Title: Fast Attribute-based Unsupervised and Supervised Table Clustering using P-Trees


Address: Department of Computer Science, North Dakota State University, Fargo, ND, 58102.

Affiliation: North Dakota State University.

Email Addresses: arjun.roy@ndsu.edu, matthew.piehl@ndsu.edu, arijit.chatterjee@ndsu.edu, mohammad.hossain@ndsu.edu, bryan.mesich@ndsu.edu, tingda.lu@ndsu.edu, g.w@ndsu.edu, shehan@uom.lk, william.perrizo@ndsu.edu

Telephone: (701)-231-8562.

Topic Area: Data Mining (Classification and Clustering).
Fast Attribute-based Unsupervised and Supervised Table Clustering using P-Trees


Department of Computer Science
North Dakota State University
Fargo, ND, 58102, USA
{arjun.roy, william.perrizo}@ndsu.edu

Abstract
Since the advent of digital image technology and remote sensing imagery (RSI), massive amount of image data has been collected worldwide. For example, since 1972, NASA and U.S. Geological Survey through the Landsat Data Continuity Mission, has been capturing images of Earth down to 15 meters resolution. Since image clustering is time-consuming, much of this data is archived even before analysis. In this paper, we propose a novel and extremely fast algorithm called FAUST_P or Fast Attribute-based Unsupervised and Supervised Table Clustering for images. Our algorithm is based on Predicate-Trees which are compressed, lossless and data-mining-ready data structures. Without compromising much on the accuracy, our algorithm is fast and can be effectively used in high-speed image data analysis.

1 INTRODUCTION

Digital Image Technology developed in 1970s has caused an exponential growth in the amount of image data throughout the world. This includes, personal photographs, medical images, scientific data such as remote sensing imagery, etc. For e.g. The Landsat Data Continuity Mission by NASA and U.S. Geological Survey has been providing image captures of Earth with moderate resolution from 15 meters to 100 meters for the last 38 years [3]. Also, consider the current scenario of usage of Unmanned Air Vehicles (UAVs) for security purpose where there is massive data collection and classifying objects of interest such as tanks, enemy hideouts, etc. is of utmost importance. Due to slowness of existing algorithms, much of these data is archived without proper analysis. Thus, there is a sudden demand for a fast clustering algorithm to cope up with massive image data collection.

In our paper, we propose an extremely fast supervised clustering algorithm for images. We use P-Trees to classify images as explained in the following sections. These data structures basically convert the attributes comprising the image into vertical strips of binary data. We then perform P-Tree specific operations to achieve extremely fast results. The overall structure of this paper is as follows: In the next section, we provide background information about P-Trees. In section 3, we propose our novel FAUST_P algorithm and explain it using the IRIS dataset available from [2]. In section 4, we give the implementation details followed by the algorithm analysis in section 5. Finally, in section 6, we state our conclusion and provide a list of future work based on our existing algorithm.

2 P-TREES

In this section, we give some background information about the P-Trees which form the basis of our algorithm. As mentioned earlier in the abstract, P-Trees are data-mining-ready, compressed and lossless data structures. Simplest forms are the Peano-Trees which are bitwise trees and can be 1, 2 or n-dimensional depending on the application. For e.g. a spatial image can be efficiently represented by 2-dimensional Peano-Trees. P-Trees have been used in a wide variety of research areas including text mining [1], DNA Microarray data analysis [7], association rule mining [4], etc.

Consider a dataset X with d attributes represented as X = (A1, A2 ... Ad) and the binary representation of any k-th attribute, Ak, be represented as bk,m−1bk,m−2bk,m−3...bk,0. Here, m is the number of bits required to represent values in Ak. For e.g., 9 can be represented by 1001, so m = 4. We then decompose each attribute into bit files, i.e. one file for
each bit position [6]. The P-Tree is simply constructed by taking bit files (one at a time) and recursively partitioning them into halves and sub-halfes until each sub-half is absolutely pure i.e. entirely 1-bits or 0-bits. For more information on P-Tree structure, construction and operations, please refer to [5]. In this paper, we only provide FAUST_P-algorithm specific details about P-Trees. More specifically, we discuss the inequality Predicate-Trees which are used in evaluating the range predictions.

