Global Journal of Computer Science and Technology
Software & Data Engineering
Volume 12 Issue 12 Version 1.0 Year 2012
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Clustering on Large Numeric Data Sets Using Hierarchical Approach: Birch

By D.Pramodh Krishna, A.Senguttuvan & T.Swarna Latha
Sree Vidyanikthan Engg.Coll., A.Rangempet, Tirupati, A.P, India

Abstract - The paper is about the clustering on large numeric data sets using hierarchical method. In this BIRCH approach is used, to reduce the amount of data, for this a hierarchical clustering method was applied to pre-process the dataset. Now a day's web information plays a prominent role in the web technology, large amount of data is consumed to communicate, but some with intruders there is loss of data or may changes occur in the interaction, so to recognize intruders they detect to build an intrusion detection system for this a hierarchical approach is used to classify network traffic data accurately. Hierarchical clustering is performed By taking network as an example. The clustering method could produce high quality dataset with far less instances that sufficiently represent all of the instances in the original dataset.

Keywords : *Hierarchical clustering, support vector machine,data mining,KDD cup.*

GJCST-C Classification: *H.3.3*

Strictly as per the compliance and regulations of:

Clustering on Large Numeric Data Sets Using Hierarchical Approach: Birch

D.Pramodh Krishna [α], A.Senguttuvan [σ] & T.Swarna Latha [ρ]

Abstract - The paper is about the clustering on large numeric data sets using hierarchical method. In this BIRCH approach is used, to reduce the amount of data, for this a hierarchical clustering method was applied to pre-process the dataset. Now a day's web information plays a prominent role in the web technology, large amount of data is consumed to communicate, but some with intruders there is loss of data or may changes occur in the interaction, so to recognize intruders they detect to build an intrusion detection system for this a hierarchical approach is used to classify network traffic data accurately. Hierarchical clustering is performed By taking network as an example. The clustering method could produce high quality dataset with far less instances that sufficiently represent all of the instances in the original dataset.

Keywords : Hierarchical clustering, support vector machine,data mining,KDD cup.

I. Introduction

Data clustering is an important technique for exploratory data analysis and has been studied for several years. It has been shown to be Useful in many practical domains such as data classification. There has been a growing emphasis on analysis of very large data sets to discover the useful patterns. This is called data mining and clustering is regarded as a particular branch. So, as the data set size increases they do not scale up well in terms of memory requirement. Hence an efficient and scalable data clustering method is proposed based on a new in memory data structure called CF Tree which serve as in in-memory summary of the data distribution.

We have implemented in a system called BIRCH (Balanced Iterative Reducing And Clustering Using Hierarchies) and studied its performance extensively in terms of memory requirement and scalability.

The SVM technique is unable to operate at such a large dataset due to system failures caused by insufficient memory, or may take too long to finish the training. Since this study used the KDD Cup 1999 dataset, to reduce the amount of data, a hierarchical clustering method was applied to pre-process the dataset before SVM training. The clustering method could produce high quality dataset with far less instances that sufficiently represent all of the instances in the original dataset.

This study proposed an SVM-based intrusion detection system based on a hierarchical clustering algorithm to pre-process the KDD Cup 1999 dataset before SVM training. The hierarchical clustering algorithm was used to provide a high quality, abstracted, and reduced dataset for the SVM training, instead of the originally enormous dataset. Thus, the system could greatly shorten the training time, and also achieve better detection performance in the resultant SVM classifier.

This study proposed an SVM-based intrusion detection system, which combines a hierarchical clustering algorithm, a simple feature selection procedure, and the SVM technique. The hierarchical clustering algorithm provided the SVM with fewer, abstracted, and higher-qualified training instances that are derived from the KDD Cup 1999 training set. It was able to greatly shorten the training time, but also improve the performance of resultant SVM. The simple feature selection procedure was applied to eliminate unimportant features from the training set so the obtained SVM model could classify the network traffic data more accurately. The famous KDD Cup 1999 dataset was used to evaluate the proposed system. Compared with other intrusion detection systems that are based on the same dataset, this system showed better performance in the detection of DoS and Probe attacks, and the be set performance in overall accuracy.

