Face Classification using Extreme Learning Machine and Deep Learning

Nurzhan Amanbayev, Danial Shokobalinov

Nazarbayev University

Abstract

This project explores a hybrid approach to face recognition by combining a deep convolutional neural network (InceptionResnetV1 pretrained on VGGFace2) for feature extraction with a shallow, non-iterative learning algorithm — the Extreme Learning Machine (ELM) — for classification. We utilized the Labeled Faces in the Wild (LFW) dataset, supplemented with custom images, to train and evaluate our model. The system achieves high classification accuracy while maintaining real-time inference speed, including live webcam predictions. The approach highlights how deep features extracted from a pretrained model can be effectively used by a simple, fast-learning classifier, offering an innovative and computationally efficient alternative to end-to-end deep learning.

1 Introduction

Face recognition is a foundational task in computer vision with widespread applications in security systems, user authentication, and human-computer interaction. Over the past decade, deep learning models—particularly Convolutional Neural Networks (CNNs)—have significantly advanced the field by enabling the extraction of high-level, discriminative facial features.

1.1 Difference and Innovation.

Notable models such as `FaceNet` and `VGGFace2` have achieved state-of-the-art performance, but they typically require large training datasets, extensive computational resources, and time-consuming iterative training. Recent work has explored more efficient alternatives, such as using pretrained networks for feature extraction in combination with simpler classifiers. In this context, the `Extreme Learning Machine (ELM)` offers a promising solution due to its non-iterative learning mechanism and rapid training. This project builds upon these developments by proposing a hybrid architecture that leverages the `InceptionResnetV1` model pretrained on `VGGFace2` for feature extraction and an ELM for fast and scalable face classification. The approach aims to balance high accuracy with computational efficiency, particularly in scenarios where training time and hardware resources are limited.

2 Methodology

This project implements a modular face classification pipeline by combining a pretrained deep convolutional neural network `(InceptionResnetV1)` for feature extraction with an `Extreme Learning Machine (ELM)[3]` for classification. The methodology consists of four major stages: data preparation, deep feature extraction, ELM training, and evaluation.

Nurzhan Amanbayev, Danial Shokobalinov. “Face Classification using Extreme Learning Machine and Deep Learning.”.

2025

2.1 Data Collection and Preprocessing

We used the Labeled Faces in the Wild (LFW) [2] dataset, which contains over 13,000 grayscale facial images of various individuals. To increase the diversity of identities in the dataset, a custom class was added using the ImageFolder interface from PyTorch. These custom images are assigned a new label index and merged with the original dataset using ConcatDataset. All images are resized to 224×224 pixels, converted to RGB, and normalized using standard ImageNet pre-processing. A stratified 80/20 split is applied to create training and test sets, and the resulting sets are wrapped in PyTorch DataLoader objects for batch processing.

2.2 Deep Feature Extraction using InceptionResnetV1

We utilized the InceptionResnetV1 model from the facenet-pytorch library, pre-trained on the VGGFace2 [1] dataset. This model is widely used for face recognition tasks due to its robust performance and compact embeddings. The model is used in evaluation mode, and its final classification layer is replaced with an identity layer, allowing us to extract 512-dimensional face embeddings from the penultimate layer. These embeddings serve as high-level feature vectors that represent input faces in a semantic space. To extract features, all images from both training and testing sets are passed through the frozen network and the resulting embeddings are stored along with their corresponding labels.

2.3 Classification with Extreme Learning Machine (ELM)

In this project, we utilize the Extreme Learning Machine (ELM) as the core classifier for face recognition. ELM is a type of single-layer feedforward neural network (SLFN) known for its remarkable training speed and simplified learning mechanism, especially when compared to traditional deep learning models. Unlike conventional neural networks that rely on iterative gradient-based optimization (e.g., backpropagation), ELM requires no weight updates in the hidden layer and instead solves a least-squares problem analytically.

Network Architecture. The ELM used in our pipeline consists of the following components:

1) Input Layer: Takes in the feature vectors (embeddings) generated by the pretrained InceptionResNetV1 model from the facenet-pytorch library. Each input vector is of fixed length and encodes discriminative facial features. Random initialization of weights and biases between the input and hidden layer.

