ABSTRACT

A heuristic incremental associative classification algorithm is proposed in this study to analyze big data. The associative classification algorithm proposed in this study builds a classifier by iterative steps, which adds some of attributes to improve the accuracy of the classifier each time, instead of using all the attributes at the same time. In addition, the proposed algorithm can identify and prioritize the discriminative attributes to minimize the number of attributes, so it can reduce the computing time significantly. The classifier can be updated based on the previous existing rules and the incremental data to avoid re-defining the existing classifier.

KEYWORDS: Big Data, Data Mining, Classification, Association Rules, Incremental Algorithm

INTRODUCTION

Big data spans from three dimensions, volume, velocity and variety [Gartner, 2012; Hopkins, 2011; IBM, 2012; Jackson, 2012], which present big challenges to manage and analyze big data. Volume means that not only the number of objects in the data set is enormous but also the dimension of data attributes is high and changeable [Novikov, 2012]. In order to analyze big data, we should find some ways to handle the ever-increasing data volume [Schadt et al., 2010]. Velocity describes that the speed of data growth is faster than ever before and will continue to accelerate in the future [Demirkana & Delen, 2012; Stackpole, 2012]. The frequent and speedy data updates mean that data is not static but dynamic, so the previous analysis result, which only reflects the state of the original data, cannot guarantee acceptable decisions [Cheung et al., 1996; Crespoa & Weber, 2005]. The last dimension of big data is variety, which describes that big data coming from different sources may contain structured and unstructured data such as text, sensor data, audio, video, image and more [Olavsrud, 2012].

Closely connected to the big data issues is the development of data mining technique, a process to discover knowledge and transform the information extracted from the data into an understandable structure for further use. One of the most popular data mining tasks is “classification” that deals with grouping data objects into predefined categories based on certain criteria. Among the classification approaches, associative classification approach is simple and
easily understood to integrate the association rule mining and the classification rule mining [Agrawal et al., 1993], where each rule can be represented in an “IF Conditions THEN Conclusion” format. This approach first mines a special subset of the association rules whose conclusion part is restricted to the classification attribute, and then builds a classifier using these rules. Since big data has above characteristics, conventional analysis tool or data mining approaches are not suitable for big data problems. We need to make some modifications on these tools and approaches to analyze big data efficiently.

Associative classification [Li et al., 2001; Liu et al., 1998; Wang et al., 2000], combining association rule mining [Agrawal et al., 1993] and classification method to form a classifier, extracts rules from relationships between items and classes that occur frequently in a given data set, where an item is formed by a pair of an attribute and its corresponding value. Associative classification approach can handle both categorical data and numerical data, and it considers the simultaneous corresponding values of different attributes, hence it yields comparable accuracy [Li et al., 2001; Liu et al., 1998; Wang et al., 2000]. Associative classification uses global search to find all the associations between items and class labels, so the accuracy of the resulting classifier is high in general. Moreover, associative classification can update a rule without affect the entire rule set and can handle the objects with different attributes without changing the structure of the classifier [Cheung et al., 1996; Thabtah & Abdeljaber, 2007].

Although associative classification has many advantages, it has the shortcoming that not all the frequent items are discriminative. If all the frequent items are used to generate the classifier, the model learning process will be slowed down and the accuracy may be decreased due to the over-fitting problem with too many attributes and rules [Han et al., 2011]. Due to the aforementioned issues, the associative classification approaches are not fitted to deal with big data problems. Therefore, this study modifies the associative classification approach to deal with big data problems.

There are several associative classification approaches, which use different methods to discover frequent rules, store rules, rank rules, and prune redundant or misleading classification rules to build a classifier, and predict the class labels for new data objects. This study proposes a heuristic algorithm called the **Heuristic Incremental Associative Classification Algorithm (HIACA)**, modified from Classification Based on Associations (CBA) algorithm, since CBA algorithm is the earliest and simplest algorithms for associative classification [Liu et al., 1998].

The rest of the paper is organized as follows. Section 2 describes the problem. Section 3 presents our two-phase iterative search heuristics, called the Heuristic Incremental Associative Classification Algorithm (HIACA), to solve the big data problem efficiently. Section 4 compares the results obtained with our heuristic algorithm to those obtained with other heuristic methods, in order to evaluate the HIACA’s efficiency and optimality. Finally, Section 5 offers our conclusions and suggestions for future research.

