CHAPTER 1

INTRODUCTION

“Informational Society” is the most proper and fitting term to describe the present day world as an astounding and ever expanding amount of information is pouring in every day. They are used for organizing the economy and the society itself. This has led to the collection of tremendous amounts of information with the aid of sophisticated technologies and tools such as computers, satellites, and remote sensors. At the outset, much more importance was given to the collection and storage of information rather than analysis of the collected information as it is an established fact that information leads to knowledge which is power and power in turn leads to success. Initially with the advent of computers and means for mass digital storage, it was believed that the powerful computers could be relied upon to collect and store the massive collection of data and analyze this amalgam of information quickly and arrive at some useful conclusions. This was not an easy task even for the powerful computers.

The massive collections of data stored on disparate structures very rapidly became overwhelming for the computers to sort through. Some methodical help was needed and this piloted the creation of structured databases and database management systems (DBMS). The efficient database management systems developed proved to be an important asset for managing a large corpus of data and especially for effective and efficient retrieval of particular information from a large collection whenever needed.

1.1 BACKGROUND

Nowadays even simple transactions such as using a credit card, a phone or browsing the web have become ubiquitous and mammoth which have resulted in more information that cannot be handled easily. Information storage and retrieval using DBMS is simply not enough anymore for decision-making. A largely untested hypothesis of modern society is that it is important to record data as it may contain valuable information. Little thought however, has gone into how this quantity of data might be analyzed. In a wide variety of applications, analysis of the collected
information becomes more essential than collection and storage. The large volumes of data have to be mined for interesting and relevant information.

Today, every organization is facing the issue of data overflow. There is too much data generated every day and a very little is used out of it. Developing the technology to distil useful knowledge from the vast amount of data has become a necessity in the present world. Corporations need better technology to make sense out of their data and turn the knowledge learned from the data into a competitive advantage.

Machine learning, the field for finding ways to automatically extract information from data, was once considered the solution to this problem. Historically machine learning has concentrated on learning from small numbers of examples, because only limited amounts of data were available when the field emerged. Some very sophisticated algorithms have resulted from the research that can learn highly accurate models from limited training examples. It is commonly assumed that the entire set of training data can be stored in working memory.

When the volume of the underlying data is very large and cannot be stored in its entirety in working memory, it leads to a number of computational and mining challenges. Traditional data analysis tools and techniques like computers, DBMS and machine learning algorithms cannot be used because of the massive size of a dataset and hence there is need to develop new methods. Data mining, the extraction of hidden predictive information from large databases, is a technology that blends traditional data analysis methods with sophisticated algorithms for processing large volumes of data [21].

Ways are investigated to reduce the computation time and memory needed to process large but static datasets. If the data cannot fit into memory, it may be necessary to sample a smaller training set. Alternatively, algorithms may resort to temporary external storage or only process subsets of data at a time. Commonly the goal is to create a learning process that is linear to the number of examples. The essential learning procedure is treated like a scaled up version of classic machine learning, where learning is considered a single, possibly expensive, operation. The
operation is the processing of a set of training examples to output a final static model.

Data mining has a great potential to help companies focus on the most important information. Most companies already collect and refine massive quantities of data. Data mining techniques can be implemented rapidly on existing software and hardware platforms to enhance the value of existing information resources, and can be integrated with new products and systems as they are brought on-line. Data mining techniques are the result of a long process of research and product development. This evolution began when business data was first stored on computers, continued with improvements in data access, and more recently, generated technologies that allow users to navigate through their data in real time [36]. Data mining takes this evolutionary process beyond retrospective data access and navigation to prospective and proactive information delivery. Data mining is applied widely in the business community because it is supported by three full-fledged technologies:

- Massive data collection
- Powerful multiprocessor computers
- Data mining algorithms

Recent emerging applications, such as security fraud detection in trading and banking, network traffic monitoring, analyzing data generated by sensor networks, Web click stream mining, measuring power consumption, and dynamic tracing of stock fluctuations generate streams of data [76]. More and more applications generate a large amount of data that streams in every day. Efficient knowledge discovery from such data streams is an emerging active research area in data mining with broad applications. In data streams huge volume of data arrive continuously and at a high speed with changes in the data distribution too over time. Traditional data mining techniques which require multiple scans of the entire datasets cannot be applied directly to mine stream data due to these unique features that allow only one scan and demand fast response time [62]. Stream data mining offers a new technology to help achieve this. When the volume of the streaming data is huge, it leads to a number of processing and mining challenges. It has also created new needs to help make better decisions. These needs are automatic summarization of data,
extraction of the essence of information stored, and the discovery of patterns from raw data.

Peter Drucker, an influential management expert, writes “From now on, the key is knowledge. The world is not becoming labor intensive, not material intensive, not energy intensive, but knowledge intensive”. It is true and the society needs all the help it can get to extract and utilize the knowledge from vast data streams. This research work aims at providing such help to mine useful knowledge from the vast amount of data streaming in everyday from various places.

1.2 SIGNIFICANCE OF DATA STREAM MINING

The data mining approach may allow larger datasets to be handled, but it still does not address the problem of a continuous supply of data. Some or all of the input data that are to be operated on are not available for random access from disk or memory, but rather arrive as one or more continuous data streams [45]. Typically, a model that was previously induced cannot be updated when new information arrives. Instead, the entire training process must be repeated with the new examples included. There are situations where this limitation is undesirable and is likely to be inefficient.

The data stream paradigm has recently emerged in response to the continuous data problem. Algorithms written for data streams should naturally cope with data sizes many times greater than memory, and should extend to challenging real-time applications not previously tackled by machine learning or data mining.

Many organizations today have more than very large databases; they have databases that grow without limit at a rate of several million records per day. Mining these continuous data streams brings unique opportunities, but also new challenges. Data Stream Mining is the process of extracting knowledge structures from continuous, rapid data records [68]. A data stream is an ordered sequence of instances that in many applications of data stream mining can be read only once or a small number of times using limited computing and storage capabilities. Data streams differ from the conventional stored relation model in several ways [49]. Some of them are listed below:

- The data elements in the stream arrive online
• The system has no control over the order in which data elements arrive to be processed
• Data streams are potentially unbounded in size
• Once an element from a data stream has been processed it is either discarded or archived
• Individual data items have to be processed extremely fast to match the stream rate
• Only limited storage is available for processing or capturing and archiving data at stream rate

Recent emerging applications, such as trade surveillance for security fraud and money laundering, network traffic monitoring, sensor network data analysis, Web click stream mining, power consumption measurement, and dynamic tracing of stock fluctuations, generate streams of data [14]. These streams can be a continuous, potentially infinite flow of information as opposed to finite, statically stored datasets. Besides querying data streams, another important application is to mine data streams for interesting patterns or anomalies as they happen [13]. Data stream mining can be considered a subfield of data mining, machine learning, and knowledge discovery. Stream mining technology can dramatically change the way corporations and governments or even individuals handle or process data [12]. Data stream mining algorithms take a continuously flowing data stream as input, mine the streaming data and produce some knowledge as output as shown in Fig. 1.1.