2) Hidden Layer: A large number (100,000 in our setup) of hidden neurons with randomly initialized weights and biases. These parameters remain fixed throughout training. A nonlinear activation function (sigmoid or ReLU) is applied to produce hidden activations. In our case, sigmoid function is implemented:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

3) Output Layer: Learns a weight matrix that maps the hidden layer output to the final class predictions. This is the only part of the model that is trained, and it is done using a closed-form solution. A closed-form solution to determine the output weights using the Moore–Penrose pseudoinverse of the hidden layer activations:

$$A^{+} = (A^{\top} A)^{-1} A^{\top}$$

We set the hidden layer size to 100,000 neurons, providing sufficient capacity to learn nonlinear mappings from embeddings to identity labels. The output layer uses one-hot encoded vectors corresponding to the class labels.

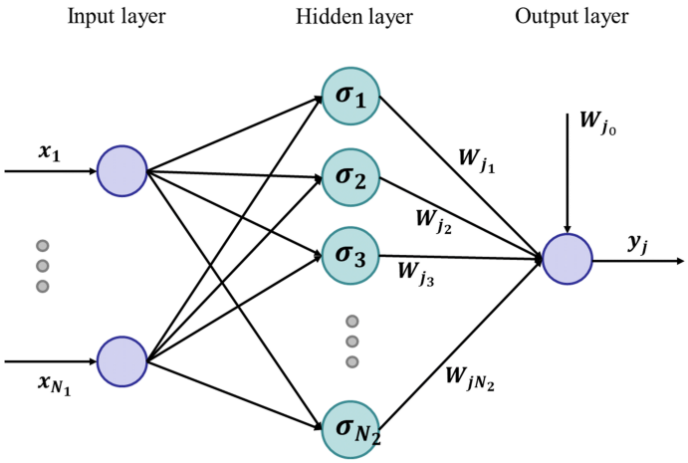

Figure 1: The architecture of the Extreme Learning Machine (ELM).

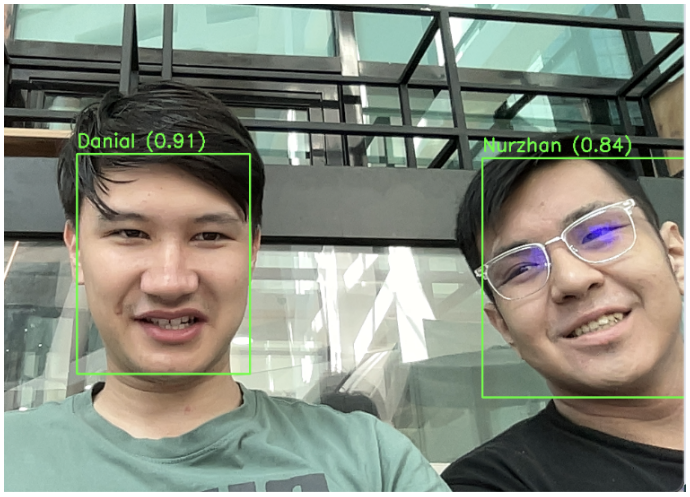

Figure 2: The real-time inference using OpenCV

2.4 Prediction and Top-K Retrieval

Once trained, the ELM model is used to predict labels for test samples. In addition to hard predictions, the model supports:

1) Top-1 classification with raw confidence values

2) Top-K identity suggestions, based on the sorted output of the ELM's final layer

These are used both for evaluating classification accuracy and for retrieving visually similar identities.

2.5 Real-Time Webcam Inference

The final stage involves integrating the model into a real-time face recognition system using a webcam feed. Faces are detected in each frame using MTCNN (Multi-task Cascaded Convolutional Networks) [4], cropped, transformed, and passed through the InceptionResnetV1 model to generate embeddings. These embeddings are then classified using the trained ELM. The predicted identity and confidence score are overlayed onto the video feed using OpenCV, allowing for interactive and continuous facial recognition.

3 Numerical Experiments

This section presents the experimental setup, evaluation procedure, and the results obtained from applying our face classification pipeline on the LFW dataset with additional custom data.

3.1 Experimental Setup

- Training setup: We used the following setup for the training: GTX 3050 GPU, 16 GB RAM and Apple Silicon based CPU setup with 8 GB of RAM.
- Dataset: We used the Labeled Faces in the Wild (LFW) dataset, filtered to include people with at least 25 images. To test real-world personalization, we also added a custom folders of personal face images, assigned to a new classes.
- Data split: 80% for training and 20% for testing.
- Image preprocessing: All images were resized to 224×224 pixels, converted to 3-channel RGB if grayscale, and normalized to [-1, 1] per channel.
- OpenCV's Haar cascade classifier was used to detect and extract the face from each image in our custom dataset, ensuring consistency in facial region input. The images were loaded using PyTorch's ImageFolder, and each was preprocessed to isolate the facial area. Labels were adjusted to align with the existing class structure, and the class list was updated accordingly.

