Clustering Millions of Faces by Identity

by

Güray Baydur, İpek Erdoğan

PATTERN RECOGNITION TERM PROJECT REPORT

17.02.2021

TABLE OF CONTENTS

1. Introduction

Clustering is a popular area where numerous studies have been conducted to derive insights from data, especially in machine learning, statistics and pattern recognition. Under this subject, face image clustering has become new focus to provide new functionalities such as searching video based on a scene description [1] or considering important areas, finding perpetrator in forensic investigations .

However, face image clustering introduces some challenges due to large number of clusters to construct and images to investigate [2]. Trivially, large dataset creates a scalability problem during computation and causes run time complexity which may require plenty of hardware requirements. Moreover, sometimes the identities that are investigated may contain rare number of samples in contrast to some of them having large number of samples. This results in low performance of some clustering algorithms such as k-means which tends to create similar sized clusters.

Considering mentioned issues, the paper that is chosen for this project proposes approximate rank order clustering. This algorithm tries to overcome the run time complexity issue by calculating rank order distance according to top-k nearest neighbors list. By doing so they reduce the computational complexity to order of n. They also point out that a good representation of face images is crucial for clustering algorithm to give good results. Therefore they used multiple convolutional neural networks (CNNs) with cross entropy loss to obtain a state-of-art face representation.

The contributions made to this paper includes converting cross entropy loss to triplet loss, where pairwise Euclidean distance is calculated between face images. Since the loss function is calculated considering anchor (can be considered as candidate image), positive (sample from true class of anchor) and negative (sample from any other class rather than ground truth) images, we did not consider classes with single image. Finally, our algorithm converged for small size of trained data due to hardware limits.

2. Background Information

2.1. Face Clustering

Several algorithms are used for face clustering problem. Threshold clustering [3] considers embedding of faces and tries to find the minimum distance between the face to be clustered and already clustered faces. When the distance is below a threshold between the new face and the clustered face, the new face is added to corresponding already created cluster. The crucial part is the choice of the threshold because high threshold value may result in false positives and low threshold value may result in false negatives.

Another approach is mean shift clustering [4]. where Euclidean space is used for face embedding representation. Kernel density estimation is chosen for underlying distribution. The aim here is moving each point to the direction of change by using a kernel. A mean shift vector is computed for each sample that points to the region with maximum increase. The benefit of this approach is that it is a non-parametric algorithm. There is no need to define specific number of clusters to create.

DBSCAN algorithm was proposed in 1996 [5] and it uses Euclidean distance to group points closer to each other. It has two parameters which are minimum points to form a dense region and minimum distance between two points. By setting the dense region to one, it prevents a face with no close neighbors from constructing a unique cluster. It is also a non parametric algorithm.

Conditional Pairwise Clustering [6] is a new method which uses pairwise similarities between faces and tries to group the faces according to identity. This method considers adjacencies between face pairs as variables to predict and tries to maximize joint posterior probability given pairwise similarities. They model the problem as Conditional Random Field which is a undirected probabilistic graphical model to find maximized posterior probability on tree-like graphical models. They also make use of Loopy Belief Propagation.

2.2. Face Representation

Early attempts to represent face images include appearance models like eigenfaces where eigenvectors of set of faces [7]. Moreover, sparse representation is also preferred which basically chooses among subset base vectors for the best expressive input signal. [8] However, representations with Deep Neural Networks are more reliable because of representational learning ability. DeepFace is one of the methods for DNN approach and it is based on minimizing the classification error and the last hidden layer output is taken as the face representation. [9] DeepId is an extension of DeepFace where Sun et al make use of joint Bayesian Framework for multiple CNNs. [10] In another work, Schroff et al [3] drops the classification layer. They utilized triplet loss to learn an embedding space. Euclidean distance is used for separation of feature vectors from different identities.