2.1 Inequality P-trees

The Inequality P-Trees are a class of Predicate trees used in evaluating range predictions. They represent data points within a dataset which satisfy the inequality predicates like \( x \geq v \), \( x < v \), etc. A P-Tree representing \( P \equiv \) can be computed using the following algorithm: Consider \( x \) to be a data point inside a dataset \( X \). Let \( x \) be an \( m \)-bit data and let \( P_{m-1} \), \( P_{m-2} \), ... \( P_0 \) be the P-Trees representing the dataset \( X \). Let \( v \) be the value represented by \( b_{m-1} \), \( b_{m-2} \), ... \( b_0 \) for which we need to find the equality. Then a Predicate Tree representing the predicate \( x \geq y \) is given by \( P_{k \geq v} = P_{m-1} \& P_{m-2} \& ... \& P_0 \) where \( i \) varies from 1 to \( m \). Also, \( op_i \) is a binary and operation if \( b_i \) is 1 and a binary or operation if \( b_i \) is 0. The operators are right binding. For e.g. \( P_{k \geq 14} \) or \( P_{k \geq 1110} \) can be computed as \((P_3 \& (P_2 \& (P_1 \& P_0))\)). Values less than 14 in the given example can by computed by taking the complement of the Predicate tree.

2.2 Computation of Mean using P-Trees

Our algorithm uses computation of mean from the vertical representation of the data. P-Trees can be efficiently used to compute the aggregate functions such as count, sum, mean, max, min, etc. For e.g. count is nothing but the RootCount of the P-Trees. Sum function can be computed using the following algorithm:

\[
\text{total} \leftarrow 0 \\
\text{for } i = n - 1 \text{ to } 0 \text{ do} \\
\text{total} \leftarrow \text{total} + 2^i \ast \text{Rootcount}(P_i) \\
\text{end for}
\]

Here, \( n \) is the number of bits representing the attribute or column of the table. The mean can be computed by total / count. A detailed research on max, min and other aggregate functions is given in [6].

3 FAUST_P Algorithm

In this section, we propose our algorithm called the FAUST_P or Fast Attribute based Unsupervised and Supervised Table Clustering and explain it using the IRIS dataset. The dataset is freely available from the UCI Machine Learning Laboratory [2]. The pseudo code of the algorithm is as follows:

Initially, let \( P_{\text{REMAINING}} \) be a pure-1 P-Tree.

From the training set:
1. For each attribute, calculate the mean for each class, sort ascending according to the mean.
2. Choose and remove a MeanTable record with maximum \( \text{gap}_{\text{RELATIVE}} \).
   - Calculate \( C_L = \text{Mean} - \text{gap}_L/2 \) to produce \( P_L = P_A > C_L \).
   - Calculate \( C_H = \text{Mean} + \text{gap}_H/2 \) to produce \( P_H = P_A > C_H \).
   - Assign class mask \( P_{\text{CLASS}} = P_L \& P_H \& P_{\text{REMAINING}} \).
3. Repeat step 2 above until all the classes have a P-Tree.
4. Repeat steps 1, 2 and 3 until means stop changing (much).

Figure 1: FAUST_P Algorithm

The algorithm functions on IRIS data in the following way. We first remove the decimal point from all the data points to use integer values. We then choose 10 random samples from each of the class, i.e. Iris-setosa, Iris-versicolor and Iris-virginica. Because of space limitation, we do not show the 30 points (10 points per class) but their means (m) in ascending order are listed in the following table.

