Cluster Optimization using Appropriate Nearest Neighbour

Nazar Imam, Sandhya Tarar

Abstract: ***In this postulation, presents the clustering of facial images using machine learning algorithm such as nearest neighbor and approximate rank order clustering. Clustering is a technique for classifying similar kind of object based on their trait. Clustering of images is challenging problems and there is still a considerable measure of work that needs to be done in this area. Problems in clustering large dataset is to choose the quantity of clusters and evaluating the obtained clusters. Clustering regard as the most important unsupervised learning as it manages finding a structure in an accumulation of unlabeled information. A loose meaning of clustering could be "the way toward sorting out articles into clusters whose people are nearby one means or another. A cluster in this manner is an accumulation of items which are "comparable" amongst them and are "divergent" to the articles which are place with different cluster. This thesis presents a work to improve clustering method to decrease the number of clusters in a LFW (Labeled face in wild) dataset. Previous work uses kd tree a nearest neighbor method and approximate rank order clustering method to find cluster on LFW dataset. our proposed method implement ball tree a better nearest neighbor algorithm to reduce the number of clusters created by previous method.***

Index Terms: Face Recognition, Face Clustering, Deep Learning, Scalability, Cluster Validity.

I. INTRODUCTION

As a research subject self-face identification by machine is extremely fascinating topic from a quite long time. In our proposed work given issues are inscribed. We have to bunch a abundant amount of facial picture according to various characteristic present in them. This issue emerges in a wide range of zones from law requirement to web-based social networking where they have to deal with the substantial accumulation of pictures arranged by several million and the names appended to them are frequently absent. Clustering lessens the computational cost and evaluation of each face in whole dataset is not required.

Clustering could be defined as "the method of sorting out items into clusters whose identities are related in some way" thus a cluster can be seen as sets of object that share some common attributes and are different from the object based on those attributes. In web based life and large database a huge unknown number of singular personalities are present, which is challenging from adaptability point of view since run time have to be determined with numbers of clusters. As in [1] challenges in bunching mean to inscribed the given issue: abundant unlabeled face pictures are given, bunching unlabeled images into discrete personalities. This situation is encounter in various situation going from web based life to legal authorization, where several million of facial image of different individual appears.

Revised Manuscript Received on 19 December 2018.

Nazar Imam, School of ICT, Gautam Buddha University, Greater Noida (U.P), India.

Dr. Sandhya Tarar, School of ICT, Gautam Buddha University, Greater Noida (U.P), India.

Often the character name of facial images either contain noise or absent. Both run time and clustering quality can be highly affected if the number of cluster or number of face image extend from a several thousand to many millions. Talking about online networking nearly 350 million images on an average is transferred as per detail from Facebook and a vast number of pictures are of individuals. Subsequently late advancement in unrestricted image identification, we try to facilitate the inconvenience of the essential face grouping issue by using a best in class convolutional neural framework based face depiction [2]. To decide the given face whether or not belong to argue identity and whether a stated face picture is coordinated with a complete pre-recorded database the problem of face recognition can be divided as either 1:1 matching (authentication problem) or 1:N (search problem).

Figure 1: How Unlabeled Faces Dataset Looks Like

Figure 2: Clustered Faces

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

II. RELATED WORKS

Anil k. jain et al, [1] Demonstrative development, enthusiastic improvement in applications for instance computerized imaging, and video, Internet quest and advances in identifying and limit development has generated some high dimensional ,high volume feature index. Arranging data is one of the most elementary mode of learning and perception for example, a strategy of classifying different items into structure of domain, kingdom, class, and object etc. Inexpensive videos and digital camera generate huge archive of videos and image which are mostly stored in electronic media providing potential for organizing, grouping and automatic data analysis. Clustering is used in many field K-mean is one of the most simple and accepted clustering algorithm. It was proposed over 50 years ago and is still widely used. K-mean algorithm focus to split up n data into K cluster with the nearest mean.

Jeffrey Ho et al. in [2] From long time in computer vision grouping or clustering pictures of 3 D object is a dynamic research field. Studies of image under varying lightning condition image from same object look very different whereas image of different object look close to each other. Viewing condition in computer vision commonly involve relative orientation between the external light under which image is taken and the object and camera. Image of identical object under varying lighting and viewing condition appears to be very different. Conversely image which appears similar may be originating from different object.

Dayong Wang et al. in [3] In spite of tremendous advancement in face identification, unconstrained face image remains a troublesome issue while looking through a huge gathering of people in a social event. Computer vision researchers are facing one challenge to formulate method to search person from billion on images on social sites.in 2013 white paper Facebook disclose that more than 350 million new images are uploaded per day, google and face book automatically detect faces from the images automatically and suggest name based on previously tagged images from their dataset and template generated from input image. To overcome this confront a face search system is proposed in a cascade framework which combine fast face search technique, ally with d with a state-of-the-art commercial off the shelf (COTS) matcher. They have discuss the two major challenge in large scale face search (i) loss in search accuracy (ii) with increase in gallery size increases the computational complexity.