II. Background

a) Data transformation and scaling

SVM requires each data point to be represented as a vector of real numbers. Hence, every non-numerical attribute has to be transformed into numerical data first. The method is simply by replacing the values of the categorical attributes with numeric values. For example, the protocol type attribute in KDD Cup 1999, thus, the value tcp is changed with 0, udp with 1, and icmp with 2. The important step after transformation is scaling. Data scaling can avoid attributes with greater values dominating those attributes with smaller values, and also avoid numerical problems in computation. In this paper, each attribute is called with linear scaling to

Author α : Assistant Professor, Dept. of CSE, Sree Vidyanikthan Engg. Coll., A.Rangempet, Tirupati, A.P, INDIA-517 102.
E-mail : pramodhkrishna.d@gmail.com
Author σ : Professor, Dept. of CSE, Sree Vidyanikthan Engg. Coll., A.Rangempet, Tirupati, A.P, INDIA-517 102.
E-mail : asenguttuvan@rediffmail.com
Author ρ : M.Tech Student, Dept of CSE, Sree Vidyanikethan Engg. Coll., A.Rangempet , Tirupati , A.P, INDIA-517 102.
E-mail : swarnalatha514@gmail.com

the range of [0, 1] by dividing every attribute value by its own maximum value.

b) Clustering feature (CF)

The concept of a clustering feature (CF) tree is at the core of BIRCH's incremental clustering algorithm. Nodes in the CF tree are composed of clustering features. A CF is a triplet, which summarizes the information of a cluster.

c) Defining the CF trees

Given n d- dimensional data points in a cluster {xi}, where i = 1, 2, . . ., n, the clustering feature (CF) of the cluster is a 3-tuple, denoted as CF = (n, LS, SS), where n is the number of data points in the cluster, LS is the linear sum of the data points, i.e., $\sum_{i=1}^{n} x_i$, and SS is the square sum of the data points, i.e., $\sum_{i=1}^{n} x_i^2$.

III. Related Work

a) Theorem (CF addition theorem)

Assume that CF1 = (n1, LS1, SS1) and CF2 = (n2, LS2, SS2) are the CFs of two disjoint clusters. Then the CF of the new cluster, as formed by merging the two disjoint clusters is

$CF_1 + CF_2 = (n1+n2, LS1+ LS_2, SS1+SS_2)$

For example, suppose there are three points (2, 3), (4, 5), (5, 6) in cluster C1, then the CF of C1 is

$CF_1 = \{3, (2+4+5, 3+5+6), (2^2+4^2+5^2, 3^2+5^2+6^2)\}$

$= \{3,(11,14),(45,70)\}$

Suppose that there is another cluster C2 with CF2 = {4, (40, 42), (100, 101)}. Then the CF of the new cluster formed by merging cluster C1 and C2 are

$CF_3 = \{3+4, (11+40, 14+42), (45+100, 70+101)\}$

By Definition and Theorem, the CFs of clusters can be stored and calculated incrementally and accurately as clusters are merged. Based on the information stored in CF, the centroid C and radius R of a cluster can be easily computed. The definitions of C and R of a cluster are given as follows. Given n d-dimensional data points, say {xi} and i = 1, 2. . . n, in a cluster:

$$\text{the centroid } C = \frac{\sum_{i=1}^{n} x_i}{n}, \quad \text{and}$$

$$\text{the radius } R = \frac{\sum_{i=1}^{n} \|x_i - C\|^2}{n}.$$

Where, R denotes the average distance of all member points to the centroid. As mentioned earlier, CF stores only the abstracted data point, i.e., statistically summary of data points that belong to the same cluster. After a data point is added into a cluster, the detail information of the data point itself is missing. Therefore, this approach can save space significantly for densely packed data.

b) CF tree

A CF tree is a height-balanced tree with two parameters, branching factor B and radius threshold T. Each non-leaf node in a CF tree contains the most B entries of the form (CF_i, $child_i$), where1≤ i≤ B and $child_i$ is a pointer to its i^{th} child node, and CF_i is the CF of a cluster pointed by the child i. An example of a CF tree with height h = 3 is shown in Fig. 1. Each node represents a cluster made up of sub-clusters, which represents its entries. It is different from a non-leaf node is that a leaf node has no pointer to link to other nodes, and contains at most B entries. The CF at any particular node contains information for all data points in that node's sub-trees. A leaf node must satisfy the threshold requirement, that is, every entry in every leaf node must have its radiuses less than threshold T.

