, we sketch our vision of some advanced patterns of data mining. Also we discussed how a data mining is a modern and powerful tool, automatizing the process of discovering relationships and combinations in raw data and using the results in an automatic decision support.

Keywords—Knowledge discovery, Patterns and Technique, Olap, OLE DB, ODBO

I. INTRODUCTION

Knowledge Discovery is the process of deriving knowledge from the input data. Some forms of knowledge Discovery create abstractions of the input data. In some scenarios, the knowledge obtained through the process of knowledge Discovery becomes further data that can be used for continuous discovery. Knowledge Discovery is a complex topic that can be further categorized according to
• What kind of data is searched; and
• In what from is the result of the search represented.

Knowledge discovery is defined as "the non-trivial extraction of implicit, unknown, and potentially useful information from data". Under their conventions, the knowledge discovery process takes the raw results from data mining (the process of extracting trends or patterns from data) and carefully and accurately transforms them into useful and understandable information. The most sophisticated definition is one according to Fayyad et al., where authors have determined that knowledge discovery in databases is interactive and iterative process with several steps and data mining is a part of this process. Process of KDD is defined as: The nontrivial process of identifying valid, novel, potentially useful, understandable patterns in data. The terms of above definition are explained as follows:-

A. *Pattern*

Any company that knows its business and its customers is already aware of many important, high-payoff patterns that its employees have observed over the years. What data mining can do is confirm such empirical observations and find new, subtle patterns that yield steady incremental improvement.
• Models or structure in data (traditional sense)
• Expression in some language describing a subset of the data or a model applicable to that subset (data comprises as set of facts)

B. *Process*
• Implies there are many steps repeated in multiple iterations

C. Nontrivial (process)
• It must involve search for structure, models, patterns, or parameters

D. Valid
• Discovered patterns should be valid for new data with some degree of certainty

E.Novel
• As least to the system and preferably to the user

F. Potentially useful
• For the user or task

G. Understandable

• Discovered pattern should be understandable if not immediately, then after some post processing.

II. KDD PROCESS

According to definition above, the KDD is an interactive and iterative process. The first two steps of the KDD process, namely task discovery and data discovery, produce the first input (goal of the KDD process). The next steps in the KDD process are data cleaning, model development, data analysis and output generation.

1). Task Discovery is one of first step of KDD. Client has to state the problem or goal, which often seems to be clear. Further investigations recommended such as to get acquainted with customer's organization after spending some time at the place and to sift through the raw data (to understand its form, content, organizational role and sources of data). Then the real goal of the discovery will be found.

2). Data Discovery is complementary to step of task discovery. In the step of data discovery, we have to decide whether quality of data is satisfactory for the goal (what data does or does not cover.

3). Domain Model plays an important role in the KDD process, thought it often remains in the mind of the expert. A data dictionary, integrity constraints and various forms of metadata from the DBMS can possibly contribute to retrieval of the background knowledge for the KDD purposes as well as some analysis techniques. Those can take advantage of formally represented knowledge when fitting data to a model (for example ML techniques such as explanation-based learning integrated with inductive learning techniques).

4). Data Cleaning is often necessary though it may happen that something removed by cleaning can be indicator of some interesting domain phenomenon (outlier or key data point?). Analyst's background knowledge is crucial in data cleaning provided by comparison of multiple sources. Other way is to clean data before loaded into database by editing procedures. Recently, the data for KDD are coming from data warehouses which contain data already cleaned on some way.

5). Model Development is an important phase of KDD that must precede actual analysis of the data. Interaction with the data leads analysts to formation of hypothesis (it is often based on experience and background knowledge).Sub-process of model development are:

- Data segmentation (unsupervised learning techniques, for example clustering);
- Model selection (choosing the best type of model after exploring several different types);
- Parameter selection (parameters of chosen model).

6). Data Analysis is in general an ambition to understand why certain groups of entities are behaving on the way they do, it is search for laws or rules of such behaviour. As first should be analyzed those parts where such a groups are already identified. Sub-process in data analysis are:

- Model specification-some formalism is used to denote specific model;
- Model fitting-when necessary the specific parameters are determined (in some cases the model is independent from data in other cases the model has to be fitted to training data);
- Evaluation-model is evaluated against the data;
- Model refinement- model is refined in interactions according to the evaluation results.

As mentioned above the model development and data analysis are complementary so it often leads to oscillation between those two steps.

