Modern face recognition with deep learning

Danylo Dembitskyi
IA et Deep Learning
ESIEE - 2023-2024
France
danylo.dembitskyi@edu.esiee.fr

***Abstract*—Facial recognition systems play a vital role in verification and security applications, yet there's room for improvement in accuracy. Challenges like occlusions, pose variations, and changes in illumination can lead to errors in facial feature detection. To address these issues, the project aim to deploy the use of Histogram of Oriented Gradients (HOG) descriptors, known for their robustness. Deep learning techniques offer a reliable method for facial measurement using the pretrained neuron network models. The final step involves training a classifier to match facial measurements from a new test image with known individuals. We are developing a Python-based application on Google Collab platform to improve accuracy and reliability.**

***Index Terms*—Deep learning, Artificial Intelligence,Facial Recognition,Convolutional Network.**

I. Introduction

Facial recognition technology has gained in popularity, finding wide-ranging applications in security, authentication, and surveillance, among others. Its deployment at major events like football matches, concerts, and upcoming Olympic Games in Paris underscores its crucial role in ensuring public safety. However, despite its prevalence, facial recognition systems encounter substantial challenges, notably in terms of accuracy and robustness. Variations in lighting conditions and facial pose pose significant obstacles, often leading to inaccuracies and false identifications.

By consequences,the ability to accurately identify individuals can help prevent crimes and enhance public safety. Similarly, in authentication systems, reliable facial recognition can provide a convenient and secure means of accessing devices, accounts, and sensitive information.

Addressing these challenges requires advancements in facial detection, recognition algorithms and scalable systems capable of handling real-world scenarios. By improving the accuracy and reliability of facial recognition technology, this work aims to streamline authentication processes and contribute to the advancement of AI-driven solutions and Deep Learning.

II. Related Works

Deep Learning Approach:

*** Paper:** *"DeepFace: Closing the Gap to Human-Level Performance in Face Verification"* by Yaniv Taigman, Ming Yang ,Marc'Aurelio Ranzato and Lior Wolf.

*** Strategy:** This work presents a strategy for face recognition, leveraging a large-scale labeled face dataset collected from online sources, known as the Social Face Classification (SFC) dataset. The authors employ deep neural networks (DNNs) for feature learning and verification, utilizing techniques such as 3D model-based alignment and multi-class classification. By training on diverse data sources and employing various network architectures, their method reaches an accuracy of 97.35 percent on the Labeled Faces in the Wild (LFW) dataset, reducing the error of the current state of the art by more than 27 percent, closely approaching human-level performance[1]. However, one potential issue lies in the reliance on a single metric for verification, which may not capture all aspects of face similarity accurately. To fix this, future work could explore the integration of multiple verification metrics or ensemble methods to enhance the generalization of the system's performance. Overall, this paper represents a significant advancement in face recognition technology, demonstrating the effectiveness of deep learning in real-world challenges for facial recognition.

Non Deep Learning Approach:

*** Paper:** *"Face recognition using Eigenface approach"* by . Marijeta Slavković and Dubravka Jevtić.

*** Strategy:** In this work, a face recognition system utilize the Principal Component Analysis (PCA) algorithm, specifically the eigenfaces approach, is implemented. The database of faces is chosen to be ORL containing 190 images of 38 individuals for training and 40 images for testing, with each image having dimensions of 92×112 pixels. The preprocessing involves transforming each 2D image into a 1D vector and centering the matrix to ensure the first principal component describes the direction of maximum variance[2]. The model relies on calculating the covariance matrix, finding its eigenvectors and eigenvalues, and projecting new images onto the eigenface subspace for classification. The results demonstrate recognition rates of up to 97.5 percent using both Euclidean and Manhattan distances with 20 principal components.A proposition for improvement could involve adopting adaptive thresholding techniques or exploring machine learning algorithms to dynamically adjust recognition thresholds based on data characteristics.

*** Paper:** *"Hierarchical classification and feature reduction for fast face detection with support vector machines"* by Bernd Heisele, Thomas Serre, Sam Prentice and Tomaso Poggio.

*** Strategy:** In the work "Hierarchical classification and feature reduction for fast face detection with support vector machines" Heisele et al. employed the CMU(Carnegie Mellon Database) face database for testing their method. They applied hierarchical classification, utilizing a cascade of classifiers

with increasing complexity to efficiently reject background regions and focus computation on potential object locations. Additionally, they utilized feature reduction techniques to select relevant image features, optimizing the computational load of the system.[3] The results demonstrated a remarkable speed-up in face detection, achieving a factor of 335 while maintaining classification accuracy. However, a potential flaw could be the limited scope of testing on a single database, which might act differently with new databases and test set. To enhance the paper, further validation on diverse datasets capturing various environmental conditions and demographics could provide a more precise evaluation of the proposed method's robustness and generalization capabilities.

