Statistical Methods in Artificial Intelligence

Assignment 2

Instructor: Prof CV Jawahar

Deadline: April 21, 2025

April 12, 2025

1 Overview

Face recognition plays a crucial role in modern-day security systems, particularly in smartphone authentication. Unlocking a phone using facial biometrics offers a neat user experience, eliminating the need for passwords or fingerprint scans. It relies on accurate, real-time detection and classification of a user's face under varying environmental conditions.
For this assignment, you will perform:

- **Part 1: Face Recognition (Binary Classification)**
- **Part 2: Emotion Recognition (Multiclass Classification)**

All experiments will be conducted on **Kaggle**, which provides GPU runtimes and limited session time. You will explore various model architectures and compare their performance.

1.1 General Instructions:

- You will be using the same dataset for both the tasks. Please go through the entire assignment once before you start the data collection.
- All the metrics and plots during training and evaluation are to be logged into **wandb**.
- You can check out the resources provided at the end of the document before you begin with the implementation.

2 Kaggle Constraints and Setup

- Kaggle offers a GPU (e.g., Tesla P100 or T4) with about 16GB memory.
- If running into CUDA Out-Of-Memory issues, consider having:

  - Lower batch sizes.
  - Possible freezing of some layers to reduce memory usage.
  - Proper usage of memory, loading only one model at a time instead of multiple together at the run-time. Delete unused variables, make use of the garbage collector properly.

- Session time can be 9–12 hours. Plan your epochs carefully.

3 Part 1: Face Recognition (50 Marks)

3.1 Task Definition

You will be training a binary classifier:

- Label 1: Your face
- Label 0: Not your face.

3.2 Data Collection (10 Marks)

- Collect a balanced dataset:
  - Multiple images of your own face in multiple lighting conditions, environment changes, slight occlusions, etc. Specifically, include the following:
    1. **Bright/Natural Light**: Outdoors in sunlight or near a window during the day.
    2. **Dim/Low Light**: Indoors with minimal lighting.
    3. **Plain Background**: e.g., a white wall or curtain.
    4. **Cluttered Room**: Indoors with furniture or other objects in view.
    5. **Partial Occlusion**: Hand over face, object like a phone etc.

    Optional:

    1. **Hair Partially Covering Face**
    2. **Different Hairstyles or Headgear**: Cap, hoodie, tied hair vs open.
  - Random/other faces to represent the “not your face” class. You may choose to collaborate with your friends to collect samples of this label (encouraged), or you may collect it from the Internet as well.

- Ensure your annotations for the images are correct, create two folders within your data folder: one for ”your face” class and one for the other class.

- Create a Dataset class which loads data and performs necessary augmentations (flips, rotations, color jitter, etc.) to ensure diversity. Please note that this is a crucial aspect if you want to train a robust model which can perform on challenging test cases as well. Experiment with different techniques and include a comparison in your report.

- Keep a separate test set with challenging cases (low light, heavy occlusions). Ensure that there are enough challenging cases in your test set as we will be using this as a metric as well to judge the performance of your models. Masks will be awarded based on how well the test images capture diversity and different environmental conditions, and how well the model performs on the test.

- You will be using this dataset for the Task-2 as well, so ensure you capture faces with varied expressions/emotions in the dataset (sad, angry, neutral, happy etc.)

3.3 Models to Compare (20 Marks)

Your implementation will include three models:

1. **VGGFace (Finetuning)**:
   - Load the pretrained VGGFace network.
   - Replace the final fully connected layer with a new layer of output size 2 for binary classification.
   - Optionally freeze early convolutional layers and finetune only the later layers and the new classifier.

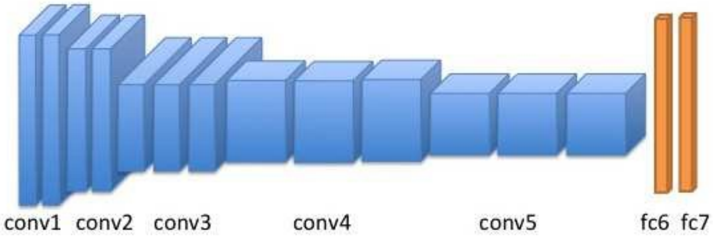

Figure 1: VGGFace architecture

2. **ResNet18 (From Scratch)**:
   - Initialize ResNet18 with random weights.
   - Replace the final fully connected layer (usually `model.fc`) with a new one of output dimension 2.
   - Train the entire network from scratch.

