Image Clustering and Retrieval in Large Photo Collection

Alessandro Artoni, Stefano Bertolani, Andrea Valentini
University of Modena & Reggio Emilia
{273791, 275735, 272554}@studenti.unimore.it

1. Abstract

Nowadays, all mobile systems provide functionalities for image and video clustering based on the identities that appear in the media gallery. In PC operating systems, like Windows, there is no native application doing this task, making photos management more challenging.

Our idea can simplify the process of organizing large collections of images by automatically identifying and grouping them.

This Computer Vision project proposes an algorithm that aims to detect all the faces in a collection of photos in order to cluster them by identities and organize them into different folders. Given a photo, the algorithm will also recognize the faces in it and search them in the file system's images, displaying only the images that contain the recognized faces of the given one. Moreover, the application provides the capability to search within the collection of images for the one that most closely matches a user-provided input string. Finally, the application labels the images based solely on their background and returns the top 5 most significant clusters.

2. Introduction

According to a study made in 2022 by Mylio, the amount of pictures that will be taken around the world by the end of that year is going to reach approximately 1.5 trillion photos. So the total number of photos taken equates to about 188 pictures per year for every man, woman, and child in the world, considering a global population of 7.95 billion in 2021. While it is important to note that these numbers can vary significantly from person to person and from country to country, based on different patterns of technology usage and cultures, they highlight 2 aspects: people around the world love taking photos and, therefore, their number is exploding. This has a serious impact on the task of organizing multimedia data on a device. When someone wants to view all their photos of a specific person, or wants to find a specific object, it could take years to find them.

For example, suppose someone wants to find a photo taken three years ago, in which he was on vacation with his girlfriend on the beach or that he wants to find an old photo of him playing soccer. If it was three years old, he would have to scroll through thousands of images before finding it, and if he doesn't have a precise time indication, he could take hours.

Thanks to our system, the number of images to examine could be significantly reduced by inserting a photo with both people, or by searching for the keyword "soccer ball" or checking if among the most significant backgrounds, there is one presenting a beach.

3. Related Work

In the field of computer vision and image management, several remarkable techniques and models have been developed to address the challenges of organizing and retrieving large collections of images. For face detection, widely recognized methods include the Haar Cascade classifier, Multi-task Cascaded Convolutional Networks [1] (MTCNN) and other Deep Learning architectures.

To address the task of Face Recognition, there are many methods, some of which are based on CNNs like FaceNet [2] and VGGFace [3]. These methods have significantly improved accuracy and efficiency in identifying individuals inside images, becoming key components in various identity verification system applications.

For text-image pair analysis and string-based image retrieval, the state-of-the-art model is CLIP [4]

While these techniques and models have individually made significant contributions to their respective domains, our proposed system aims to integrate these concepts into a comprehensive solution. It combines the capabilities of face detection, facial recognition, and text-image translation to offer to the users an application with many functionalities. This surely helps to solve the challenge of managing and accessing the vast and ever-increasing volume of digital photographs in today's image-centric world.

3.1. Apple on-device Photo Organization

In the field of computer vision and image organization, the experimental study performed by Apple introduces a comprehensive approach to enhance the management of images for Apple devices [5]. Photos application is designed to help users organize and manage their photo collections using machine learning algorithms, with a specific focus on recognizing individuals in photos.

To provide it, the application uses deep neural networks to detect and extract feature vectors from faces and upper bodies analyzing images under various lighting, pose, and expression conditions. Moreover, Apple's system prioritizes on-device performance: the entire recognition process runs locally and so employs a custom-designed neural network architecture that combines lightweight and efficient modules.

During our project development, we considered their idea but opted for simpler techniques and architectures that could be better aligned with our aim. For example, their recognition structure involves two phases: the first one involves the construction of a gallery of known individuals and the assignment of new observations to these individuals. On the other hand, our model is aimed at assigning an identity to each face, making it more suitable for retrieval purposes, especially in those cases that need to manage multiple identities simultaneously; moreover, it also takes into account individuals who appear less frequently.

4. Data

To train the neural network responsible for extracting features, we needed a dataset with a large number of faces, and the choice fell on CelebA. CelebFaces Attributes Dataset [6] is a large-scale face attributes dataset with 202,599 celebrity images, composed of 10,177 different people. Each image in this dataset has RGB format and 178x218 pixels resolution, making it a valuable resource for various computer vision and facial analysis tasks. We have splitted the dataset into two subsets, with 80% of the data for the train set and 20% for the test set. Furthermore, thanks to its considerable number of photos per identity, it was useful in providing robustness to the network regarding scales, poses, and lighting conditions. It was also a valuable dataset for testing the subsequent face clustering component.