7). Output Generation- output can be in various forms. The simplest form is a report with analysis results. The other, more complicated forms are graphs or in some cases it is describable to obtain action descriptions which might be taken directly outputs. Or there should be a monitor as the output, which should trigger an alarm or action under some certain condition. Output requirements might determine task of designed KDD application.

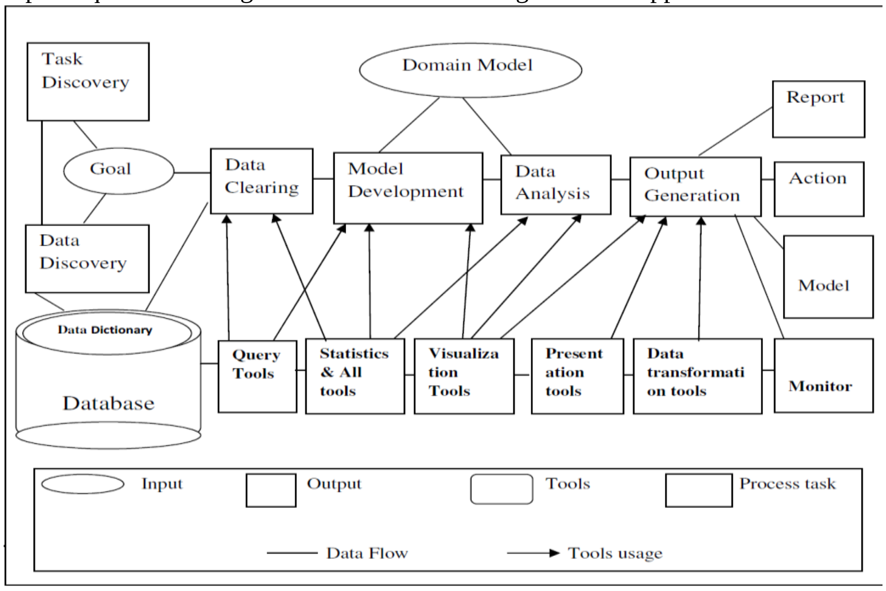

Fig. Integration of Data mining with Databases

III. KDD TECHNIQUES

Learning algorithms are an integral part of KDD. Learning techniques may be supervised or unsupervised. In general, supervised learning techniques enjoy a better success rate as defined in terms of usefulness of discovered knowledge. According to, learning algorithms are complex and generally considered the hardest part of any KDD technique. Machine discovery is one of the earliest fields that have contributed to KDD while machine discovery relies solely on an autonomous approach to information discovery; KDD typically combines automated approaches with human interaction to assure accurate, useful, and understandable results.

There are many different approaches that are classified as KDD techniques. There are quantitative approaches, such as the probabilistic and statistical approaches. There are approaches that utilize visualization techniques. There are classification, approaches such as Bayesian classification, inductive logic, data cleaning pattern discovery, and decision tree analysis, genetic algorithms, neural networks, and hybrid approaches that combine two or more techniques. Because of the ways that these techniques can be used and combined, there is a lack of agreement on how these techniques should be categorized. For example, the Bayesian approach may be logically grouped with probabilistic approaches, classification approaches, or visualization approaches. For the sake of organization, each approach described here is included in the group that it seemed to fit best. However, this selection is to intend to imply a strict categorization. The major techniques are explained below:

A. Probabilistic Approach

This family of KDD techniques utilizes graphical representation models to compare different knowledge representations. These models are based on probabilities and data independencies. They are useful for application involving uncertainty and applications structured such that a probability may be assigned to each "outcome" or bit of discovered knowledge. Probabilistic techniques may be used in diagnostic systems and in planning and control, systems. Automated probabilistic tools are available both commercially and in the public domain.

B. Statistical Approach

The Statistical approach uses rule discovery and is based on data relationships. An "inductive learning algorithm can automatically select useful join paths and attributes to construct rules from a database with many relations". This type of induction is used to generalize patterns in the data and to construct rules from the noted patterns. Online Analytical Processing (OLAP) is an example of a statistically-oriented approach. Automated statistical tools are available both commercial and in the public domain. An example of a statistical application is determining that all transactions in a sales database that start with a specified transactions in the database only 60% are cash sales. Therefore, the system may accurately conclude that 40% are collectibles.

C. Classification Approach

Classification is probably the oldest and most widely-used of all the KDD approaches. This approach groups data according to similarities or classes. There are many types of classification techniques and numerous automated tools available.