Richard Hartley et al. in [4] Matching image descriptor image recognition and retrieval is one particular problem that they look in this paper. Based on multiple randomize KD-trees they have explain and examine a search method and improve result from KD-tree suggested. We take a gander at enhancing the KD-tree for a specific utilization ordering a substantial number of SIFT and different kinds of picture descriptors. We have broadened priority search, to priority search among numerous trees. By making numerous KD-trees from similar informational index and at the same time seeking among these trees, we have enhanced the KD-tree's search execution significantly. We have likewise utilized the structure in SIFT descriptors to lessen the time spent in backtracking. By utilizing Principal Component Analysis we have additionally expanded the KD-tree's search performance.

Radha Chitta et al. in [5] For organizing data a commonly used data is clustering, clustering algorithm based on kernel takes the non-linear data structure, when number of the object are large(more than tens of thousands) kernel based clustering do not performs well in terms of memory requirement and speed. While many other clustering algorithm do not perform well on real world data which are designed to deal large data set accept linear reparability of data. As it is predicted by EMC Corp IDC by the end of 2020 there will be 35 trillion gigabytes of data (pictures, videos, text) and most of them will be generated by social media and blogs. To let trouble free access by user and organising large amount data logically large-scale clustering is principal tool. To solve this obstacle kernel-based clustering method is used that insert the data points into high-dimensional non-linear manifold and defining their similarity using a nonlinear kernel distance function.

Xiaoshe Dong et al. in [6] Kd-tree is the most recent method for determining approximate nearest neighbor (ANN) field. In ANN framework descending search is needed for each query patch and nearby patch is needed propagation search and this gives scope to save search time. To compute ANN speedily they presented edge propagation kd-tree and shows that their proposed search technique is 2-3 time quicker and accuracy is nearly same as in propagation assisted kd-tree search technique. Traditional and productive method for arranging data is Kd-tree but in tree based technique search iteration is more efficient than backtracking phase. Traditional method for ANN append kd-tree and locality sensitive hashing (LSH). in many visual and computer vision application ANN is vital part and is a very difficult and complicated issue because for two images ANN are calculated in real time or near real time and number of patch is very large for two images.

III. FACE CLUSTERING AND EVALUATION

Smart phones and reconnaissance systems are spreading rapidly and keeps on developing in response to it number of pictures and their size also grows rapidly to handle this large image it is very time consuming and complex especially in social media and forensic investigation, hence we need to cluster the data based on different identities present in the given data set which reduces the analysis and investigation time to find the perpetrators and victims for example in Boston Marathon bombing where lacs of images are investigated in time sensitive investigation. For clustering we need to determine the faces of each individual after that we need to determine facial feature then nearest neighbor algorithm is applied on each face after that final clustering is done.

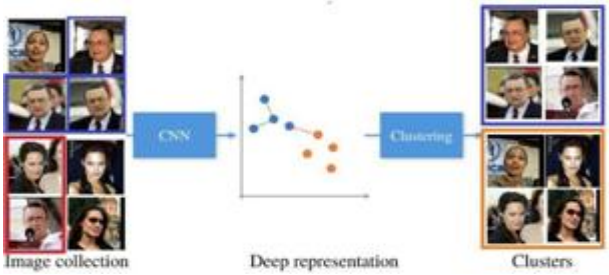

Fig.3 Clustered Data

A. Extracting Deep Feature

Motivated from [12, 13] a specific architecture of the system proposed here (illustrated in Fig. 3.3). There are four primary contrasts between the proposed system and the one in [13]: I)Rather than greyscale pictures color picture is given as input to the system; ii) utilization of a powerful face arrangement technique; iii) that haphazardly cut a 100×100 areas out of 110×110 from given picture, trailed even by its after-image to produce extra pictures to prepare the framework; and iv) For fast calculation at the time of training the contrastive cost layer are deleted.

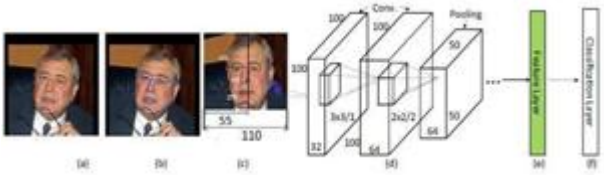

Fig.4 : Working of CNN: A shading picture is embedded a) facial imprints are uncovered, (b) picture is institutionalize, (c) the institutionalize picture is given to a CNN, (d) 320-aspects yield of the final average pooling layer is utilized as the face portrayal, (e) Grouping layer, (f) is utilized amid preparing as it were.

Table 1. Construction of Light CNN

Name	Filter size / Stride	Type	Output Size
Data	-	Data	110×110×3
Data Augmentation	-	Random Crop	100×100×3
Conv1-1	3×3/1	convolution	100×100×32
Conv1-2	3×3/1	convolution	100×100×64
Pool1	2×2/2	max pooling	50×50×64
Conv2-1	3×3/1	convolution	50×50×64
Conv2-2	3×3/1	convolution	50×50×128
Pool2	2×2/2	max pooling	25×25×128
Conv3-1	3×3/1	convolution	25×25×96
Conv3-2	3×3/1	convolution	25×25×192
Pool3	2×2/2	max pooling	13×13×192
Conv4-1	3×3/1	convolution	13×13×128
Conv4-2	3×3/1	convolution	13×13×256
Pool4	2×2/2	max pooling	7×7×256
Conv5-1	3×3/1	convolution	7×7×160
Conv5-2	3×3/1	convolution	7×7×320
Feature	7×7/1	average pooling	1×1×320
Dropout	dropout(keep-60%)		1×1×320
FC	Fully connection		10,575
Classification	softmax		10,575

We embrace a extremely deep architecture [14] (10 convolution layers altogether)for the system's convolution layers, and utilize filter of lesser width 3×3. This 3x3 array lessens the aggregate features to be studied, and the profound design improves the alignment of the system [7] and the obtained result is a 320 aspect of each picture which are employed in our classification system.

