Abstract

The Crime investigation department have very significant role in police system in all countries in the world. Computer systems are placed in almost all police stations to store and retrieve the crimes and criminal data and for subsequent reporting. For crime analysis, it is a challenging, time-consuming to determine who have been committed crimes from the large set of crimes that are happening every year. Our goal is to identify recent popular crime patterns from large set of crime databases that are updating every day. To do this, we are proposing a popular crime pattern detection algorithm from incremental crime databases. Such data mining technologies are helpful to design proactive services to reduce crime incidences in the police stations jurisdiction. Our experiment results show the efficiency in time consuming for mining popular crime patterns.

Keywords: crime patterns, popular patterns, crime database, sliding window

1. Introduction

Data mining can be used to model crime detection problems in a simple and easy way. Crimes are social nuisances and cost our society in various ways with respect to human values. The high volume of crime datasets and also the complexity of relationships [1] between these kinds of data have been made criminology as an appropriate field for applying data mining techniques. The first step is to identify the crime characteristics for developing further analysis. From the knowledge that is gained from data mining approaches is a very useful tool which can help and support police forces. The idea here is to try and capture years of human experience into computer models via data mining models. Now these days the criminals are becoming technologically sophisticated in committing crimes. So police needs such a crime analysis to catch criminals and to remain ahead in the eternal race between the criminals and the law enforcement [1]. In this paper we proposed a new algorithm called IncPCrime-growth algorithm to mine popular crime patterns in incremental databases.

The rest of the paper is organized as follows. In section 2 related work is discussed. In section 3 we define the problem of popular crime patterns in incremental crime databases. Then we describe the process of mining popular crime patterns by IncPCrime-growth algorithm and how to construct IncPCrime-tree structure for mining popular crime patterns in section 4. Our experimental results have been shown in section 5. Finally the paper has been concluded in section 6.

2. Related Work

Association rule mining is one of the most significant techniques in data mining. Market basket analysis is one of the best example for association rule mining. Frequent pattern mining and sequential pattern mining techniques [8] are core components for association analysis. Classic Apriori algorithm is one tool to learn association rules between different sets of data and to find patterns in data. But Apriori takes number of scans of database for processing and it produces large number of candidate keys. Han et.al [12] proposed FP-growth algorithm to avoid the problem of number of scans of the database and generation of candidate keys. FP-growth algorithm requires two scans of database and tree structure was used in the mining process. Leung C.K.S. et. al [13] [14] introduced popular pattern mining in transactions and in popular friends in social groups.

J. De Bruin et.al [2] introduced a tool which generates profiles of criminals based on nature of crime, frequency, severity and duration. Regional Crime Analysis Program (RECAP) used data mining techniques and data fusion to extract knowledge about crime activities [3]. This provides significant information about crime activities and also alerts the user if activity is unmanageable. They used Kernal density technique for determination of crime in different locations. et.al [4] proposed a model by using K-Means clustering to mine crime patterns in real time data where data is collected from Sheriff’s office. This model is used for clustering of large data sets and it provides support to get knowledge from pattern analysis. L.Ding et.al [5] proposed Perp.
search based on LETS and mainly it has four components as social network analysis, geographic profiling, physical matching and crime patterns. B Chandra et.al [6] proposed dynamic wrapping method for traditional clustering approach for multi dimensional data. It has the drawback of multivariate data with different values. To overcome this Parametric Minkowski model is introduced for finding distance matrix. V. H. Bhat et.al [7] proposed frame work to collect data from various devices. Data is collected from flash drive and ETL is used for data processing. In this K-Means algorithm is used for clustering and C4.5 algorithm is used for classification.

3. Problem Statement

A crime database usually contains some distinct crime attributes which cover the location of the incident, suspect, victim information, day of the week, time, date, weapons used, and crime scene status etc., a few can be seen in our crime database (CDB) [9] can be seen in Table 1. From the existing crime data base we have to manage and identify the attributes of interest that are available. Then the database is queried to extract the interesting attributes, to create the Crime Transaction Database (CTDB). Table 2 has to be mined for extracting the latest popular crime patterns. Now we define the basic definitions of our problem in this section [10]:

Crime Transaction Popularity: The crime transaction popularity i.e., Pop(C, ct) of a pattern C in crime transaction ct measures the membership degree of C in crime transaction ct. The membership degree will be computed based on the difference between the crime transaction length |ct| and pattern size |C|.

\[ Pop(C, ct) = |ct| - |C| \]

Long Crime Transaction Popularity: The Long Crime Transaction Popularity i.e., Pop(C, c\text{\text{max}CTL(C)}) of a pattern C in crime transaction c\text{\text{max}CTL(C)} measures the membership degree of C in c\text{\text{max}CTL} where c\text{\text{max}CTL}(C) is the crime transaction having maximum length in the database DB_C.