<table>
<thead>
<tr>
<th>SL</th>
<th>m</th>
<th>SW</th>
<th>m</th>
<th>PL</th>
<th>m</th>
<th>PW</th>
<th>m</th>
</tr>
</thead>
<tbody>
<tr>
<td>se</td>
<td>51</td>
<td>ve</td>
<td>32</td>
<td>se</td>
<td>14</td>
<td>se</td>
<td>2</td>
</tr>
<tr>
<td>vi</td>
<td>63</td>
<td>vi</td>
<td>32</td>
<td>ve</td>
<td>47</td>
<td>ve</td>
<td>14</td>
</tr>
<tr>
<td>ve</td>
<td>70</td>
<td>se</td>
<td>35</td>
<td>vi</td>
<td>60</td>
<td>vi</td>
<td>25</td>
</tr>
</tbody>
</table>

Table 1: IRIS data mean

The successive mean gap (mg) is calculated for each class and all the attributes.
In the table, cl and att represent the class and the attribute respectively while gapL and gapH are the low and high side of the mean. grelative in the algorithm is abbreviated as gr in the table. Once the table has been constructed, we sort it descending on the gr parameter. Since 0.621 and 0.44 are the highest relative gaps among their class, they represent the cut point for classification. It may be noted that PW is the single most relevant attribute in separating out one class from the other. It is not surprising to get this kind of result in our current "single attribute cutoff" approach. Of course, in case of image classification, this would be different since we would expect red attribute to be the most relevant attribute in separating out red colored objects, blue attribute for blue objects, etc. Back to our IRIS dataset, clearly, Iris-setosa is the first class to be classified since it has the maximum relative gap. We now calculate the \( C_L = 2 - 12/2 = -4 \) and \( C_H = 2 + 12/2 = 8 \). Since \( C_L \) is negative, it is assigned a pure-1 P-Tree. For \( C_H \), we convert 8 to binary 01000 and apply the Predicate Tree formula, i.e. \( P_{A>C_H} = (P_{A>4} \land (P_{A>3} \land (P_{A>2} \land (P_{A>1} \land P_{A>0})))) \). Here, \( P_{A>4}, P_{A>3}, P_{A>2}, P_{A>1} \) and \( P_{A>0} \) are the P-Trees representing the PW attribute for the entire dataset. Thus \( P_{setosa} = P_{A>C_L} \land P_{A>C_H} \land \text{remaining} \). \( \text{remaining} \) now holds only those data points which have not been classified as \( P_{setosa} \). We repeat the above procedure to classify \( P_{versicolor} \) and \( P_{virginica} \).

### 4 IMPLEMENTATION

We execute our algorithm on a standard Ubuntu Linux machine with Java and GNU g++ compiler installed on it. We use an extractor program written in Java to extract the red, green and blue values along with the x,y pixel coordinates of the image in raster order. The values are then converted into P-Trees using the APIs as defined in [5]. The P-Trees are then fed to the FAUST-P algorithm. The algorithm has been implemented in standard C++ language. It outputs the new P-Trees corresponding to the classes defined in the input. This is then sent back to the JAVA program to output the resultant image. The entire application is generalized to take any number of classes and attributes as input. To calculate the algorithm speed, we use the standard gettimeofday system call with a resolution of 1 microsecond. We conducted tests of our algorithm on several images. Most primitive among them is the aerial view of a car park to distinguish between cars of different color as well as road.

### 5 ALGORITHM ANALYSIS

As can be analyzed from the algorithm defined in section 3, FAUST-P algorithm has a complexity of \( O(k) \) where \( k \) is the number of attributes or columns. This is extremely fast considering the fact that all the horizontal methods have at least \( O(n) \) assuming no suitable indexing. The value of \( k \) is generally small ranging from 2 to 7 (in case of Landsat data). Even high-attributed images with \( k \) of for e.g. 200 can be rapidly classified in comparison to horizontal methods where \( n \) are of the order of 1 billion or even more. Our algorithm achieves an accuracy of 95% on IRIS dataset with only 1 epoch. Higher accuracy can be achieved at the cost of time.

### 6 CONCLUSION AND FUTURE WORK

In this paper, we propose a fast attribute-based unsupervised and supervised clustering algorithm for images. Our algorithm is extremely fast with a small compromise on the accuracy. This can be effectively
used for Landsat data or any other kind of image data where rate of data generation is much higher than time to analyze it. In our future work, we plan to propose a divisive method which considers all data points in one cluster initially and splits depending on maximum gap. We are also in the process of using standard deviation for better accuracy.

References