From an adjusted face picture only RGB pixel value are accepted by the input layer. Facial appearance are adjusted as pursues: i. Utilize the DLIB2 execution of Kazemi and Sullivan regression trees technique [15] to identify 68 facial marks ii. Depending on the eye positions images is rotated to make it position upright; iii. Averaging every facial marks of the eye and mouth area separately the middle point of the mouth and eyes are marked by red; by taking mid-point between the furthest left and furthest right facial marks pivotal locus in the image of the face is marked using blue iv. In view of the central point (blue point) face appearance is centered along x-axis; v. Determine the location along the vertical scale by putting the eye at central location at forty-five percent out of highest point in the picture and the mouth is focused below it at twenty-five percent the from the base of the picture. vi. Adjust the image both horizontally and vertically to 110 X 110. Even after this we can observe that the calculated pivot location isn't reliable crosswise over posture. In appearances displaying noteworthy yaw, the processed midpoint will be unique in relation to the one registered in a frontal picture, so facial milestones are not adjusted reliably crosswise over yaw.

We used ten layers in CNN, four maxpool layers along with one aggregated pool layer after the input layer. Each combine of conv layers is gathered \& associated consecutively. Initial 4 gatherings of conv layers are trailed with the maximum pool layer of filter width 2 X 2, whereas the final set of conv layers is trailed by a normal pool layer of filter width 7 X 7. In the last conv layer total number of filter and number of aspect in it are equal. As examined in [13], the ReLU [16] hub creates a sparse vector, which is unwanted for a face portrayal layer. In our system, we utilize ReLU hubs [16] in all the conv layers, aside from the last one, which is joined with a normal pooling layer to produce a 320 features of face portrayal.

B. Finding Approximate Nearest Neighbors

In a given dataset of N sample finding nearest neighbour is one of the common problem in clustering method, for large the rum time is $O(N^2)$. To overcome this problem approximate method is present in literature.

For calculating full k-NN graph an approximate nearest neighbor method is given by Chen et al.[7]. In our work the first-element of the dimensional tree suggests the complete simulation quantity. Alternative elements constitute square sub-sections which comprise weights, center-of-weight, and objects surrounded under these areas. It is one of the foremost proposed systems working on cataloging collection of all structures. Every stage of KD tree segments distance into halves, as appeared in figure 5, the dividing is achieved along one degree of the sud-division at the fine degree of the tree, alongside some other dimension inside the sud-divisions at the following stage, et cetera, repeating through the measurements. The partitioning keeps such that, at every sud-division, approximately one 1/2 of the focuses located away within the subtree that show in one aspect, and the rest placed on the opposite. Apportioning ends whilst a sud-division has now not as a notable deal as a given max no. of focuses.

a) *Randomize Kd-Tree*

Tree-based partitioning algorithm is one traditional approach among various approximate nearest neighbor algorithm. A k-d tree algorithm that split the data by developing pointer which further divide given data after capturing small portion of data through non-exhaustive search of a tree. k-d tree is a binary tree and its development is fast because no D-dimensional is not required only partition is done along the data axis. Once the k-d tree is developed only distance calculation is done to find the closest neighbor of an enquiry point. The k-d tree algorithm is fast only for low-dimension (D<20) neighbor search as D increases its efficiency decreases.

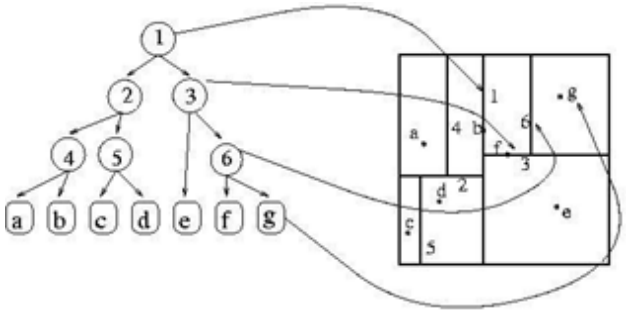

Figure 5: Illustrate a Randomize KD-Tree

In spite of the fact that the KD tree approach is quick for low-dimensional (D<20) neighbors looks, it ends up wasteful as D develops substantial: this is one indication of the purported "revile (curse) of dimensionality". To improve the efficiency the randomize k-d tree method [9] is proposed that produce multiple randomize k-d tree indices and in parallel search those indices and after visiting a fixed number of tree nodes the search operations stops

b) Ball Tree

To search the neighborhood nodes especially when number of dimension is very large an algorithm named Ball Tree data structure is introduce. Ball Tree method is highly effective in condition when the dimensions are very high in number and structure is hierarchical (binary) in structure. Ball tree is a multi-dimensional space it start with the formation of two cluster each represent a ball appropriately called a hypersphere. In a given n-dimensional space a given point belong to only in one hypersphere and in those hypersphere whose centroid distance from the given point is less. The point may be included in any one of the sphere the distance from each hypersphere centroid is same. Two sub-cluster are formed from each of the ball each forming a centroid again and the distance from the centroid of the sub-cluster decide the family of a given point. Most probably some hypersphere intersect each other but their distance from the centroid decide the cluster in which the given will be included.