To conduct a meaningful test in line with the environment in which the application will be used, we also extracted approximately 500 images from our devices in order to test the background clustering and the string search functionalities. This allowed us to include noisy images, screenshots, and images of various sizes and orientations, mimicking real-world usage scenarios.

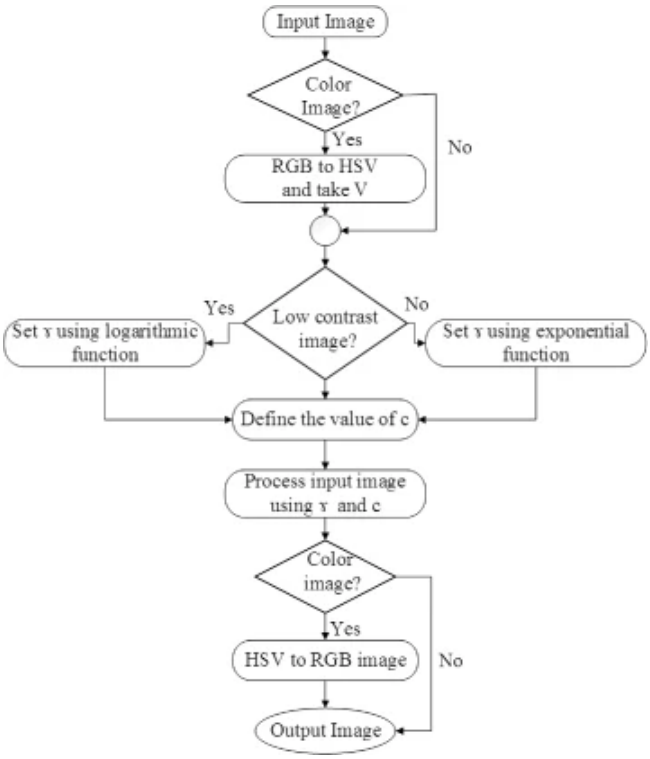
4.1 Image Processing

To work on the images, we had to resize the detected faces to 224x224, making them compatible with the input of the CNN that extracts features from them. In addition, in some cases, we needed to use image processing techniques. In particular, we addressed the problem of exposure to sunlight, as some faces are not detected if they are backlit.

By using OpenCV Gamma Correction, it is possible to increase the gamma value to enhance the brightness and contrast of the image and make it more readable.

CNN with Gamma Correction (Gamma = 2.0)

Building upon this concept, we employed the Adaptive Gamma Correction (AGC) algorithm [7], that automatically assesses which gamma level to apply, based on whether an image has high or low contrast. This approach offers flexibility and can properly handle various types of photos that may exist in our galleries.



This functionality is implemented only during the inference phase to correct faces that are underexposed to light and to improve their detectability.

5. Technical Approach

Our application provides four primary features:

Face Clustering: The entire gallery is segmented into clusters based on identities, and a folder is created for each one of the clusters containing more than 5 images. To accomplish this task, each image undergoes face detection; the detected faces are then processed through a feature extraction network and organized using a clustering algorithm.

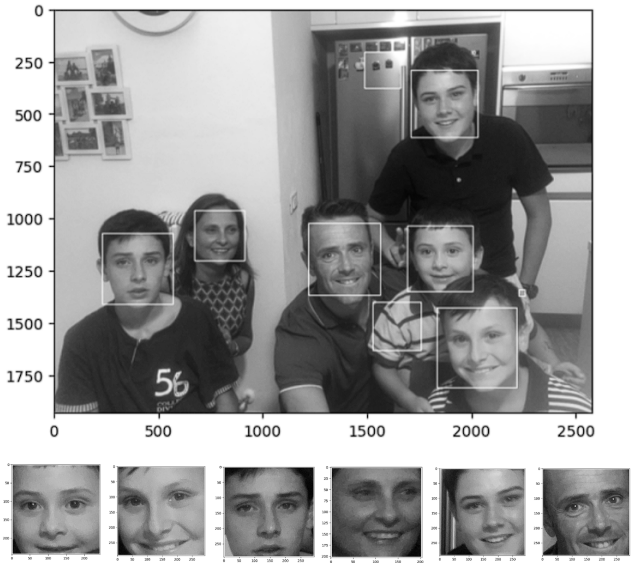
Image Retrieval: Given an input image, the program identifies the identities inside of it, if any, and, using the previously clustered images, returns only the photos where all of these identities are present, i.e. the intersection between each identity cluster.

Background Clustering: The complete gallery is partitioned into clusters based on background characteristics, with folders generated for the most significant clusters. For each image, background extraction is performed and the extracted backgrounds are fed into a neural network for feature extraction, followed by clustering.

