Face Recognition Challenge 2024

1. Introduction

Our project centers on creating a Python-based automatic face recognition system to perform a challenge based on a dataset of 1200 images. These images encompass individuals identified by 80 unique numbers (labeled as "users"), other individuals categorized as "impostors," and images lacking facial features. In addition to machine learning techniques learned in the Face and Gesture Analysis course (like the Viola-Jones algorithm), to enhance accuracy we employ various deep learning techniques, including a model inspired by the famous VGG. Additionally, multiple data augmentation strategies were utilized to refine the system's performance. Our goal is to improve our model's capabilities to classify the images into our classes.

2. Concepts Explanation

Data augmentation techniques [1] are essential in deep learning tasks, especially when faced with limited training data. By increasing the diversity of the training dataset, data augmentation helps prevent overfitting and improves the generalization ability of the model. This augmentation can be both dynamic, when applying mathematical transformations to the input images, and status, when new data similar to the existing one is added to the dataset,

The VGG convolutional neural network [2] architecture is highly regarded for its effectiveness in image classification tasks. Developed by the Visual Geometry Group (VGG) at the University of Oxford, this model is known for its deep structure but small convolution filters followed by max-pooling layers. As its main focus was in classification problems, it is a good fit to use as a basis for this challenge’s model.

3. Practical Procedure

3.1 Data Analysis

Our procedure started with an in-depth analysis of the dataset provided by the course, comprising 1200 images classified in the following three classes:

- Images from 80 people identified by a number, which we will refer to as “users”.
- Images from other people, different from the 80 users that we will refer to as “impostors”.
- Images without a face.

Upon examining the dataset, we encountered a significant imbalance, particularly in the presence of "unknown" or "unidentified" images (those labeled as -1). Specifically, we observed:

- Identified faces: 480 images
- Unknown (label -1): 720 images

This motivates the neural network to prioritize learning the “-1” label, as correctly predicting that value represents an accuracy of 60% (and a first training of our model with this data confirmed this). Additionally, most of the identities have lackbuster data (4 images of low quality, for example). We decided that it was needed to increase or replace the dataset to achieve better results.

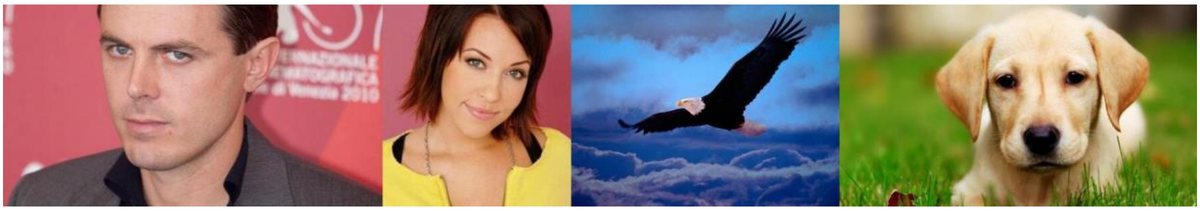

Figure 1: Example images from dataset provided by the course.

3.2 Data Augmentation

Recognizing the insufficiency of data for many identities, with most having only four images, we opted to augment the dataset by downloading additional images from Google for each identity. To prepare the images for subsequent processing and training, we applied the Viola Jones algorithm (which we fine-tuned during Lab 1) to accurately crop the input images to solely include faces.

But another problem arises, as we were not taking into account the class representing unrecognized identities. To further diversify the dataset, we introduced fake face images sourced from "thispersondoesnotexist.com" and generic images (representing objects or activities without faces) to the "-1" label. Following this, we also run the Viola-Jones algorithm on this data, to get a good representation of what the neural network will be receiving as input. This step aimed to enable the Convolutional Neural Network (CNN) to independently predict unknown identities.

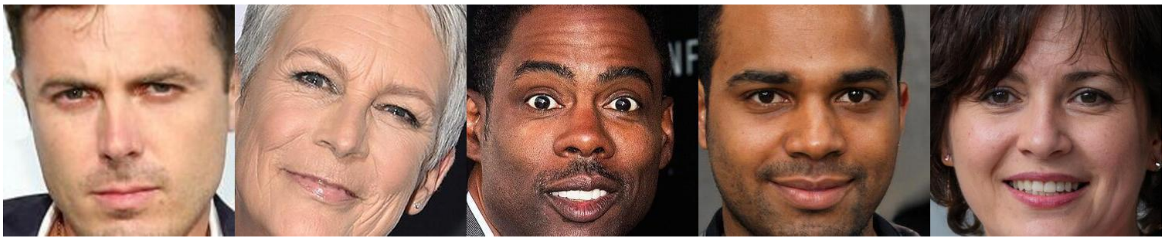

Figure 2: Example images from our augmented dataset

Furthermore, to ensure the integrity of our testing process, we removed the images from the TRAINING directory provided by the course. These excluded images will be reserved for unseen testing using the challenge script, to get a better representation of our real accuracy for the challenge.