The ball tree algorithm partition the given data in a progression of settling hyper-circles. This makes tree development more exorbitant than that of the KD tree, yet results in an information structure which can be exceptionally effective on exceedingly organized information, even in high measurements. Until the tree reaches the required depth segmentation takes place again and again forming a sub-sub-ball. During the formation phase of ball tree it required a high amount of space and time but nearest neighbour location becomes easy task after generation of nested hyper-sphere. A ball tree is described by a centroid C and radius r that recursively divides the given data into nodes so that every point in the nodes belong to a hypersphere determined by r and C. With the use of triangle inequality the number of points for a neighbour search is reduced.

Ball Tree Pseudo-code:

1. Procedure BALL TREE (M,y)
2. if |M|<k then stop end if Return leaf containing S
3. pick $K_0 \varepsilon M$ uniformly at random
4. pick $K_1 = \text{argmax}_{K \varepsilon S}\, d(K_0, K)$
5. pick $K_2 = \text{argmax}_{K \varepsilon S}\, d(K_1, K)$
6. for all i=1,...|M|, $z_1 = (X_1, X_2)$ T Xi project data onto $(X_1 - X_2)$
7. K= median(z_1,...,$z_{|M|}$)
8. $S_L = \{x \varepsilon M : z_i < K\}$
9. $S_R = \{x \varepsilon M : z_i >= K\}$
10. Return Tree
    center c = mean (M)
    radius r = $\max_{x \varepsilon S} d(X, c)$)
    childern: BallTree (M_L,y),BallTree(M_R,y)
11. end procedure

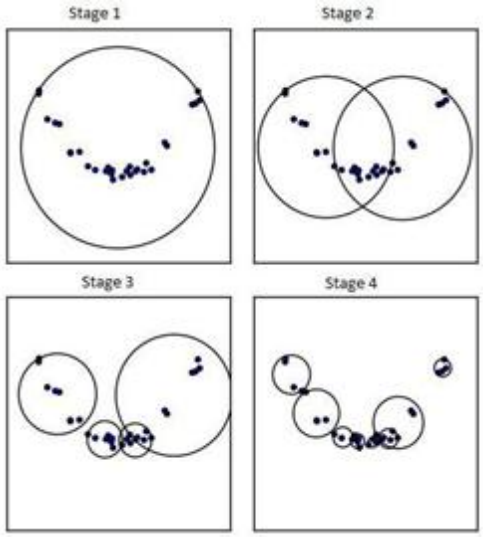

Fig.6 Illustrate a Ball-Tree

C. Rank-Order Clustering Distance

The R.O.C method recommended in [21], is similar to strategy discussed in [22], is approximately a type of aggregate ordered bunching, utilizing closest neighbour positioned separation estimation. General stream for calculation is as follows, first initialize every illustrative as independent bundle, calculate gap (distance) between sets of cluster, integrate those cluster for which the calculated gaps (distance) are underneath an threshold, at that point iteratively recalculate new arrangement of group-to-group separation after that separations are calculation again.The least separation among two illustrative in the bunches is chosen while calculating the separation among two separate bundles.

In Rank-Order clustering the primary distance calculation is given by

$$d(y,z) = \sum_{i=1}^{O_y(z)} O_z(f_y(i))$$

where $f_y(i)$ is the i-th look in the neighbor rundown of y, and $O_z(f_y(i))$ indicates the position of picture fy(i) in z's closest elements, where the closest elements set is produced using some basic separation measure. Equation given underneath is utilized to decide symmetric separation of 2 images, y & z with the utilization of above given unbalanced capacity.

$$D(y,z) = \frac{d(y,z) + d(z,y)}{min\left(O_y(z), O_z(y)\right)}$$

To merge local neighbourhood successfully an new distance calculation is used to normalise the cluster-level. Specifically, in a pair of cluster across any two point the minimum distance is calculated, and is divided by the mean distance, as:

$$D^{N}\left(C_r, C_s \right) = \frac{1}{\phi\left(C_r, C_s \right)} \times d\left(C_r, C_s \right)$$

$$\phi\left(C_r, C_s \right) = \frac{1}{\left| C_r \right| + \left| C_s \right|} \times \sum_{y \in C_r \cup C_s} \frac{1}{K} \sum_{K=1}^{K} d\left(y, f_y(K) \right)$$

In cluster y and z the minimum distance is $DN(C_r,C_s)$ between two point is split using mean of every feature in C_r or C_s to their K closest acquaintances. For this function, we consistently take threshold to be 1 and K is the size of local neighborhood, in a cluster among given two illustrative, the minimum distance is lower than the mean separation of every point in the pair of cluster to their K closest neighbor.

For cluster-normalized distance a neighborhood size is stated and a distance threshold is defined to calculate the rank order distance and the number of cluster formed will be determined by these parameter together. We have use our own values for neighborhood size and distance threshold.

Run-time cost for each and every sample while calculating nearest neighbor list is $O(N^2)$ addition to this clustering required iterative step and cost of per iteration is proportional total number of current cluster squared(initial cluster N which decreases after every iteration),therefore with the increase in dataset size both clustering step and nearest neighborhood calculation are costly.

