Hybrid Scalable Action Rule: Rule Based and Object Based

Jaishree Ranganathan, Sagar Sharma, Angelina A Tzacheva
{jrangan1,ssharam52,aatzache}@uncc.edu
University of North Carolina at Charlotte
Charlotte, North Carolina

ABSTRACT
Action Rule mining is a method to extract actionable pattern from datasets. Classification rules are those which helps predict the object’s class, whereas Action Rules are actionable knowledge that provide suggestions on how an object’s class can be changed to a more desirable state to benefit the user. In the internet era, digital data is wide spread and growing tremendously is such way that it is necessary to develop systems that process the data in a much faster way. The literature of Action Rule mining involves two major frameworks; Rule-Based method: where extraction of Action Rules is dependent on the pre-processing step of classification rule discovery, and Object Based Method: extracts Action Rule directly from the database without the use of classification rules. Object based method extracts Action Rule in an apriori like method using frequent action sets. Since this method is iterative it takes longer time to process huge datasets. In this work we propose a novel hybrid approach to generate complete set of Action Rules by combining the Rule-Based and Object-Based methods. Our results show a significant improvement, where the existing algorithm does not span for the Twitter dataset. On the other hand the proposed hybrid approach completed execution and produces Action Rules in less than 500 seconds on a Cluster.

CCS CONCEPTS
• Computing methodologies → Neural networks; • Applied computing → Collaborative learning.

KEYWORDS
Classification, Educational Data Mining, Neural Networks, Student Evaluations, Teaching Methods

ACM Reference Format:

1 INTRODUCTION
In the last decade, the dramatic progress in digital data accumulation resulted in growing volumes of data in huge databases. The tools and techniques that support humans in the process of identifying useful information (knowledge) from such data falls under Knowledge Discovery in Databases with application of Data Mining methods [10], [7]. It is important to discover useful information in the Knowledge Discovery process. Rule based systems helps to discover useful knowledge as well support the process of turning that knowledge into actions for the benefit of the user. In literature of Data Mining there exists many rule based methods including but not limited to Action Rules mining, Association Action Rules, and Classification Rules. Author Mendel [12] suggests rule-based systems can be used to solve a broad range of problems like forecasting, classification, diagnosis, judgment making, and control, etc. The structure of rule is of the form, IF x THEN y, where x is the antecedent of the rule and y is the consequent. Rule antecedent x is set of variables that can be observed in the data and gives the user the detail about the desired output or consequent y.

Action Rule mining is a rule based data mining method that helps extract Action Rules. Action Rules are extracted from a decision system that suggest possible transition of data from one state to another [14]. Such rules can be used to benefit the user. The formal definition of Action Rule [14] is defined as in "(1)"

\[ (\omega) \land (\alpha \rightarrow \beta) \Rightarrow (\phi \rightarrow \psi) \tag{1} \]

Where, \( \omega \) is conjunction of fixed condition features shared by both groups, (\( \alpha \rightarrow \beta \)) represents changes in flexible attributes, and (\( \phi \rightarrow \psi \)) is the desired change in the decision attribute or attribute whose change benefits the user.

Association Action Rules is an apriori [1] based method that extracts all set of Action Rules from the dataset using iterative procedure. This method takes advantage of the frequent action sets and then merge and prune to generate the set of association rules [14]. Association Action Rules method takes longer time to complete because of the iterative procedure. In this paper we propose a novel algorithm that combines the loosely coupled and tightly coupled methods of Action Rule extraction. Here we combine the strategy of using Action Rule schema and then apply the apriori based Association Action Rule method to extract more complete rules and faster than the iterative procedure.

2 RELATED WORKS
In Data Mining literature, we see two pre-dominant frameworks for Action Rule generation: Rule based (loosely coupled) and Object based (tightly coupled) methods.