III. Proposition

A. Face detection

- The provided data for face detection comprises 225 elements, consisting of JPEG, PNG, and JPEG format images in folder 'data' with overall size 91.9 Mo. All images are in RGB format. These images feature various characters from the "Jurassic Park" movie, including Ian Malcolm, Alan Grant, Claire Dearing, Ellie Sattler, John Hammond, and Owen Grady. The dataset is diverse, with a range of characters and facial expressions captured in different scenes. The images vary in dimensions, with the smallest image measuring 100x100 pixels belonging to Ellie Sattler(00000022.jpg), and the largest image measuring 2160x980 pixels belonging to Ian Malcolm(00000007.jpg). On average, each image is approximately 126.82 KB in size and has dimensions of approximately 785.16x567.62 pixels.
- This dataset offers important insights into various aspects such as different shapes, angles, and lighting conditions. Detecting faces enables the analysis of characters' facial expressions and emotions, which favors an understanding of scenes and character interactions within the storyline. In the realm of video editing and visual effects, face detection facilitates the application of effects or adjustments to characters' facial features. Moreover, analyzing this dataset provides insights into character dynamics, distinguishing between protagonists and antagonists, and identifying main characters. This detailed analysis enriches statistical understanding and develops facial feature analysis across various challenges.
- The code employs two methods for face detection, namely Histogram of Oriented Gradients (HOG) and Convolutional Neural Network (CNN) based detectors from the dlib library, applied to input images to locate faces **(face_locations(image, model="cnn")** function) Figure Following detection, the faces are aligned using facial landmarks, ensuring right positioning and orientation. Facial landmarks, extracted via pre-trained models, capture key facial points like eyes, nose, and mouth, utilized for alignment and feature extraction **(face_landmarks(face, model="large")** function) Figure 3. Then, each aligned face is encoded into a fixed-length vector **(face_encoder()** function) Figure 5.
  Finally, the pre-processed data, comprising aligned faces and their encodings, undergoes preparation for neural network training by splitting into training and testing sets, encoding class labels, and formatting for tasks such as classification via one-hot encoding.

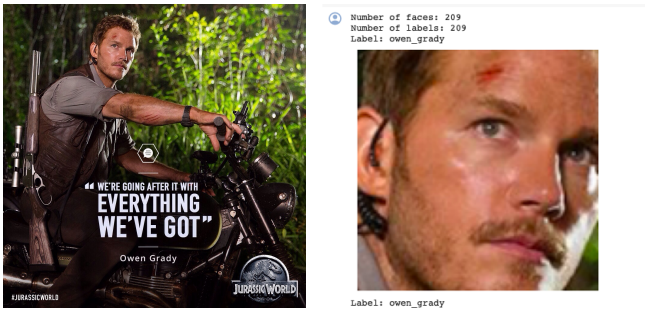

Fig. 1. Face before and after detection.

- The 70-30 split is a practical choice that balances the need for sufficient training data with the importance of unbiased evaluation, leading to more reliable and generalizable deep learning model.
  With a smaller testing set, there's a risk of **overfitting**, where the model performs well on the testing data but poorly on new, unseen data. However, the 70-30 split strikes a good balance between training and testing data, helping to mitigate overfitting by providing enough data for the model to learn general patterns while still having a separate dataset for evaluation[4].
- The chosen convnet is comprised of a series of **separable convolutional layers** followed by **max-pooling layers**. The project involves pooling layer to reduce the spatial size and reduce the noice of the input . It is used a stochastic pooling random pooling mask at each pass. Max pooling aids in making the model invariant to feature location and orientation, ensuring the network can identify faces in images regardless of their placement.Through downsampling, max pooling cuts down on parameters and computations within the network, accelerating learning and mitigating overfitting risks. The rectifier neuron Rectified Linear Unit (**ReLU**), is one of several keys to the recent success of deep networks. It expedites convergence of the training procedure and leads to better solutions than conventional sigmoid-like units [5][6]. **Dropout** tecnhique is used and requires training multiple neural networks, which can increase the training time significantly. Therefore, the random deactivation of the neurons during training encourages the network to learn more robust and generalized representations of the input data. This regularization method not only enhances generalization but also facilitates faster convergence during training. By preventing the network from relying too heavily on specific weights, dropout promotes the learning of diverse features, leading to improved performance,

particularly in scenarios with limited training data.