D. Approximate Rank-Order Clustering Distance

The R.O.C algorithm possess undeniable adaptability issue for each sample in dataset calculation of nearest neighbor list is required, which has an $O(N^2)$ run time cost. Albeit different approximation algorithms also resides to calculate closest acquaintances, they are just skilled to calculate K closest neighbors effectively, instead of thoroughly positioning the database. To calculate a nearest neighbor list for executing randomized KD tree FLANN library is executed [23]. Before calculating nearest neighbor with approximate method Original Rank-Order clustering algorithm required some modification. Instead of summing all the neighbor as in equation below the top most K neighbor are taken. Further instead of applying cluster level normalization as given in original algorithm [21]. We execute locally between pair of sample for calculation approximate rank order clustering which lies only within the top-200 nearest neighbor. Only a short rundown of best K closest acquaintances is assumed which become more remarkable than samples numerical rank. As such instead of rank based we implement a distance calculation method based on summation of presence or absence of shared nearest neighbors, as stated below distance function:

$$D_m(y,z) = \frac{d_m(y,z) + d_m(z,y)}{min\left(O_y(z), O_z(y) \right)}$$

$$d_m(y,z) = \sum_{i=1}^{min\left(O_y(z), k \right)} I_z\left(O_z\left(f_y(i) \right), k \right)$$

here value of indicator function Iz(x,k) is 0 if image of face x is the image of face z top k nearest neighbour else 1. Practically speaking, it is found that above equation prompts good grouping precision.

Adequately, given separation(gap) formula infers that the absence or presence of same acquaintances in the short rundown of its closest acquaintances list for both y & z (for example inside the 200 closest acquaintances) is imperative, whereas the actual estimations for the positions for those acquaintances does not matter the normalisation plan utilized in the original function (just summing up to the rank of the other sample being compared, and dividing by min(Oy(z), Oz(y)) is still powerful, and adds to more precise clustering results even with this change to the original function.

$$D_m(y,z) = \frac{d_m(y,z) + d_m(z,y)}{min\left(O_y(z), O_z(y) \right)}$$

Moreover, to enhance the execution seed for bunching process, we (1) just calculate distance between samples which show up in one another's nearest neighbor list, and

(2) as in original algorithm multiple merge iteration are performed for clustering contrary to this only one round merges are done to form clusters. This implies contrasted with the original calculation which has a runtime of (C^2) for every repetition, just execute single cycle of bundling, & also examine for fusion for every single conceivable match. Thus O(N) is the concluding execution speed for bunching (expecting pre-calculated closest acquaintances)

Bundling finally proceed in given sequence:

1. For each and every facial image in dataset deep features are extracted.
2. For each picture in collection, best K closest acquaintances are calculated.
3. Following equation pairwise distance are calculated between each facial image and the facial image in top-k nearest neighbor list.
4. All pairs of facial image are merges with the distance below a given threshold.

To decide the number of cluster a threshold is selected. In clustering an all-time issue is threshold in a given database. We cannot assure the true number of cluster in particle application, therefore we have just assume several different value of threshold to compute our algorithm and record best result observed in our experiment.

E. Evaluation of Clustering

In surveying grouping execution, as we are using a pre-described significance of "right" bunching (grouping by character), we can evaluate exactness to the extent groups identifying with acknowledged personality names. Outside measures for evaluating bunching quality rely upon character marks; we will use pairwise precision/recall as it very well may be determined effectively. Execution speed is also a basic assessment metric.

To evaluate the clustering results F1-measure is used, which computed with the help of Recall & Precision defined below:

Pairwise Precision Those number of illustrative set inside the group (taking every single reliable pair) which belong to similar individuals (having same features), divided by aggregate number of exact cluster set within the given collection is known as pairwise precision. In Fig. 7, (P1, P2) are from same group, and (P1, Q1) and (P2, Q1) are from different group.

Pairwise Recall Those number of illustrative set inside the group (taking every single reliable pair) that belong from same set divided by aggregate sum of pairs in the collection is known as pairwise recall. In Fig. 7 (P1, P3) and (P2, P3) are placed in the separate group but belong from same set, whereas (P1, P2) are placed in same group and also belong from same group.

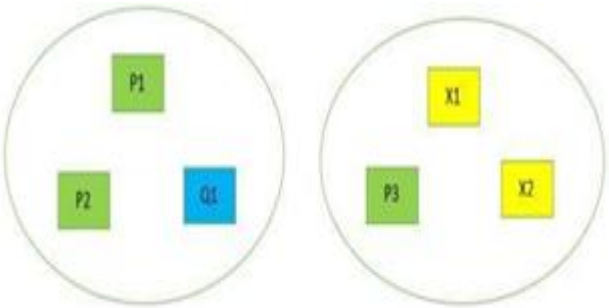

Fig. 7 Illustrating 2 clusters with P1,P2,P3 objects having same identity and Q1 object have different identity and X1,X2 are unlabeled objects.

