Face Identification using Clustering

EEE 511: ANC - Clustering Assignment

Viraj Savaliya
ASU ID: [redacted]
Email: Viraj.Savaliya@asu.edu
Arizona State University, Tempe, AZ

***Abstract*—In this assignment, the problem of having a large collection of unlabelled images and clustering them into unknown number of faces identified is addressed. This problem is tackled daily by social media sites like Facebook and also by Google photos, in forensic investigations, law enforcement, etc. The algorithm used to solve this problem is Approximate Rank-Order clustering algorithm [1] which is also the state-of-the-art. The algorithm performs far better than popular ones with F-measure of 0.87 on LFW benchmark (13k faces, 6508 clusters) and does this in just 8 seconds.**

***Index Terms*—Face clustering, rank-order, scalability**

I. Application

On social media platforms like Facebook where millions of images are uploaded daily and also Google photos where people upload their pictures, there are unknown number of faces of people in this dataset. To organise this data, one very common way to do is grouping them by the people identified in these photos. This number of people is unknown and can range from hundreds to millions which leads to difficulties of clustering the data in real-time and with good accuracy. The example of face clustering can be shown in Fig. 1, where there are 2 clusters formed from the data on the left.

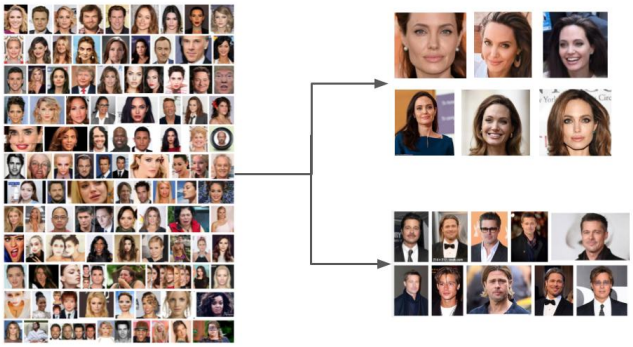

Fig. 1. **Face Clustering example**

The key challenges in face clustering are often the number of discrete clusters (identities) to be formed are unknown and can usually be a large number. Also, the run-time of the algorithm is directly related to this number. The other issue is that the number of images per identity in the dataset can be unbalanced as some people might appear in most of the images while some not that often. The clusters can also be classified wrongly if the face recognition is incorrect due to poor illumination, noisy image, pose, occlusion, etc. Just using strong face recognition model or very accurate clustering algorithm doesn't alone solve the problem as both should be used in such a way that they are compatible, leverage the benefits of the other and improve the accuracy and scalability.

II. Approximate Rank-Order Clustering

The overview of the steps followed in Rank-Order clustering algorithm is that first the samples from the dataset are separated in different clusters, then distance between the clusters is computed and then the distances of samples below a certain threshold are merged together. The Fig. 2 shows a high level view of the steps followed.

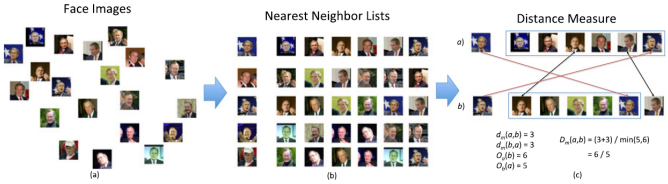

Fig. 2. **Approximate Rank-Order Clustering Algorithm [1]**

Inorder to address the scalability issue, some approximation is required for computing the nearest neighbor for every sample in the dataset. Randomized k-d tree algorithm of FLANN library is used to compute a set of nearest neighbors. The presence or absence of a particular sample in the near neighbors is considered rather than a numerical rank and summation of all these samples is used as a distance for a sample. The equation can be given by,

$$d_m(a,b) = \sum_{i=1}^{min(O_a(b),k)} I_b(O_b(f_b(i)),k) \tag{1}$$

In the eqn. 1, $I_b(x,k)$ is an Indicator function stating the presence or absence of sample. Then this distance measure is normalized in order to obtain accurate clustering results. This normalized distance or symmetric distance between two faces in sample a and b is given by,

$$D_m(a,b) = \frac{d_m(a,b) + d_m(b,a)}{min(O_a(b), O_b(a))} \tag{2}$$

Further, the run-time and the accuracy for clustering is improved in this algorithm by only computing the distances between pairs of samples which appear in the set of nearest neighbor for both. Also, only a single round of merges is performed rather than iterating it for multiple times to reduce the run-time. Therefore, the runtime is now just one iteration rather than C^2 to merge by iterating over all pair of clusters. Additionaly, the run-time is further reduced as the distance

check is only for top-k nearest neighbors (k=200 is selected by authors). Hence, the runtime is of complexity $O(n)$.