D. Bayesian Approach **to KDD** "is a graphical model that uses exclusively to form a directed acyclic graph". Although the Bayesian approach uses probabilities and a graphical means of representation, it is also considered a type of classification. Bayesian networks are typically used when the uncertainty associated with an outcome can be expressed in terms of a probability. This approach relies on encoded domain knowledge and has been used for diagnostic systems. Other pattern recognition applications, including the Hidden Markov Model, can be modeled using a Bayesian approach. Automated tools are available both commercially and in the public domain.

E. Pattern Discovery and Data Cleaning is another type of classification that systematically reduces a large database to a few pertinent and informative records. If redundant and uninteresting data is eliminated, the task of discovering patterns in the data is simplified. This approach works on the premise of the old adage, "less is more". The pattern discovery and data cleaning techniques are useful for reducing enormous volumes of application data, such as those encountered when analyzing automated sensor recordings. Once the sensor readings are reduced to a manageable size using a data cleaning technique, the patterns in the data may be more easily recognized. Automated tools using these techniques are available both commercially and in the public domain. The **Decision Tree Approach** uses production rules, builds a directed graph based on data premises, and classifies data according to its attributes. This method requires the data classes are discrete and predefined. The [primary use of this approach is for predictive models that may be appropriate for either classification of regression techniques. Tools for decision tree analysis are available commercially and in the public domain.

F. Deviation and Trend Analysis

Pattern detection by filtering important trends is the basis for this KDD approach. Deviation and trend analysis techniques are normally applied to temporal databases. A good application for this type of KDD is the analysis of traffic on large telecommunications network. AT&T used such a system to locate and identify circuits that exhibit deviation (faulty behavior). The sheer volume of data requiring analysis makes an automated technique imperative. Tr4end-type analysis might also prove useful for astronomical and oceanographic data, as they are time-based and voluminous. Public domain tools are available for this approach.

G. Other Approaches

Neural Networks may be used as a method of knowledge discovery. Neural networks are particularly useful for pattern recognition and are sometimes grouped with the classification approaches. There are tools available in the public domain and commercially. Genetic algorithms, also used for classification are similar to neural networks although they are typically considered more powerful. There are tools for the genetic approach available commercially. A hybrid approach to KDD combines more than one approach and is also –called a multipara- digmatic approach. Although implementation may be more difficult, hybrid tools are able to combine the strengths of various approaches. Some of the common. Only used methods combine visualization techniques, induction, neural networks, and rule-0base3d systems to achieve the desired knowledge discovery. Deductive databases and genetic algorithms have also been used in hybrids approaches. There are hybrid tools available commercially and in the public domain.

IV. KNOWLEDGE DISCOVERY METAMODEL

*Knowledge Discovery Metamodel (KDM***)** is publicly available specification from the Object Management Group (OMG). KDM is a common intermediate representation for existing software systems and their operating environments that defines common metadata required for deep semantic integration of Application Lifecycle management tools. KDM was designed as the OMG's foundation for software modernization, IT portfolio management and software assurance. KDM uses OMG's Meta-Object facility to define an XMI interchange format between tools that work with existing software as well as an abstract interface (API) for the next generation assurance and modernization tools. KDM standardizes existing approaches to knowledge discovery in software engineering artifacts, also known as software mining. The goal of KDM is to ensure interoperability between tools for maintenance, evolution, assessment and modernization. KDM is defined as a Metamodel that can be also viewed as ontology for describing the key aspects of knowledge related to the various facets of enterprise software. KDM support means investment into the KDM ecosystem a growing open-standard based cohesive community of tool vendors, service providers, and commercial components. KDM represents entire enterprise software systems, not just code. KDM is a wide spectrum entity-relationship representation for describing existing software. KDM represents structural and behaviour elements of existing software systems. The key concept of KDM is a container: an entity that owns other entities. This allows KDM to represent existing systems at various degrees of granularity. KDM defines precise semantic foundation for representing behaviour, the so-called micro-KDM. It provides a high-fidelity intermediate representation which can be used, for example, for performing static analysis of existing software systems. Micro-KDM is similar in purpose to a Virtual machine for KDM, although KDM is not an executable model, or a constraint model, but a representation of existing artifacts for analysis purposes. KDM facilities incremental analysis of existing software systems, where the initial KDM representations is analyzed and more pieces of knowledge are extracted and made explicit as KDM to KDM transformation performed entirely within the KDM technology space. The steps of the knowledge extraction process can be performed by tools, and may involve the analyst. KDM is the uniform language-and platform-independent representation. Its extensibility mechanism allows addition of domain-, application and implementation-specific knowledge.