The pairwise precision and recall records two types error, high precision is achieved when all samples are placed as individual clusters, but it will achieve low recall, whereas recall is achieved when all samples are placed in same clusters, but it will achieve low precision. An F measure is described with the help of pairwise precision and recall, defines as F=formula or plotted on precision recall graph. To use with partially labelled dataset we have extend the pairwise recall and precision in large scale clustering complication by eliminating the unlabelled data as much as possible from evaluation.

$$F = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)}$$

To use with partially labelled dataset we have extend the pairwise recall and precision in large scale clustering complication by eliminating the unlabeled data as much as possible from evaluation.

In our calculation instead of including every possible pair we have use improved modified recall by just excluding unmarked pair whether present or not. For precision, we have included unlabeled-labelled mismatch pairs (e.g.(P3,X1) and (P3,X2) given in figure) and excluded unlabeled - unlabeled pairs (e.g.(X1,X2)) from total pairs. Utilize (P3, X1) & (P3, X2) to compute the revised precision. Practically speaking, when unmarked collection of data and marked collection of data are combine, some unmarked collection of data might be existing in marked collection of data. Hence, one must examine all unmarked-marked pair to achieve better efficiency.

F. LFW Dataset

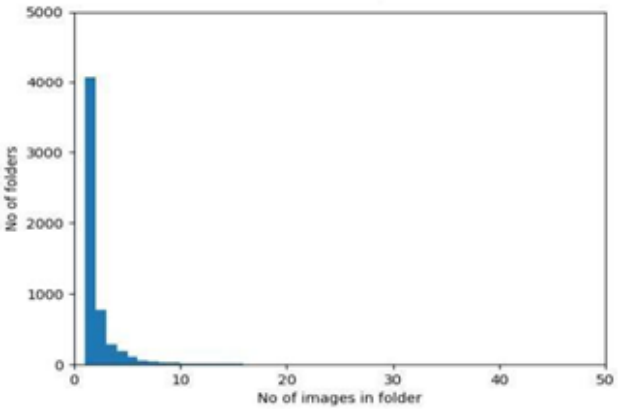

Fig. 8 Number of Image and Number of Folder in LFW Dataset.

A LFW is a database of facial photos intended for learning the issue of unconstrained face recognition. The LFW contains in excess of 13,233 facial image of 5749 individual gathered through internet. Pictures in LFW as remarkable difference in light brightness, expression & pose. Each face has been marked with the name of the individual picture. In given LFW dataset 1680 of the individual have at least two or more different facial picture recognize by viola Jones locater [18, 19].

IV. RESULT

In simulation results, we have used two method of knn one is kd-tree and the other one is ball tree then we have applied our clustering algorithm and we observed that by using kd tree total no. of clusters obtain is nearly 7000 and by using ball tree no. of cluster obtain is nearly 1500 in order to see the results from of each algorithm we have generate a bar graph as mentioned below. LFW database have been used in our approach, where pairwise precision, recall and f1 measure is used to evaluate cluster and find the accuracy. There are various feature classification techniques to detect Human Face. Proposed clustering algorithm performs better than the given algorithm final comparison will be performed to assess the outcomes result.