![Data Stream Mining Diagram](image)

Fig. 1.1 Data Stream Mining

A data stream is a real-time, continuous, ordered sequence of items where the ordering is either implicitly by the time of arrival or explicitly by the timestamp included. It is not possible to control the order in which items arrive, nor is it feasible to locally store a stream in its entirety [18]. Data stream mining is the process of extracting knowledge structures from continuous rapidly-changing
streams of data. Stream mining has its solutions based on well-established statistical and computational approaches that can be categorized into data-based approaches and task-based approaches [13]. Data-based solutions focus on examining a subset of the whole dataset or to reduce it vertically by means of feature removal or horizontally by reducing the number of records, both resulting in reduced data [76]. Task-based solutions engage techniques from computational theory that have been adapted to achieve time and space efficient solutions.

Data-based techniques include sampling, load shedding, sketching, synopsis data structures and aggregation. Sampling is concerned with the selection of a subset of data from within the data stream to estimate characteristics of the entire data stream. Load shedding involves discarding some fraction of the unprocessed data to handle the peak load in the streaming data [86]. Both sampling and load shedding have a problem that they are dropping out chunks of data streams that might represent a pattern of interest in time series analysis.

Sketching involves randomly projecting to a subset of features from a stream. The major drawback of sketching is its accuracy. Creating synopsis from data may include wavelet analysis (Fourier transform), histograms, quartiles and frequency moments which result in compressed form of data [91]. When using these techniques, one needs to be aware of the inaccuracies that might occur due to the possibly incomplete representation of data. Aggregation involves computing statistical measures, such as means, variance, minimum, maximum, and count that can be used by mining algorithms to achieve their objective without handling the data in its entirety. Again, such a method does not perform well with highly fluctuating data distributions.

Task-based techniques modify existing techniques and introduce new techniques for stream processing that are able to cope with the computational challenges of data stream processing [98]. These include approximation algorithms, sliding windows, and algorithm output granularity. Approximate algorithms are one pass mining algorithms to approximate the mining results according to some acceptable error margin [5]. Sliding windows concentrate only on recent data by ignoring old data [34]. Algorithm output granularity is the amount of mining results
that fits in main memory before any incremental integration and it controls the output rate of the algorithms according to the available memory.

The intuition behind sliding windows technique is in the fact that user is more concerned with the analysis of most recent data. Detailed analysis is done over the most recent data items and a summarized version of historic data. The sliding windows approach is widely implemented and can be combined with data-based techniques for even higher efficiency [43]. This is the approach tried in this research work.

Stream mining algorithms are being developed to discover useful knowledge from data as it streams in [53]. This rapid generation of continuous streams of information has challenged the storage, computation and communication capabilities in computing systems. As data stream is continuous flow of data, volume of data to be processed is enormous.

With increasing volume of the data, it is no longer possible to process the data efficiently by using multiple passes. Rather, one can process a data item at most once. This leads to constraints on the implementation of the underlying algorithms. Therefore, stream mining algorithms typically need to be designed so that the algorithms work with one pass of the data [72]. In most cases, there is time component associated to the stream mining process. This is because the data may evolve over time. Therefore, a straightforward adaptation of one-pass mining algorithms may not be an effective solution to the task. Stream mining algorithms need to be carefully designed with a clear focus on the evolution of the underlying data.

1.3 CHALLENGES IN DATA STREAM MINING

Some of the major challenges in data stream mining are handling the continuous flow of data, minimizing the energy consumption, unbounded memory requirement, transferring data mining results, modeling changes of results over time, real time response, and visualization of data mining results [37]. These challenges and some of the methods suggested to overcome them are discussed in the following sections.
Handling the continuous flow of data

This is a data management issue. Traditional database management systems are not capable of dealing with such continuous high data rate. As a consequence, in many cases it is impractical to store all data in persistent media and in other cases it is too expensive to randomly access data multiple times. The challenge is to discover frequent datasets while the data can only be accessed once. Novel indexing, storage and querying techniques are required to handle this fluctuating, non-stopping flow of information streams [75].

Minimizing energy consumption

Large numbers of data streams are generated in resource-constrained environments. Sensor networks represent a typical example. Wireless sensor networks (WSNs) have recently captured the world-wide attention due to its enormous potential for commercial as well as military applications. A WSN consists of low-power, low-cost, and energy-constrained sensors with limited communication and computation ability. These devices have short life batteries. They have to monitor certain phenomena like temperature rise and report them to a central site where further processing is done by the end user. The design of techniques that are energy efficient is a crucial issue in such cases given that sending all the generated stream to a central site is energy inefficient in addition to its lack of scalability problem.

Unbounded memory requirements

Another feature of data streams is that data are unbounded but storage that can be used to discover or maintain the frequent datasets is limited. Machine learning techniques represent the main source of data mining algorithms. Most machine learning methods require data to be resident in memory while executing the analysis algorithm. Owing to the huge amounts of the generated streams, it is absolutely important to design space efficient techniques that can have only one look or less over the incoming stream [10]. Another consequence of unbounded data is that the frequency of the itemset depends on time. An itemset that is infrequent now may become frequent at a later instant. The challenge is to use limited storage to discover dynamic frequent itemsets from unbounded data and also maintain
information about itemsets that may become frequent in future [27]. All itemsets cannot be maintained due to limited memory, so decisions have to be taken regarding what to maintain and what to discard. This is an important issue in data stream mining.

**Transferring data mining results**

Knowledge structure representation is another essential research problem. Given that transfer of all data to a central site is not feasible and scalable, nowadays extraction of interesting patterns are done locally and after extracting models and patterns locally from data stream generators, it is essential to transfer these structures to the user. This reduces the traffic and the load on the central site. Many researchers have addressed this problem by using different transformations to efficiently send mining results over limited bandwidth links. Thus transferring data mining results using efficient transformations to limit the usage of bandwidth and time is an important issue in data stream mining.