2.3. Approximate Nearest Neighbor Methods

Finding nearest neighbor sets for n samples in a dataset is one of the problems in well-known clustering approaches since its time complexity. The runtime of this operation is $O(n^2)$ which can be a problem especially with large numbers. There are two different approaches in literature to solve this problem: n approximate nearest neighbor search and k-NN graph construction.

For k-NN graph construction, Chen et al. [11] proposed an approach which divides the feature space by using Recursive Lanczos algorithm [12]. The time complexity of this method is changes depends on the common region in the divide step. It changes between O(n) and $O(n^2)$. There is also a parallelized version of this approach proposed by Otto et al. [13] which recursively divides process into branches and handles each parts in a different thread.

Instead of considering this problem as a k-NN graph construction, we can think "building nearest neighbor lists" problem as a discrete nearest neighbor search problem. Approximate nearest neighbor search is one type of the solutions and most popular approaches for this solution are partitioning tree-based approaches. These are usually k-d tree algorithm which they create an index which divides the feature space. Selecting sets of features creates this division. By creating multiple k-d trees and search across them in parallel, searching performance increases significantly [14].

2.4. Clustering Evaluation

In terms of clustering evaluation, the authors of this paper used accuracy, pairwise precision, pairwise recall and run time. Pairwise Precision is fraction of sample pairs which belong to the same cluster and class over the total same-clustered sample pairs. Pairwise Recall is fraction of sample pairs which belong to the same class and cluster over the total sample pairs which belong to the same class.

These errors have a trade-off relation in between. If we think extremely, when we cluster all samples in individual clusters, we will have high precision yet low recall. Clustering all the data in individual clusters doesn't make sense about generalizing aim in clustering. When we cluster all samples in the same cluster, we will have high recall yet low precision. That also doesn't make sense in terms of clustering the data correctly. In this point of view, a metric to measure these errors both is used which is called F-score.

$$F = \frac{2 * Precision * Recall}{Precision + Recall}$$

To deal with the partially labeled data, authors just omitted the unlabeled data from evaluation.

3. Proposed Method

The algorithm proposed in this paper consists of two parts: extracting face representation and clustering. First part stars with detecting the facial landmarks of the input image. To estimate facial landmarks, Sullivan and Kazemi's [15] method has been used which is based on ensemble of regression trees. Image normalization has been performed considering the facial landmarks. The output of this process is a 100x100 image.

This output is passed through a 10 layered convolutional neural network to extract the representations, which is inspired by the architecture proposed by Zisserman and Simonyan [16]. The architecture consists of 4 repeating convolutional layer and max-pooling layer pairs and two final convolutional layers. An average pooling layer follows these last convolutional layers that is used as feature representation. During training phase, this output feeds a linear layer and output of this linear layer is input for a softmax loss. To sum up, the feature extractor which is a convolutional neural network, has been trained with a classification approach.

There are different clustering algorithms with different approaches. Considered the comparison authors did in their previous work [13], they decided to implement an approximate version of the rank-order clustering algorithm which is proposed by Wen et al. [17]. Since Rank-Order clustering method computes nearest neighbot lists for every sample individually, it has time complexity of $\mathcal{O}(n^2)$. There are some approximation approaches that compute top N nearest neighbors rather than ranking all of the samples.

This paper inspired by this approach, too. Instead of considering all the neighbot samples in the summation equation, authors suggest to sum up top N nearest neighbors. Furthermore, authors suggests to consider the presence or absence of a sample in the top N list is more important than this sample's numerical place. So as a summation function, this paper proposes a summation of presence and absence of commmon nearest neighbors. This approach eliminates the ranking approach.

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

To define a distance between a and b, following function is being used. This distance function is the same in the original Rank Order Clustering algorithm.

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

This approach computes the distances only if samples have a common neighbor. Also with one time of merging of individual faces into clusters, final run time of clustering part decreases to $\mathcal{O}(n)$ (with pre-computed nearest neighbors).