A. Architecture of KDM

Knowledge Discovery Metamodel consists of 12 packages arranged into four layers. KDM packages are arranged into the following four layers:

1). Infrastructure Layer

The KDM Infrastructure Layer consists of the **Core, KDM**, and **Source** packages which provide a small common core for all other packages, the inventory model of the artifacts of the existing system and full traceability between the meta-model elements as links back to the source code of the artifacts, as well as the uniform extensibility mechanism. The Core package determines several of patterns that are reused by other KDM packages. Although KDM is a meta-model that uses Meta-Object facility there is an alignment between the KDM Core and Resource Description Framework (RDF).

2). Program Elements Layer

The program Elements layer consists of the Code and Action packages. The **Code package** represents programming elements as determined by programming languages, for example data types, procedures, classes, methods, variables, etc. This package is similar in purpose to the Common Application Meta-model (CAM) from another OMG specification, called Enterprise Application Integration (EAI). KDM Code package provides greater level of detail and is seamlessly integrated with the architecturally significant views of the software system. Representation of data types in KDM is aligned with ISO standard ISO/IEC 11404.

3). The Action package captures the low level behaviour elements of applications, including detailed control-and data flow between statements. Code and Action package in combination provide a high-fidelity intermediate representation of each component of the enterprise software system.

4). Resource Layer

The Resource layer represents the operational environment of the existing software system. It is related to the area of Enterprise Application Integration (EAI).

4.4.1 Platform Package represents the operating environment of the software, related to the operating system; middleware etc. including the control flows between companies as they are determined by the runtime platform.

4.4.2 UI Package represents the knowledge related to the user interfaces of the existing software system.

4.4.3 Event Package represents the knowledge related to the event and state- transition behaviour of the existing software system.

4.4.4 Data Package represents the artifacts related to persistent data, such as indexed files, relation databases and other kinds of data storage. These assets are key to enterprise software as they represent the enterprise metadata. The KDM Data package is aligned with another OMG specification, called Common Warehouse Metamodel (CWM).

5). Abstractions Layer

The Abstraction Layer represents domain and application abstractions.

5.1 Conceptual Package represents business domain knowledge and business rules, insofar as this information can be mined from existing applications. These packages are aligned with another OMG specification, called Semantics of Business Vocabulary and Rules (SBVR).

5.2 Structure package describes the meta-model elements for representing the logical organization of the software system into subsystems, layers and components.

5.3 Build Package represents the engineering view of the software system.

V. ON LINE ANALYTICAL PROCESSING

On Line Analytical Processing or OLAP is an approach to quickly providing answers to analytical queries that are multidimensional in nature. OLAP is part of the broader category business intelligence, which also includes Extract transform load (ETL), relational reporting and data mining. The typical applications of OLAP are in business reporting for sales, marketing, management, reporting, business process management (BPM), budgeting and forecasting, financial reporting and similar areas. The term OLAP was created as a slight modification of the traditional database term OLTP (**On Line Transaction Processing**). Databases configured for OLAP employ a multidimensional data model, allowing for complex analytical and ad-hoc queries with a rapid execution time. Nigel Pendse has suggested than an alternative and perhaps more descriptive term to describe the concept of OLAP is **Fast Analysis of Shared Multi-dimensional Information** (FASMI). Data mining is different from OLAP because rather than verify hypothetical patterns, it uses the data itself to uncover such patterns. It is essentially an inductive process.

A). OLAP Types

OLAP systems have been traditionally categorized using the following taxonomy:

1). MOLAP

MOLAP stands for **Multidimensional Online Analytical Processing. MOLAP** is the 'classic' form of MOLAP and is sometimes referred to as just OLAPO. MOLAP uses databases structures that are generally optimal for attributes such as time period, location, product or account code. The way that each dimension will be aggregated is defined in advance by one or more hierarchies. MOLAP is an alternative to the ROLAP (Relational ROLAP) technology. While both ROLAP and MOLAP analytical tools are designed to allow analysis of data through the use of a multi-dimensional data model, MOLAP differs significantly in that it requires the precomputation and storage of information in the cube – the operation known as processing. MOLAP stores this data in optimized multi-dimensional array storage, rather than in a relational database (i.e. in ROLAP).