2.1 Rule Based Action Rule Mining
In Rule based method, extraction of Action Rules or actionable knowledge is dependent on the pre-processing step of classification rule discovery. These methods use pre-existing classification rules
or generate rules using algorithms like Learning Based on Rough Sets (LERS) [8] and Extracting Rules from Incomplete Decision (ERID) [5] Systems. Rule based methods are further sub-divided into methods generating Action Rules from certain pairs of classification rules like Discovering Extended Action Rules (DEAR) [15, 18], and methods that generate Action Rules from single classification rule Action Rules Based on Agglomerative Strategy (ARoSA) [16].

2.2 Object Based Action Rule Mining
Action Rule Extraction from Decision Table (ARED) [11], Association Action Rule [14] method extracts Action Rule directly from the database without the use of classification rules.

2.3 Algorithms Computational Efficiency
The above mentioned algorithms work faster for datasets of considerable size, but the Association Action Rule method is computationally extensive and time consuming because of the iterative nature. Scalability and processing time is one of the main attributes needed for Association rule mining with data increasing in terms of both dimensions and size.

Agrawal and Shafer [2] proposed three parallel distribution algorithms for association rule mining namely: Count distribution, Data set distribution and Candidate distribution algorithms. But each of these algorithms have its own disadvantages as follows. Count distribution algorithm does not efficiently utilize the aggregate system memory; Data distribution algorithm suffers communication overhead; Candidate distribution algorithm redistributes the database while scanning the local partition repeatedly and is worse compared to Count distribution algorithm [21]. Shintani and Kirsugawa [17] used a hybrid approach of Hash Partitioned Apriori - Extremely Large stemset duplication (HPA-ELD) combined with the non-partitioned apriori method to ensure almost same amount of communication overhead. Another hybrid distribution method that combines Count distribution and intelligent data distribution is proposed by Han et al. [9]. This method reduces the database communication cost.


In this paper we propose a novel approach of hybrid Association Action Rule generation, combining the rule based and object based approach of Action Rule mining to reduce the overhead of Association Action Rule iterative procedure.

This section gives basic information about the terminologies related to the methods adopted in the paper.

2.4 Information System and Decision System
Information system “Table 1” is perceived as a system $Z = (X, M, V)$, where $X$ is set of objects $(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8)$ in the system; $M$ is non-empty finite set of attributes $(A, B, C, E, F, G, D)$; $V$ is the domain of attributes in $M$, for instance the domain of attribute $B$ in the system $Z$ is $(B_1, B_2, B_3)$.

The information system in “Table 1” is denoted as Decision system if the attributes $M$ are classified into flexible $M_{i,t}$, stable $M_{s,t}$ and decision $d = (M_{i,t}, M_{s,t}, \{d\})$. From “Table 1” $M_{i,t} = \{A, B, C\}$, $M_{s,t} = \{E, F, G\}$, and $d = D$.

2.5 Action Term
The expression $(y, y_1 \rightarrow y_2)$ is an atomic action term, where $y$ is an attribute and $y_1, y_2 \epsilon V_y$. If $y_1 = y_2$, then $y$ is stable on $y_1$. In this case action term is denoted as $(y, y_1)$ for simplicity.

- If $t$ is an atomic action term, then $t$ is an action term.
- If $t_1 \cdot t_2$ are action terms, then $t_1 \cdot t_2$ is an action term.
- If $t$ is an action term containing $(y, y_1 \rightarrow y_2)$, $(z, z_1 \rightarrow z_2)$ as its sub-terms, then $y \neq z$.
- Domain of action term is denoted by Dom(t), which includes all attributes listed in $t$.

2.6 Action Rule
The expression $r = [t_1 \rightarrow t_2]$ is an Action Rule where, $t_1$ is an action term and $t_2$ is an atomic action term. The following is an example Action Rule from “Table 1”. $[B_1 \land C_1 \land (F, F_3 \rightarrow F_1) \land (G, \rightarrow G_1) \rightarrow \{D, D_2 \rightarrow D_1\}$.