\[ Pop(C, c_{\text{max}CTL}(C)) = \max_{ct \in DB_C} |ct| - |C| \]

Popularity: The popularity i.e., Pop(C) of a pattern C in the CTDB measures an aggregated membership degree of C in all crime transaction in CTDB, which is defined as an average of all crime transaction popularities of C.

\[ Pop(C) = \frac{1}{|DB_C|} \sum_{ct \in DB_C} Pop(C, ct) \]

Popular Crime: The user given minimum popularity threshold is min_pop. The crime pattern C is said to be popular if its popularity is greater than or equal to min_pop i.e., Pop(C) \geq min_pop.

The Crime-item={c_1, c_2, c_3, ….., c_m} be a set of m domain items. The set of n crime transactions in

### Table 1. A Crime Database (CDB)

<table>
<thead>
<tr>
<th>Location</th>
<th>Crime Incident</th>
<th>VictimID</th>
<th>Gender</th>
<th>Time</th>
<th>Culprit</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area 1</td>
<td>Robbery</td>
<td>A2</td>
<td>M</td>
<td>Noon</td>
<td>gang</td>
<td></td>
</tr>
<tr>
<td>Area 2</td>
<td>Kidnapping</td>
<td>D4</td>
<td>M</td>
<td>Noon</td>
<td>gang</td>
<td></td>
</tr>
<tr>
<td>Area 3</td>
<td>Robbery</td>
<td>Z8</td>
<td>F</td>
<td>Night</td>
<td>individual</td>
<td></td>
</tr>
<tr>
<td>Area 4</td>
<td>Rape</td>
<td>P3</td>
<td>M</td>
<td>Night</td>
<td>gang</td>
<td></td>
</tr>
</tbody>
</table>

### Table 2. Crime Transactional Database (CTDB)

<table>
<thead>
<tr>
<th>Crime Id</th>
<th>Crime Incidents</th>
</tr>
</thead>
<tbody>
<tr>
<td>ct 1</td>
<td>c_2, c_3, c_4</td>
</tr>
<tr>
<td>ct 2</td>
<td>c_1, c_6, c_7, c_8</td>
</tr>
<tr>
<td>ct 3</td>
<td>c_2, c_4, c_5, c_6, c_7, c_8</td>
</tr>
<tr>
<td>ct 4</td>
<td>c_1, c_3, c_5, c_9</td>
</tr>
<tr>
<td>ct 5</td>
<td>c_2, c_5, c_7</td>
</tr>
<tr>
<td>ct 6</td>
<td>c_4, c_6, c_7, c_8</td>
</tr>
<tr>
<td>ct 7</td>
<td>c_1, c_3, c_6, c_9</td>
</tr>
</tbody>
</table>
crime transactional database (CTDB) are \(\{ct_1, ct_2, ct_3, \ldots, ct_n\}\) where each \(ct_j\) in CTDB is a subset of each Crime-item in crime database. Let \(|ct_j|\) represents the length of crime transaction \(ct_j\). \(C = \{c_1, c_2, \ldots, c_k\}\) \(\subseteq\) Crime-item or Crime-pattern consisting of \(k\) items (\(k\)-itemset) where \(|C| = k \leq m\). The projected database of \(C\) denoted as \(DB_C\) and it is a set of CTDB transaction that contain \(C\). Here \(maxCTL(C)\) and \(sumCTL(C)\) are used to represent maximum crime length and total crime length respectively, for all crime transactions in \(DB_C\).

**Popular Crime Patterns in Incremental Crime Databases**: Every day or every week new crimes or existing crimes may happen and be added to the existing crime databases. As the database is updating frequently, the popularity of crimes may change, so there is a need to mine recent popular crime patterns from the updating crime database. Let us consider that new crimes have been added to the current crime database \(CDB\). The updated crime database is denoted as \(UCDB\). In order to find latest popular crime patterns, it is time consuming to mine the crime database from the scratch. So, to avoid such time consuming environment, let us consider \(I\) be the number of crime records \(UCDB (c_8 - c_{10})\) added to the \(CDB (c_1 - c_{7})\), Table 3. After querying \(UCDB\), the latest crime records with interesting attributes will be updated in \(CTDB\) and becomes \(ICTDB\). Initially the first transaction i.e., null transaction \(t_{\text{first}}\) will be zero. Whenever \(I\) transactions are added, the \(t_{\text{first}}\) transaction will be \(t_{\text{first}} + I\) and the last transaction \(t_{\text{last}}\) will be \(t_{\text{last}} + I\). In this way the crime initial transaction and last transaction will change whenever the crime database updates. Now the newly defined crime database will be considered for mining to obtain new \(\text{IncPCrime-tree}\). From the new \(\text{IncPCrime-tree}\), recent popular crime patterns will be mined.