Additionally, we applied dynamic transformations to the dataset for training, to further increase our performance. The transformations were: random rotation in the [-30, 30] degrees range, Gaussian blur, resizing the image to a lower resolution, changing its brightness and contrast, and randomly cropping a section of it. As a result, the training dataset now comprises 20688 images, while the validation dataset contains 862 images as transformations are not applied to them.

3.3 Defining the Model

Our model, developed using PyTorch, is a simplified version inspired by the VGG architecture, which we called “VGGSimple”. We used VGG as the basis as it was specifically designed for image classification tasks across a diverse range of different output classes. It uses six convolutional layers with kernel size 3, followed by max pooling and dropout layers in between some convolutions, to mitigate overfitting. The model culminates in two fully connected layers responsible for classifying images, producing outputs for each of the 81 classes. The total number of parameters in the model amounts to **996,017 parameters** with **8 layers** in total (not counting dropout and max pooling).

We would also like to mention that we tested other types of models, utilizing techniques as separable convolutions and bottleneck layers, and with various numbers of convolutional layers and filters, but this “VGGSimple” model (after 5 iterations from our base design) provided the best results.

3.4 Training the model

The training process involves optimizing our model across multiple epochs and mini-batches of training data. It's important to note that the data used for training is the augmented dataset we created, which utilizes external images obtained through data augmentation techniques. Within each epoch, the model undergoes a series of forward and backward passes, where predictions are compared against ground truth labels to compute the loss. This loss is then used to update the model's parameters via gradient descent optimization, utilizing the specified optimizer. For this project, we employed the Stochastic Gradient Descent (SGD) optimizer [3] with a learning rate of 0.01, momentum of 0.9, and weight decay of 0.001, during 25 training epochs. Additionally, at the end of each epoch, the model's performance is evaluated on the separate validation dataset to assess its generalization ability; this evaluation includes calculating both the validation loss and accuracy. The state dictionary of the model is saved after each epoch iteration, so we can apply early stopping if needed utilizing the best weights we obtained.

3.5. Building the identification method

With the previous steps, we can obtain a CNN capable of identifying our 80 identities or indicating if the identity is unknown, but with the tradeback that it receives as input a mostly perfectly cropped image of a human face. As such, for the final identification method, we add a first step to our input images: detecting and cropping faces using the Viola-Jones algorithm, and passing them to the CNN. If this first part of the method does not detect a face, it returns the label for unidentified images. Finally, to the output of the CNN we can apply the softmax algorithm to obtain probabilities for each class, and utilize a threshold value to decide how big the max probability should be to decide on that identity/class. We also applied a fine tuning of the Viola-Jones parameters and softmax threshold to get the best F1 score possible for the challenge script.

4. Result Analysis

A first iteration of our model was trained on the original training dataset provided by the course. This resulted in a 92.62% accuracy on this dataset using the challenge script, but in reality the model was overfitted to this data. The real accuracy using the testing dataset by the course instructors was closer to 16%.

With the modified model and the improvements we mentioned above, our final evaluation on the validation dataset yielded a **validation accuracy of 95.5%,** on the augmented dataset that we prepared ourselves. We generated two different plots in order to monitor the training steps and performance of our model, and apply early stopping if needed to avoid over-fitting. In the end, we found this last epoch to be the best performance, as when we tried to continue with the training we did not obtain a higher accuracy.

Loss Plot

This plot illustrates the evolution of the training and validation losses over the course of training, computed using the cross-entropy loss function. The x-axis represents the training steps, while the y-axis represents the loss value. A lower loss indicates better model performance.