4. Experiments and Results

4.1. Dataset

In the original paper [2], CASIA-WebFace [18] dataset has been used for training the face representation extractor CNN. Youtube Faces [19] dataset and Labeled Faces in the Wild(LFW) [20] dataset have been used for evaluation of the clustering part. In our experiments, we used LFW. LFW is a public dataset which is commonly used for face recognition and verification researches. It includes 13233 images from 5749 people.

4.2. Results and Contribution

In our project, we mostly focused on the face representation part. We could run the experiments on LFW dataset in kind of a different approach. We followed a more recent paper called A light CNN for deep face representation with noisy labels [21]. It is really a light CNN (comparing to FaceNet it has 85 less amount of parameters. Due to our source limitation in terms of hardware, it made sense for us. Also its results seemed good and there were some open source implementations we could follow.

Differences between CNNs	
A light CNN for deep face representation	Clustering Millions of Faces by Identity
Grayscaled, aligned to 144,144 by 5 landmarks Randomly cropped to 128,128 as inputs Trained with CASIA-WebFace, MS-Celeb-1M 29 layered CNN	Aligned to 110,110 by 68 landmarks Centered cropped to 100,100 as inputs Trained with CASIA-WebFace 10 layered CNN

You can see the base differences between the model that authors referred in the Clustering Millions of Faces by Identity paper and the one we used. They both use cross entropy a loss function.

The quantitative results of our experiments are down below. F1 score here is a combination of Pairwise Precision and Recall. Results here are average of 5 experiments. The runtime here is the clustering run time. the Our runtime is higher which make sense when you consider the hardware difference between ours and theirs.

Results on LfD Dataset		
Model	F1 Score	Run Time
Paper Result	0.87%	00:00:19
Our Result with LightCNN29	0.82%	00:01:28
Our Result with LightCNN29[v2]	0.88%	00:01:20

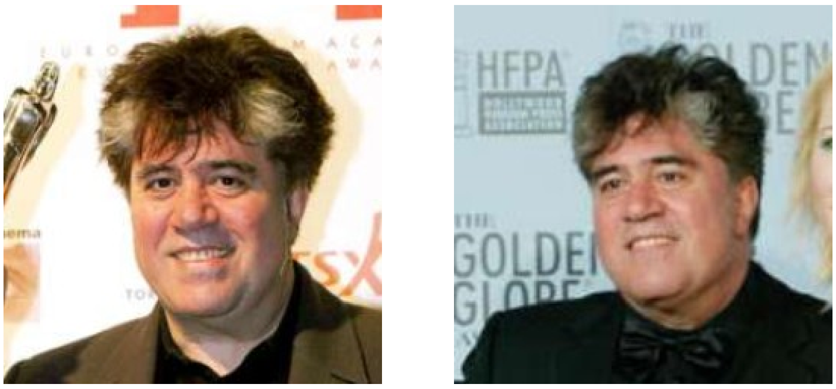

Figure 4.1. Example of Correct Clustering

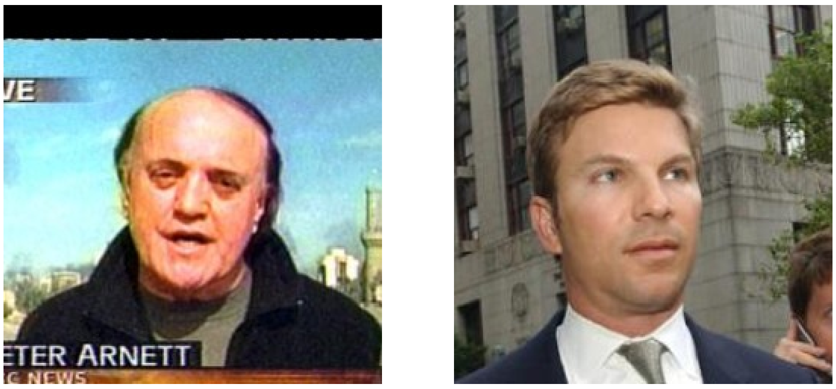

Figure 4.2. Example of Wrong Clustering

These face representations will be used for clustering and in clustering part, distance based ranking metrics will be used. So considering your comments at out project proposal presentation, we said how would it be if we use a distance based metric for training in face representation extraction part. Then we selected triplet loss which has been proposed in FaceNet [22] paper. Triplet loss takes 3 inputs: Anchor, positive and negative. The Anchor will be the reference input, the positive will be an input that has the same class as the anchor, while negative must be input with a different class from the anchor.