**PROBLEM DESCRIPTION**

Associative classification proposes the integration of association rule mining [1] and classification. Associative classification mines association rules, which satisfy some minimum support and minimum confidence constraints [Agrawal & Srikant, 1994; Liu et al., 1998], for building a classifier. This process is a special case of association rule mining [Agrawal & Srikant, 1994; Han et al., 2000; Zaki et al., 1997]. In a general association rule mining process, the target attribute of
rules is not pre-determined, while in associative classification the target attribute of rules is one and only one pre-determined attribute and is named classification attribute [Liu et al., 1998].

Before describing the detail of CBA [Liu et al., 1998], we define the associative classification problem as follows.

Definition 1: The training data set $T$ has $m$ distinct attributes $A_1, A_2, ..., A_m$ and one classification attribute $C$, and $n$ is the number of training objects in $T$.

Definition 2: Each training object in $T$ consists of the values of the attributes and one class. The training object $t$ can be described as $<(A_1, a_1), (A_2, a_2), ..., (A_m, a_m), c>$, where $a_i$ is the value of $A_i$ in $t$ and $c$ is the class of $t$.

Definition 3: An item is a pair of attribute $A_i$ and the value of $A_i$, denoted as $(A_i, a_i)$.

Definition 4: An itemset $i$ with $k$ items can be described as $\{ (A_{i_1}, a_{i_1}), (A_{i_2}, a_{i_2}), ..., (A_{i_k}, a_{i_k}) \}$, where all the $k$ attributes are distinct.

Definition 5: A rule $r$ has the form $<i, c>$, where $i$ is an itemset or a rule body and $c \in C$ is the rule class label. Each rule represents $i \rightarrow c$ which means that if the object contains all the items in $i$, then the object is labeled as $c$.

Definition 6: The size of a rule $<i, c>$ is the number of items in the rule body $i$.

Definition 7: A training object $t$ is said to be covered by a rule $r$, if $t$ contains all the items in the rule body of $r$.

Definition 8: A training object $t$ is said to match a rule $r$, if not only $t$ is covered by $r$ but also the class of $t$ is same as the class of $r$.

Definition 9: The actual occurrence of a rule $r$ is the number of training objects in $T$ that are covered by $r$, and it is donated as $actoccr(r)$.

Definition 10: The support count of a rule $r$ is the number of training objects in $T$ that match $r$, and it is donated as $suppCount(r)$.

Definition 11: The support of a rule $r$ is $supp(r) = suppCount(r) / n$.

Definition 12: The confidence of a rule $r$ is $conf(r) = suppCount(r) / actoccr(r)$.

Definition 13: The frequent rules are the rules with supports higher than the minimum support threshold, where the minimum support threshold is donated as $minSupp$.

Definition 14: The association rules are frequent rules with confidences higher than minimum confidence threshold, where the minimum confidence threshold is donated as $minConf$.

CBA consists of two parts: a rule generator (CBA-RG) and a classifier builder (CBA-CB). CBA-RG uses the Apriori algorithm [Agrawal & Srikan, 1994] to discover all the frequent rules. The Apriori algorithm finds the frequent rules by iterative steps. First, the Apriori algorithm finds
all the frequent rules with size 1, and then uses these rules to generate the candidate rules with size 2, which are possibly frequent rules with size 2. By passing over the training objects, the Apriori algorithm computes the supports of the candidate rules and define the candidate rules as the frequent rules if the supports are higher than \( \text{minSupp} \). Then, the Apriori algorithm uses the same method to generate the frequent rules with larger size until no more candidate rules can be found. In brief, Apriori finds candidate rules with \( k \) size based on the frequent rules with \( k-1 \) size, and computes the supports of the candidate rules by passing over the training objects. After discovering all the frequent rules by the Apriori algorithm, CBA-RG extracts the association rules, which are the frequent rules with confidences higher than \( \text{minConf} \).

