An Enhanced Similarity Index Approach to optimize clusters in Data

Birinder Singh Sarao
1Mata Gujri College, Fatehgarh Sahib, Punjab, India

Abstract
Data Clustering being a serious issue as it may lead to major flaws in data sets. As clustering between various documents is based on the similarity index between the data files. This research paper uses cosine similarity and Gaussian similarity to calculate the radius of the clusters and then the performance of the same is analysed with the other existing algorithms like K-Means, DBOD etc based on various parameters like efficiencies, accuracy etc. Proposed algorithm is an enhanced version of the improved similarity indexes on the depth based clustering algorithm, which is implemented on real data sets. The clustering of proposed approach is done in such a manner that every document gets a second chance to be adjusted in some cluster and hence the chances of being a document to be an outlier are minimal.

Index Terms: Clustering, K-Means, Dice-Coefficient, Gaussian Similarity, Outliers

1. Introduction
Clustering algorithms are one of the types of unsupervised machine learning. Unsupervised learning uses unlabelled data. Most of the clustering algorithms work on the distance calculation evaluation. Details are given as:-

a. Calculation by Distance Measures
The distance similarity measure calculates the distance between two or more data elements in a list. The distance can only be calculated between the vector values and hence the data is converted into numeral vectors. There are lot of conversion methods like word to vector model, ASCII conversion, and intermediate word values solutions [1, 2, 18].

i. Cosine Similarity: It is the cosine of the two vectors values.
   \[ \cos_{\text{Sim}} = \frac{A \cdot B}{|A||B|} \]  

ii. Soft Cosine: It is soft distance measure in which the data element is dependent upon the second vector value. It is a modified form of Cosine Similarity

iii. Dice-Coefficient: It is a measure of distance which is based on common elements in two given document vectors [3].
   \[ \text{DiceCoefficient} = 2 \times \frac{|A \cap B|}{|A||B|} \]

2. Proposed Architecture
The main objective of the algorithm is to calculate the similarities between documents. Cosine and Gaussian similarity mutually develops improved similarity algorithm. After combining both similarities, resultant similarity is computed.

Clustering of an algorithm is divided into two sections as follows:
1) In this section, find radius function covers the evaluated radius [3].
2) This section covers the radius for creating cluster [4, 5, 6].

Initially two clusters are created by the proposed algorithm whose algorithmic formation is presented below.

---

Cluster Formation of Algorithm

```python
[clust1, clust2] = function to create initial clusters(R1, R2, R3)
 ImproveSim
 Clust1 = []; At first both clusters would be vacant
 Clust2 = [];                  
 Clust1count = 0;
 Clust2count = 0;
 // At initial stage, first element is considered in first cluster
 Clust1[clust1count] = ImproveSim(0, 1);
 Clustcount = Clustcount + 1;
 R = connection numbers in enhanced similarity
 For i=0: R
    Present_elem = ImproveSim(i, 2) // 1st connection collection
    allpositions = []; // holds the alternating position of element existing presently
    allpositions = find([ImproveSim(i, 2) == Present_elem])
    altsumvalues = ImproveSim(allpositions, 3);
```
kp = findminimum (Altrvalues) 
ks = Altrposition (ks); 
If kp > R3 
Clust1[Clust1count] = ImproveSim (ks, 2) 

2.1 Graphical User Interface (GUI) of Similarity Index Calculation

Figure 1 shows the processing of GUI (Graphical User Interface). In this figure, “Click here to upload the Audio” in this button we upload the Audio data. We have used only Audio data for this research.

3. Tool Utilized

In this research work, different algorithms are implemented on a same platform using MATLAB R2016a. The parametric based results and some details about the tool used in the work are discussed below:

<table>
<thead>
<tr>
<th>Table1 Hardware and Software Detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer</td>
</tr>
<tr>
<td>Random Access Memory</td>
</tr>
<tr>
<td>Platform</td>
</tr>
<tr>
<td>Another Hardware</td>
</tr>
<tr>
<td>Software</td>
</tr>
</tbody>
</table>

The hardware as well as software required to simulate the entire process is listed in Table1. The full name for MATLAB is Matrix Laboratory, which basically presents the LINPACK and EISPACK (package system project) (linear system packages) established by Matrix Software. Matrix Laboratory is a language of high-performance technical computing. It has a variety of visualization, programming environment and computing capabilities. MATLAB is the latest programming language, it also has a sophisticated data structure, including built-in tools for debugging, editing and help with object-oriented programming. These MATLAB capabilities make it an educational and excellent search tool. The unified application is collected in a package called the Toolbox[11,12].