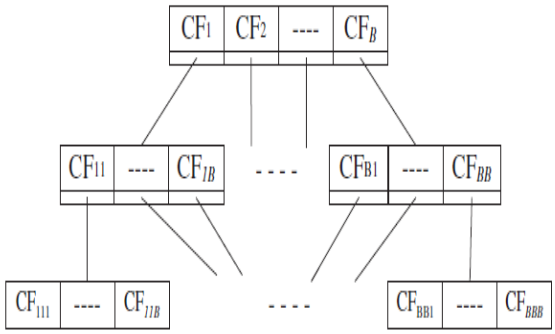

Fig. 1 : A CF Tree with height h=3

A CF tree is a compact representation of a dataset, each entry in a leaf node represents a cluster that absorbs many data points within its radius of T or less.

A CF tree can be built dynamically as new data points are inserted. The insertion procedure is similar to that of a B+-tree to insert a new data to its correct position in the sorting algorithm. The insertion procedure of CF tree has the following steps.

1. Identify the appropriate leaf
2. Modify the leaf
3. Modify entries on the path to the leaf

S no.	String Value	Equivalent Numerical Value
1	Tcp	0
2	Udp	1
3	Icmp	2
4	http	0
5	domain_u	1
6	Auth	2
7	Smtp	3
8	finger	4
9	telnet	5
10	ecr_i	6

Table 1 : Replacement of the string values

IV. Experimental Results

Here the scenario is mainly responsible in the task of clustering. The process of data clustering is performed using the WEKA 3.7 tool's Hierarchical clustered. The testing in this test case can be done in the stage of mentioning the parameters of the clustering such as distance parameter i.e. Euclidean distance, link type such as centroid and number of clusters value. The testing strategies here mainly the setting of those parameters regarding the hierarchical clustered in the WEKA tool. The clustered output can be visualized on the results for the test results.

This test case deals with the training of the clustered dataset. Both the train set and the test set are givenas input for classification and regression/prediction of the accuracy values with the following option lists.

1	2	3	4	5	6
0	0	0	0	0.320175	0.188728
0	0	0	0	0.087719	0.018894
0	0	0	0	0.412281	0.005072
0	0	0	0	0.399123	0.00844
0	0	0	0	0.425439	0.001964

Table 2 : Dataset values in .csv format

This is the first step in the process of data transformation. Here the KDD data set which is in the text format is changed in to the comma separated values i.e., from .txt to .csv. The reason for the format change is the incompatibility of the tool or the language that we use in the further processing steps.

After the format conversion the data set can be view in the office excel as it support csv format viewing in windows environment. Here the data present in the KDD data set before transformation is seen in the excel sheet i.e. in the csv format.

The java code has been written for the data transformation, all the non integer values should be transformed to integer values in the data set. So that program is executed in this screen shot.

The data set has been transformed i.e. all the variables present in the data set has been transformed to the integer values which is in the text format. The data set which is transformed is present in the text format we have to convert that in to .csv format.

After the data transformation the data scaling has to be performed. So in order to perform the data scaling we use the WEKA tool here and load the transformed file into WEKA pre- processing task.

a) Data Clustering

WEKA provides a wide range of clusterers for the process of data clustering. Here we use the hierarchical clusterer for our clustering process which has been clearly mentioned in the screen shot. The testing regarding the clustering has been discussed in the test case 3 in the system testing chapter.

Once the type of clustering has been selected now the query tab is double clicked for the setting the hierarchical clustering parameters. The paramaters of hierarchical clustering that we see here are,

1. Distance Function
2. Link Type
3. NumClusters

The process of clustering is continued once these parameters are correctly set by the user.

The clustered output shown in WEKA mainly show the percentage values of the clusters made by it and the cluster values. The percentage mainly show the amount of dataset those are regarding a single property i.e. based on a single attack present in the considered test set. The output of the clustered is also compared to this test set values in the further procedures such as data training etc.