The threshold value for determining the number of Clusters C for a dataset is evaluated by the authors by experimenting at certain effective values chosen and then reporting the best result from the experiment. This is a difficult task as the number of clusters cannot be known before-hand and it is left as a future task to be addressed.

III. Implementation

The two key aspects of face clustering are Face recognition and then clustering them in groups of unique identities. For face recognition here first feature extraction is done using DLIB implementation [2] and then the normalized image is sent to the state-of-the-art deep CNN model [3] for Face recognition having 96.96% accuracy on popular LFW and IJBA benchmarks and then the approximate Rank-Order Clustering algorithm is used to form clusters. The main steps can be summarised as follows:

1) Extracting deep features for every identity.
2) Per identity, compute set of top-k nearest neighbors.
3) Computing distance for each pair from the set.
4) Merging all the pairs below threshold into a cluster.

The implementation by original authors was using CASIA-webface dataset for training and LFW dataset for testing. They also used Intel Xeon CPUs of 20 cores clocked at 2.5 GHz for the experiments. For re-implementing it locally the GitHub repository [4] can be used which also has the steps clearly stated. The packages required for implementation are FLANN library, Flask, profilehooks, etc listed under setup.py code. Also CPU with atleast 8 cores and using python's multiprocessing module is recommended to obtain good speed using the model. For top-k nearest neighbors computation, both the authors have used 200 as the value for k which if altered can modify the accuracy and run-time of the model as accordingly the number of computations for distance for every pair will change.

IV. Results

Clustering Results on the Complete LFW Dataset

Clustering Algorithm	# Clusters	F-measure	Run-Time
k-Means	100	0.36	00:00:16
k-Means	6,508	0.07	04:58:49
Spectral (Euclidean)	200	0.20	00:11:18
Spectral (Approx. ROD)	200	0.43	00:14:04
Hierarchical	1,000	0.005	00:00:25
Rank-Order	7,059	0.65	00:00:33
Approx. Neighborhood Rank-order	6,440	0.83	00:00:09
Approx. Rank-Order (proposed)	6,508	0.87	00:00:08

Fig. 3. **Comparison of Clustering Algorithms [1]**

The evaluations for the algorithm were done by the authors on the LFW dataset which has large scale datasets and unequal number for images for each subject. The comparison of various clustering algorithms on LFW dataset is shown in Fig. 3. Approximate Rank-Order Obtains the best accuracy with minimal run-time and most number of clusters. Evaluation for large-scale datasets for the algorithm is shown in Fig. 4. By adding huge number of unlabelled data to the LFW dataset, the F1-measure reduces and the computation time increases even if we increase the number of cores. This can be stated as a current limitation of the algorithm.

Large-Scale Clustering Results Using the Proposed Approximate Rank-Order Clustering, with Randomized K-D Tree Nearest Neighbor Approximation

Dataset	F-measure	# Clusters	# Cores	Run-Time
LFW	0.87	1,463	1	00:00:19
LFW + 1M	0.79	94,740	1	01:03:25
LFW + 5M	0.67	445,880	5	06:28:42
LFW + 10M	0.56	933,278	10	12:11:33
LFW + 30M	0.42	2,800,202	30	30:44:58
LFW + 123M	0.27	10,619,853	123	289:04:53
LFW + 121M	0.32	10,281,612	123	245:12:54

Fig. 4. **Large-Scale Clustering Results [1]**

The paper [1] shows more results for the algorithm. Some of the key results are that F1-measure for around 7000 clusters was the highest when the distance threshold was varied during experimentation. Approximation method of randomized k-d tree performs the best when compared to other techniques. The authors also tested the algorithm for video frame clustering on the YTF dataset and the algorithm performed well again with 0.71 F1-measure. Also, per-cluster quality measures were evaluated for determining the effectiveness of internal measures and external measures with precision.

V. Conclusion

The effect of high volume of data of images consisting for unknown number of identities is addressed in the assignment. The key challenges faced for solving the problem are discussed and an appropriate state of the art solution for face clustering is explained. The implementation details with key steps involved are stated and discussed with respect to the decisions made by the original authors. Also, a verified link for the model to re-implement it locally is provided. The key results for the algorithm are that it performs the best compared to other existing models and leverages the benefits of the design decisions. The main task to be further addressed in future are to incorporate even better nearest neighbor method, append automatic selection of threshold and improvise per-cluster quality to improve the clustering accuracy.

References

[1] C. Otto, D. Wang, and A. K. Jain, "Clustering millions of faces by identity," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 2, pp. 289–303, 2018.

[2] B. Shi, X. Bai, W. Liu, and J. Wang, "Face alignment with deep regression," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 1, pp. 183–194, 2018.

[3] X. Wu, R. He, Z. Sun, and T. Tan, "A light cnn for deep face representation with noisy labels."

[4] V. Suresh, "varun-suresh/clustering." [Online]. Available: https://github.com/varun-suresh/Clustering