Chapter 4

DESCRIPTION OF THE WORK

This research introduces a new approach called Vector quantization for Preserving Privacy during Data Mining. The proposed work deals with the concepts of Vector quantization techniques. VQ Codebook Generation Algorithms are implemented and the performances of the algorithms were compared in terms of accuracy and Distortion parameters between the Original Data and Transformed Data. Vector Quantization is generally used in Signal processing, pattern recognition and Computer based applications. However, vector quantization is found to be a Lossy Compression technique.

It is been proposed in this thesis that Vector quantization is used for Privacy Preservation by approximating each vector (point of data) to the other [23]. More over Data is not compressed and only Quantization of the data is performed so that the privacy is preserved.

4.1 Types of quantization:

There are mainly two types of quantizations.

1) Scalar Quantization
2) Vector Quantization

“Scalar quantization deals with the quantization of samples on a sample by sample basis”, while “vector quantization deals with quantizing the samples in groups called vectors”. Vector quantization Provides optimal performance at increased cost of computational complexity and memory requirements.

4.1.1 Scalar Quantization:

Scalar Quantization is one of the Quantization techniques. It plays essential role in speech coding systems. In Scalar quantization, the input sample are quantized independently. An N-level scalar quantizer may be viewed as a one-dimensional mapping of the input range R onto an index in a mapping table (or codebook) C. As stated in the methodology chapter, quantization is viewed as Constructing code book,
encoding and reconstruction. So, The receiver (decoder) uses index table to reconstruct an approximation to the input level.

\[ Q : \mathbb{R} \rightarrow \mathbb{C} \quad \mathbb{C} \subseteq \mathbb{R} \quad (4.1) \]

4.1.2 Vector Quantization:

The Ancient and best example of Quantization is rounding off, It was first introduced and implemented by Sheppard for many applications. Number ‘S’ can be rounded off to the nearest integer, say Q(s), with quantization error e=Q(s)-S. In reality, Quantization used for data compression. The key point of quantization is to divide large set of points (vectors) into groups (or regions) having approximately the equal number of points closest to them. Each group (or region) is represented by its centroid, as in k-means clustering and some other clustering algorithms.

Vector Quantization (VQ) is an efficient and simple approach for data compression. Since it is simple and easy to implement, VQ has been widely used in different applications, such as pattern recognition, image compression, speech recognition, face detection and so on [97].

Vector quantization consists of following steps:

1. Generating Codebook from Training Data.
2. Encoding the original data with the help of Codebook by nearest neighbor search;
3. Using the index table, data will be reconstructed by looking up in the codebook.

Vector quantization [96] is a process whereby the elements of a vector of k data samples are jointly quantized. Vector quantization (VQ) is generally used for data compression. In previous days, the design methodology of a vector quantizer (VQ) is treated as a big problem in terms of the need for multi-dimensional integration. Linde, Buzo, and Gray (LBG) introduced an algorithm for Vector quantization design based on training sequence. A VQ that is designed based on this algorithm are referred as LBG-VQ.
Figure 4.1 : Components of Vector Quantizer

The main component of a Vector Quantizer (VQ) is a codebook CB of size N*K, Vector Quantizer maps the K-dimensional space $R^k$ to the reproduction vectors (also called code vectors or code words)[94][95]:

$$Q : R^k \rightarrow CB, \quad CB = (Y_1, Y_2, ..., Y_N)^T, \quad Y_i \in R^K$$

(4.2)

The codebook contains finite set of vectors also called as code words, $Y_i$; $i = 1, 2, ..., N$. The codebook vectors are obtained through a clustering or training process in order to represent the training sequence. In the basic approach to VQ, the encoder reduces the distortion $D$ to give the optimal estimated vector $X_i$;

$$X_i = \min_{j \in CB} D(X_i, Y_j)$$

(4.3)

The above formula is for nearest neighbour encoding. The code rate or rate of a vector quantizer is described in terms of bits per component.

$$r = \frac{\log_2 N}{K}$$

(4.4)