2.7 Support and Confidence
Support and confidence of rule $r$ is given as below:

- $sup(r) = \min(card(Y_1 \cap Z_1), card(Y_2 \cap Z_2))$.
- $conf(r) = \frac{card(Y_1 \cap Z_1)}{card(Y_1)} \land \frac{card(Y_2 \cap Z_2)}{card(Y_2)}$.
- $card(Y_1) \neq 0, card(Y_2) \neq 0, card(Y_1 \cap Z_1) \neq 0, card(Y_2 \cap Z_2) \neq 0$.
- $conf(r) = 0$ otherwise.

2.8 Learning from Rough Sets (LERS)
LERS [8] is classic bottom-up strategy that constructs rules with a conditional part of the length $k + 1$ after all rules with a conditional part of length $k$ have been constructed. This method finds the certain and possible rules describing the decision attribute in terms of other attributes in the system. Let us assume that “Table 1” as Decision system with the following attributes $M = \{M_{i,t}, M_{s,t}, \{d\}\}$, where $M_{i,t} = \{A, B, C\}$, $M_{s,t} = \{E, F, G\}$, and $d = D$.

This is the list of certain and possible rules that LERS strategy finds from “Table 1”.

- Certain Rules

<table>
<thead>
<tr>
<th>X</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>A1</td>
<td>B1</td>
<td>C1</td>
<td>E1</td>
<td>F2</td>
<td>G1</td>
<td>D1</td>
</tr>
<tr>
<td>x2</td>
<td>A2</td>
<td>B1</td>
<td>C2</td>
<td>E2</td>
<td>F2</td>
<td>G2</td>
<td>D2</td>
</tr>
<tr>
<td>x3</td>
<td>A3</td>
<td>B1</td>
<td>C1</td>
<td>E2</td>
<td>F2</td>
<td>G3</td>
<td>D2</td>
</tr>
<tr>
<td>x4</td>
<td>A4</td>
<td>B1</td>
<td>C2</td>
<td>E2</td>
<td>F2</td>
<td>G1</td>
<td>D2</td>
</tr>
<tr>
<td>x5</td>
<td>A5</td>
<td>B1</td>
<td>C1</td>
<td>E3</td>
<td>F2</td>
<td>G1</td>
<td>D2</td>
</tr>
<tr>
<td>x6</td>
<td>A6</td>
<td>B1</td>
<td>C1</td>
<td>E2</td>
<td>F2</td>
<td>G1</td>
<td>D1</td>
</tr>
<tr>
<td>x7</td>
<td>A7</td>
<td>B1</td>
<td>C2</td>
<td>E2</td>
<td>F2</td>
<td>G2</td>
<td>D2</td>
</tr>
<tr>
<td>x8</td>
<td>A8</td>
<td>B1</td>
<td>C1</td>
<td>E3</td>
<td>F2</td>
<td>G3</td>
<td>D2</td>
</tr>
</tbody>
</table>
- $E_1 \rightarrow D_1$
- $G_3 \rightarrow D_2$
- $F_3 \rightarrow D_2$
- $E_3 \rightarrow D_2$
- $A_2 \land G_1 \rightarrow D_2$
- $A_1 \land E_2 \rightarrow D_2$
- $A_2 \land C_1 \rightarrow D_2$
- $A_1 \land C_2 \rightarrow D_2$
- $E_2 \land C_1 \rightarrow D_2$
- $A_2 \land E_2 \land B_1 \land C_2 \rightarrow D_3$
- $A_2 \land F_2 \land E_2 \land B_1 \rightarrow D_3$
- $A_1 \land G_1 \land C_1 \land B_1 \rightarrow D_3$
- $A_2 \land F_2 \land B_1 \land C_2 \rightarrow D_3$
- $A_1 \land F_2 \land C_1 \land B_1 \rightarrow D_3$
- $G_2 \land C_1 \land B_1 \rightarrow D_3$
- Possible Rules
- $A_2 \land G_2 \land F_2 \land E_2 \land C_2 \rightarrow D_1$
- $A_2 \land G_2 \land F_2 \land E_2 \land C_2 \rightarrow D_2$
- $A_2 \land G_2 \land F_2 \land E_2 \land C_2 \rightarrow D_3$

2.9 Action Rules Based on Agglomerative Strategy (ARoGs)

ARoGs [16] uses LERS [8] to extract Action Rules without the need to verify the validity of the certain rules. By using LERS as preprocessing step the overall complexity of ARoGs method is reduced when compared to DEAR [15][18] method.