- The convnet performs well during training, but its accuracy drops to around 67% on the test set, which is below today's standards Figure 2. After adjusting the **learning rate** from 1e-4 to the standard value, We could make an observation about significant improvement in performance. This highlights the importance of allowing the network to adaptively select its learning rate. The accuracy surged by over 12% for the test set and by 20% for the validation set. Furthermore, selecting the **batch size** judiciously is crucial. While these adjustments may appear time-consuming initially, fine-tuning them in my network led to achieving a 90% accuracy level in a shorter timeframe. Excessive layering proved to be counterproductive; despite initially incorporating numerous layers, the results failed to match those of a smaller yet carefully crafted network.

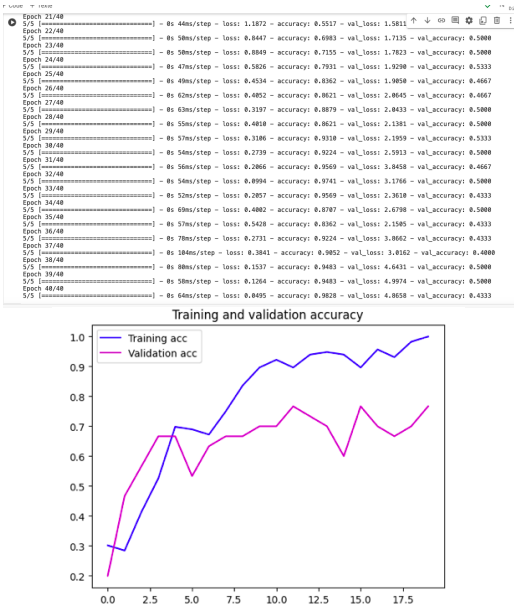

Fig. 2. Model and its training and validation tests.

B. Pose estimation

- Pose estimation refers to the process of determining the spatial orientation and position of an object or entity within an environment, typically in the context of computer vision and image processing[7]. In our case, the basic idea is to locate **68 specific points**, called **landmarks** (which are represented in model **pose68**) that exist on every face: such as the top of the chin, the outside edge of each eye, the inner edge of each eyebrow, and so on[8]. After extracting these landmarks, we rotate the image to obtain aligned and centralized faces for better and more accurate treatment analysis.In the Figure 3 we can see assigned labels,which were predicted by our network to corresponding face with landmarks.The process of **normalization** was crucial as it gave an opportunity to improve the results in the model training process.For this model, **one hot encoding** was used.

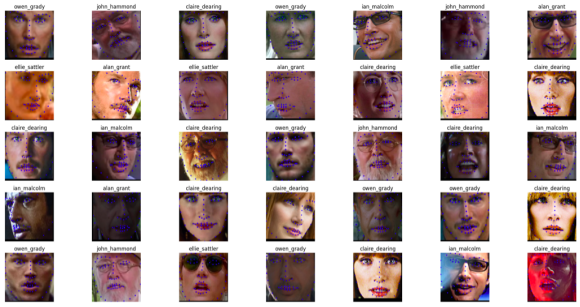

Fig. 3. Model and its training and validation tests.