The above formula (4.4) measures the number of bits per vector component and that can be used to represent the input vector and mainly indicates the accuracy or precision that can be achievable with the vector quantizer, if the codebook constructed effectively[98][73][97]. During the codebook generation process, codeword will be generated until the desired number of codeword’s has been generated. Suppose it is $N = 2^r$, then encoding search complexity and codebook storage size (i.e., memory complexities) can grow exponentially with dimension k and rate r.
Vector quantization is a procedure which requires a rich combination of training samples to produce codebooks which can be sufficiently robust for quantization of data not presented in the training set. Examples of some of the areas which might enrich the training set include varying microphones, relational data, acoustic background noise, different languages and gender.

Generally, a large training sequence or large number of input vectors will produce a effective codebook, codebook plays important role. Codebook is used for encoding and decoding the codes and can be used in data transmission and compression. Limitation in designing the codebook is optimization problem. There is several number of codebook design algorithms were there like Mean-distance-ordered Partial Codebook Search (MPS), Principal Component analysis (PCA), Generalized Lloyd Algorithm (GLA). Generally GLA yield only locally optimized codebooks. More advanced methods, such as deterministic annealing and genetic optimization have promised to overcome the drawback of local optimal at the expense of greater computational requirements and memory requirements.

A VQ is just like an approximator. The Key point is just ``rounding-off'' (say to the nearest integer). An example of a one-dimensional VQ is shown below:

```
<table>
<thead>
<tr>
<th>00</th>
<th>01</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>-4</td>
<td>-3</td>
<td>-2</td>
<td>-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 4.2 : one-dimensional Vector Quantizer

In figure 4.2, numbers less than -2 are rounded (i.e, approximated) to -3. Numbers between -2 and 0 are rounded to -1. Numbers between 0 and 2 are rounded to +1. All the numbers greater than 2 is approximated by +3. Note that the approximate values are uniquely represented by 2 bits. It is called as one-dimensional, 2-bit VQ and it has a rate of 2 bits/dimension.
An example of a Two-dimensional VQ is shown above. Here, pair of numbers falling in a particular region is grouped by a red star associated with that region. In the Figure 4.3, there are 32 regions and 32 red stars, each of which can be uniquely represented by 4 bits. Hence, this is a two-dimensional, 4-bit VQ and its rate is 2 bits/dimension.

### 4.2 DESIGN PROBLEM:

The VQ design problem[11][12] can be stated as follows.

**Problem Statement:** For the given Training sequence with set of vectors, distortion measure i.e., $\epsilon$, and the number of code vectors, Find a codebook which gives the smallest average distortion among the encoding regions.

Let us assume that the training sequence is having $M$ source vectors

\[
TV = \{V_1, V_2, V_3, ..., V_M\}
\]  

(4.4)

$TV$ represents Training sequence and $V_1, V_2, V_3, ..., V_M$ is having $M$ Vectors This work takes training sequence from UCI Repository and also real world company data. Suppose, if the source is a speech signal, then the training sequence can be obtained by recording several long telephone conversations. This Work Takes Input data from UCI Repository. Generally $M$ is sufficiently large so that one can design effective codebook with best approximations and efficient codebook depends on codebook design algorithm. Let us assume that the source vectors are K-dimensional, e.g.,
Let \( N \) be the number of code vectors and
\[
\text{Let } CB = \{ C_1, C_2, C_3, \ldots, C_N \}
\]  
(4.6)

CB represents the codebook. Generally Code book contains set of code words or cluster centroids, each code word is \( K \)-dimensional, e.g.,
\[
CB_n = \{ C_{(n,1)}, C_{(n,2)}, C_{(n,3)}, \ldots, C_{(n,K)} \}, n = 1, 2, 3, \ldots, N