2). MOLAP Advantages

• Fast query performance due to optimized storage, multidimensional indexing and caching.
• Smaller on-disk size of data compared to data stored in relational database due to compression techniques.
• Automated computation of higher level aggregates of the data.
• It is very compact for low dimension data sets.
• Array model provides natural indexing.
• Effective data extract achieved through the pre-structuring of aggregated data.

3). MOLAP Disadvantages

• The processing step (data load) can be quite lengthy, especially on large data volumes. This is usually remedied by doing only incremental processing, i.e. processing only the data which has changed (usually new data) instead of reprocessing the entire data set.
• MOLAP tools traditionally have difficulty querying models with dimensions with very high cardinality (i.e., millions of members).
• Certain MOLAP tools (e.g., Essbase) have difficult updating and querying models with more than ten dimensions. This limit differs depending on the complexity and cardinality of the dimensions in questions. It also depends on the number of facts or measures stored. Other MOLAP tools (e.g., Microsoft Analysis Services or Applix TMI) can hundreds of dimensions.
• MOLAOP approach introduces data redundancy.

4). MOLAP Trends

Most commercial OLAO tools now use a "Hybrid OLAP" (HOLAP) approach, which allows the model designer to decide which portion of the data will be stored in MOLAP and which portion in ROLAP.

5). MOLAP Products

Examples of commercial products that use MOLAP are Microsoft Analysis Services, Essbase, MIS Alea and TMI. There is also an open source MOLAP server Palo.

6). ROLAP

ROLAP stands for **Relational Online Analytical Processing. ROLAP** works directly with relational databases. The base data and the dimension tables are stored as relational tables and new tables are created to hold the aggregated information depends on a specialized schema design. ROLAP is an alternative to the MOLAP (Multi-dimensional OLAP) technology. While both ROLAP and MOLAP analytic tools are designed to allow analysis of data through the use of a multidimensional data model, ROLAP differs significantly in that it does not require the precomputation and storage of information. Instead, ROLAP tools access the data in a relational database and generate SQL queries ti calculate information at the appropriate level when an end user requests it. With ROLAP it is possible to create additional database tables (Summary tables or aggregations) which summarize the data at any desired combination of dimensions. While ROLAP uses a relation database source, generally the database must be carefully designed for ROLAP use. A database which has designed for OLTP will not function well as a ROLAP database. Therefore, ROLAPM still involves creating an additional copy of data. However, since it is a database, a variety of technologies can be used to populate the database.

6.1. Advantages of ROLAP

• ROLAP is considered to be more scalable in handling large data volumes, especially models with dimensions with very high cardinality (i.e. millions of members).

• With a variety of data loading tools available, and the ability to fine tune the ETL code to the particular data model, load times are generally much shorter than with the automate MOLAP loads.

• The data is stored in a standard relation database and can be accessed by any SQL reporting tool (the tool does not have to be an OLAP tool).

• ROLAP tools are better as handling non- aggregately factors or (e.g. textual descriptions). MOLAP tools tend to suffer from slow performance when querying these elements.

• By decoupling the data storage from the multidimensional model, it is possible to successfully model data that would not otherwise fit into a strict dimensional model.

• The ROLAP approach can leverage database authorization controls such as row-level security; whereby the query results will be filtered depending on a respect criteria applied for example to a given user or group of users (SQL WHERE clause).

6.2. Disadvantages of ROLAP

• There is a general consensus in the industry that ROLAP tools have slower performance than MOLAP tools.

• The loading of aggregate tables must be managed by custom ETL code. The ROLAP tools do not help with this task. This means additional development time and more code to support.

• When the step of creating aggregate tables is ski8pped, the query performance then suffers because the larger detailed tables must be queried. This can be partially remedied by adding additional aggregate tables, however it is still not practical to create aggregate tables for all combinations of dimensions/attributes.

• ROLAP relies on the general purposes database for querying and caching, and therefore several special technique employed by MOLAP tools are not available (such as special hierarchical indexing). However, modern ROLAP tools take advantage of latest improvements in SOL language such as CUBE and ROLLUP operators, DB2 Cube Views, as well as other SQL OLAP extensions. These SQL improvements can mitigate the benefits of the MOLAP tools.

• Since ROLAP tools rely SQL for all of the computations, they are not suitable when the
model is heavy on calculations which don't translate well into SQL. Examples of such models include budgeting, allocations, financial reporting and other scenarios.