$$TripletLoss = \sum_{i=1}^{N} [\|f_i^a - f_i^p\|_2^2 - \|f_i^a - f_i^n\|_2^2 + \alpha]$$

The idea behind triplet loss is calculate the embeddings of these three by passing them through the same CNN with same weights. With triplet loss, we maximize the distance between the anchor and the negative and minimize the distance between the anchor and the positive. That's what makes triplet loss more powerful. Triplet Loss considers both negative and positive samples where contrastive loss considers positive and negative samples seperately.

We accomplished to train a CNN which has 5 convolution layers followed by max-pooling layers with triplet loss. Yet we couldn't pass through all of the LFW dataset through this model. The reason behind this may lie under high time complexity of Triplet loss, which is $\mathcal{O}(n^3)$ [23]. We were able to pass very little amount of data. As a consequence, we could not connect the outputs of Triplet Loss model to the clustering part. However, we had a chance to investigate different models and loss functions. To exemplify, we first tried to run triplet loss on all of the LFW dataset which also includes one sampled classes. We considered how it would be if we choose the same image as a positive example for itself and tried it. Our attempt resulted in models including weights with NaN values.

We strongly believe in that's because when we select the same image as a positive image for itself, positive distance becomes zero and margin cannot compansate the negative effect of the negative distance so our loss value easily becomes zero [24]. Also we tried different regularization functions for our model weights and different margins for triplet loss, but these did not make any difference. When we checked the FaceNet experiments, we saw that they select the paired, multi-exampled classes of LFW for training.

Even if we could run the all dataset with triplet loss, according to our researches, we needed to make a solid normalization on images. Otherwise, the negative difference between images may be so high that it would prevent our loss function to be converged [24]. For now, we are enforcing a very basic normalization. It is more likely that it would not be enough to successfully train triplet loss model with a huge dataset like LFW.

5. Conclusion and Future Work

In this project, we demonstrated it is possible to obtain a CNN constructed with a different loss function and can be fed into approximate rank order clustering algorithm even though it is not implemented in this paper. What is promising is that better representations of faces by CNNs helps clustering algorithms to work better [2].

The future work may include implementing a different loss function such as Git loss [25], which utilizes softmax and center loss functions. Additionally, we could not use whole LFW dataset also due to hardware constraints therefore the triplet loss can be used with more samples from LFW dataset with sufficient hardware.

The clustering algorithm can also be improved by using Conditional Pairwise Clustering [6] which takes benefit from Conditional Random Field model and Loopy Belief Propagation to find an the closest solution for maximizing posteriors of adjacency matrix.

6. Acknowledgment

During the experiments, we trained two different models which you can reach from here:
LightCNN + Approximate Clustering Algorithm:
https://github.com/erdoganip/lightcnn-and-clustering
Triplet Loss:
https://github.com/erdoganip/tripletloss

As we referred in Readme files of the Github repositories, we modified these base codes while doing the experiments:
https://github.com/varun-suresh/Clustering
https://github.com/AlfredXiangWu/LightCNN
https://github.com/sanku-lib/image_triplet_loss

REFERENCES

1. Tapaswi, M., M. Law and S. Fidler, “Video Face Clustering With Unknown Number of Clusters”, , 08 2019.

2. Otto, C., D. Wang and A. K. Jain, “Clustering Millions of Faces by Identity”, , 2016.

3. Schroff, F., D. Kalenichenko and J. Philbin, “FaceNet: A unified embedding for face recognition and clustering”, pp. 815–823, 06 2015.