**Modeling changes of results over time**

In some cases, the user is not interested in mining data stream results, but to know how these results change over time. For example the number of clusters generated by a data stream changes over time, which might represent some changes in the dynamics of the arriving stream. Identifying the dynamics of data streams by analyzing the changes in the knowledge structures generated would benefit many temporal-based analysis applications like video-based surveillance, emergency response, disaster recovery, stock monitoring, fluctuations in fashion, fluctuations in seasonal purchases and many more.

**Real-time response**

Because data stream applications are usually time-critical, there are requirements on response time. For some restricted scenarios like emergency response in flight navigation at air traffic control facilities, algorithms that are slower than the data arriving rate are useless. Modern Air-Traffic Control systems give at a glance a clearer, more complete picture of the congestion status of a given airspace. The mouse hovering over each track symbol allows the controller to see a plethora of
data and issue commands that can get relayed to the aircraft via satellite. The new commands pop up in the on board navigation and communication instruments and the pilot can decide whether to implement it or not.

The commands and subsequent execution, together with key flight data are to be processed and communicated at high-speed to be useful to the pilots and controllers for effective management. For such systems, the streaming data has to be processed and responses produced in real-time. Thus real-time response is an important issue in stream mining.

**Visualization of data mining results**

Visualization of traditional data mining results on a desktop is still a research issue. Visualization in small screens of a Personal Digital Assistant (PDA) for example is a real challenge. Imagine a businessman at his job analyzing the data streaming in from his head office and various point-of-sales on his Personal Digital Assistant. The results of his analysis should be efficiently visualized in a way that enables him to take a quick decision.

If the results are viewed as pages and pages of reports it is useless to him, instead if the results are visualized in terms of graphs then it shall enable him to take quick and correct decisions. Thus visualizing the results of data stream mining efficiently plays an important role in mining.

**Fluctuating data rates**

Scientists have to deal with the fact that the data rate of the stream isn't constant, leading to a condition called burstiness, and the patterns of the data stream and scheduling resources are continuously evolving. Most of the data streams available for mining today exhibit changes in underlying process that generate the data. For example changing weather conditions, changing economic conditions, a poorly calibrated sensor, all could lead to changes in data distribution or collection methods. New and efficient methods are essential to extract useful information from such data streams that arrive at high speed and exhibit significant changes as time progresses.
Temporal locality

In most cases, there is an inherent temporal component to the stream mining process. This is because the data may evolve over time. This behavior of data streams is referred to as temporal locality. Many applications such as news group filtering, text crawling, and document organization require real time clustering and segmentation of text data records. The categorical data stream clustering problem also has a number of applications to the problems of customer segmentation and real time trend analysis. Such applications often exhibit temporal locality which is not taken into account by most batch processing algorithms. The clustering problem presents a number of unique challenges in an evolving data stream environment. For example, the continuous evolution of clusters makes it essential to be able to quickly identify new clusters in the data. While the clustering process needs to be executed continuously in online fashion, it is also important to provide end users with the ability to analyze the clusters in an offline fashion.

Data stream mining algorithms have to naturally cope with all issues discussed earlier and handle data sizes many times greater than memory, and extend to challenging real-time applications not previously tackled by machine learning or data mining [20]. The core assumption of data stream processing is that training examples can be inspected in brief a single time only, and then must be discarded to make room for subsequent examples. The volume of data is usually too big to be stored on permanent devices or to be scanned thoroughly more than once. The algorithm processing the stream has no control over the order of the instances seen, and must update its model incrementally as each instance is inspected [11]. Incremental updates are essential to avoid building the model from scratch every time a new transaction arrives in the data stream. Arrival of new transactions is a continuous never-ending process which makes incremental updates of the model mandatory. An additional desirable property, the so-called anytime property, requires that the model is ready to be applied at any point between training examples. Shared execution of many continuous queries is needed to ensure scalability in data stream processing. Thus the ability to approximate and the ability to adapt are very important key ingredients for performing mining tasks over rapid data streams.
1.4 STREAM MINING ALGORITHMS

A number of algorithms have been proposed for extracting knowledge from streaming information. Some of the most popular techniques are frequent pattern mining, clustering, classification, and Outlier analysis techniques.

**Frequent Pattern Mining**

Frequent pattern mining is a core data mining operation and has been extensively studied over the last decade. Recently, mining frequent patterns over data streams has attracted a lot of research interests [91]. Frequent pattern mining focuses on discovering frequently occurring patterns from different types of datasets. There are mainly three types of datasets, unstructured, semi-structured and structured datasets. Transactions streaming in and text documents available on the Internet are typical unstructured datasets. Extensible Markup Language (XML) documents communicated by the Internet are typical semi-structured datasets and graph datasets are typical structured datasets. The patterns mined from these datasets can be itemsets, sequences, sub-trees, or sub-graphs, depending on the mining tasks and targeting datasets. Frequent pattern mining from data streams poses great challenges due to high memory and computational costs, high speed data and accuracy requirement of the mining results.

**Clustering**

The clustering problem is defined as follows: for a given set of data points, one wishes to partition them into one or more groups of similar objects. The number of groups to be formed is determined dynamically. Thus clustering is considered as an unsupervised learning activity. The similarity of the objects with one another is typically defined with the use of some distance measure or objective function. Intra-cluster similarity should be very high and inter-cluster similarity should be very low for a good clustering algorithm. Clustering is a widely studied problem in the data mining literature. However, adapting arbitrary clustering algorithms working on static datasets to dynamic data streams is very difficult mainly because of the one-pass constraint imposed on the data streams. In the context of data streams, it is more desirable and practical to determine clusters in
specific user defined horizons rather than on the entire dataset as data streams in continuously at a very high speed.

**Data Stream Classification**

The problem of classification is perhaps the most widely studied one in the context of data stream mining. Classification problem aims to identify the characteristics that indicate the group to which each instance belongs. It can be used both to understand the existing data and to predict how new instances will behave. Classification is thus considered as a supervised learning activity as the number of groups is known before the classification process begins. But the problem of classification in data streams is made more difficult by the evolution of the underlying data stream. The concept and distribution of data may vary according to the time and this may result in new classes being added. Therefore, effective algorithms need to be designed in order to take temporal locality into account.