Using the "Table. 1", below is the sample Action Rule that ARoGs algorithm generates considering the change of decision value from $D_2 \rightarrow D_1$.

ARoGs method uses the certain rules extracted by LERS strategy and generates Action Rule schema. Then for each Action Rule schema the Action Rules are constructed. For instance let us take the classification rule "(2)".

$$G_1 \land F_2 \land C_1 \land B_1 \rightarrow D_1 \tag{2}$$

The Action Rule schema associated with the above Equation 2 for the reclassification task $D_2 \rightarrow D_1$ is given as "(3)" and the corresponding Action Rule "(4)".

$$[B_1 \land C_1 \land (F_2 \land F_1) \land (G \rightarrow G_1)] \rightarrow (D, D_2 \rightarrow D_1). \tag{3}$$

$$[B_1 \land C_1 \land (F_2 \land F_1) \land (G \rightarrow G_3)] \rightarrow (D, D_2 \rightarrow D_1). \tag{4}$$

2.10 Apriori Based Association Action Rule Mining (AAR)


- **Merging step**: The algorithm merges the previous two frequent action sets into a new action set.

- **Pruning step**: The algorithm discards the newly formed action set if it does not contain the decision action (e.g. the user desired value of decision attribute).

For our example, using the data from "Table. 1", the primary action sets generated by AAR are shown in "Table. 2". The frequent action sets generated by AAR are shown in "Table. 3".

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Primary Action Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>(A, A_1), (A, A_2), (A, A_3)</td>
</tr>
<tr>
<td>B</td>
<td>(B, B_1), (B, B_2), (B, B_3)</td>
</tr>
<tr>
<td>C</td>
<td>(C, C_1), (C, C_2)</td>
</tr>
<tr>
<td>E</td>
<td>(E, E_1), (E, E_2), (E, E_3), (E, E_4)</td>
</tr>
<tr>
<td>F</td>
<td>(F, F_1), (F, F_2), (F, F_3)</td>
</tr>
<tr>
<td>G</td>
<td>(G, G_1), (G, G_2), (G, G_3)</td>
</tr>
<tr>
<td>D</td>
<td>(D, D_1), (D, D_2), (D, D_3)</td>
</tr>
</tbody>
</table>

Table 3: Frequent Action Sets

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Frequent Action Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iteration 1</td>
<td>(A, A_1) \land (D, D_2 \rightarrow D_3)</td>
</tr>
<tr>
<td>Iteration 1</td>
<td>(A, A_2) \land (D, D_2 \rightarrow D_3)</td>
</tr>
<tr>
<td>Iteration 1</td>
<td>(A, A_3) \land (D, D_2 \rightarrow D_3)</td>
</tr>
<tr>
<td>Iteration 2</td>
<td>(B, B_1) \land (D, D_2 \rightarrow D_1)</td>
</tr>
<tr>
<td>Iteration 2</td>
<td>(B, B_2) \land (D, D_2 \rightarrow D_1)</td>
</tr>
<tr>
<td>Iteration 2</td>
<td>(B, B_3) \land (D, D_2 \rightarrow D_1)</td>
</tr>
</tbody>
</table>

In our example, the action set is discarded if (D, 2 \rightarrow 1) is not present in it. From each frequent action set, the association Action Rules are formed. Therefore, the algorithm generates frequent action sets and forms the association Action Rules from these action sets. For our example, using the data from the Information system in "Table. 1", the algorithm generates Association Action Rules, an example is shown below:

$$(B, B_1 \rightarrow B_1) \land (C, C_1 \rightarrow C_1) \land (E, E_1 \rightarrow E_1) \rightarrow (D, D_2 \rightarrow D_1)$$
2.11 Spark

Apache Spark [20] is a cluster computing framework that provides scalability and fault tolerance properties similar to MapReduce [6] but efficient in two aspects: iterative jobs and Interactive analytics. Spark supports parallel programming through abstractions like Resilient Distributed Datasets (RDD) and Parallel operations. All of these operations are controlled and launched by the driver program of the application. Two main stages in Spark programming are Transformation - where the data is processed and saved across nodes in the cluster and Action - is when operations like reduce, collect, foreach, map, flatmap collect results from the RDD and send to the Driver.