3.2 Feature Extraction

We used a pretrained InceptionResnetV1 from the facenet-pytorch library (trained on VGGFace2) to extract high-dimensional embeddings. Output feature size: 512-dimensional embedding vector per image. ResNet weights were frozen to avoid backpropagation and retain pretrained identity-representative features.

3.3 Training the ELM Classifier

- **Input dimension**: 512
- **Hidden units**: 100,000
- **Activation function**: Sigmoid
- **Training time**: 17 seconds on CPU due to the non-iterative closed-form solution of ELM.

The ELM was trained using the pseudoinverse of the hidden layer activations to solve for output weights. One-hot encoding was used for labels.

3.4 Evaluation Metrics

We evaluated our model using:

- **Accuracy**: Fraction of correctly classified test images.
- **F1-Score**: The macro F1-score was calculated across all classes, treating each class equally regardless of the number of samples. This provides a better sense of overall performance, especially in datasets with class imbalance.

3.5 Results

- **Accuracy**: 99.00%
- **F1-Score**: 0.99 (both macro and weighted)

These results show that ELM, when paired with strong feature embeddings from InceptionResnetV1, can achieve competitive accuracy with minimal training time and no backpropagation. Additionally, the top-k face ranking functionality makes the model suitable for face retrieval or similarity-based applications.

3.6 Comparison

During the comparison of ELM-based InceptionResnetV1 and InceptionResnetV1 with its own Dense layer, we can see the improvement in the results. Here are the results of basic pretrained InceptionResnetV1 with its own results:

- **Accuracy**: 91.00%
- **F1-Score**: 0.90

3.7 Discussion

The results demonstrate that the proposed pipeline, combining a fixed deep feature extractor (InceptionResnetV1) with a non-iterative ELM classifier, achieves strong performance on the face classification task. With a test accuracy of 99% and a macro F1-score of 0.99, the system shows the potential for accurate and efficient recognition.

One key advantage is the training speed of the ELM model, which is 17 seconds. Unlike traditional deep networks that require iterative optimization, the ELM solves the weights in a single-matrix operation, making it highly suitable for rapid deployment and real-time applications.

However, during experimentation, it was observed that each class needs at least around 25 images to be classified reliably. When fewer samples were provided for a class (e.g., custom-added identities), the classifier struggled to learn meaningful representations, even when supported by strong features. This highlights a limitation of ELM in low-data regimes, where other methods like fine-tuning or few-shot learning might perform better.

Overall, the method is highly effective in balanced and moderately-sized datasets and offers an excellent trade-off between speed and accuracy for face recognition systems.

4 Conclusion

This project introduced a face classification pipeline that combines deep feature extraction via a pretrained InceptionResnetV1 with a fast and scalable Extreme Learning Machine (ELM) classifier. The model achieves strong performance on the LFW dataset, reaching 17 seconds of training time, 99% accuracy and a macro F1-score of 0.99, while significantly reducing training time due to ELM's closed-form solution.

Our results demonstrate that deep embeddings, when paired with ELM, offer an effective balance between accuracy and computational efficiency. However, we observed that the model requires at least 25 samples per class to generalize well, indicating limitations in low-data regimes.

Overall, the proposed approach offers a compelling alternative for face recognition tasks, particularly in settings where fast training and deployment are critical. Future work may explore methods to improve robustness under class imbalance and extend applicability to few-shot learning scenarios.

References

[1] Cao, Qiong et al. “VGGFace2: A dataset for recognising faces across pose and age.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence 41*., no. 8 (2018), pp. 1962–1974. DOI: 10.1109/TPAMI.2018.2858254. URL: https://ieeexplore.ieee.org/document/8401875.

[2] Huang, Gary B. et al. “Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments.” In: , no. 07-49 (2007). URL: http://vis-www.cs.umass.edu/lfw/lfw.pdf.

[3] Huang, Guang-Bin, Zhu, Qin-Yu, and Siew, Chee-Kheong. “Extreme learning machine: Theory and applications.” In: *Neurocomputing 70*., no. 1-3 (2006), pp. 489–501. DOI: 10.1016/j.neucom.2005.12.126. URL: https://www.sciencedirect.com/science/article/pii/S0925231205003915.

[4] Zhang, Kaipeng et al. “Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks.” In: *IEEE Signal Processing Letters 23*., no. 10 (2016), pp. 1499–1503. DOI: 10.1109/LSP.2016.2603342. URL: https://ieeexplore.ieee.org/document/7553523.