To implement research work, different categories of datasets are used. These datasets are used in all the attained objectives of the research work. The description of one category of data sets is described below:

3.1 Description of Audio Data

The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) have 7356 total number of files (total size: 24.8 GB). The storage contains 24 professional actors (12 females, 12 male), vocalizing two lexically-matched statements in a neutral North American accent. Speech includes calm, happy, sad, angry, fearful, surprise, and disgust expressions, and song contains calm, happy, sad, angry, and fearful emotions. Every expression generated two different level of emotional intensity (normal, strong), with an additional neutral expression. All conditions are present in different modality formats: Audio-only (16bit,
48 kHz .wav), Audio-Video (720p H.264, AAC 48 kHz, .mp4), and Video-only (no sound). Note, there are no song files for Actor_18.

**Audio-only files:**
Audio-only files of all actors (01-24) are present in two different zip files (~200 MB each):
- Speech file (Audio_Speech_Actors_01-24.zip, 215 MB) have 1440 files: 60 trials per actor x 24 actors = 1440.
- Song file (Audio_Song_Actors_01-24.zip, 198 MB) contains 1012 files: 44 trials per actor x 23 actors = 1012.

4. Simulation Results

4.1 Efficiency

Efficiency is calculated with required average execution time to complete execution of an algorithm [7,8].

\[
Efficiency = \left( \frac{ClusteredDocs \times 2}{TotalDocs} \right) \times 100
\]

4.2 Accuracy

It is the proximity of a computation to the true value which is calculated by taking true positive and true negative with a fraction of true positive, true negative and false positive with false negative [7,9,10].

\[
Accuracy = \frac{Tp + Tn}{Tp + Tn + Fp + Fn}
\]

Where \( Tp = \text{Truepositive} \), \( Tn = \text{Truenegative} \), \( Fp = \text{falsepositive} \) and \( Fp = \text{falsenegative} \)

Table 2 shows the evaluation of parameter Efficiency and Accuracy with datasets on different algorithms. It represents that accuracy and efficiency of the proposed algorithm is far better than other existing algorithms.

<table>
<thead>
<tr>
<th>Datasets used</th>
<th>Efficiency of Proposed Approach</th>
<th>Accuracy of Proposed Approach</th>
<th>Efficiency of K-Mean</th>
<th>Accuracy of K-Mean</th>
<th>Efficiency of DBOD</th>
<th>Accuracy of DBOD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset 1</td>
<td>169.64</td>
<td>68.35</td>
<td>103.35</td>
<td>55.13</td>
<td>100.15</td>
<td>45.21</td>
</tr>
<tr>
<td>Dataset 2</td>
<td>182.21</td>
<td>82.26</td>
<td>113.13</td>
<td>61.63</td>
<td>102.19</td>
<td>47.31</td>
</tr>
<tr>
<td>Dataset 3</td>
<td>194.73</td>
<td>91.02</td>
<td>124.52</td>
<td>64.24</td>
<td>103.42</td>
<td>49.10</td>
</tr>
</tbody>
</table>

Table 2: Comparative Analysis on different datasets based on various parameters

![Efficiency Comparison Graph](image)

**Figure 2** Comparison of Efficiencies
Figure 2 shows the comparison of parameter Efficiency with different algorithms including proposed algorithm on audio dataset. It depicts that proposed approach is more efficient as compared to others.

![Accuracy Comparison Graph](image)

**Figure 3** Comparison of Accuracies

Figure 3 shows that accuracy of proposed algorithm is better than DBOD and K-Means algorithms [15]. The proposed algorithm holds an average accuracy of 89% whereas DBOD and K-Means hold 45% and 56%, respectively.

![Time Consumption in Minutes](image)

**Figure 4** Time Consumption in Minutes
Figure 4 represents the total time (minutes) consumed in order to cover all the text files. It depicts that on an average the total time consumed by the proposed approach is about 10 minutes i.e the time taken by DBOD (11.5 minutes) and K Means (12.9 minutes). Thus the time efficiency of the proposed algorithm is 15 % better than the other algorithms.

![Figure 5 Outlier Vs Datasets](image)

5. Conclusion

The objective of the proposed algorithm is to efficiently form the clusters from the data sets and further which will result in efficiently detecting the outliers. Various parameters like accuracy, efficiency, time consumed and detection rate of outliers based on the cluster formation is computed.

Current research work has a lot future responsibilities. The optimization in the current research can be achieved by the Swarm Intelligence Category algorithm’s like Ant Colony Optimization. Deep learning can be taken as an option in machine learning.

References