3 METHODOLOGY

In this paper, we propose a novel approach of hybrid Action Rule generation, combining the rule based and object based approach of Action Rule mining to reduce the overhead of Action Rule iterative procedure. The algorithm pseudocode is given in "Fig. 1."

![Algorithm](image)

**Table 4: Suitable for Action Rule Schema**

<table>
<thead>
<tr>
<th>X</th>
<th>B</th>
<th>C</th>
<th>F</th>
<th>G</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>B1</td>
<td>C1</td>
<td>F2</td>
<td>G1</td>
<td>D1</td>
</tr>
<tr>
<td>x3</td>
<td>B1</td>
<td>C1</td>
<td>F2</td>
<td>G3</td>
<td>D2</td>
</tr>
<tr>
<td>x5</td>
<td>B1</td>
<td>C1</td>
<td>F3</td>
<td>G1</td>
<td>D2</td>
</tr>
<tr>
<td>x8</td>
<td>B1</td>
<td>C1</td>
<td>F2</td>
<td>G3</td>
<td>D2</td>
</tr>
</tbody>
</table>

Association Action Rule extraction is an exhaustive Apriori based method which extracts complete set of Action rules by taking all possible combinations of the action terms. It is an iterative procedure and does not scale very well in case of dense and high dimensional dataset. In this work we create subtables by using the Action Rule Schemas in a highly dense data as explained above. We perform Association Action Rule extraction algorithm on each of the subtables in parallel which allows the algorithm to complete and generate rules in a much faster time compared to the existing algorithms and systems. In our sample dataset example, the algorithm generates following Action Rules " 7" based on the subtable "Table 4."

\[
[B_1 \land C_1 \land (F_1 \rightarrow F_1) \land (G_2 \rightarrow G_1)] \rightarrow (D_2 \rightarrow D_2). \tag{7}
\]

The hybrid algorithm is implemented in Spark [20], runs separately on each subtable and does transformations like map(), flatmap(), join() and other distributed operations. The overall operating methodology is shown in "Fig. 2."

![Figure 2: Hybrid Action Rule Algorithm - Flowchart](image)

4 EXPERIMENTS AND RESULTS

To experiment our method, we use the Twitter dataset [13] which is densely populated with values. The Twitter dataset consists of
records describing the emotions in the tweet like the intensity of emotions in a tweet text and other tweet attributes. We choose Emotion as the decision attribute and collect Action Rules that help identify changes that are required for the emotion to be more positive. For instance, to change the emotions from 'Sadness' to 'Joy'. Table 5 gives an overview of the dataset used for the experiments. The minPartitions in Spark scala is given as 30.

Table 5: Properties of Dataset Used for Experiments

<table>
<thead>
<tr>
<th>Property</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attributes</td>
<td>25 attributes</td>
</tr>
<tr>
<td>Emotions</td>
<td>Joy, Sadness, Anger,</td>
</tr>
<tr>
<td>Decision Attribute Values</td>
<td>Anticipation, Trust, Disgust,</td>
</tr>
<tr>
<td></td>
<td>Surprise, Fear</td>
</tr>
<tr>
<td># of Instances</td>
<td>174890</td>
</tr>
</tbody>
</table>

The decision problem here is to suggest possible recommendations to the user on how to be more positive in terms of Twitter users. Some promising applications in this context include but not limited to the following: Education, to benefit students, institution and faculty in terms of Teaching models, learning environment and how to improve them based on student evaluations, Customer Care Service based on emotions from customer feedback, these actionable patterns can suggest what aspects of the service could be improved or changed for better customer satisfaction. Table 6 gives information about the stable, and decision attributes, Number of attributes and instances used for each of the experiments to extract Action Rules using the algorithm "Fig. 1".