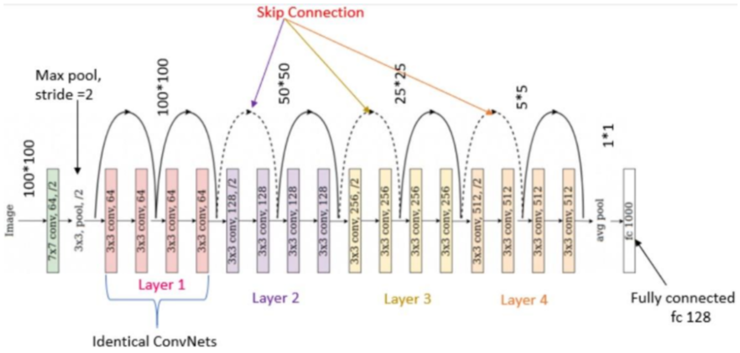

Figure 2: ResNet architecture

3. **ResNet18 (Pretrained on ImageNet)**:
   - Load a ResNet18 model pretrained on ImageNet.
   - Replace the final fully connected layer with a new layer of output dimension 2.
   - Train the network for the custom dataset.

Compare convergence, accuracy, and resource usage. Highlight performance qualitatively and quantitatively as well in your report. Document the challenges faced while training the models on your collected data.

Ensure that model performance is robust to detecting your face in different environments as well as with varied expressions.

3.4 Evaluation (10 Marks)

- Report Accuracy, Precision, Recall, F1-score on a dedicated test set.
- Do qualitative analysis of the models as well and see if there exists a consistent pattern in performance in certain conditions.
- Create a video of your face frames and corresponding predicted labels. Ensure that you test it on all the different conditions mentioned above as this will be primarily used for qualitative analysis of your model performance during evaluations.

3.5 Report (10 Marks)

- Plot train and validation curves for loss over the epochs for each of the three models.
- Summarize your results and insights. Also include the challenges faced while training the different models on the custom dataset.

3.6 Unlock Simulation (Scaffolding) Bonus (5 Marks)

- Demonstrate a naive phone-lock simulation within Kaggle.
- E.g., Create a video from different test images. Loop over the frames and predict 0/1; display "Unlocked" (for your own face) or "Locked" (for any other face).

4 Part 2: Emotion Recognition (50 Marks)

4.1 Task Definition

Extend face recognition to a **k-way** classifier for emotions (e.g., happy, sad, angry, neutral). Ensure that there are atleast 3 emotion classes - **happy**, **sad** and any other of your choice. Having $k < 3$ would make your submission for this part invalid.

4.2 Data (10 Marks)

- Use the same face images from Part 1, but label each image with an emotion category. It is sufficient to use just your own face images for this. However, you are welcome to try it on other faces as well.
- Use the same augmentations as defined in Part 1. Please, ensure that you label the images appropriately. Feel free to discard images that do not encode the emotions from the categories you have defined.

4.3 Model Architecture (20 Marks)

For all three model variants, adapt the final classification layer as follows:

- Replace the binary classifier (2 output neurons) with a new fully connected layer that outputs **k** neurons, where **k** is the number of emotion classes.
- The same backbone (either VGGFace finetuning, ResNet18 from scratch, or ResNet18 pretrained) is used to extract features.

Ensure that the performance of the model is robust in correctly detecting emotions from your face under different surrounding conditions.

4.4 Evaluation (10 Marks)

- Provide accuracy per emotion class.
- Present a confusion matrix to highlight any misclassifications.

4.5 Report (10 Marks)

- Show training curves (loss vs. epoch, accuracy vs. epoch).
- Summarize your results and insights. Also include the challenges faced while training the different models on the custom dataset.

4.6 Creative Element Bonus (5 Marks)

- If an emotion is detected (say "happy"), display a snippet or a relevant message.
- On Kaggle, you could link to a short audio track or embed a text-based reaction.

5 Resources

- VGGFace Oxford
- Transfer Learning: Finetuning
- Freezing layers in PyTorch
- Loading Pretrained models in PyTorch
- ResNet Original Paper
- ResNet Architecture Explained!
- Wandb Logging

Good luck with the assignment!