CBA-CB uses the association rules generated by CBA-RG to build a classifier, which is used to predict the class of each object whose class is unknown. In a single rule prediction way, each object will use only one rule to predict the class. Since an object may be covered by many different rules, the CBA should rank the rules in the classifier to determine the precedence of the rules to avoid rule conflicting problem. The object is labeled by the rule with the highest precedence among all the rules covering the object. The method that CBA-CB ranks the rules is described as follows. Given two rules, \( r_i \) and \( r_j \), \( r_i \) has a higher precedence than \( r_j \) if (1) the confidence of \( r_i \) is higher than that of \( r_j \), or (2) their confidences are the same, but the support of \( r_i \) is greater than that of \( r_j \), or (3) both their confidences and supports are the same, but \( r_i \) is generated earlier than \( r_j \).

However, including all the association rules generated by CBA-RG into the classifier may affect the performances of the classifier because some rules are over-fitting and some are redundant. The rule \( r \) is redundant if there is a more general rule with a higher precedence than \( r \). In order to reduce the size of the classifier, improve classification efficiency, and maximize the accuracy, those redundant or misleading rules should be excluded from the classifier.

The association rules in the classifier should satisfy two conditions. First, each training object is covered by the rule with the highest precedence among the rules covering the object. Second, every rule in the classifier correctly classifies at least one training object that has not been covered by any previous rules. However, not all the rules satisfying the conditions will be added into the classifier. The CBA-CB determines a break-even point that obtains the maximum expected accuracy, and the rules that rank behind the break-even point are dropped from the classifier. Furthermore, to avoid the problem that the objects are not covered by any rules in the classifier, a default rule is added into the classifier to label these objects as the default class. The default class is determined by the major class in the training objects that are not covered by any rules. CBA proposes two versions of CBA-CB: M1 or M2. CBA-CB: M1 has to scan the data set one time for each association rule. CBA-CB: M2 is an improved version, which needs to scan the data set only slightly more than one pass.

This study attempts to use associative classification approach to analyze big data. Due to the three-V characteristics of big data described before, this study modifies the associative classification approach in two perspectives. First, the data set contains a large-volume and high dimensional structured data. In order to obtain mining result in a reasonable time, the efficiency of the classification approach should be improved. CBA adopts the Apriori algorithm [Agrawal & Srikant, 1994] to find the frequent rules by searching all the possible combinations of all attributes. However, not all the attributes have discriminative abilities to classify the objects. Including items of these non-discriminative attributes into a rule not only risks lowering the confidence of the rules but also wastes a lot of computing time to generate redundant rules. Therefore, we should identify the discriminative attributes and filter out the non-discriminative
attributes to reduce the number of attributes used for finding the frequent rules. Furthermore, the classifier may contain the rules generated by only a part of the discriminative attributes. In other words, we can further reduce the number of attributes by removing the discriminative attributes that are not essential to the classifier. Once the number of attributes is reduced, the number of candidate rules can be cut down, which in turn reduces the number of times to scan the data set for computing the supports.

Secondly, the perspective of the incremental data should be handled in the modified associative classification approach. Since this study considered a big data problem, large numbers of objects are continuously added into the data set and different objects may contain different attributes, which make the classifier unable to make an acceptable decision under the dynamic environment. In other words, some of the new objects may be classified by the classifier incorrectly because the classifier does not include the new attributes, and the accuracy of the classifier may be unacceptable when facing the incremental data.

There are two ways to obtain the new useful classifier. One is to re-execute the associative classification algorithm to build a whole new classifier with the new data, which is more intuitive but also wastes a lot of time and computation cost obtaining the information that has already known, such as the support counts of the association rules in the original data set. Another way to generate a new useful classifier is the incremental approach, which updates the classifier based on the previous mining results and new objects to avoid wasting computing time and resources. Since the speed of data generation is fast, the method to obtain a new useful classifier must be efficient. If the data patterns do not change drastically, it is obvious that the latter of generating a new useful classifier is more suitable. Therefore, the modified associative classification approach proposed by this study must be able to identify and keep the existing rules that is still useful for the incremental data. It should also be capable to generate new rules to reflect the new data patterns efficiently.