Table 6: Parameters Used for Action Rule Experiments

<table>
<thead>
<tr>
<th>Property</th>
<th>Experiment</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Attributes</td>
<td>9 attributes</td>
</tr>
<tr>
<td>Stable Attributes</td>
<td>user language</td>
</tr>
<tr>
<td>Decision Attribute Values</td>
<td>Sadness → Joy</td>
</tr>
<tr>
<td># of Instances</td>
<td>174888</td>
</tr>
</tbody>
</table>

We use the University Research Cluster - Taipan (Hadoop) for the experiments, which includes the following services: HBase, Hive, Hue, Impala, Kudu, Oozie, Spark and Spark2, Sqoop 2, and YARN. It has 16 nodes with dual Intel 2.93 GHz 6-core processors. "Fig. 3" gives run time analysis of the new hybrid Association Action Rule generation algorithm. The existing algorithm [19] does not span for the Twitter dataset and unable to complete extracting Action Rules. It fails due to the iterative overhead when run in the similar environment using the same parameters.

Let us consider the action rule $ARC_3$ from "Table. 7", this rule suggest possible changes to achieve a desirable emotional state of 'Joy'. If user tends to reduce use of negative words as denoted by $(SadnessScore, Medium → VeryLow)$ and if the TweetSource is iPhone then it is possible to change the emotion from 'sadness' to 'joy'. In that case, the emotion associated with this user tweet can be classified as 'joy'. This example is more intuitive in case of applications where it is required to monitor people emotions in a particular city or county in order to understand the life satisfaction and help in public policy making and societal well-being measures.

5 CONCLUSIONS

Action Rules are actionable recommendations that suggest possible transition from one state to another for benefit of the user. Emotion mining from text has its root in many application disciplines namely, Psychology, Neuroscience, Social Science, Computer Science, and others. Systems that can detect emotions and suggest actionable recommendations has many potential applications. In Education, to benefit students, institution and faculty in terms of Teaching models, learning environment, in Customer Care Service based on emotions from customer feedback, these actionable patterns can suggest what aspects of the service could be improved or changed for better customer satisfaction, in Technology of Future like smart phones, to predict user emotions and suggest suitable movies, music or call a family member or friend. In this work we propose a novel approach of hybrid association action rule algorithm by combining the rule based and object based approach to reduce the overhead of the iterative procedure. We test our algorithm using Twitter dataset and compare with existing method of Association Action Rule. Twitter dataset is labeled with emotions based on the tweet text and is a densely populated data with more number of attributes. It is observed that the proposed algorithm

![Figure 3: Hybrid Action Rule Algorithm - Run Time.](image)

Table 7: Sample Action Rules

<table>
<thead>
<tr>
<th>Twitter Emotions Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) $ARC_1$ : (FearScore, Medium → VeryLow) ∧ (SadnessScore, Medium → VeryLow) ∧ (MediaEntities, 0) → (Emotion, Sadness → Joy)[Support : 526, Confidence : 96%]</td>
</tr>
<tr>
<td>(2) $ARC_2$ : (AngerScore, VeryLow) ∧ (SadnessScore, Medium → VeryLow) ∧ (UserLanguage, en) → (Emotion, Sadness → Joy)[Support : 553, Confidence : 96%]</td>
</tr>
<tr>
<td>(3) $ARC_3$ : (SadnessScore, Medium → VeryLow) ∧ (TweetSource, 1) → (Emotion, Sadness → Joy)[Support : 520, Confidence : 94%]</td>
</tr>
</tbody>
</table>

111
is able to generate complete set of Action Rules given the entire dataset consisting of 174888 instances in less than 500 seconds. On the other hand the existing algorithm could not handle the entire dataset, it fails due to memory overhead of the iterative procedure. In future we plan to identify emotions from student evaluations of teaching and extract actionable patterns to help improve the teaching models and learning performance.

REFERENCES