4. Comaniciu, D. and M. Peter, “Mean shift: A robust approach toward feature space analysis”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, pp. 281–288, 01 2002.

5. Ram, A., J. Sunita, A. Jalal and K. Manoj, “A Density Based Algorithm for Discovering Density Varied Clusters in Large Spatial Databases”, *International Journal of Computer Applications*, Vol. 3, 06 2010.

6. Shi, Y., C. Otto and A. Jain, “Face Clustering: Representation and Pairwise Constraints”, *IEEE Transactions on Information Forensics and Security*, Vol. PP, 06 2017.

7. Turk, M. A. and A. P. Pentland, “Face recognition using eigenfaces”, *Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 586–591, 1991.

8. Wright, J., A. Y. Yang, A. Ganesh, S. S. Sastry and Y. Ma, “Robust Face Recognition via Sparse Representation”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 31, No. 2, pp. 210–227, 2009.

9. Taigman, Y., M. Yang, M. Ranzato and L. Wolf, “DeepFace: Closing the Gap to

Human-Level Performance in Face Verification", *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1701–1708, 2014.

10. Chen, D., X. Cao, L. Wang, F. Wen and J. Sun, "Bayesian Face Revisited: A Joint Formulation", pp. 566–579, 10 2012.

11. Chen, J., H.-r. Fang and Y. Saad, "Fast Approximate k NN Graph Construction for High Dimensional Data via Recursive Lanczos Bisection", *The Journal of Machine Learning Research*, Vol. 10, pp. 1989–2012, 09 2009.

12. Lanczos, C., "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators", *Journal of research of the National Bureau of Standards*, Vol. 45, pp. 255–282, 1950.

13. Otto, C., B. Klare and A. Jain, "An Efficient Approach for Clustering Face Images", , 05 2015.

14. Silpa-Anan, C. and R. Hartley, "Optimised KD-trees for fast image descriptor matching", *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

15. Kazemi, V. and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees", *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1867–1874, 2014.

16. Simonyan, K. and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", *arXiv 1409.1556*, 09 2014.

17. Zhu, C., F. Wen and J. Sun, "A rank-order distance based clustering algorithm for face tagging", *CVPR 2011*, pp. 481–488, 2011.

18. Yi, D., Z. Lei, S. Liao and S. Z. Li, "Learning Face Representation from Scratch", , 2014.

19. Wolf, L., T. Hassner and I. Maoz, “Face recognition in unconstrained videos with matched background similarity”, *CVPR 2011*, pp. 529–534, 2011.

20. Huang, G. B., M. Ramesh, T. Berg and E. Learned-Miller, *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*, Tech. Rep. 07-49, University of Massachusetts, Amherst, October 2007.

21. Wu, X., R. He, Z. Sun and T. Tan, “A Light CNN for Deep Face Representation with Noisy Labels”, , 2015.

22. Schroff, F., D. Kalenichenko and J. Philbin, “FaceNet: A unified embedding for face recognition and clustering”, *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2015, `http://dx.doi.org/10.1109/CVPR.2015.7298682`.

23. Do, T.-T., T. Tran, I. Reid, V. Kumar, T. Hoang and G. Carneiro, “A Theoretically Sound Upper Bound on the Triplet Loss for Improving the Efficiency of Deep Distance Metric Learning”, , 2019.

24. Yu, B., T. Liu, M. Gong, C. Ding and D. Tao, “Correcting the Triplet Selection Bias for Triplet Loss”, *ECCV (6)*, pp. 71–86, 2018, `https://doi.org/10.1007/978-3-030-01231-1`$_5$.

25. Calefati, A., M. K. Janjua, S. Nawaz and I. Gallo, “Git Loss for Deep Face Recognition”, , 2018.