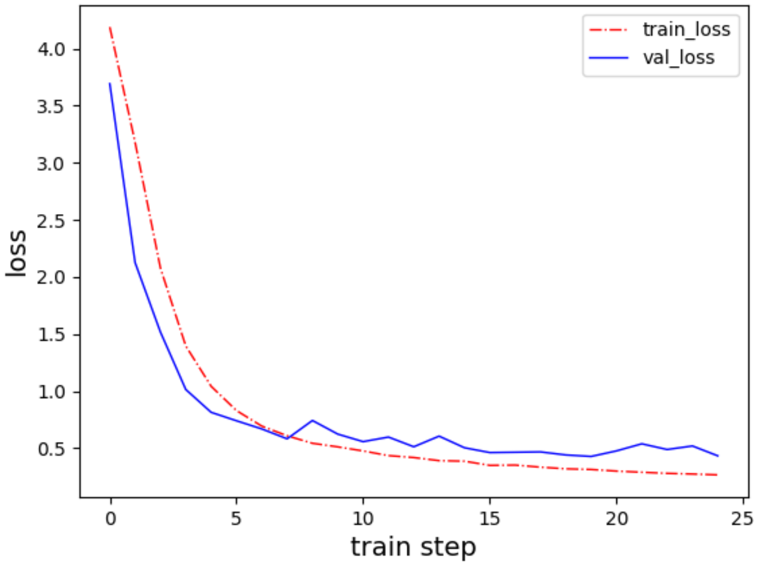

Figure 3: Training and validation loss evolution vs. training epochs

Accuracy Plot

This plot showcases the validation accuracy achieved by the model throughout the training process. We calculated this value using the F1 Score method provided by the course. The x-axis represents the training steps, while the y-axis represents the accuracy value. Higher accuracy values indicate improved model performance.

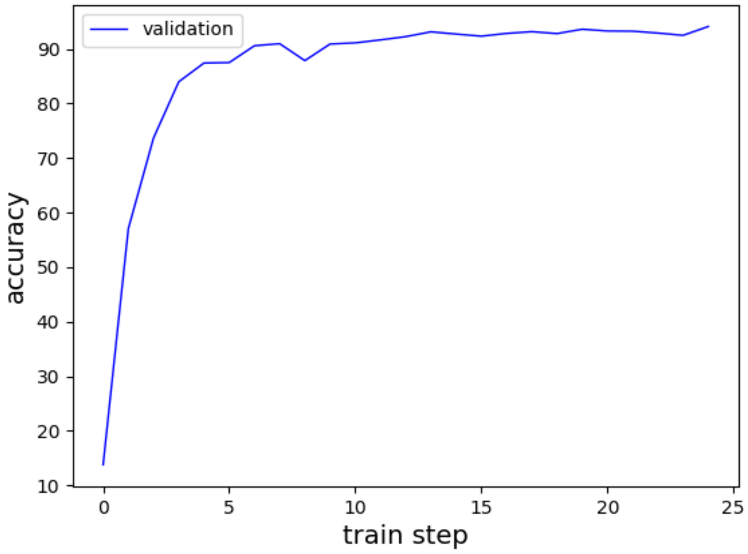

Figure 4: Validation accuracy evolution vs. training epochs.

Challenge Script

With the best CNN we could train, we tested the challenge script provided by the course, which runs on the unbalanced testing data (the original training dataset provided by the course, as our model never learned from these images), and uses Viola-Jones and Softmax to augment the model predictions. We obtained a F1 score of **83.64% accuracy**, in **48.61 seconds**. These values are inside the conditions indicated in the statement of the lab.

As a way to analyze our model's performance, we also counted the number of wrong predictions for our model. Out of 144 wrongly classified images, 32 were false positives (we predicted an identity where there was none) and 88 false negatives (the model filtered known identities as unknown). The rest of the mistakes were 24 wrongly identified celebrities. These numbers show that our model could be improved in the future, for example utilizing a more powerful CNN or an alternative to Viola-Jones for face detection.

Our insight we gathered while testing various combinations to build our method is that the Viola-Jones parameters and the chosen softmax threshold can have a big impact on the performance. We obtained (using the same CNN model) an F1 score as low as 77% versus the high of 83% just by changing these parameters. Additionally, we tested our model without filtering with Viola-Jones first, and obtained an accuracy of only 0.96%. This is expected as the model was trained with cropped face images as input, but this cropping helped to make the model focus on the facial features as a means of identification, and to filter the input images that do not include a human face.

5. Conclusion

These results demonstrate the effectiveness of our trained model in accurately classifying unseen data. The trends observed in the plots validate the learning dynamics of our model, showcasing its ability to learn from the training data and generalize well to unseen instances. We also demonstrated that the aiding of other algorithms aside from the CNN helped with giving accurate predictions

.

6. References

[1] Perez, L., & Wang, J. (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning. arXiv (Cornell University). https://arxiv.org/pdf/1712.04621.pdf

[2] Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR 2015. http://export.arxiv.org/pdf/1409.1556

[3] Wang, L., Yang, Y., Min, M. R., & Chakradhar, S. (2016, 17 March). Accelerating Deep Neural Network Training with Inconsistent Stochastic Gradient Descent. arXiv.org. https://arxiv.org/abs/1603.05544

[4] Universitat Pompeu Fabra. (n.d.). Anàlisi de Gestos i Cares 2023-24389-T1: Theory 6 - Deep learning & Face Recognition. https://aulaglobal.upf.edu/course/view.php?id=61511