### Table 3. New Crime Records (UCDB)

<table>
<thead>
<tr>
<th>Crime Id</th>
<th>Crime Incidents</th>
</tr>
</thead>
<tbody>
<tr>
<td>ct_8</td>
<td>c_4, c_5, c_6, c_9</td>
</tr>
<tr>
<td>ct_9</td>
<td>c_1, c_2, c_3, c_5, c_8, c_9</td>
</tr>
<tr>
<td>ct_10</td>
<td>c_2, c_7, c_8</td>
</tr>
</tbody>
</table>

Three transactions are added to \(CTDB\). For time reduction we will not mine the crime database from the scratch. For that we

### Table 3. Incremental Crime Transactional Database (ICTDB)

<table>
<thead>
<tr>
<th>Crime Id</th>
<th>Crime Incidents</th>
</tr>
</thead>
<tbody>
<tr>
<td>ct_1</td>
<td>c_2, c_3, c_4</td>
</tr>
<tr>
<td>ct_2</td>
<td>c_1, c_6, c_7, c_8</td>
</tr>
<tr>
<td>ct_3</td>
<td>c_2, c_4, c_5, c_6, c_7, c_8</td>
</tr>
<tr>
<td>ct_4</td>
<td>c_1, c_3, c_5, c_9</td>
</tr>
<tr>
<td>ct_5</td>
<td>c_2, c_5, c_7</td>
</tr>
<tr>
<td>ct_6</td>
<td>c_4, c_6, c_7, c_8</td>
</tr>
<tr>
<td>ct_7</td>
<td>c_1, c_3, c_6, c_9</td>
</tr>
<tr>
<td>ct_8</td>
<td>c_4, c_5, c_6, c_9</td>
</tr>
<tr>
<td>ct_9</td>
<td>c_1, c_2, c_3, c_5, c_8, c_9</td>
</tr>
<tr>
<td>ct_10</td>
<td>c_2, c_7, c_8</td>
</tr>
</tbody>
</table>

4. **Mining of Popular Crime Patterns in Incremental Databases**

Frequent pattern mining satisfies the downward closure property i.e., if an item is considered as infrequent then its superset is also considered as infrequent. Such property will helps in reducing the search space by deleting the infrequent patterns to make the process of mining patterns to be fast. While mining popular patterns sometimes we can see that a crime pattern \(\{c_i\}\) is unpopular but its superset \(\{c_c, c_i\}\) can be popular which does not satisfy downward closure property [12] [13]. Therefore the mining of popular crime patterns may be a treated as a challenging step. The complete process of mining popular crime patterns was discussed in [12]. So in this paper we are going to mine the latest or recent popular crime patterns by using \(\text{IncPCrime-growth}\) algorithm whenever the crime database increments or updates. The proposing \(\text{IncPCrime-growth}\) algorithm also having two phases like \(\text{FP-growth}\) algorithm (i) construction of a \(\text{IncPCrime-tree}\) (ii) mining of popular crime patterns from \(\text{IncPCrime-tree}\).

Let us consider that three crime records are going to add the original crime database. We can see in Table 3.
increments. Now the latest defined crime database i.e., from c_{t4} to c_{t10} will be considered for constructing new pop tree and from the new pop tree the recent popular crime patterns will be mined. This process continues whenever the crime database updates.

Let the user given minimum popularity threshold, \textit{min\_pop} is 3.2. First calculate the popularity of each individual pattern with the below equation

\[
\text{Pop}(C) = \frac{\text{sum\_CTL}(C)}{|DB_C|} - |C|
\]

- \text{Pop}(c_1) = \frac{14}{3} - 1 = 3.6
- \text{Pop}(c_2) = \frac{12}{3} - 1 = 3
- \text{Pop}(c_3) = \frac{14}{3} - 1 = 3.6
- \text{Pop}(c_4) = \frac{8}{2} - 1 = 3
- \text{Pop}(c_5) = \frac{17}{4} - 1 = 3.2
- \text{Pop}(c_6) = \frac{12}{3} - 1 = 3
- \text{Pop}(c_7) = \frac{10}{3} - 1 = 2.3
- \text{Pop}(c_8) = \frac{13}{3} - 1 = 3.3
- \text{Pop}(c_9) = \frac{18}{4} - 1 = 3.5

Now compare with the given \textit{min\_pop} i.e., 3.2. The crime patterns \(c_2\), \(c_4\), \(c_6\), \(c_7\) are not popular crime patterns because the popularity is less than the given \textit{min\_pop}. So the above crime patterns are unpopular but its superset’s may be popular, which does not satisfy downward closure property. To handle such type of challenge we can redefine the popularity of a crime pattern \(C\) by super-pattern popularity check.