**Outlier analysis**

One of the central tasks in managing, monitoring and mining data streams is that of outliers. Outliers also called deviants are based on the fundamental statistical concept of standard deviation or variance. Deviants are values whose removal from dataset leads to an improved compressed representation of the remaining items. Identifying deviants on a massive data stream using very small space at any instant is a challenge for researchers. The goal in outlier or anomaly detection is to find data points that are most different from the remaining points in the dataset. Most outlier detection algorithms are schemes in which the distance between every pair of points is calculated, for example the Euclidean distance, and the points that are most far apart are marked as outliers. Algorithms that use such an approach for identifying outliers are quadratic, i.e. their time efficiency is in the order of $O(n^2)$ and also they operate on a static dataset. Such approaches are difficult to extend to the distributed streaming datasets. Points in these datasets arrive at multiple distributed end-points (nodes), which may or may not be the compute nodes, and must be processed incrementally. Such constraints lead us away from purely distance-based approaches, and towards more heuristic techniques. The central issue in many anomaly detection systems is to identify anomalies in real-time.
or as close to real time as possible thus making it a natural candidate for many streaming applications. Moreover, most often the data is produced at geographically disparate sites making distributed stream mining a more probable candidate for this domain.

1.5 FREQUENT PATTERN MINING IN DATA STREAMS

Frequent patterns are itemsets, subsequences, or substructures that appear in a dataset with frequency no less than a user-specified threshold. An item (I) is defined based on the context of data. In market basket data, ‘I’ is an item in the store like milk, bread, and butter. In relational data, ‘I’ is an attribute-value pair, like gender-male, gender-female. If there are some numeric attributes like age, they should be discretized. Numeric attribute age can be discretized as age-youth, age-middle, age-old. An itemset is a conjunction of items, I_1 \land I_2 \land \ldots I_n. A frequent itemset is a set of items whose frequency of occurrence is not less than a minimum threshold specified by the user. A subsequence, such as buying a PC first, then a digital camera, and then a memory card, which occurs frequently in a shopping history database, is a sequential pattern and it is considered to be a frequent pattern if its frequency of occurrence is greater than or equal to a user-specified threshold.

A substructure can refer to different structural forms, like sub-graphs, sub-trees, or sub-lattices, which may be combined with itemsets or subsequences. Let g_1 = (V_1, E_1, L_1) and g_2 = (V_2, E_2, L_2). g_1 is sub-graph of g_2, written g_1 \subseteq g_2, if

1. V_1 \subseteq V_2,
2. E_1 \subseteq E_2,
3. L_1(v) = L_2(v), for all v \in V_1 and
4. L_1(e) = L_2(e) for all e \in E_1

where V_1, V_2 are set of vertices, E_1, E_2 are set of edges, L_1(v), L_2(v) are set of vertex labels and L_1(e), L_2(e) are set of edge labels.

If a substructure occurs frequently in a graph database, it is called a frequent structural pattern. Specifically a sub-graph is frequent if its support (occurrence frequency) is no less than a minimum support threshold. A sub-tree of an undirected graph G is an acyclic connected sub-graph of G. Substructure mining can be used to mine biochemical structures, perform program control flow analysis, mine
XML structures and to perform graph classification, clustering, comparison and correlation analysis.

Finding frequent patterns plays an essential role in mining associations, correlations, and many other interesting relationships among data. Moreover, it helps in data indexing, classification, clustering, and other data mining tasks as well [16]. Thus, frequent pattern mining has become an important data mining task and a focused theme in data mining research.

Tremendous and potentially infinite volumes of data streams are often generated by real-time surveillance systems, communication networks, Internet traffic, online transactions in the financial market or retail industry, electric power grids, industry production processes, scientific and engineering experiments, remote sensors, and other dynamic environments. Unlike traditional datasets, stream data flow in and out of a computer system continuously and with varying update rates. It may be impossible to store an entire data stream or to scan through it multiple times due to its tremendous volume. To discover knowledge or patterns from data streams, it is necessary to develop single-scan and on-line mining methods [54]. Such algorithms can guarantee only approximate results. In the case of data streams, one may wish to find the frequent datasets either over a sliding window or the entire data stream. The problem of frequent pattern mining can be studied under several Models.

1.5.1 Entire Data Stream Model

In this model, the frequent patterns need to be mined over the entire data stream. Thus, the main difference from a conventional pattern mining algorithm is that the frequent patterns need to be mined in one pass over the entire data stream. Most frequent pattern mining algorithms require multiple passes in order to estimate the frequency of patterns of different sizes in the data. A natural method for frequent pattern counting is to use sketch-based algorithms in order to determine frequent patterns. Sketches are often used in order to determine heavy-hitters in data streams, and therefore, an extension of the methodology to the problem of finding frequent patterns is natural. This model does not allow false negatives, but may miss some of the frequent patterns. The main advantage of such a technique is that it is possible to
provide a more concise set of frequent patterns at the expense of losing some of the patterns with some probability which is quite low for practical purposes.

1.5.2 Sliding Window Model

In many cases, the data stream may evolve over time, as a result of which it is desirable to determine all the frequent patterns over a particular sliding window. The main assumption of this approach is that the numbers of frequent patterns are not very large, and therefore, it is possible to hold the transactions in each sliding window in main memory. Here window is divided into small basic windows and only store a synopsis and timestamp for each portion. When the time stamp of oldest window expires, its synopsis is removed, a fresh window is added to the front and the aggregate is incrementally recomputed [52]. The design of sliding window is based on the fact that people are often interested in recent changes at a fine granularity, but long term changes at a coarse granularity.

1.5.3 Damped Window Model

Pure sliding windows are not the only way by which the evolution of data streams can be taken into account during the mining process. A second way is to introduce a decay factor into the computation. Specifically, the weight of each transaction is multiplied by a factor of $f < 1$, when a new transaction arrives [56]. The overall effect of such an approach is to create an exponential decay function on the arrivals in the data stream. Such a model is quite effective for an evolving data stream, since recent transactions are counted more significantly during the mining process [87]. Specifically, the decay factor is applied only to those itemsets whose counts are affected by the current transaction. However, the decay factor will have to be applied in a modified way by taking into account the last time that the itemset was touched by an update. This approach works because the counts of the itemsets reduce by the same decay factor in each iteration, as long as a transaction count is not added to it. Such an approach is also applicable to other mining problem, where statistics are represented as the sum of decaying values.
1.6 CHALLENGES OF FREQUENT PATTERN MINING IN DATA STREAMS

Frequent pattern mining in data streams faces many challenges. Some of the important challenges are discussed below.