Image Search: Our system allows users to submit a text string, which is then compared to the images by a CLIP model to identify and retrieve the most similar to the entered text.

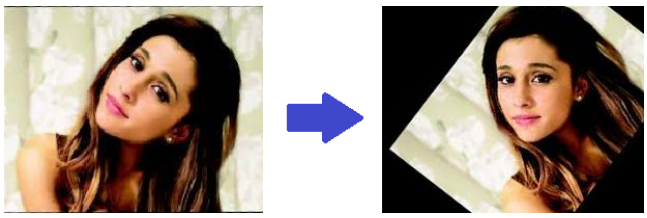
5.1. Face Detection and Alignment

The first challenge we tackled was face detection. For each image in our gallery, we need to extract all the faces inside of it. We have studied and tested various options for the detection part, and we ultimately opted for a pre-trained convolutional network provided by Dlib, taking into account only detections with at least 90% of confidence.



Before proceeding with the feature extraction, we made two modifications to the detection results. First, if it was the case, we extended the bounding box to increase a bit the number of pixels detected by the network. This will be useful for applying the clustering algorithm.

Then we apply a face alignment algorithm using the shape predictor from Dlib [8] to obtain landmarks of detected faces. These are points on the face such as the corners of the mouth, along the eyebrows, on the eyes, and so forth. Dlib provides a pretrained model on the iBUG 300-W face landmark dataset, which allows us to acquire the positions of 68 landmarks. Using these landmarks, we extracted the centers of the eyes and used these values to compute the rotation matrix to align the face so that the eyes-line becomes horizontal.



5.2. Feature Extraction

Once all faces have been detected, it is necessary to process them in order to extract the significant features that allow us to distinguish one person from another. This process is executed using a ResNet50 [9], that we have trained to extract the embedding space of features instead of a probability for each class. So we replaced the last linear layer with a layer that projects the extracted features into a 4096-dimensional space.

Considering that our goal is not classification but rather than obtaining similar features for similar faces and highly dissimilar features for different ones, we have employed the Contrastive Learning paradigm, specifically utilizing the Triplet Loss.

During training, for each epoch, we divide the training set data into multiple 32-images mini-batches, each manipulated as follows:

We randomly extract one image, which is marked as 'anchor'.

We then arbitrarily select n other images of the same identity, referred to as positive samples, which replace n images in the batch.

This process is repeated three more times, ensuring that the selected labels and replaced images are different each time. At this point, for each batch, we compute all possible triplets composed of an anchor, a positive element, and a negative element. These triplets must satisfy the constraint that the anchor and the positive element must have the same identity, while the anchor and negative element must have different identities. Among these triplets, we select the hard ones, meaning those for which the anchor is farther from the positive element than it is from the negative one. In this way, our loss will be positive, enabling us to learn how to obtain an embedding space where similar examples are closer together, and different samples are far apart.

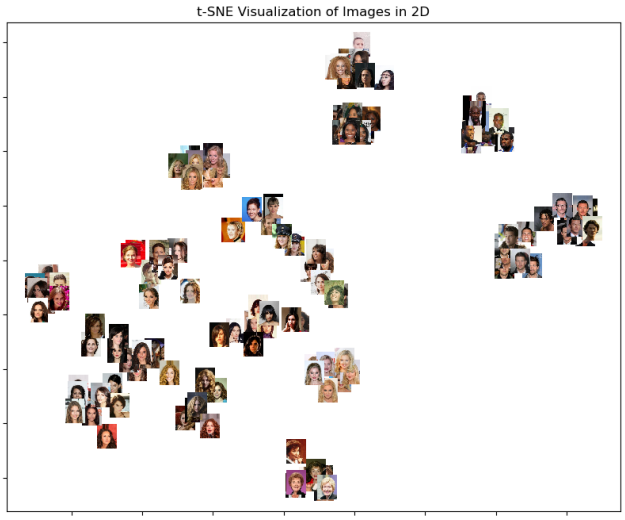
Thus the Loss used is:

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

To improve performance, we decided to use transfer learning by initializing the weights with those extracted from ImageNet1K [10].