This is an additional option that is being provided in WEKA tool for visualizing the clustered output in the form of graphs with different color representations. To select this option right click on the result list that is being displayed on the left of the WEKA Explorer sorted with their timing status in hh:mm:ss

This is how you visualize the clustered output in the form of a graph. Here we come across a slider labelled Jitter, which is a random displacement given to all points in the plot. Dragging it to the right increases the amount of jitter, this is useful for spotting concentrations of points. Without jitter, a million instances at the same point would look no different to just a single lonely instance. Here we have the option save to save our clustered output in the external memory to use that in our training procedure.

Once the clustering has been visualized and saved, the clustered output is saved in the external memory with a new column known as cluster of the type nominal which clearly mentions the complete dependency of the particular row with the respective cluster.

This is treated as the clustering output in dataset format which is in turn given as the input for the process of data training.

This is the visualization graphs for each of the column values that is being seen in the KDD dataset. The graphs are drawn according to the level of values present in the complete column. This helps the user in accessing the column present in the particular column in the dataset.

```
=== Run information ===
Scheme:
weka.clusterers.HierarchicalClusterer -N
2 -L SINGLE -P -A
"weka.core.EuclideanDistance -R first-
last"
Relation:     testsample-1_clustered
Instances:    1000
Attributes:   44
```

```
                Instance_number
                1
                2
                3
                4
                5
                6
Cluster
Test mode:    evaluate on training data
=== Clustering model (full training set)
===
Cluster 1
Time taken to build model (full training
data) : 2.81 seconds
=== Model and evaluation on training set
===
Clustered Instances
0         1 (  0%)
1       999 (100%)
```

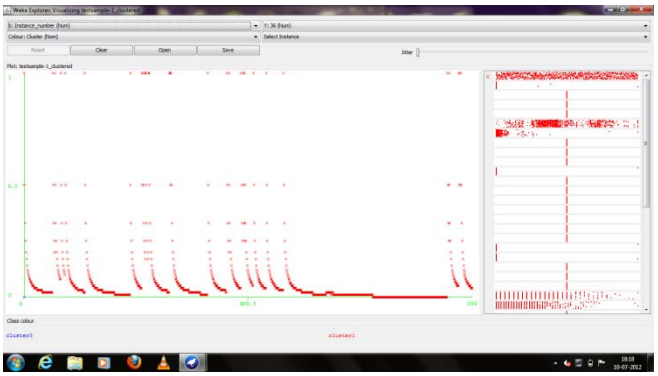

Fig. A : Visualization of the clustered data

V. Conclusion

In this paper, we have proposed an hierarchical clustering approach using BIRCH algorithm it proposed an SVM-based network intrusion detection system with BIRCH hierarchical clustering for data pre-processing. The BIRCH hierarchical clustering could provide highly qualified, abstracted and reduced datasets, instead of original large dataset, to the SVM training. Thus, in addition to a significant reduction of the training time, the resultant SVM classifiers showed better performance than the SVM classifiers using the originally redundant dataset.

However, in terms of accuracy, the proposed system could obtain the best performance. Some new attack instances in the test dataset, which never appeared in training, could also be detected by this system.

References Références Referencias

1. Abraham, A., Grosan, C., & Martin-Vide, C. (2007).Evolutionary design of intrusion detection programs. International Journal of Network Security, 4(3), 328–339.
2. Bouzida, Y., & Cuppens, F. (2006). Neural networks vs. decision trees for intrusion detection. <http://www.rennes.enstbretagne.fr/~fcuppens/articles/monam06. pdf>.
3. Guha, S., Rastogi, R., & Shim, K. (1999). Rock: A robust clustering algorithm for categorical attributes. In Proceedings of the international conference on data engineering (ICDE'99) (pp. 512–521).
4. Guha, S., Rastogi, R., & Shim, K. (1998). Cure: An efficient clustering algorithm for large databases. In Proceedings of the ACM SIGMOD (SIGMOD'98) (pp. 73–84).
5. Hsu, C. -W., Chang, C. -C., & Lin, C. -J., (xxxx). A practical guide to support vector classification. <http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf>.
6. Karypis, G., Han, E.-H., & Kumar, V. (1999). Chameleon: A hierarchical clustering algorithm using dynamic modelling. Computer, 32, 68–75.
7. KDDCup, (1999). Intrusion detection data set. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
8. Khan, L., Awad, M., & Thuraisingham, B. (2007). A new intrusion detection system using support vector machines and hierarchical clustering. The International Journal on Very Large Data Bases, 16(4), 507–521.