Frequent pattern mining in data streams needs to search a space with an exponential number of patterns. The number of patterns in the answer set itself is very large. To reduce the huge set of frequent patterns generated in data mining while maintaining the high quality of patterns, recent studies have been focusing on mining a compressed or approximate set of frequent patterns [81]. In general, pattern compression can be divided into two categories: lossless compression and lossy compression, in terms of the information that the result set contains, compared with the whole set of frequent patterns. Mining only Maximal frequent itemsets is a lossy compression and mining closed frequent itemsets is lossless compression [52]. This pattern compression becomes mandatory while handling data streams as the amount of data streaming in is unlimited and as a result the number of frequent patterns keeps increasing in leaps and bounds.

Frequent pattern mining relies on downward closure property to prune infrequent patterns and mine frequent ones. Downward closure property, otherwise called Apriori property, states that every subset of a frequent itemset is also frequent. This rule essentially says that there is no need to find the count of an itemset, if all its subsets are not frequent [69]. This is made possible because of the anti-monotone property of support measure which states that the support of an itemset never exceeds the support of its subsets. Applying this property for pruning is a very computation intensive process even in static databases and this becomes a very serious problem in data streams.

Data stream mining algorithms are designed to provide only approximate results. More accurate results need more memory and computations. When data is the key focus, then the idea is to examine only a subset of the whole dataset or to transform the data vertically or horizontally to an approximate smaller size data representation. Sampling, sketching, and load shedding are some of the data based techniques. Sampling refers to the process of probabilistic choice of a data item to
be processed or not. Sampling is an old statistical technique that has been used for a long time [59]. Many data stream problems cannot be solved with just a sample. Instead, data structure can be used which, in effect, include a contribution from the entire input, rather than just the items picked in the sample.

For example, consider trying to count the number of distinct objects in a stream. It is easy to see that unless almost all items are included in the sample, then we cannot tell whether they are the same or distinct. Sketch is referred as a compact data structure which summarizes the stream for certain types of query. For classes of data stream queries where no exact data structure with the desired properties exists, one can often design an approximate data structure that maintains a small synopsis or sketch of the data rather than an exact representation, and therefore is able to keep computation per data element to a minimum.

Load shedding refers to the process of dropping a sequence of data streams. In this, data points are picked selectively from the stream, without losing accuracy. It has been used successfully in querying data streams. All these techniques provide only approximate results. So a mining algorithm needs to provide users the flexibility to control the accuracy of the final mining results.

Many mining algorithms like Apriori make multiple passes over the data to mine frequent itemsets. But frequent pattern mining in data streams should be accomplished in a single-pass, since streaming data cannot be stored permanently making multi-pass algorithms impossible [55]. Permanent storage is impossible because of the huge unbounded volume of streaming data. So algorithms used for mining frequent patterns from static databases cannot be employed in data streams. Modified algorithms that require only one pass have to be designed.

Trying to maintain and deliver the potentially frequent patterns at any time involves very high computational costs. If the data stream arrives very rapidly, this could be unrealistic too. When the stream of data arrives at a very quick rate, processing and extracting patterns from them becomes impossible. Some information may be lost in this process. Owing to this fact special attention has to be paid to make the processing of the flowing information very fast to handle the rapid arrival of data.
Mostly stream mining involves approximate algorithms [53]. Specifying an error bound that is practical and providing adaptability in the mining algorithm is an important issue in such algorithms. The user has to specify an error bound which makes it possible to mine frequent patterns from the data streams but still ensure sufficient accuracy of the results. Great care has to be taken to choose an appropriate error bound according to the required accuracy level of the underlying application.

Handling fixed size and varying sized tuples in the transactions of the data stream is also an important issue. When the tuples are of fixed-size the number of 2-itemsets in each transaction is also fixed. So handling of the first part of frequent pattern mining algorithm, generation of 2-itemsets, is simplified. But in varying length transactions, which is more common in real world data streams, the number of 2-itemsets depends on the length of the transactions and it keeps varying with the arriving transactions. This leads to a more complex procedure for identifying the 2-itemsets in the transactions as the number of 2-itemsets per transaction is not constant.

1.7 MOTIVATION FOR THIS RESEARCH

Some real world applications which motivated this research work are described in this section.

Warehouse Monitoring System in an industry

A warehouse containing temperature-sensitive merchandise like crackers and explosive materials may deploy temperature sensors to report the current temperature regularly to a monitoring system. Each sensor is assigned its own identifier (ID), and generates its own data stream and sends it to the monitoring system. A data item from a sensor contains the sensor ID, the current time and the current temperature. The monitoring system takes action and reports immediately if the count of distinct sensors that report temperatures over 75F per hour crosses a predefined threshold value. The monitoring system has to read the temperature report from all sensors, group them by time reported (reporting hour is used for grouping), eliminate duplicate entries. If the sensor readings are taken more than once in an hour, entries with same sensor id within the group are eliminated. While eliminating such entries, care is taken not to eliminate an entry with reported temperature above
75F. Finally the number of sensors reporting more than 75F in each hour is counted. This sort of processing is not possible in standard DBMS. First, duplicate elimination may require an unbounded amount of information to be maintained. Thus, if the information required for the query is kept in memory, the system will ultimately run out of memory. This makes data stream mining an essential process in this application.

**Network Traffic Monitor**

Communication networks produce vast amounts of data that help network operators manage and plan their networks. They also help the researchers study a variety of network characteristics. The data is rich and it varies from the minute to the massive. Minute data may be in the size of kilobits per hour giving the periodic average link utilization. Massive data may be in the size of gigabits per second of live traffic captures. This data can help in network monitoring, trouble shooting, and reactive routing. On long runs, this data may be used for network traffic engineering, in which the operators try to shift the flow of traffic away from over utilized links onto less busy links for load balancing and effective bandwidth utilization. This data may be accumulated for months and may be used for capacity planning also.

Common network protocols define structure over data in the form of packets. For example, an Internet Protocol (IP) packet contains the source IP address, the destination IP address, the fragment offset (if fragmented), flags, and a packet identifier. A Transmission Control Protocol (TCP) packet includes everything in an IP packet as well as source port number, destination port number, and other information. Many applications process streams of network packets to monitor usage, analyze performance, and detect security issues such as intrusions. Also they may be analyzed to identify peak time traffic, identify bottlenecks and overcome them to improve bandwidth usage and response time of packets.

