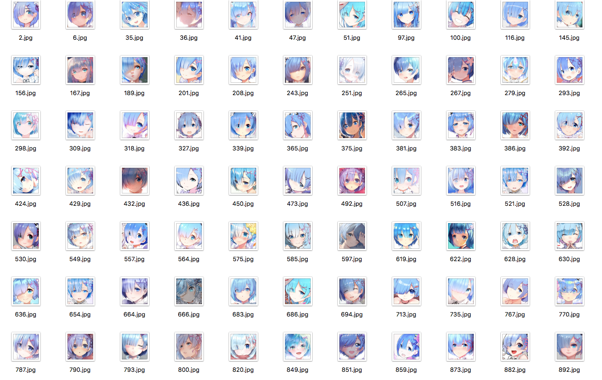
**Implementation of self-supervised learning to cluster unlabeled images**

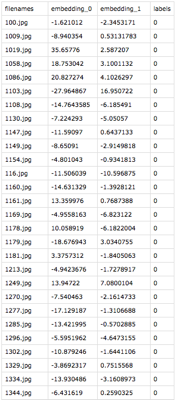
**Tim(shiyun) Kong**

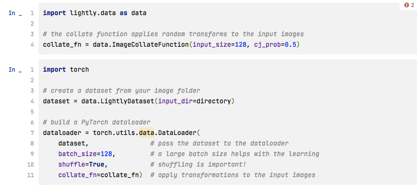
The problem statement of my project is: How to cluster, and hence classify a set of unlabeled images. This has importance in providing hints, and helping people identifying the classes amongst images. In my case, using the anime face from Rem is an implementation to help people decide their online profile by choosing the picture they like the most from a reduced amount of images.

I start with Rem face dataset that contains 725 images without labels:

https://www.kaggle.com/datasets/andy8744/rezero-rem-anime-faces-for-gan-training

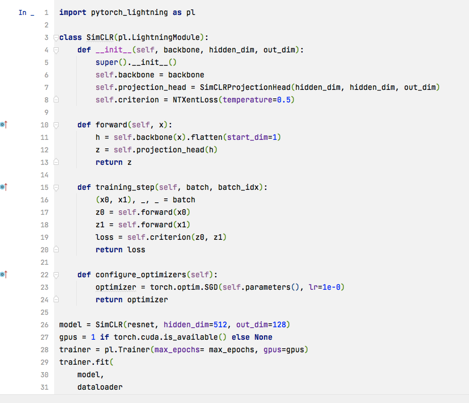


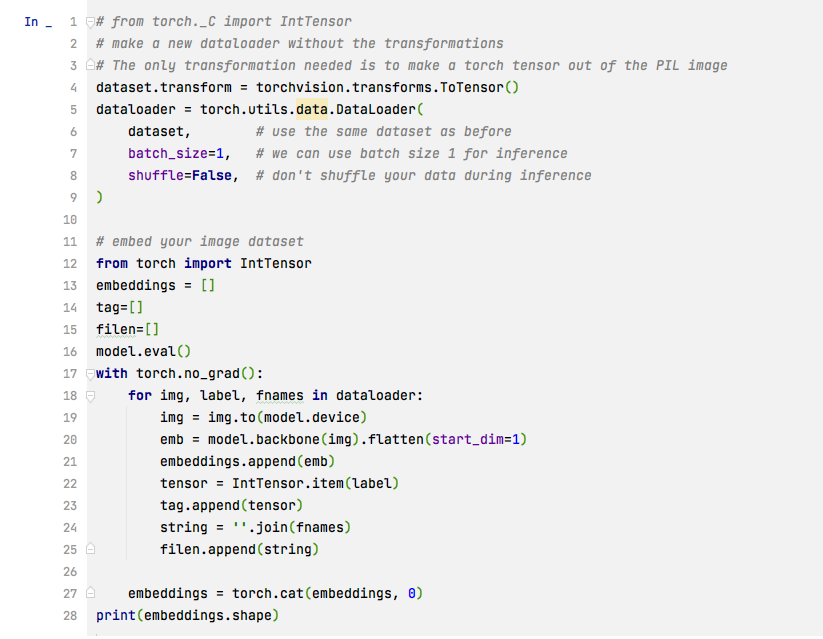
Then I introduce all the images to Collate function, which will undertake image transformations such as rotation, reflection, cropping, altering colors, etc. This step endeavors to minimize the effect of minor details on image to influence the result of clustering.



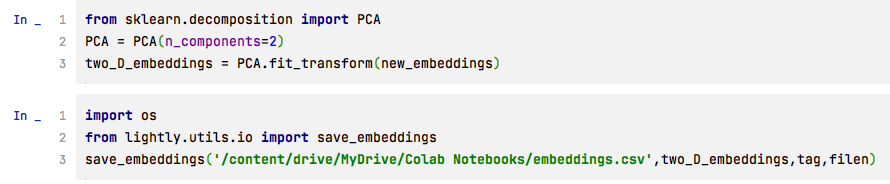
Then, I will take the SimCLR model (which is a simple framework for contrastive learning of visual representations) with backbone of residual neural network(ResNET)(which is a artificial neural network that has no drop-outs, so it minimize the effect of losing gradients and explores more of the featured space; and ResNET is able to skip through layers to make learning faster) to project all the images into a 512 dimension space.