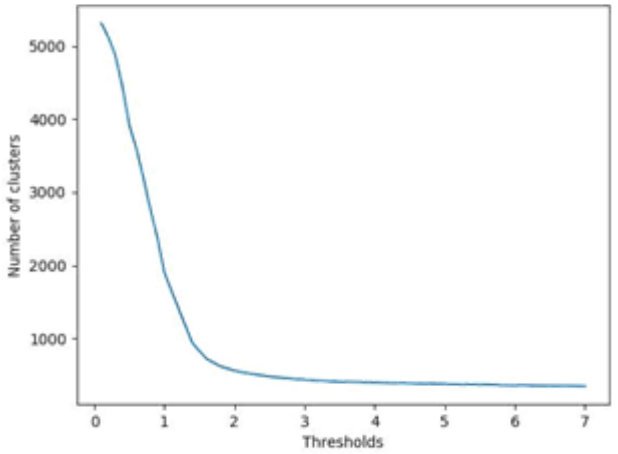

Fig.9 Curve Between Number of Cluster and Threshold in Ball Tree

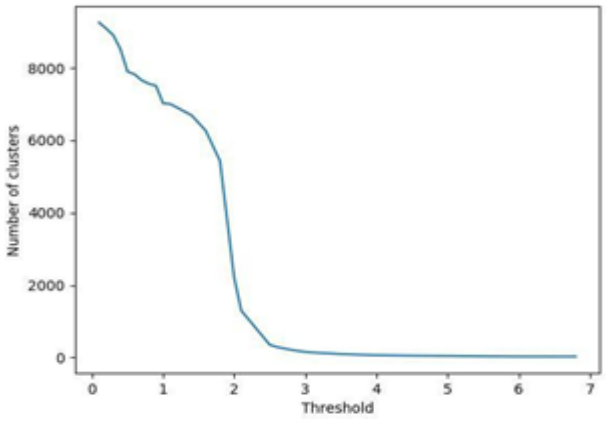

Fig.10 Curve Between Threshold and Number of Cluster in Kd Tree

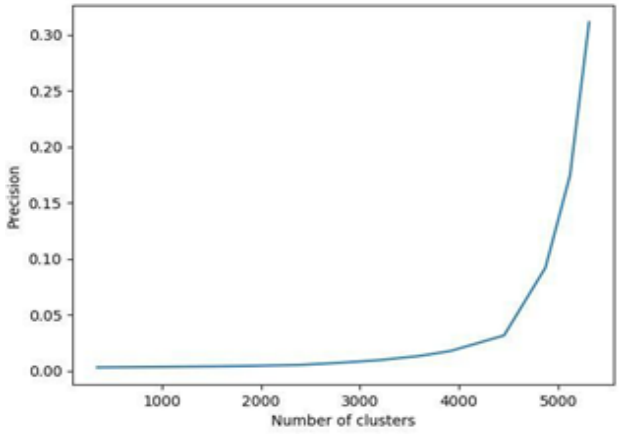

Fig 11 Curve Between Number of Cluster and Precision in Ball Tree

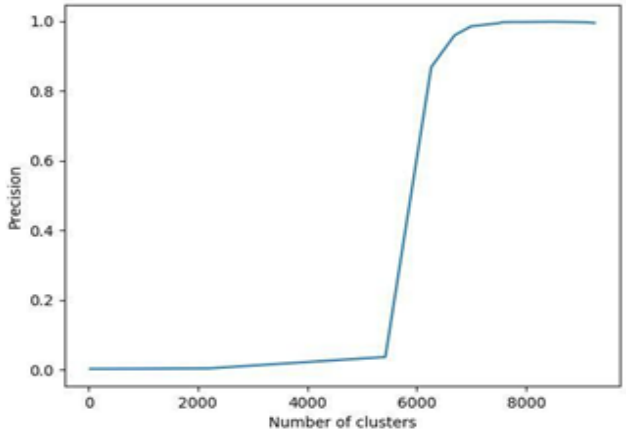

Fig 12 Curve Between Precession and Number of Cluster in Kd Tree

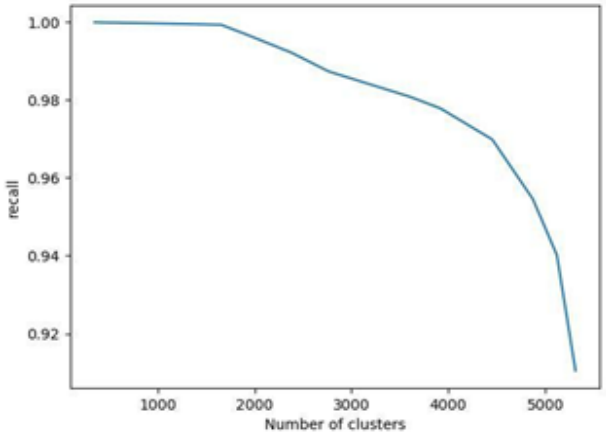

Fig 13 Curve Between Number of Cluster and Recall in Ball Tree

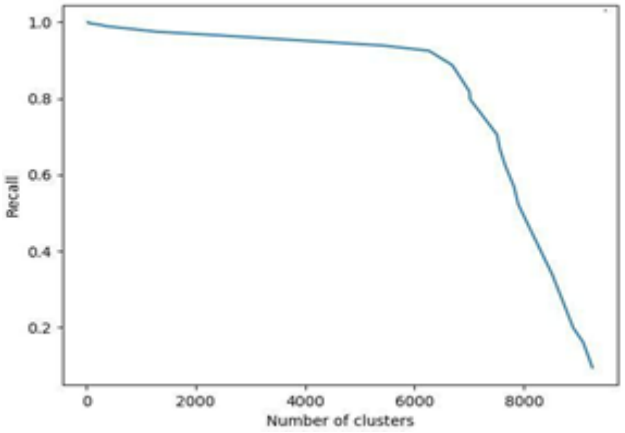

Fig 14 Curve between Recall Score and number of cluster in Kd Tree

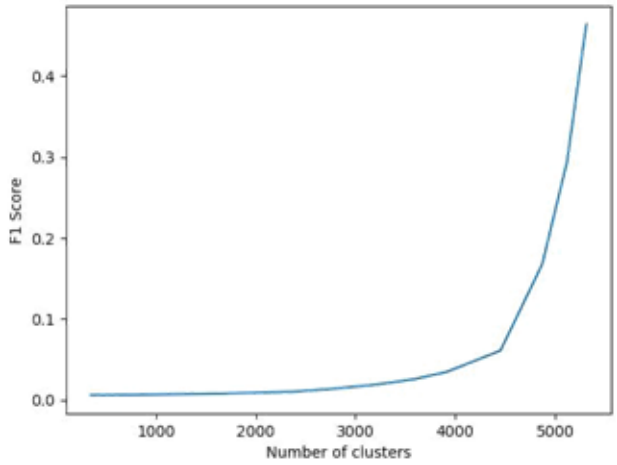

Fig 15 Curve between f1 Score and number of cluster in Ball Tree

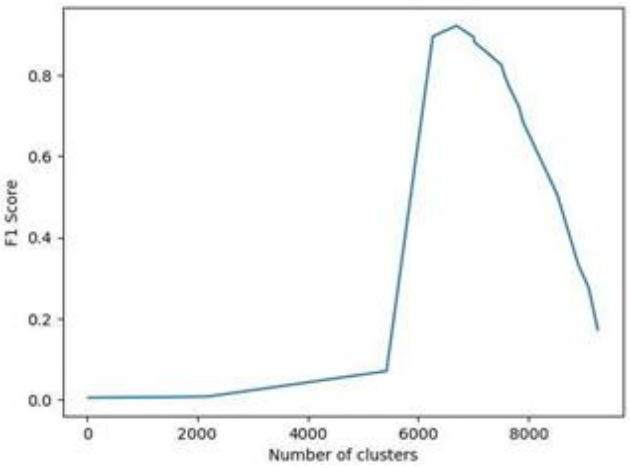

Fig 16 Curve between f1 Score and number of cluster in Kd Tree

In figure numbers 9 and 10 we can see that, using threshold vs number Of clusters graph for both cases the curve for ball tree is smooth i.e. we can see the direct dependency between threshold and no of clusters whereas for KD tree the curve is not smooth indicating that it also depends on some other feature and it also indicate that on increasing dataset there would be more variation in graph. Also no. of clusters for KD tree is much greater than that in ball tree thus it concluded that for LFW dataset KD tree

Is creating many clusters with only one image which is not an indication for good algorithm.

In figure numbers 11 and 12 we can see that, for Precision vs number Of clusters for KD tree since many of clusters have only a single image thus precision suddenly increases to 1 whereas since no. of clusters in ball tree is lesser so its precision is not going that high.

In figure numbers 13 and 14 we can see that, for recall vs number of clusters because of so high number of clusters its recall is almost 0 which is very poor whereas for ball tree we can see that because no. of clusters is lesser so recall is very good which also signifies that most of the similar faces are clustered together.

Also in figure numbers 15 and 16 we can see that, for F1 score which is dependent on precision and recall for kd tree drastically increases then decreases but for ball tree it does not went so high but is increasing continuously.

V. CONCLUSION AND FUTURE SCOPE

With the approach of internet, everyday billion and trillion of images is being transferred online has become very difficult to ensure its identity. While finding culprits or finding a particular person from a very large dataset is a very difficult task which involve a lot of searching in a very limited time. To overcome this difficulty we try to improve the existing clustering with the use of an appropriate nearest neighbor called Kd-Tree.

Our aim always would be to modify this algorithm to increase the accuracy of clusters by improving given nearest neighbor methods. Given work can be enhanced in various ways (1) utilizing bigger preparing sets (2) By upgrading deep model architecture. (3) Automatically selecting of the number of clusters to improve clustering accuracy

REFERENCES

1. Charles Otto, Dayong Wang, and Anil K. Jain. Clustering millions of faces by identity. arXiv preprint arXiv:1604.00989 , 2016.