\]  
(4.7)

Let \( ER \) be the encoding region corresponding to the code vector \( C_n \)
\[
P = \{ ER_1, ER_2, \ldots, ER_N \}
\]  
(4.8)

Represents the partition of the whole space and \( P \) consist of all encoding regions.

If the source vector \( V_m \) is in the encoding region \( ER_n \), then its approximation (denoted by \( Q(V_m) \)).
\[
Q(V_m) = C_n \quad \text{if } V_m \in ER_n
\]  
(4.9)

\( D_{\text{ave}} \) is the mean squared-error used as distortion measure; the average distortion is given by:
\[
D_{\text{ave}} = \frac{1}{MK} \sum_{m=1}^{M} \| V_m - Q(V_m) \|
\]  
(4.10)

Privacy preserving data mining problem deals with Accuracy and distortion for performance evaluation, if \( D_{\text{ave}} \) is low then one can achieve better accuracy for data mining results. During the transformation process, it is required to analysis and evaluates the optimal condition for best approximation of code words

### 4.2.1 Optimality Criteria:

Let \( N_N \) and \( C_n \) are the two parameters that can be used for denoting minimization problem, and it should satisfy the two conditions such as nearest neighbour condition and centroid condition.

### 4.2.2 Nearest Neighbor Condition:

\[
N_N = \{ V : \| V - C_n \|^2 \leq \| V - C_{n'} \|^2 \quad \forall n' = 1, 2, \ldots, N \}
\]  
(4.11)

Where \( ER_n \), encoding region and \( C_n \) is the code vector (centroid). The Nearest neighbor condition says that each encoding region \( ER_n \) should consist of set of vectors that are near to \( C_n \) than any of the other code vectors.
4.2.3 Centroid condition (or code vector):

\[ C_n = \frac{\sum_{m} V_m}{\sum_{x_n=s_n} 1} \]  \hspace{1cm} (4.12)

This condition says that the code vector or centroid \( C_n \) is average of all the training vectors that are in encoding region \( ER_N \). In general, minimum one training vector should belongs to each encoding region \( ER_N \) (so that the denominator in the above equation will be never 0).

4.3 CODEBOOK GENERATION ALGORITHMS:

Traditional definition for codebook is, it is a document for gathering and storing of codes. The term codebook can be referred in cryptography, social science and data compression. Usually codebook contains Lookup table. Encoding and decoding can be performed with the help of lookup table. Lookup table is generally used for security purpose. So, it is been proposed in this work for preserving privacy during data mining with the help of Lookup table.

4.3.1 LBG Algorithm:

The VQ-LBG design algorithm [12] is a recursive algorithm which provides solution for the above two optimality criteria. This algorithm requires an initial codebook \( C^{(0)} \). This initial codebook is obtained by taking the average of entire training dataset this method can be called as splitting method. In this process, an initial code vector is set as the mean of the entire training sequence. This code vector is then split into two.

The recursive algorithm is run with these two vectors as the initial codebook. The two code vectors are then divided into four code vectors by individually and the process is repeated until the desired number of code vectors is obtained. The algorithm design is described in below.
4.3.2 LBG Algorithm design:

1. Given T Fixed $\varepsilon > 0$ to be a small number
2. Let $N=1$ and

$$C_i^* = \frac{1}{M} \sum_{m=1}^{M} V_m$$

Calculate

$$D_{\text{ave}}^* = \frac{1}{MK} \sum_{m=1}^{M} \| V_m - C_i^* \|^2$$

3. Splitting  For I=1,2…N, set

$$C_i^{(0)} = (1+ \varepsilon) C_i^*$$

$$C_{N+i}^{(0)} = (1- \varepsilon) C_i^*$$

Set $N=2N$

4. Iteration

Let $D_{\text{ave}}^0 = D_{\text{ave}}^*$. Set the iteration index $i=0$

a. For $m=1, 2, …, M$, find the minimum value of $\| V_m - C_n^{(i)} \|^2$, over all $n=1, 2, …, N$ Let $n^*$ be the index which achieves the minimum.

Set $Q(V_m) = C_n^{(i)}$

b. For $n=1, 2, …, N$, update the code vector

$$C_n^{(i+1)} = \frac{\sum_{n=1}^{M} V_m}{Q(V_m)}$$

Set $i=i+1$

d. Calculate

$$D_{\text{ave}}^{(i)} = \frac{1}{MK} \sum_{m=1}^{M} \| V_m - Q(V_m) \|^2$$

e. If $(D_{\text{ave}}^{(i-1)} - D_{\text{ave}}^{(i)}) / D_{\text{ave}}^{(i-1)} > \varepsilon$, go back to step (a)