THE HEURISTIC INCREMENTAL ASSOCIATIVE CLASSIFICATION ALGORITHM (HIACA)

The heuristic incremental associative classification algorithm (HIACA) proposed in this study can be divided into two procedures: (1) initial procedure; and (2) incremental procedure. The initial procedure is executed to build a new classifier when there is no previous association rules and classifier, and the algorithm proposed in this study is named the Associative Classification based on Potential Items (ACPI); while the incremental procedure deals with the update of the classifier for an ongoing time-variant data set, the updating is based on the previous mining results and the new added objects. The algorithm of the incremental procedure is named the Incremental Associative Classification based on Potential Items (IACPI).

This study proposes a heuristic algorithm, called Associative Classification based on Potential Items (ACPI), by modifying the CBA [Liu et al., 1998]. The ACPI finds the frequent rules by the Apriori approach [Agrawal & Srikant, 1994] and uses the CBA-CB: M2 algorithm [Liu et al., 1998] to build the classifier. Step 1 of the ACPI is to sort the attributes and partition the attributes into subsets according to the sequence. Steps 2 to 5 of the ACPI are the iterative steps to build and improve the classifier. Step 2 finds the new potential items within each attribute subset. Step 3 of ACPI finds the frequent rules within the potential item sets and builds the classifier by using the CBA-CB: M2 algorithm [Li et al., 2001]. The ACPI then evaluates the accuracy of the classifier with the development test data in Step 4. If the accuracy of the classifier is higher than targeted accuracy or there are no remaining attribute subsets to improve the classifier, the current classifier is the resulting classifier. Else, if the classifier does not achieve the targeted accuracy
and there are remaining attribute subsets, the ACPI goes to Step 2 and starts the next iteration to improve the classifier until no more attribute subset exists.

In order to minimize the number of iterations before achieving the targeted accuracy, the ACPI should arrange the attributes that are more likely to distinguish the objects by classes in the front in Step 1. The discriminative ability of an item is measured by the entropy [Shannon, 1948]. The discriminative ability of an attribute is determined by the overall discriminative ability of the items belonging to the attribute. Base on this idea, the score of an attribute is the weighted average of the scores of all the items belonging to the attribute. The weight of an item is the number of the training objects containing the item. That fact that the score of an attribute is low implies that the discriminative ability of the attribute is high and thus should be selected first. Since the ACPI improves the classifier at each iteration with an attribute subset, it needs to partition all the attributes into many subsets based on the sequence of the attributes, and each subset contains $\alpha$ attributes.

The potential items are defined as the items that can generate rules with a high confidence and a small size, where the size of a rule is the number of items used in the rule. The ACPI identifies the new potential items by a three-phase procedure in Step 2: (1) finds the frequent rules within the attribute subset; (2) extracts the association rules whose confidences are higher than the filtering confidence threshold; and (3) collects the items in these association rules and adds these new potential items into potential item sets. If the potential item is collected from an association rule of Class $c$, then it is added into the potential item set for the Class $c$, which is denoted as $PI_c$. The frequent rules are defined as the rules with supports higher than the minimum support threshold, and the association rules are defined as the frequent rules with confidences higher than the minimum confidence threshold. The minimum support threshold is denoted as $minSupp$ and the confidence threshold for the potential items is denoted as $filterConf$, both of which are determined based on the data set.

After finding all the frequent rules, the ACPI builds the classifier by using the CBA-CB: M2 algorithm [Li et al., 2001], which adopts only the association rules, the frequent rules with confidences higher than the minimum confidence threshold, donated as $minConf$. The CBA-CB: M2 algorithm ranks the association rules by the confidences in descending order, and then the supports in descending order, and then the sizes of the rules in ascending order. The rules are selected into classifier by following two conditions: (1) each training object is covered by the rule with the highest precedence among the rules covering the object; and (2) every rule in the classifier correctly classifies at least one remaining training object, which is not covered by the other rules with the higher precedence.

This study also proposes an Incremental Associative Classification based on Potential Items (IACPI) to modify the existing classifier if necessary. IACPI modifies the classifier based on the original items in the original association rules and the supports and confidences of the rules are computed from the incremental objects. In Step 1, IACPI retrieves the items in the original association rules and uses these items to generate the frequent rules for the incremental objects, and then builds the Preliminary Classifier based on the confidences and supports computed from the incremental objects. IACPI evaluates the accuracy of the Preliminary Classifier with the development test data in Step 2. If the accuracy of the Preliminary Classifier reaches the target, the resulting classifier is found. Else, when the accuracy does not reach the target, IACPI has to keep improving the classifier by going to Step 3.

