



## Performing Facial Recognition with Deep Learning

**Objective:** Create a facial recognition tool using a relevant deep learning algorithm, leveraging the provided resources.

**Context:** You are working for Face2Gene, an American AI company that has developed a healthcare app for doctors. The app utilizes deep learning algorithms to aid in diagnosing patients for genetic disorders and their variants. It converts patient photos into de-identified mathematical facial descriptors, which are then compared to syndrome-specific computational-based classifiers to determine similarity. The app provides a prioritized list of syndromes with similar morphology and suggests phenotypic traits and genes for feature annotation and syndrome prioritization.

Management has given priority to empowering and entrusting the in-house AI team. As a new member of the team, your task is to build a baseline model for facial recognition. The goal is to further enhance the app's existing features and add more value to the business based on this baseline model.


**Dataset Details:** The ORL Database of Faces consists of 400 images from 40 different subjects. The images were captured at different times, under varying lighting conditions, with different facial expressions (open, closed eyes, smiling, not smiling), and with or without glasses. All the images have a dark homogeneous background, and the subjects are positioned upright and frontal with some tolerance for side movement. Each image has a size of 92x112 pixels and 256 grey levels per pixel.

**Data can be downloaded from the following link:**

<https://www.kaggle.com/datasets/kasikrit/att-database-of-faces>

**Steps to be followed:** The following steps will guide you in building the model.

1. Import the relevant packages and collect all the necessary dependencies.
2. Upload and import the data.
3. View a few images to get a sense of the data.
4. Create a validation framework and split the data into train, test, and validation datasets.

- 
5. Perform necessary transformations to prepare the data for input to the CNN model.
  6. Build a CNN model with three main layers: a convolutional layer, a pooling layer, and a fully connected layer. You can also consider utilizing state-of-the-art architectures using transfer learning.
  7. Train the model using the prepared data.
  8. Plot the results to evaluate the model's performance.
  9. Iterate on the model, making adjustments and improvements, until you achieve an accuracy above 90%.

Note: Please refer to the Chicago Manual of Style for any specific formatting or citation requirements.