Network packets may be fragmented after they leave their source and they can be reassembled at the destination. A network packet is assigned an identifier by its source so that the receiver can identify which fragments belong to which packet. Each fragment of the packet contains the identifier of the packet, an offset
specifying in bytes where it belongs in the reconstructed packet, and a flag
specifying if it is the last fragment for a packet. One known attack to bypass a
network firewall takes advantage of fragmented network packets. Network firewalls
are designed to only allow network traffic for specific ports to pass through. For
example, a firewall might allow packets destined for port 80, the port designated for
Hyper Text Transfer Protocol (HTTP) to pass through, but block packets destined for
port 23, the port designated for Telnet. One known strategy for bypassing a firewall
from a remote system in order to send packets to a blocked port proceeds as follows:

- The first fragment for a network packet is output with fragment offset set to 0
  and with the destination port for the entire packet set to 80
- The firewall allows this fragment to pass through, along with any subsequent
  fragments with the same network packet identifier, to be reassembled by the
  host inside the firewall
- One of the subsequent fragments has a fragment offset of 0 (again), this time
  with a destination port of 23, which is a blocked port for this particular
  firewall
- The destination port is not checked as it is a fragment of a previously
  permitted package. The packet is reassembled inside the host and sent to port
  23 instead of port 80, and the hacker has reached the blocked port

Since packets will arrive from many sources, it cannot be assumed that
the identifier values will arrive in order. Further, fragments from many sources (with
different identifiers) may arrive at the same time. Therefore, fragments with different
identifiers will be mixed together in a data stream. It cannot be assumed that
fragments for the current packet have all arrived just because a fragment with a new
identifier has arrived. This leads to the need for special data stream mining
algorithms to overcome the above reported issues. For such applications, network
operators must ensure availability and good performance in the face of constantly
changing conditions like changing traffic patterns, link failures and security threats
like worms, viruses, Trojan horses, hackers, and Denial of Service attacks. A
real-time stream mining analysis facility is thus a necessary requirement for such
applications.
**Online Auction**

An online auction is an auction which is held over the internet. Online auctions come in many different formats, but most popularly they are ascending English auctions, descending Dutch auctions, first price sealed bid, Vickrey auctions, or sometimes even a combination of multiple auctions, taking elements of one and forging them with another. The scope and reach of these auctions have been propelled by the Internet to a level beyond what the initial purveyors had anticipated. This is mainly because online auctions break down and remove the physical limitations of traditional auctions such as geography, presence, time, space, and a small target audience. This influx in reach ability has also made it easier to commit unlawful actions within an auction [40]. In 2002, online auctions were projected to account for 30% of all online e-commerce due to the rapid expansion of the popularity of electronic commerce.

There are a number of examples of commercial and research systems for processing and monitoring online auctions, including EBay, uBid, Bidz.com, Online Auction, WeBidz, and the eBid. Software agents may be used to represent humans in the auction to bid on or sell items. A user first registers with the system through an agent, then participates in auctions as a buyer or seller. The online auctions can be modeled with three kinds of stream sources (which can be implemented as software agents) that supply data to an auction monitoring system. This architecture is shown in Fig.1.2. Bids for items currently for sale arrive on the Bid stream.

New items for sale arrive on the Auction stream. New users arrive on the Person stream. A relational database is used to store auction history. However, systems that try to use a database to track active auctions face two issues: First, because all auctions read and store information in the same database, the database itself becomes a bottleneck. Second, synchronization issues arise when multiple system agents try to further the progress of an auction. For example, bids may arrive for an auction that has expired but not yet marked as closed by the system. When the system does mark the auction closed, care must be taken to ensure that only those bids that arrived while the auction was active should be considered. These two issues can be avoided by processing active auction data as it arrives.
The increasing popularity of using online auctions has led to an increase in fraudulent activity. This is usually performed on an auction website by creating a very appetizing auction, such as a low starting amount. Once a buyer wins an auction and pays for it, the fraudulent seller will either not pursue with the delivery, or send a less valuable version of the purchased item (replicated, used, or refurbished). Protection to prevent such acts has become readily available, most notably Paypal's buyer protection policy. As Paypal handles the transaction, they have the ability to hold funds until a conclusion is drawn whereby the victim can be compensated. Online auction websites are used by thieves or fences to sell stolen goods to unsuspecting buyers.

According to police statistics there were over 8000 crimes involving stolen goods, fraud or deception reported on eBay in 2009. It has become common practice for organized criminals to steal in-demand items, often in bulk. These items are then sold online as it is a safer option due to the anonymity and worldwide market it provides. Auction fraud makes up a large percentage of complaints received by the FBI’s Internet Crime Complaint Center (IC3). This was around 45% in 2006 and 63% in 2005. For handling such frauds, continuous monitoring of the

---

Fig.1.2 Simple architecture for an online auction monitoring system
online auction data streaming in every instant from the websites is essential. Thus stream mining is essential to prevent frauds in online auctioning and also plan for future enhancements in auctioning.

E-commerce and E-business applications

Electronic commerce, commonly known as E-commerce or eCommerce, is trading in products or services conducted via computer networks such as the Internet. Electronic commerce draws on technologies such as mobile commerce, electronic funds transfer, supply chain management, Internet marketing, online transaction processing, electronic data interchange (EDI), inventory management systems, and automated data collection systems. Modern electronic commerce typically uses the World Wide Web at least at one point in the transaction's life-cycle, although it may encompass a wider range of technologies such as e-mail, mobile devices, social media, and telephones as well.

Electronic commerce is generally considered to be the sales aspect of any e-business. It also consists of the exchange of data to facilitate the financing and payment aspects of business transactions. This is an effective and efficient way of communicating within an organization and one of the most effective and useful ways of conducting business. It is a Market entry strategy where the company may or may not have a physical presence.

Data stream mining can help organizations to identify the information that would be most suitable to put on the Web by conducting an analysis of competitors and potential customers, and determining the expertise of the organization. An organization in order to get onto the World Wide Web needs to decide what information to put on its web site. The web site will assist online users in their decision making by listing for instance other books that have been bought by people who also bought the particular book that the online user is currently looking at. One of the most useful applications of data mining in an e-commerce environment can be found on the Amazon.com web site. The web site also makes use of data mining to bundle books that are often bought in pairs together and to sell or market these at reduced price if they are bought together. Amazon.com also uses data mining to profile its customers, and online users who often return to the web site after having
bought a book in each of their visit, will be provided with lists of other book titles that they might be interested in based on the category of book that they bought [71].