IACPI drops the useless items and retains the useful items as the potential items as well as uses
the wrongly classified incremental objects to determine the order of the attributes instead of all
the incremental objects, and then partitions the attributes into subsets in Step 3. IACPI then finds
the new potential items within the next attribute subsets in Step 4. The IACPI generates the new
frequent rules and uses these rules to modify the classifier for all the incremental objects in Step
5. Steps 4 and 5 of IACPI are the same as Steps 3 and 4 of ACPI, but IACPI uses the
incremental objects rather than the original objects. Finally, IACPI evaluates the accuracy of the
classifier and decides whether the classifier needs further improvement in Steps 6 and 7.

To improve the accuracy of the Preliminary Classifier, IACPI needs to find new potential items
from the incremental data and apply them to modify the Preliminary Classifier. The idea of the
improvement method is the same as the iterative steps in ACPI. However, since accuracy is what
needed to be achieved when facing the incremental data, IACPI should focus on the incremental
data that were wrongly classified by the Preliminary Classifier. IACPI retrieves and uses the
wrongly classified incremental objects to determine the order of the attributes instead of all the
incremental objects, and then partitions the attributes into subsets in Step 3.

COMPUTATIONAL ANALYSIS

This study demonstrates the performances of ACPI and IACPI with the data set from the Third
International Knowledge Discovery and Data Mining Tools Competition, which was held in
conjunction with KDD-99 the Fifth International Conference on Knowledge Discovery and Data
Mining. This data set is retrieved from UCI Machine Learning Repository [KDD, 1999]. All the
experiments are performed on a desktop PC with 16 GB RAM, CPU (Intel® Core™ i7-2600
3.40GHz), and running Windows® 7 Professional.

This data set contains the intrusion detection data, including a wide variety of simulated
intrusions. The data set contains 4,898,431 network connection records, with 34 continuous
attributes, 4 binary attributes, and 3 categorical attributes. This study designs two scenarios to
examine the incremental procedure and hence, this study generates three data sets, one for the
initial procedure and two for the incremental procedure, by sampling from the intrusion detection
data set, and names them data set 0 to data set 2. Data set 0 is used for initial procedure and
has 1,504,875 objects with 42 attributes and 10 classes. The rest two data sets are used as
incremental data set to examine the incremental procedure and each differs from data set 0 in
different perspectives. Data set 1 has 1,505,000 objects with 42 attributes and 10 classes and is
used for the first scenario in which one new class is added into and one of the original classes is
eliminated from the incremental data set. Data set 2 has 1,504,500 objects with 43 attributes and
10 classes and is used for the second scenario in which the incremental data set contained a
new attribute which does not appear in the original data set.

In order to verify the efficiency and the validity of the proposed ACPI, we design four experiments.
The first experiment demonstrates the effect of increasing the numbers of attributes upon the
efficiency of ACPI. The second experiment displays the effect of increasing the numbers of
objects upon the efficiency of ACPI. The performance is measured by the run time, the result of
ACPI is compared with that of CBA [Liu et al., 1998], which finds frequent rules with all the
attributes at the same time by the Apriori approach [Agrawal & Srikant, 1994].

Both experiments for ACPI use data set 0, which has 1,504,875 objects with 42 attributes, \(A_1\) to
\(A_{42}\), and 10 classes, \(C_0\) to \(C_9\). The 10% of objects are used as test data to evaluate the accuracy
of the resulting classifier. The accuracy is computed by the number of test objects classified
correctly dividing by the number of test objects. The 5% of objects are used as development test
data to evaluate whether the classifier generated at each iteration of ACPI achieves the targeted accuracy. The rest 85% of objects are used as the training data set.

Since the run time of CBA grows exponentially as the number of attributes increases and CBA uses 2 days and 17 hours to train the classifier with 20 attributes, this experiment does not provide the run time of CBA when the numbers of attributes are more than 20 because the run times are longer than 1 month. The CBA generates a classifier for data set 0 by using only small size rules (the number of items \( \leq 4 \)) even when all the 42 attributes are included in the classification process. However, the run time is 14 hours and 45 minutes when the CBA search all the 42 attributes for data set 0. It is still much longer than the run time of ACPI (16.17 seconds) when using all the 42 attributes. Thus, even when the CBA discovers only the rules with sizes that are not larger than four, ACPI is more efficient than the CBA.