To construct a IncPCrime-tree we need to scan the ICTDB to find the support(\(c\)), maximum crime transaction length maxCTL(\(c\)) and the popularity Pop(\(c\)) for each item \(c\) in the ICTDB. Then, we can perform the super-pattern popularity check and safely delete a pattern \(c\) if PopUB(\(c') < \text{min\_pop}\) (where \(c'\) is an extension of \(c\)). We then scan the ICTDB the second time to insert each transaction into the IncPCrime-tree in a similar fashion as the insertion process of FP-tree.

While constructing the conditional database from a projected database, a super-pattern popularity check for extensions of any unpopular crimes can take place and deletes the crime \(c\) when it fails the check. Such pruning technique is called as the lazy pruning.

Now calculate for the \text{Pop}(\(c_4'\)) to check for their super-pattern popularity check with the below equation.

\[
\text{Pop}(C') = \text{maxCTL}(X) - |C'|
\]

- \text{Pop}(c_{1}' = 6 - 2 = 4
- \text{Pop}(c_{2}' = 4 - 2 = 2
- \text{Pop}(c_{3}' = 4 - 2 = 2
- \text{Pop}(c_{4}' = 4 - 2 = 2

\text{Pop}(c_{2}'\) is greater than \textit{min\_pop}, so except \(c_1, c_6, c_7\) remaining all are popular. Therefore \(c_1, c_2, c_3, c_5, c_8, c_9\) are popular. To construct IncPCrime-tree we need the H-table with \(c:\) support(\(c\)), sumCTL(\(c\)), maxCTL(\(c\)). \((c_1: 3, 14, 6), (c_2: 3, 12, 6), (c_3: 3, 14, 6), (c_5: 4, 17, 6), (c_6: 3, 13, 6), (c_8: 4, 18, 6)\).

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{h_table.png}
\caption{H-table and IncPCrime-tree}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{projected_database.png}
\caption{Projected Database of \(c_8\)}
\end{figure}

Once again we scan the ICTDB database to compute the length of each transaction and to remove all items that are not in the H-table and sort the remaining items in each transaction according to the H-table order. Figure 1 shows the contents of the H-table \((c: \text{support}(c), \text{sum\_CTL}(c), \text{max\_CTL}(c))\) and the IncPCrime-tree structure after inserting \(c_8\) to \(c_{10}\) crime transactions of ICTDB. Figure 2 shows the \(c_8\) header table and projected database of \(c_8\). Therefore \(c_1, c_3, c_5, c_9\) are popular crime patterns with \(c_8\).
The IncPCrime-growth finds popular crime patterns from the IncPCrime-tree in which each tree node captures the number of occurrences, the total crime transaction length, and the maximum crime transaction length. Our algorithm finds popular crime patterns by constructing the projected database for potential popular crime itemsets and recursively mining their extensions. This lazy pruning technique ensures that no popular crime patterns will be missed by IncPCrime-growth algorithm and popular crime patterns can be found by mining our IncPCrime-tree.

5. Experiment Results

We did our experiments on the crime datasets which are available as open source in National Crime Records Bureau, India http://data.gov.in, and few datasets of UK and USA are also considered http://data.gov.uk/~/crime_statistics, http://data.vancouver/~details.htm. Some other datasets like city of Chicago crime datasets from the year 2001 to present, it contains 58,71,658 number of crime records with 20 different attributes.

![Figure 3. Execution Time over Chicago Crime Data](image)

![Figure 4. Execution Time over HartFord Data](image)

By using this data we obtained consistent results. With the help of space constraint, we represent the experimental results on a subset of these datasets.

All our programs are written in Java and run in windows environment and processor is 1.3 GHz. The programs which describes the run time indicates the total execution time along with selecting interesting attributes from large number of attributes in the database i.e., query processing. The results that are reported are based on average multiple run in every case.

6. Conclusion

In this paper we introduced a new algorithm called IncPCrime-growth algorithm that finds popular crime patterns from Crime datasets. IncPCrime-tree will constructs with first database scan and also computes support of the crime data, maximum crime transaction length and popularity of each crime have been calculated. After that the algorithm performs for super-pattern popularity for unpopular crimes to prune the crime database. In the second scan it’s going to compute length of each crime transaction by eliminates unpopular crimes and will extracts popular crime patterns from the projected crime databases. Our experimental results showed that our IncPCrime-tree is time efficient for mining popular crime patterns from crime datasets.

References