f. Set $D_{\text{ave}}^* = D_{\text{ave}}^{(i)}$. For $n=1, 2, …, N$ set $C_n^* = C_n^{(i)}$ as the final code vectors

g. Repeat steps (c) and (d) until the desired no of code vectors are obtained. Here the LBG design algorithm is run with $\varepsilon = 0.001$. 
This Design algorithm can be used for constructing codebook with $\epsilon=0.001$, this value is not constant and it is up to the requirement of the application. Suppose, if we would like to have more distortion between the original data and transformed data then the value of $\epsilon$ can be increased.

4.3.3 Code Book Generation using LBG:

Codebook construction is the main step in the Quantization process. Basic thing which is required for generation of codebook is the training sequence. The training sequence is obtained from UCI Data repositories water plant treatment data set[12].

1. Let R represents the region of the training data.
2. Let T represents Training vectors.
3. Produce an initial codebook from the training data, now it will be the centroid or mean of the training dataset and let the initial codebook be C.
4. Divide the initial codebook C into $C^+_n$ and $C^-_n$; where $C^+_n$ is calculated with $C^+_n = C(1+\epsilon)$ and $C^-_n$ is calculated with $C^-_n = C(1-\epsilon)$; $\epsilon=0.01$ is the minimum expected distortion between old and new code words.
5. Evaluate the difference between the training data and each of the codeword’s $C^-_n$ and $C^+_n$ and call the difference as D
6. Now divide the training data into two regions called R1 and R2 based on the difference D between the training data and the codeword’s $C^+_n$ and $C^-_n$. The training vectors which are closer to $C^+_n$ falls under the region R1 and the training vectors which are closer to $C^-_n$ falls under the region R2.
7. The training vectors which are there in region R1 is called as TV1 and the training vectors which are there in region R2 is called as TV2.
8. Obtain the new centroid or mean for TV1 and TV2. Let the new centroids be CR1 and CR2.
9. Replace the old centroids $C^-_n$ and $C^+_n$ by the new centroids CR1 and CR2
10. Compute the difference between the training sequence and the new centroids CR1 and CR2 and let the difference be $D^f$.
11. Repeat steps 6 to 11 until $\frac{D^f - D}{D} < \epsilon$. 

12. Repeat steps 5 to 12 till the required number of codewords in the codebook are obtained. where \(N=2^b\) represents the number of codewords in the codebook and \(b\) represents the number of bits used for codebook generation, \(D\) represents the difference between the training sequence and the old codewords and \(D'\) represents the difference between the training sequence and the new codewords [96][97].

In vector quantization process, code book plays a very important role and accuracy of data mining results depends mainly on the effective design of code book, because quantization performed with the help of indices of codebook.

This Thesis uses LBG algorithm for designing of codebook initially, then Modified LBG. Here code book contains centroids of entire training sequence and it is a Transformed version of training data set. This new data set contains approximated data values, not exact values like original data set.

4.3.4 Flow chart representation of LBG:

The LBG algorithm designs works in M-stages i.e, LBG design algorithm is used in constructing codebook LBG design algorithm takes M-vectors. In this work, matrix representation is considered for representing the vectors. After constructing 1-vector codebook, then construct 2-vector codebook by using splitting process, and continues the splitting process until the desired M-vector codebook is obtained.

Figure: 4.4 below shows, in a flow diagram, the detailed steps of the LBG algorithm. "Cluster vectors" is the nearest-neighbor search procedure which assigns each training vector to a cluster associated with the closest codeword. Using the concept of centroid condition and nearest neighbour search, training vectors will be replaced with nearest neighbour codeword in the codebook.