This study also compares the performances of the incremental procedure with that of non-incremental approach. Since we have shown that the accuracy of ACPI is comparable to that of CBA and ACPI is much more efficient than CBA in the previous subsections, we compare the run time of IACPI with that of ACPI.

This study demonstrates that IACPI can handle the problem of simultaneously adding a new class to and eliminating an original class from the incremental data set. IACPI updates the classifier based on the association rules generated from data set 0 by ACPI and data set 1, used as the incremental data set, excludes one class, \( C_2 \), from and includes a new class, \( C_{10} \), into data set. The performances of IACPI are measured by the accuracy and the run time and compared with those of ACPI, which generates a whole new classifier with data set 1 directly without considering the previous results generated from data set 0.

The result indicates that IACPI has the ability to find the new potential items from the new discriminative attribute to update the classifier, when some rules in the original classifier are no longer useful to classify the objects. This experiment not only examines whether IACPI updated the classifier with the new discriminative attribute, but also compares the accuracy and the run time of IACPI with those of ACPI, which generates a whole new classifier with data set 2 without considering the previous result generated from data set 0.

CONCLUSION

This study proposes the Heuristic Incremental Associative Classification Algorithm (HIACA) based on the CBA to analyze big data. In addition, not only the number of objects but also the attributes are ongoing time-variant. The HIACA finds a classifier with accuracy higher than the pre-specified targeted accuracy rather than makes a great effort to generate the best classifier with all the attributes. The HIACA consists of two procedures: (1) initial procedure; and (2) incremental procedure. The initial procedure, the Associative Classification based on Potential Items (ACPI), generates the initial classifier by using the potential items extracted from some discriminative attributes instead of all the attributes and thus avoids wasting time to find the rules with non-discriminative attributes. The discriminative ability of an item is determined by the entropy of classes which this item appears, and the discriminative ability of an attribute is determined by the average discriminative abilities of the items belonging to this attribute. Once the valid classifier is found, the iterative steps of ACPI stops.

This study has shown that by reducing the number of attributes and items used for training, the ACPI is much more efficient than the CBA because the ACPI generates much fewer rules than
Moreover, the accuracies of the classifiers generated by ACPI and CBA are similar, which indicates that although the ACPI does not generate rules by global search, the ACPI does not sacrifice the accuracy of the classifier for its efficiency. Therefore, the ACPI is capable of analyzing the big data with a large volume and a high dimension.

The incremental procedure, the **Incremental Associative Classification based on Potential Items** (IACPI), updates the classifier when there is incremental data and the data patterns have changed. The IACPI first uses the items found in ACPI to build the Preliminary Classifier and then finds some new potential items if necessary. This study has shown that the IACPI can update the classifier when the classes or the attributes in the incremental data are different from those in the original data. Moreover, the study also shows that since the IACPI reuses the existing information, it is more efficient than generating a whole new classifier with the incremental data. Overall, IACPI can update the classifier by using the existing information found in ACPI and the new information found in the incremental data efficiently and effectively, since it take advantage of reusing the existing information. Thus, HIACA proposed in this study suits for analyzing the big data.

**ACKNOWLEDGEMENTS**

This research was sponsored by the Ministry of Science and Technology of Taiwan, under the grant: NSC 100-2410-H-002-022-MY3.

**REFERENCES**


Han, J., Pei, J., & Yin, Y. (2000). Mining Frequent Patterns without Candidate Generation. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (Dallas, Texas, USA), 1-12.

Hopkins, B. (2011). Beyond the Hype of Big Data. CIO.


Jackson, J. (2012). The Big Promise of Big Data. CIO.


Li, W., Han, J., & Pei, J. (2001). CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules. In Proceedings of 2001 IEEE International Conference on Data Mining (San Jose, California, USA), 369-376.


Olavsrud, T. (2012). Big Data Causes Concern and Big Confusion. CIO.


Stackpole, B. (2012). 5 Things IT Should Do to Prepare for Big Data. Computerworld US.