6.3. ROLAP Trends

The undesirable trade-off between additional ETL cost and slow query performance has ensured that most commercial OLAP tools now use a "Hybrid OLAP" (HOLAP) approach, which allows the model designer to decide which portion of the data will be stored in MOLAP and which portion in ROLAP.

6.4. ROLAP Products

Examples of commercial products using ROLAP include Microsoft Analysis Services, Microstrategy and Business Objects, Oracle BI (the former Siebel Analytics). There is also an open source ROLAP server-Mondrian.

6.5. HOLAP

HOLAP (**H**ybrid **O**nline **A**nalytical **P**rocess) is a combination of ROLAP and MOLAP which are other possible implementation of OLAP. There is no clear agreement across the industry as to what constitutes "Hybrid OLAP", expect that a database will divide data between relational and specialized storage. For example, for some vendors, a HOLAP database will use relational tables to hold the larger quantities if detailed data, and use specialized storage for at least some aspects of the smaller quantities of more-aggregate or less-detailed data. HOLAP allows to store part of the data in the MOLAP store and another part of the data in ROLAP store. The degree of control that cube designer has over this partitioning varies from product to product:

6.6. Vertical partitioning

In this mode HOLAP aggregations in MOLAP for fast query performance and detailed data in ROLAP to optimize time of cube processing.

6.7. Horizontal partitioning

In this mode HOLAP stores some slice of data, usually the more recent one (i.e. sliced by time dimension) in MOLAP for fast query performance, and older data in ROLAP.

6.8. HOLAP Products

Examples of commercial products which support HOLAP storage mode are Microsoft Analysis Services, Micro Strategy and SAP AG BI Accelerator.

B). Comparison

Each type has certain benefits, although there is disagreement about the specifics of the benefits between providers. Some MOLAP implementations are prone to database explosion. Database explosion is a phenomenon causing vast amounts of storage space to be used by MOLAP databases when certain common conditions are met: high numbers of dimensions, precalculated results are sparse multidimensional data. The typical mitigation technique for database explosion is not to materialize all the possible aggregation, but only the optimal subset of aggregation based on the desired performance vs. storage trade off. MOLAP generally delivers better performance due to specialized indexing and storage optimizations. MOLAP also needs less storage space compared to ROLAP because the specialized storage typically includes compression techniques. ROLAP is generally more scalable. However, large volume pre-processing is difficult to implement efficiently so it is frequently skipped. ROLAP query performance can therefore suffer. Since ROLAP relies more on the database to perform calculations, it has more limitations in the specialized functions it can use. HOLAP encompasses a range of solution that attempt to mix the best of ROLAP and MOLAP. It can generally preprocess quickly, scale, well, and offer good function support.

C. Other Types

The following acronyms are also used sometimes, although they are not as widespread as the ones above:
• **WOLAP** – Web-based OLAP
• **DOLAP-** Desktop OLAP
• **RTOLAP** – Real Time OLAP
• **SOLAP** – Spatial OLAP

OLAP APIs and Query languages

Unlike relational database- which had SQL as the standard query language, and widespread APIs such as ODBC, JDBC and OLEDB – there was no such unification in the OLAP worlds for a long time. The first real standard API was OLEDB for OLAP (ODBO) specification from Microsoft which appeared in 1997 and introduced the MDX query language. Several OLAP vendors- both server and client- adopted it. In 2001 Microsoft and Hyperion announced the XML for Analysis specification, which was endorsed by most of the OLAP vendors. Since this also used MDX as a query language, MDEX became the de-facto standard in the OLAP world.

*D.OLE DB for OLAP (ODBO***)**

OLE DB for OLAP (abbreviated ODBO) is a Microsoft published specification and an industry standard for multi-dimensional data processing. ODBO is the standard application programming interface (API) for exchanging metadata and data between OLAP servers as a client on a Windows Platform. ODBO extends the availability of OLEDB to access multidimensional (OLAP) data stores. ODBO is the most widely supported, multi-dimensional API to data. Platform specific to Microsoft Windows, ODBO was specifically designed for Online Analytical Processing (OLAP) systems by Microsoft as an extension to Object Linking and Embedding Database (OLE DB). ODBO uses Microsoft's Component Object Model. ODBO permits independent software vendors (ISVs) and corporate developers to create a single set of standard interfaces that allow OLAP client to access multi-dimensional data, regardless of vendors or data source. ODBO is currently supported by a wide spectrum of server and client tools.