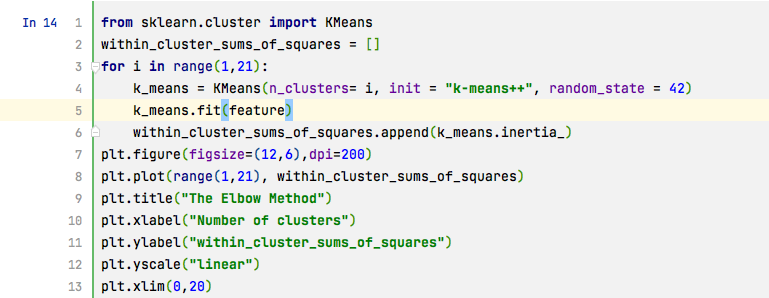
And in the featured space, the training happens to determine the locality of each images’ relative position. Then the model will extrapolate 512 dimension embeddings.



Because 512 dimension is far too big for any machine learning models, therefore, I used Principle component analysis to reduce the dimensionality to 2.



Then, I can undertake k-means to calculate within clusters of squares sums (the Euclidian distance between images) by projecting the embedding into a 2D latent space. By implementing the Elbow method, which shows the relation between average Euclidian distance and number of clusters, I find the latest obvious turning point and obtained the optimum number of clusters for the images.



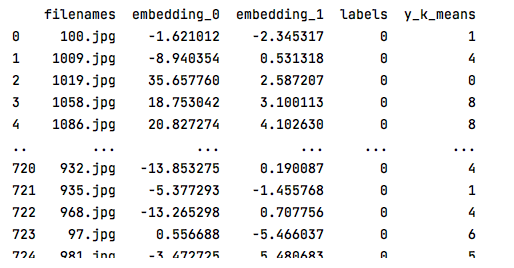


Then, the result is being verified through K-nearest Neighbor:





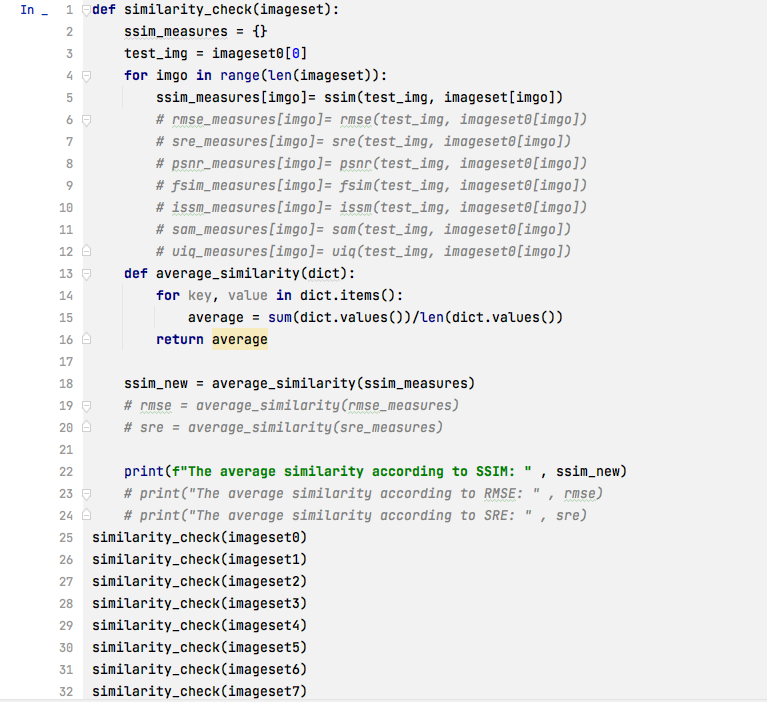
Which shows the 5 images with smallest Euclidian distance between each other are very similar. Then, I printed examples according to their label in the names of variable y\_k\_means obtained from k-means algorithm:

****

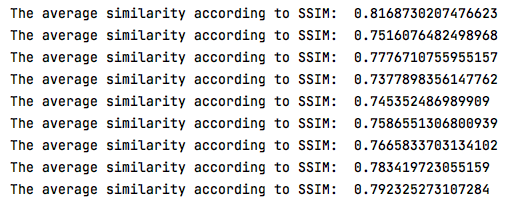
****

****

Numerical validation is also applied to this project:



Using the ssim image similarity comparison package, I successfully rendered the image similarity: showing that a random image in dataset 0 has most similarity with pictures in the same cluster as its.



Until, this step, I can conclude the need to cluster images according to newly generated labels is finished. Future works may be using style Generative Adversarial Networks to create high quality images that follow the classified pattern, and hence make feasible for people to construct their own online profile (not taking the artwork of others).