Marketing and sales were two of the early business functions that drove the development of the data mining. The following are some of the data mining application relevant to these functions:

- Performing targeted marketing
- Determining the marketing strategies of competitors
- Prediction of sales trends
- Market segmentation
- Lifestyle behavior analysis
- Online sales support
- Analyzing or predicting customer reaction to promotions
- Market basket analysis
- Document automation in supply chain and logistics
- Domestic and international payment systems
- Group buying
- Automated online assistance
- Online shopping and order tracking
- Shopping cart software

For many of these applications, stream mining analysis is very essential to make them beneficial to the organizations. Thus stream mining plays an important role in e-commerce and e-business applications.

*Telecommunication industry*

The telecommunications industry was one of the first to adopt data mining technology. This is most likely because telecommunication companies routinely generate and store enormous amounts of high-quality data, have a very large customer base, and operate in a rapidly changing and highly competitive environment. Telecommunication companies utilize data mining to improve their marketing efforts, identify fraud, and better manage their telecommunication networks. The telecommunication industry offers an attractive domain for data stream mining applications due to the data intensive nature of applications employed
in this industry and can be viewed as a means of automatically generating knowledge directly from the data. Also these companies face additional challenges due to the sequential and temporal aspects of their data, and the need to predict very rare events like customer fraud and network failures in real-time.

Telecommunication companies maintain data about the phone calls that traverse their networks in the form of call detail records, which contain descriptive information for each phone call. Millions of such records are generated every day and, because they are kept online for several months, this meant that billions of call detail records are readily available for data mining. Such call detail records contain data that is useful for marketing and fraud detection applications.

Telecommunication companies also maintain extensive customer information, such as billing information, as well as information obtained from outside parties, such as credit score information. This information can be quite useful and often is combined with telecommunication-specific data to improve the results of data mining. For example, while call detail data can be used to identify suspicious calling patterns, a customer’s credit score is often incorporated into the analysis before determining the likelihood that fraud is actually taking place.

Telecommunications companies also generate and store an extensive amount of data related to the operation of their networks. This is because the network elements in these telecommunication networks have some self-diagnostic capabilities that permit them to generate both status and alarm messages. These streams of messages can be mined in order to support network management functions, namely fault isolation and prediction.

The following are some of the major applications of data stream mining in the telecommunications industry:

- Determining travel routes and giving advice to wireless users during peak hour travel times
- Data mining can be used to assist network operators with network intrusion detection and fault monitoring
- Identifying and understanding the critical issues that determine client loyalty
- Narrowing the focus and increasing the effectiveness of marketing campaigns
- Forecasting network behavior
- Call tracking
- Churn management
- Fraud detection

**Law Enforcement Industry**

Data mining tools can be used to detect unusual patterns, suspicious behavior and unauthorized intrusions, all of which can be extremely helpful not only in aiding companies and government in their law enforcement efforts, but also in their counter-terrorism efforts. The following includes some of the specific applications of data mining in counter-terrorism:

- Intrusion detection
- Insider threat analysis
- Identifying terrorists

Law enforcement systems often need to perform searches on person and organization names. The names could be those of criminals, suspects, witnesses, missing persons, registered offenders, or gangs. Regardless of format, country of origin, or quality and completeness of the data, the identity resolution in law enforcement should support both fast on-line inquiries and comprehensive batch matching to be effective. This necessitates stream mining algorithms to be applied to such data for quick results.

**Financial Services Industry**

Across the global financial system, data doesn't just come in waves but as an incessant blizzard. In fact, as one example of the volumes of data, some financial organizations consume market data at rates exceeding one million messages per second, twice the peak rates they experienced only a year ago. But without a business intelligence architecture that can process vast amounts of data in any format, in real time, the financial services enterprise could be lost in such whiteouts.
The major benefit offered by data mining to the financial services industry is trying to predict the future. The financial services industry can benefit a great deal from data mining, especially with regard to the major role that Business Intelligence and Customer Relationship Management play in this industry. Typical data mining applications in the financial services industry include the following:

- Analyzing stock market quotes
- Providing clients with customized investment advice
- Loan approvals
- Risk classification
- Identifying suspicious transactions
- Risk management
- Provide security and information confidentiality for shared information
- Respond quickly to events and changing requirements
- Adapt rapidly to changing data forms and types
- Continuously analyze data at rates that are orders of magnitude greater than existing systems

1.8 PROBLEM STATEMENT

As frequent pattern mining is an essential data mining task, developing efficient frequent mining techniques has been an important research direction in data mining. The general goal of this research is to develop algorithms to:

- Construct Synopsis of data stream of transactions
- Mine frequent patterns
- Prune infrequent patterns
- Compress frequent patterns

As an inherent problem, frequent pattern mining over data stream has to deal with enormous data. Practically it is not possible to store the whole data, so synopsis of data stream is constructed using numeric representation technique. Once the synopsis is constructed it is analyzed for mining frequent itemsets. Mining association between these frequent itemsets gives frequent patterns. Not only
frequent patterns but infrequent patterns also have important applications like intrusion detection or outlier detection. The number of frequent itemsets generated as part of the pattern mining grows exponentially as the volume of data increases and the threshold is reduced. To handle this frequent itemsets have to be compressed and stored. For this two options are available (Maximal and closed itemsets). Closed itemsets representation is chosen in this research work as it is a lossless form of compression and also it is essential to obtain the exact support of all frequent itemsets that is not possible from maximal itemsets. From maximal itemsets all frequent itemsets can be extracted but their exact support is not available. Closed itemsets provide both.

In this research datasets are simulated as data stream of transactions. Data stream is offline and arrival only model of data stream is considered for analysis. Both landmark window model and sliding window model have been used to handle the data streams and the pros and cons of each model have been analyzed in this research work.

1.9 SCOPE OF THE RESEARCH

This research work aims at improving the performance of frequent pattern mining algorithms over data streams specifically web click stream data. For this sliding window model is used. The memory usage is largely reduced by using bit-representation of the transactions. To reduce the need for storing explosively large number of frequent patterns closed frequent itemset mining from landmark and sliding windows are proposed. These algorithms can be applied in a variety of data mining applications. Web browsing experience can be improved by deciding the pages to cache using the frequent patterns retrieved by mining the web log data.