- The data is similar as in Face Detection section. In movies like Jurassic Park, pose estimation is essential for seamlessly integrating computer-generated imagery (CGI) with live-action footage. By accurately estimating the pose and orientation of actors, props, and creatures within a scene, filmmakers can ensure realistic interactions between real and virtual elements. Pose estimation ensures that virtual objects align correctly with the live-action elements, enhancing the overall believability and immersion of the cinematic experience for audiences.
- The division of training and test sets is crucial for assessing the models' performance accurately, ensuring that they can effectively generalize to new, unseen data and are not **overfitting** to the training set.For this section the data is also divided in **70 percent of training and 30 percent of validating data.**
  Aligning faces reduces variability caused by different head orientations, scales, or perspectives. This reduction in variability can make it easier for the model to learn discriminative features and generalize better to unseen data.By improving the input data, the overall results are improved as well.
- We had to modify the number of **hyperparameters** due to the augmented loss during the test. The number of layers was reduced Figure 4.

```
#2.2 Re-train your convnets on the modified dataset.
from keras import models,layers,optimizers
from keras.utils import to_categorical
from sklearn.preprocessing import LabelEncoder
# Define the network
input_dim = [128,128,3] #As our image is in 128*128 format.   3 is a RGB colour.

# Convnet
model2 = models.Sequential()
model2.add(layers.Dense(64, activation='relu', input_shape=(input_dim)))
model2.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=input_dim))
model2.add(layers.MaxPooling2D((2, 2)))
model2.add(layers.Dropout(0.1))
model2.add(layers.Dense(32, activation='relu'))
model2.add(layers.Flatten())
model2.add(layers.Dense(32, activation='relu'))
model2.add(layers.Dense(6, activation='softmax'))

model2.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Fig. 4. Model 2.

C. Face encoding

- In face encoding, each detected and aligned face is encoded into a numerical representation, typically a **high-dimensional feature vector**. This encoding captures unique facial characteristics, such as the arrangement of facial landmarks, texture, and shape[9]. These encoded representations not only reduce the **dimensionality** of the data but also capture discriminative information essential for tasks like face verification, identification, and clustering. Additionally, face encoding enables the development of robust and scalable face recognition systems capable of handling variations in pose, illumination, expression, and occlusion[10].
- After face encoding,the raw facial images are transformed into numerical representations or feature vectors that capture the characteristics of each specific face. The output could be seen as a signal.Figure 5. These feature vectors generally possess **lower dimensionality** than the original image data but retain crucial facial information essential for recognition tasks. The resultant encoded data establishes a high-dimensional feature space where akin faces are closely grouped, facilitating efficient similarity assessment and classification.

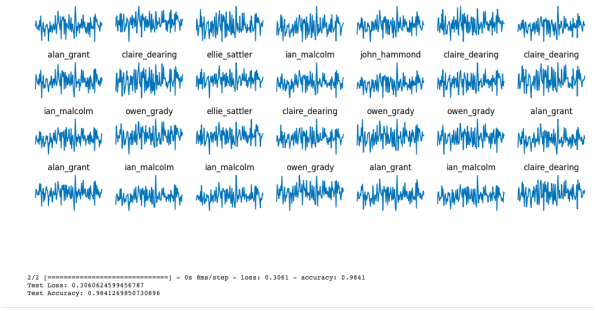

Fig. 5. Model and its training and validation tests.

- For this new dataset, a convnet is not employed since the input data consists of a single vector of size 128. This represents the **simplest model** among all the models discussed in this study, as depicted in Figure 6.

```
# Network architecture
model3 = models.Sequential()
model3.add(layers.Dense(128, activation='relu', input_shape=(128,))) # Input shape is a 128-length 1D vector
model3.add(layers.Dropout(0.5))
model3.add(layers.Dense(128, activation='relu'))
model3.add(layers.Dense(6, activation='softmax'))  # Output layer with 6 classes