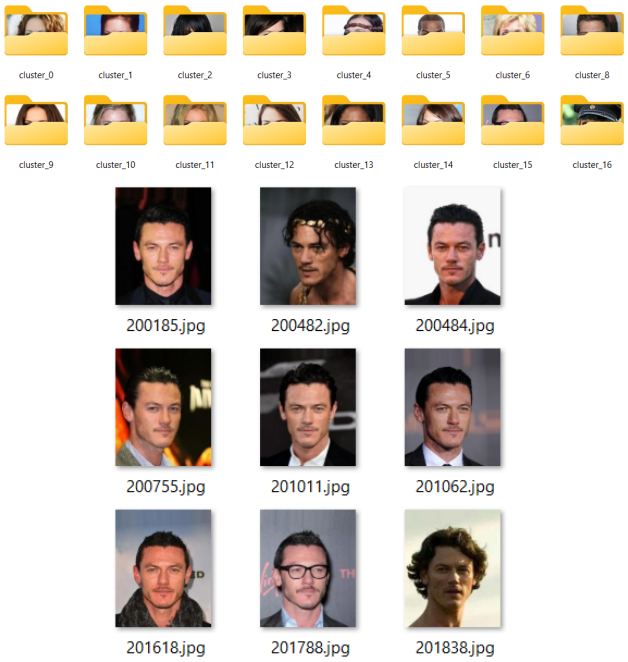
5.3. Face Clustering

After performing feature extraction, the next step is to cluster the features in order to assign each entity to a unique cluster. Among many algorithms, we chose Agglomerative Clustering [11], the most common type of hierarchical clustering used to group objects in clusters based on their similarity, without knowing the number of groups to create. The algorithm starts by treating each object as a singleton cluster. Subsequently, pairs of clusters are iteratively merged until all clusters have been merged into one big cluster containing all objects or until the distance between different clusters reaches a given threshold.



5.4. Folder organization

At the end of the clustering process, a folder is created for every cluster containing more than 5 elements. Inside of it, we can find all the images of that person and the associated cluster label as the name of the folder.



CLUSTER_15

5.5. Saving Features to a file

To mitigate a high computational load each time the application is accessed, we have considered storing certain intermediate results. Once the features of the faces are extracted, they are compressed together with the image path and cluster label, then saved in a binary file.
A similar procedure is applied after the background clustering. The chosen module for file saving is "pickle."

5.6. Image Retrieval

The user has the option to provide an image as input to the system, which will serve as the basis for retrieval. To obtain all the faces present in the image, it is passed through the face detector, and from there, two paths are followed:

If the image contains only one face, feature extraction is performed on that face, and subsequently, a K Nearest Neighbor [12] (KNN with k=1) algorithm is used to find the cluster most similar to the input image.

If the initial image contains multiple faces, feature extraction is performed on all the detected faces and successively the system proceeds with the KNN algorithm. This time, the computed clusters are intersected to obtain only the images in which all the people present in the input image appear simultaneously.

The KNN algorithm fits the features of the entire collection, contained in the file created in the previous step, in order to calculate distances and similarities between the elements and the input image. It returns the single closest element, from which the cluster is then extracted.

5.7. Background Clustering

Our proposed approach to background clustering follows this guideline: starting with an image or a batch of images, we extract the background pixels and pass them through a neural network to obtain a background feature vector. Subsequently, we apply a clustering algorithm to group similar backgrounds.

To perform image cropping, we utilized the "rembg" tool, which leverages the U2-Net [14] neural network to extract the foreground. The background was then constructed by selecting black pixels if they belonged to the foreground and using pixels from the original image otherwise.

The set of images is then fed into the ResNet50 model from CLIP to extract a 1024-dimensional feature vector for each image.

Starting from these feature vectors, the DBSCAN [13] clustering algorithm ensures that photos with the same background are assigned to the same label and that isolated images are managed as noise. Subsequently, the system will display on the screen only the top 5 clusters with the highest number of elements.

5.8. Image Search Engine

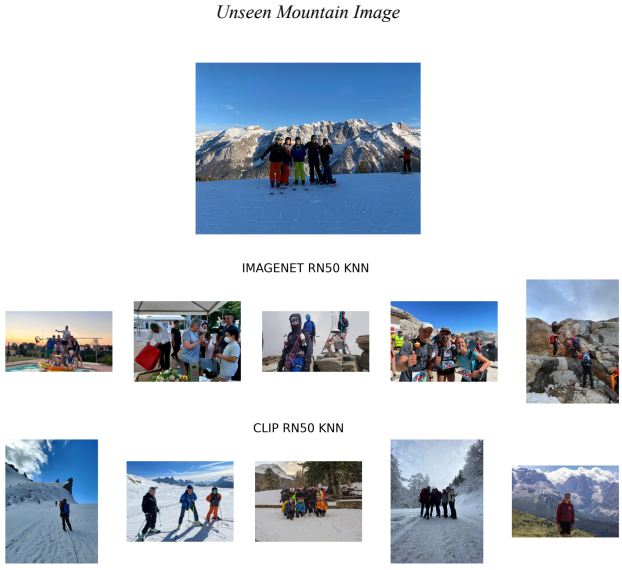
The latest feature of our application is designed to simplify image searching. Thanks to CLIP, we have enabled image search using text input. The user will input a sequence of words, which will be passed to the CLIP model along with the entire image collection. The model will return two tensors containing the logit scores corresponding to each image and text input, where the values are cosine similarities between the corresponding image and text features, times 100. Hereafter, all the images with a similarity score exceeding a certain threshold will be displayed on the screen (or only the top 10 if more). The system will also prompt the user if he wishes to load more images.