Find centroids" is the centroid update procedure. "Compute D (distortion)" sums the distances of all training vectors in the nearest-neighbor search so as to determine whether the Procedure has been converged. From the Training data, codebook will be constructed like one which is explained in Chapter 3.
Figure 4.4 Flow chart representation of LBG
Constructing a codebook is a recursive process. The first step is to divide the whole training set into two half’s based on average of that training set. As shown in the flow diagram, the process will be repeated until the desired number of code words has been generated. After constructing the code book, next step is to perform encoding operation. As a result of encoding, index table will be generated. This index table is the compressed version of original data. Sometimes, this compressed version can be given for transmission. In privacy preserving data mining, data will be reconstructed using index table. Finally, comparison between the original data and reconstructed takes place and measures the distortion also.

4.3.5 Code book Generation using Modified LBG Algorithm:

Approach 2 Proposes Modified LBG Algorithm in codebook generation step for Privacy preserving Data mining. In Modified LBG, Local optimum problem of LBG is solved by using the concept “utility of the codeword”.

4.3.5.1 What is Utility of codeword?

LBG codebook generation algorithm is not having proper codeword adjustment. In general they can move only through contiguous regions, so a bad initialization leads to bad final quantizer because of this limitation. So, it is possible that LBG quantizer can produce a region with empty cells i.e, no codeword’s some times a region with large number of codeword’s. So, it is possible to adjust the codeword’s between the regions it is called as distortion equalization or utility of codeword.

Let us consider a scenario,

![Utility problem in LBG algorithm](image)

Figure 4.5 : Utility problem in LBG algorithm
The above figure represents three regions Called as R1, R2 and R3 two codeword’s C1 and C2. R2 is a region with empty cells and R3 is region with number of codeword’s. The solution is to move the codewords so that the distortion among the regions can be uniformed.

Let us consider another scenario after solving the utility of the codeword’s problem,

![Figure 4.6: Modified LBG: Distortion equalization](image)

**Vq-Modified LBG Algorithm:**

Step1: \( U_t = \frac{D_j}{D_{\text{mean}}} \)

Step2: \( D_{\text{mean}} = \frac{1}{N} \sum_{j=1}^{N} D_j \)

Step3: Divide the codewords in the codebook into two regions R1 and R2.

Step4: Code words whose \( U_t \) is higher than 1 will be placed in R1

Code words whose \( U_t \) is lower than 1 will be placed in R2

Step5: codeword with smallest distortion in R1 will be shifted to a nearby codeword in R2.

Centroids will be rearranged if required based on the above conditions.
4.3.6 Flow chart representation of Modified LBG

- Obtain the Training Sequence
- Find Centroid
- Split Centroid
- Compute D
- Split the Sequence into Regions
- Obtain Centroids for Regions
- Replace old Centroids with new centroids
- Compute $D'$
- $\frac{(D' - D)}{D} < \epsilon$?
  - Yes
  - New code book calculation
  - MLB G Block
  - $\text{Is size } = \log_2 n$?
    - Yes
    - Stop
    - No
    - No

Figure: 4.7 Flow chart representation of Modified LBG Algorithm
Modified LBG algorithm gives effective codebook when compared to traditional LBG. Figure 4.7 has M-LBG block, which exactly performs codeword equalization, resulting in a new codebook. This process is repeated until desired numbers of codeword’s are assigned. In the Modified LBG algorithm, centroids are arranged properly with respect to rearrangement of objects into the regions. Hence, the objects are arranged properly when compared to the traditional LBG algorithm.

Modified LBG algorithm produces an intended result in the codebook when compared to the traditional LBG algorithm where mean square quantization error is less in Modified LBG. Sub optimal quantization is achieved while calculating the process of voronoi partition. The Overhead occurred by Modified LBG is negligible when compared with the time complexities of the traditional LBG algorithm.