**Face Recognition**

**Problem Statement:**

1. Explore the application of Convolutional Neural Networks (CNN) in matching extracted faces in images to a known identity in database using feature extraction, transfer learning, fine tuning, data augmentation, regularization, and CNN embeddings. We have 11 identities selected from the publicly available CASIA-WebFace dataset. First, 8 models are created repurposing pretrained models on the VGGFace2 dataset using VGG16 and Resnet50 architectures. VGGFace2 dataset contains 3.31 million images of 9131 subjects across different poses and ages. Then, a pretrained model that uses on MS-celeb-1M dataset with 1 million subjects and one image per person in the database is repurposed to classify an extracted face to one of the 11 subjects or ‘Not in database’. Performance of a total 9 models are compared and visualized.

**Overview of Technology:**

1. Do existing pretrained models correctly recognize faces in images?
2. How to improve performance of existing pretrained models using deep learning techniques?
3. What if we only have one image per person in our database? What if we want to add another person to the existing database for face recognition, or if the person in the image is not in our database?
4. What if there are multiple faces in the images?

**Data set:** Selected 11 classes of celebrities in CASIA-WebFace publicly available dataset([here](https://drive.google.com/file/d/1Of_EVz-yHV7QVWQGihYfvtny9Ne8qXVz/view)) This dataset is created by the Institute of Automation, Chinese Academy of Sciences (CASIA) gathered from IMDb profiles. CASIA WebFace Facial dataset includes 494,414 images over 10,575 identities.

**High Level Overview of steps:**

1. Create new environment, install softwares, download dataset from [here](https://drive.google.com/file/d/1Of_EVz-yHV7QVWQGihYfvtny9Ne8qXVz/view).
2. Clean images by extracting only one face from each image such that the area of the face is the largest.
3. Select 11 classes of dataset and split each class to three folders for training, validation, and testing.
4. Use pretrained models (VGG16 and Resnet50) to predict faces of the 11 subjects.
5. Train 8 models using feature extraction, transfer learning, and fine tuning with keras.
6. Use data augmentation and regularization to improve model performance and avoid overfitting.
7. Compare training and validation accuracy across 8 models and predicted class of faces for the 11 subjects.
8. Download FaceNet pretrained model and repurpose it to predict one of the 11 subjects or ‘Not in database’.
9. Create a function that classifies all faces in an image to one of the 11 subjects or ‘Not in database’.

**References:** Lecture notes: 5.5, 5.2, 5.3 and labs, Deep Learning with Python, Francois Chollet, Chapter 5.

**Hardware:** processor**:**  2.3 GHz Intel Core i9 , memory: 16 GB 2400 MHz DDR4

**Software:** Python 3.7.6 / Keras 2.1.0/ Keras-facenet/keras-vggface/tensorflow 2.1.0/Anaconda (jupyter notebook)

**Lessons Learned & Pros/Cons:**

1. Repurposing existing models, using a moderate size dataset, to match an extracted face to a known individual can significantly improve the prediction accuracy of pretrained models.
2. The performance of repurposed models strongly depends on the underlying network architecture and not necessarily the depth or number of layers. VGG16 yields higher validation accuracy with less layers retrained. VGG16 and Resnet50 models are both pretrained on the same VGGFace2 dataset. However, when more of the layers are retrained, model using ResNet50 improved more.
3. The performance of repurposed models strongly depends on the number of images in the underlying datasets and the learning algorithm of pretrained models. Repurposing FaceNet model with 1 million identities and one-shot learning algorithm strongly over performs repurposed models using VGG16 and ResNet50 with 9,131 identities that use many images per identity.
4. Fine tuning improved the validation accuracies more than transfer learning using models’ convolutional bases.
5. Models using VGG16’s convolutional base responded better than ReNet50 to the transfer learning when 1 or 2 layers added. Model using ReNet50’s convolutional base achieved the same accuracy by adding 2 hidden layers.
6. Model using Resnet50’s convolutional base responded better than VGG16 to finetuning.
7. There are many hyperparameters that could be optimized using hyperparameter-tuning technique. The selection of those parameters can impact the results and accuracies. Optimizing such parameters requires more time and computational power.
8. With the selected parameters, the validation and training accuracy and loss depicted enough selected epochs. Therefore, AWS was not used in this project.

**YouTube URLs:**

**Short video:** <https://www.youtube.com/watch?v=JfPWQ8RJ6LM>

Long video: <https://www.youtube.com/watch?v=wrOLiOygwJU>