Considering that our application operates in real-time, we opted to use the RN50 version of the CLIP model, which still achieves good results but with significantly reduced processing times compared to the ViT model.

6. Experiments

In order to evaluate the effectiveness of our proposed Computer Vision system for image organization and retrieval, we conducted a series of experiments that aimed to demonstrate its capabilities and advantages. To assess the accuracy and efficiency of our system in detecting and recognizing faces within a collection of photos, we compared it with established face detection methods, including Haar Cascade, MTCNN. However, we chose the Dlib CNN due to its excellent balance between speed and accuracy. For clustering, we explored various clustering methods to identify the most effective one. We conducted tests using other algorithms but ultimately chose agglomerative clustering for face clustering and DBSCAN for background clustering.

Despite not having developed a complete application that encompasses all our functionalities, we conducted individual tests on a dataset constructed using our personal images and examined the results. For instance, in the extraction of features from the background, we employed a KNN algorithm to determine whether the features extracted from a pre-trained ResNet on ImageNet or from the Clip model were visually more consistent.



Through these real-life experiments, we observed that our approach is effective and would significantly simplify image retrieval within very large collections.

Nevertheless, there are certain limitations, mainly associated with computational load and the time required for the initial access, during which the images must undergo the initial processing through the neural networks, which might take a considerable amount of time.

We have also observed that the feature space obtained through our training, while achieving good results, is less performant than the one extracted from a state-of-the-art architecture such as a ResNet50 pre-trained on VGGFace, that produces more homogeneous and consistent clusters.

7. Conclusions

Our project offers several possibilities for extension, such as background classification and clustering into useful categories (e.g., snow, sunset, beach, most famous cities) or adapting the pipeline to work with videos.

In conclusion, our Computer Vision system presents a comprehensive solution to the challenge of managing and retrieving images in the era of exponentially growing digital photo collections, offering users a wide range of functionalities for image organization and retrieval, such as facial recognition and clustering, background clustering, or text-based and face-based image retrieval.

References

[1] Zhang, K., Zhang, Z., Li, Z., and Qiao, Y. (2016). Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Processing Letters.

[2] Schroff, F.; Kalenichenko, D. & Philbin, J. (2015), FaceNet: A unified embedding for face recognition and clustering., in 'CVPR' , IEEE Computer Society, , pp. 815-823 .

[3] Parkhi, O.M., Vedaldi, A. and Zisserman, A. (2015) Deep Face Recognition. Proceedings of the British Machine Vision Conference (BMVC).

[4] Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger and Ilya Sutskever. “Learning Transferable Visual Models From Natural Language Supervision.” International Conference on Machine Learning (2021).

[5] Apple, “Recognizing People in Photos Through Private On-Device Machine Learning”, (2021)

[6] Liu, Ziwei, Ping Luo, Xiaogang Wang, and Xiaoou Tang. "Deep Learning Face Attributes in the Wild." In Proceedings of International Conference on Computer Vision (ICCV), December 2015.

[7] Rahman, S., Rahman, M.M., Abdullah-Al-Wadud, M. et al. An adaptive gamma correction for image enhancement. J Image Video Proc. 2016, 35 (2016).

[8] Davis E. King. Dlib-ml: A Machine Learning Toolkit. Journal of Machine Learning Research 10, pp. 1755-1758, 2009

[9] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.

[10] J. Deng, W. Dong, R. Socher, L. -J. Li, Kai Li and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 2009, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

[11] Müllner, Daniel. (2011). Modern hierarchical, agglomerative clustering algorithms.

[12] Guo, G., Wang, H., Bell, D., Bi, Y., Greer, K. (2003). KNN Model-Based Approach in Classification. In: Meersman, R., Tari, Z., Schmidt, D.C. (eds) On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE. OTM 2003. Lecture Notes in Computer Science, vol 2888. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-39964-3_62

[13] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96). AAAI Press, 226–231.

[14] Qin, Xuebin, et al. "U2-Net: Going deeper with nested U-structure for salient object detection." Pattern recognition 106 (2020): 107404.