E. Selecting an OLAP Application

While selecting an OLAP that can fit your organization's need requires a closer inspection of your requirements.

1). Platform Choice

The first step in narrowing the list of vendors is the strength of the OLAP tool on the platform that your organization will be using. Some vendors tend to be only Microsoft centric or have products that are better supported on a particular.

2). Vendor History

There are many new OLAP vendors that have cropped up since the mid-nineties. Some of the newer companies have rewarding and rich OLAP tools that can be cost effective. Investigate how many installs the vendor has made and, if possible, interview some of the vendor's clients.

3). OLAP Cube Size and Transaction Speed

There are products that are optimized to your organization's needs. Fine tune your vendor list to your operational needs. How large is your cube? What would be a tolerable usage of storage space and transaction speed? How will the OLAP cube be created – i.e. HOLAP, MOLAP, and ROLAP?

4). User Interface (GUI)

Will you prefer a thin client or desktop interface to interact with the OLAP cube? What is the trade-offs? List potential security concerns.

5). Consulting

How much consulting would be requiring installing the product? If Changes are needed after installation, will this require the need of external consultants?

6). Integration to Database

How tightly integrated is the OLAP tool to your database? Can you perform SQL queries within the OLAP tool?

7). Integrating to other Analysis Tools

Does your organization have or plan to have other analysis tools such as data mining, reporting and data visualization? How will the OLAP application integrate with the other analysis tools?

IV. CONCLUSIONS

In this paper, we surveyed major challenges for data mining in the years ahead. First, we started with the type of "patterns in data" which knowledge discovery is examining. While original data mining concentrated on vectorial data, future data will predominantly be stored in much more complex data types and data mining will have to cope with this increasing volume of structured data. Another aspect of "patterns in data" in the future is the increasing importance of studying their evolution over time. If future data mining methods have to handle all this complex input and intelligent preprocessing, it is very likely that the user will have to adjust more and more switches and knobs before getting any result. Hence, achieving user-friendliness with transparent or even reduced parameterizations a major goal. Usability is also enhanced by finding new types of patterns that are easy to interpret, even if the input data is very complex. Although no human being can foretell the future, we believe that there are plenty of interesting new challenges ahead of us, and quite a few of them cannot be foreseen at the current point of time. Data mining and knowledge discovery are emerging as a new discipline with important applications to science, engineering, health care, education, and business. Data mining rests firmly on 1) research advances obtained during the past two decades in a variety of areas and 2) more recent technological advances in computing, networking and sensors. Data mining is driven by the explosion of digital data and the scarcity of scientists, engineers, and domain experts available to analyze it. Appropriate privacy and security models for data mining must be developed and implemented in future. Although no human being can foretell the future, we believe that there are plenty of interesting new challenges ahead of us, and quite a few of them cannot be foreseen at the current point of time.

ACKNOWLEDGMENT

I would like to thank to all my co-authors to support in Lab and to publish this paper.

REFERENCES

[1] Chen, M. S., Han, J., & Yu, P. S., Data Mining: An Overview from Database Perspective, IEEE Trans. on Knowledge and Data Engineering, Vol. 8, No. 6, pp. 866-883, 1996

[2] Fayyad, U, Djorgovski, S., & Weir, N. Automating the Anal-ysis and Cataloging of Sky Surveys. In Advances in Knowledge Discovery and Data Mining, pp. 471-493, AAAI Press, 1996.

[3] Agrawal, R., Imielinski, T., & Swami, A., Mining association rules between sets of items in large databases, Proc. of Int. Conf. ACM SIGMOD, Washington D. C. pp. 207-216, 1993.

[4] Agrawal, R., Gehrke, J., Gunopulos, & D., Raghavan, P., Auto-matic Subspace Clustering of High Dimensional Data for Data Mining Applications, Proc. of the ACM SIGMOD Int. Conf. onManagement of Data, Seattle, Washington, 1998.

[5] Agrawal, R., Metha, M., Shafer, J., & Srikant, R., The Quest Data Mining System, Proc. of the 2nd Int. Conf. on Knowledge Discovery in Databases and Data Mining, Portland, Oregon, 1996.

[6] Agrawal, R., & Shim, K., Developing Tightly-Coupled Data Mining Applications on a Relational Database System, Proc. of 2nd Int. Conf. on Knowledge Discovery in Databases and Data Mining, Portland, Oregon, 1996.