# Compilation of the model
model3.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
#model3.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
# Training of the model
history3 = model3.fit(X_train3, Y_train_encoded3, epochs=60, batch_size=128, validation_split=0.2)
```

Fig. 6. Model of convnet 3.

Despite its simplicity, this model has yielded the best results for face detection in this work. Achieving a 100% test result, as shown in Figure 7, is a remarkable accomplishment.

D. Face recognition

- Face recognition is a biometric technology that aims to identify or verify individuals based on their unique facial features. It involves analyzing facial images or video frames to match them against a database of known faces. The process relies on the premise that each person has distinct facial characteristics, such as the arrangement of facial landmarks, overall facial structure, and skin texture, which can be used for identification purposes. Face recognition systems have gained popularity in various sectors, including security, law enforcement, and personal device authentication, due to their non-intrusive nature and potential for high accuracy[2].

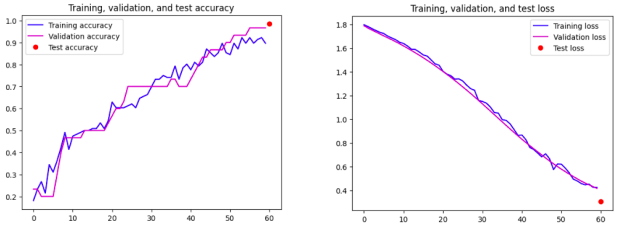

Fig. 7. Training and validation tests loss and accurancy.

- In contrast, face detection is the process of locating and identifying the presence of faces within an image or video frame, without necessarily identifying specific individuals. Face detection algorithms typically focus on identifying regions of an image that contain faces based on visual cues such as color, texture, and geometric patterns. Unlike face recognition, which involves matching detected faces to known individuals, face detection is primarily concerned with detecting the presence and location of faces for further analysis or processing.[11] While both face recognition and face detection are important components of computer vision systems, they serve different purposes and employ distinct algorithms and techniques.
- As the classifiers were indicated directly in the assignment, **logistic regression, SVM, kNN** and **neural network** were chosen. First 3 of those classifiers were studied in Data Science unit, which made the training even more fascinating. In our case, the logistic regression classifier is the most accurative one and the most rapid one is kNN, therefore its lack in precision. **Logistic Regression Accuracy: 94% , Neural Network Accuracy: 98% , kNN Speed (seconds): 0.0049 , Neural Network Speed (seconds): 0.00055**.

E. Personal dataset

- To create personal database, photos of my friends and family were taken in different angles,situations and lightnings. Personal database is larger than the original data,therefore its size is not excessive, with 10 persons and approximately 50 photos of each member, which results in 513 photos in total. Due to high level performance of the encoded faces model, I have decide to continue using this model, which includes encoding the image into one vector of the size 128.

- The predefined model of face encoding is working on the new personalized dataset.As the **datasize increased**, its performance to detect and reckognize faces has improved. However, its comparable huge size of 1.27Go made my system processor suffer a little bit.

Conclusion

This work focused on developing and evaluating a facial recognition system using deep learning techniques. I began by downloading an existing dataset of facial images and preprocessing them to ensure consistency and compatibility with the neural network model through different steps such as extracting faces, aligned them and encoding into a single vector.Subsequently, we trained multiple classifiers, including logistic regression, support vector machine (SVM), k-nearest neighbors (kNN), and neural networks, on the encoded faces. We evaluated the performance of each classifier on the test set in terms of accuracy and speed. Notably, the neural network classifier outperformed the others in terms of accuracy, making it the best classifier for facial recognition tasks. Additionally, we encountered challenges such as limitations in Google Colab GPU resources, misconceptions regarding variable types, and other technical issues related to model complexity, all of which required careful tuning and optimization. Overall, completing this work was an enlightening experience that provided a deeper understanding of facial recognition systems. It underscored the importance of selecting appropriate models and optimizing them for real-world applications.As an added bonus, I discovered how Overleaf works and completed the PDF file using its software. The discovery of functioning of the neural network was the most fascinating part of the work. I honestly enjoyed completing this assignment.

References

[1]Taigman, Y. et al. (2014) Deepface: Closing the gap to human-level performance, Toronto Edu. Available at: https://www.cs.toronto.edu/ ranzato/publications/taigman_cvpr14.pdf (Accessed: 28 April 2024). [2] Jevtić, D. and Slavkovic, M. (2012) Face recognition using Eigenface approach, Research Gate. Available at: https://www.researchgate.net/publication/260880521_Face_Recognition_Using_Eigenface_Approach (Accessed: 28 April 2024). [3]Heisele, B. et al. (2003) Hierarchical classification and feature reduction for fast face detection with support vector machines, Science Direct. Available at: https://www.sciencedirect.com/science/article/pii/S0031320303000621(Accessed: 02 May 2024).

[4]Gholamy, A., Kreinovich, V. and Kosheleva, O. (2018) Why 70/30 or 80/20 Relation Between Training and Testing Sets: A Pedagogical Explanation, Scholar Works Utep. Available at: https://scholarworks.utep.edu/cgi/viewcontent.cgi?article=2202&context=cs_techrep (Accessed: 20 April 2024).

[5]He, K. et al. (2015) Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification, Computer Vision Foundation. Available at: https://www.cv-foundation.org/openaccess/content_iccv_2015/papers/He_Delving_Deep_i (Accessed: 23 April 2024).

[6]Croce, F., Andriushchenko, M. and Hein, M. (2018) Provable robustness of RELU networks via maximization of Linear Regions, Proceedings. Available at: http://proceedings.mlr.press/v89/croce19a/croce19a.pdf (Accessed: 17 April 2024).

[7]Zhu, X. and Ramanan, D. (2012) Face detection, pose estimation, and landmark localization in the Wild — IEEE Conference publication — IEEE Xplore, Vision. Available at: https://ieeexplore.ieee.org/document/6248014/ (Accessed: 27 April 2024).

[8]Najman, L. (2021) TP7_FaceRecognition,Google colab.Available at: https://colab.research.google.com/drive/1FROdNGFv3LRS z20MRO (Accessed: 04 April 2024).

[9]CM, B. (2023) Face recognition in Python: A comprehensive guide, Medium. Available at:https://basilchackomathew.medium.com/face-recognition-in-python-a-comprehensive-guide-960a48436d0f#: :text=Face%20recognition%20systems%20differentiate%2 (Accessed: 04 May 2024).

[10]Dong, Q., Sun, J. and Hu, Z. (2019) Face representation by Deep Learning: A linear encoding in a parameter space?, arXiv.org. Available at: https://arxiv.org/abs/1910.09768 (Accessed: 04 May 2024).

[11]Colmenarez, A.J. and Huang, T.S. (1998) Face detection and recognition, SpringerLink.Available at: https://link.springer.com/chapter/10.1007/978-3-642-72201-1_9 (Accessed: 28 April 2024).