Analyzing navigation history of a web site and that of a user reported in log files and web click stream helps in determining the components to be solicited in future. Frequent Itemset Mining from such data enhances the process. Processing and analyzing the vast amounts of click stream data and web logs that are generated at very high rates in a smart and cost-efficient way is a daunting challenge to the data mining community. Mining frequent itemsets (MiFI) plays an important role in analyzing such data streams. In spite of the existence of many such mining
algorithms, more time and space efficient algorithms for mining frequent patterns are the need of the hour and are attracting wide attention in web click-stream analysis in recent years [8].

Given the high traffic in the Internet, frequently visited web pages by a client may be stored in a cache which helps to reduce the perceived latency when the user requests a page. These pages may be pre-fetched and this proves to be an important technique to reduce latency, bandwidth consumption and server load thereby increasing the system performance. Building and maintaining a relevant cache with a suitable pre-fetching policy can enhance performance to a significant extent. The frequently visited pages by a client may be identified by studying the user-profile in the web log of the servers and counting the occurrence of web pages visited by the user in a particular session. The navigation history of a web site which is maintained in its log files can be analyzed to determine the web pages that will be requested in the near future and pre-fetch it into the cache.

Frequent access patterns of a client can be recognized by studying the pages accessed by the user in a session. Session information is not available in the web log. Individual sessions of a client can be identified by using a suitable timeout mechanism. A gap-based approach may be suitable for this. It has been observed after extensive study on web traces that if the time interval between two requests by the same user is larger than twenty minutes, then the second request is not related to the first. Also this time interval is sufficient as it reflects the way session timeouts are measured on many websites.

This pre-fetching and storing of web pages in the cache may be performed by the Internet Service Providers (ISP). The ISPs maintain two sets: always-pre-fetch set and miss-pre-fetch set. If the web pages in the always-pre-fetch set are not in the cache maintained by the ISP, then during the light-traffic time interval, say early morning or late night, the ISP may pre-fetch the desired web pages and cache them. The always-pre-fetch set is determined by identifying the frequent patterns in a session by different users. The common frequent patterns of many clients may be identified; Association Rules (ARs) can be formed from them. From these association rules, the rules that are of the following form may be identified.
**User-id → url1**

If a significant number of ARs of this form with different user-ids in the antecedent are identified then ‘url1’ is definitely in the always-pre-fetch set. The miss-pre-fetch set is constructed by identifying ARs of the form shown below

**User-id → set of URLs**

If one Uniform Resource Locator (URL) request in the consequent part of the rule results in a cache-miss, then all URLs in the consequent part may be pre-fetched, assuming that the user with the user-id in the antecedent part will be requesting them in the near future. The association rules thus formed should be refreshed periodically to reflect the changing user-access patterns. To achieve this, time sensitive sliding window model used in the FIM-CQTimeSWin algorithm will be most appropriate. One session of a client may be considered as one transaction. Each web page requested by the user is an item. The time unit is decided by the ISP. When the sliding window is full, the oldest time unit is cut off and the new time unit is added.

Some preprocessing is required for Mining Frequent Itemsets (MiFls) from this dataset. The duplicates, which represent repeated visits to the same page, have to be removed in each sequence and then the sequence should be ordered. If a time-sensitive sliding window based algorithm is used for processing the web log data, the time duration for a time-unit has to be fixed. This is taken as one hour. One time unit consists of all sessions of different users in a one-hour time period. Each user’s session may last for different time periods. If a user’s session exceeds one hour, the current session is terminated and a new session is commenced to enable the construction of the time unit. This causes each time unit to have different number of sessions depending on the time for which each session lasts.

In web logs, each entry will have the time of access recorded as one of the fields. This can be used to separate the entries into different time units. The size of the sliding window is set according to the user’s wish depending on how the analysis is to be performed. If the user wants to analyze three hour details then the sliding window size may be set to three, so that the frequent itemsets in every 3 hour period of the dataset can be analyzed.
New algorithms are proposed in this research work that employ sliding windows or landmark windows as dictated by the applications. They may be used to identify web pages frequently visited by analyzing the preprocessed web log files appearing as web click stream data [99]. The algorithms generate a list of frequent itemsets (frequently visited web pages). And if this list is very huge, a compressed form of frequent itemsets called Closed Frequent Itemsets (CFIs) is generated.

1.10 ORGANIZATION OF THESIS

The remainder of the thesis is organized as follows: The second chapter gives the literature survey conducted as an essential preliminary work of this research. The third chapter describes FI mining in data streams. It elaborates on the issues involved in frequent itemset (FI) mining from data streams. New algorithms for mining FIs from data streams (FIM-CQTransSWin and FIM-CQTimeSWin) are proposed. These algorithms employ transaction compression technique. Both are incremental algorithms that employ the sliding window model and use a circular queue to represent the sliding window.

The fourth chapter describes a new incremental algorithm, called HATCI, which mines Closed Frequent Itemsets (CFI) from a data stream using a sliding window model. It also elaborates on the need for closed itemsets (CI). HATCI employs hash tables to store CIs, its supersets and subsets that provide efficient retrieval. These tables also help efficient updates when new transactions arrive and the existing CIs have to be updated in an incremental fashion instead of redoing the entire processing from the scratch.

The fifth chapter is also devoted to Closed Frequent Itemset (CFI) mining in data streams but uses the landmark window model to cater to the needs of applications that need to query past data. A new algorithm to mine the CFIs, called FOCIT is proposed, which uses tree representation of the Closed Itemsets and an incremental update mechanism to handle new arrivals on the data stream. Efficient schemes for building and maintaining the trees are employed.

The sixth chapter provides a description of an interesting application, web log mining that can be made more efficient and effective by applying the algorithms developed in the current work. As millions of visitors interact daily with
Web sites around the world, massive amounts of data are being generated. And this information could be very precious to the company hosting the web site in the fields of understanding customer behavior, improving customer services and relationship, launching target marketing campaigns, measuring the success of marketing efforts, and so on. Web server log files are a huge repository of such data which help to trace the visitors’ on-line behaviors. For example, after some basic traffic analysis, the log files can help to identify the search engines frequently used by the visitors to the website and the pages that are the most and least popular in a particular website. Mining such web log files and analysis of the information mined can be made more efficient and effective by applying the algorithms developed in the current work.

This chapter also describes a data mining tool, Waikato Environment for Knowledge Analysis (WEKA) to prepare the input file in the required format. The last chapter provides a conclusion on this research work and also discusses some future directions for extending the research work.