2. Dayong Wang, Charles Otto, and Anil K Jain. Face search at scale. IEEE Trans. on PAMI , 2016.
3. Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face description with local binary patterns: Application to face recognition. IEEE Trans. on PAMI , 28(12), 2006.
4. David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In Proc. of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms , 2007
5. Ting Liu, Charles Rosenberg, and Henry A Rowley. Clustering billions of images with large scale nearest neighbor search. In Proc. WACV, 2007.
6. Duane M. Blackburn, Mike Bone, and Jonathon P. Phillips. Face recognition vendor test 2000: evaluation report. Technical report, DTIC Document http://www.dtic.mil/ dtic/tr/fulltext/u2/a415962.pdf, 2001.
7. Jie Chen, Hawren Fang, and Yousef Saad. Fast approximate k-NN graph construction for high dimensional data via recursive lanczos bisection. The Journal of Machine Learning Research , 10, 2009.
8. Charles Otto, Brendan Klare, and Anil Jain. An efficient approach for clustering face images. In Proc. ICB , 2015.
9. Chanop Silpa-Anan and Richard Hartley. Optimised kd-trees for fast image descriptor matching. In Proc. CVPR , 2008.
10. K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing humanlevel performance on Imagenet classification. arXiv preprint arXiv:1502.01852 , 2015.
11. Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In Proc. International Conference on Machine Learning , 2010.
12. J. Wan, D. Wang, S. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li. Deep learning for contentbased image retrieval: A comprehensive study. ACM Multimedia , 2014.
13. Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. arXiv preprint arXiv:1411.7923 , 2014.1. Charles Otto, Dayong Wang, and Anil K. Jain. Clustering millions of faces by identity. arXiv preprint arXiv:1604.00989 , 2016.
14. Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for largescale image recognition. arXiv preprint arXiv:1409.1556 , 2014.
15. Vahdat Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In Proc. CVPR , 2014.
16. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems , 2012.
17. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1), 2014.
18. Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.
19. Paul Viola and Michael J Jones. Robust real-time face detection. International Journal of Computer Vision , 57(2), 2004.
20. Anil K Jain. Data clustering: 50 years beyond k-means. Pattern Recognition Letters , 31(8), 2010.
21. C. Zhu, F. Wen, and J. Sun. A rank-order distance based clustering algorithm for face tagging. In Proc. CVPR , 2011.
22. C. Zhu, F. Wen, and J. Sun. A rank-order distance based clustering algorithm for face tagging. In Proc. CVPR , 2011.
23. Marius Muja and David G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. IEEE Trans. on PAMI , 36, 2014.
24. P. Grother and M. Ngan. Face recognition vendor test (FRVT): Performance of face identification algorithms. NIST Interagency Report 8009 , 2014.
25. Cheng Cheng, Junliang Xing, Youji Feng, Deling Li, and Xiang-Dong Zhou. Bootstrapping joint bayesian model for robust face verification. In Proc. ICB, 2016.