Multi-Face Recognition Based on Convolutional Neural Networks

Tingting Zhu, Zhou Li

School of Computer and Software, Jincheng College of Sichuan University, Chengdu 611731, Sichuan, China

Abstract: ***With the continuous advancement of technology, facial recognition is being applied more and more widely in this era of big data. Techniques based on deep-learning convolutional neural networks are increasingly recognized and adopted. Training facial recognition with convolutional neural networks does not require complex feature extraction; it only needs to use the OpenCV library to detect faces and then employ a suitable network model for automatic training to achieve good recognition performance.***

Keywords: Facial recognition OpenCV convolutional neural network.

1. INTRODUCTION

Facial recognition is extensively used in various fields, such as attendance systems, social networking, payment verification, suspect tracking, and transportation check-ins. At a macro level, facial recognition is divided into two categories: one is face detection, which identifies the position and size of faces in an image; the other is face recognition, which compares the detected face with images in an existing database to determine whether it is the same person, thereby completing identity verification.

There are many facial-recognition algorithms. Traditional methods rely on manually designed features and some machine-learning algorithms, such as extracting edges, textures, lines, and boundaries from images and then processing these features further. Such approaches are inefficient. Therefore, methods involving manual feature extraction and machine-learning techniques have been largely replaced by the widely used convolutional neural networks for training data.

The characteristics and advantages of deep-learning neural-network algorithms lie in their ability to train on a relatively large dataset and learn its surface features. In addition to facial recognition, CNNs are also widely applied to facial-expression recognition, object recognition, age analysis, and other areas.

Ding and Wu (2024), who systematically reviewed self-supervised learning for ECG/PPG signal processing [1], while Restrepo et al. (2024) developed multimodal deep learning with embedding alignment for low-resource healthcare [2]. In interactive systems, Xie and Chen (2025) created InVis for human-centered data interpretation [3], and Zhu (2025) introduced RAID for reliability automation in ad systems [4]. Developer tooling innovations include Zhang's (2025) InfraMLForge for LLM deployment [5] and Hu's (2025) GenPlayAds for procedural 3D ad generation [6]. Healthcare applications feature Qin et al. (2025) optimizing deep learning against ALS progression [7], complemented by Weng et al. (2025) proposing SafeGen-X for LLM security [8]. LLM research evolves through Zhao et al. (2025) with KET-GPT's knowledge updating [9], while Li et al. (2025) fused Vision Transformers and LLMs in MLIF-Net for image detection [10]. Foundational data science includes Chen (2023) applying data mining [11], Chen et al. (2024) contributing the Bimcv-R CT retrieval dataset [12], and Sun et al. (2025) building AutoML frameworks on LLMs [13]. Fintech applications feature Pal et al. (2025) implementing AI credit risk assessment [14]. Seminal works provide critical foundations: Koutrintzes et al. (2022) pioneered multimodal activity recognition [15]; Plexousakis (2005) analyzed recommendation algorithms [16]; Schwegler and Challacombe (1996) advanced quantum chemistry computations [17]; van den Brink et al. (1996) studied ecological impacts of insecticides [18]; Karp and Paley (1996) integrated biological data access [19]; Theologos et al. (1997) simulated catalytic reactors [20]; Brazier et al. (1996) formalized cooperation models [21]; Martin et al. (1987) explored parsing algorithms [22]; Feng and Mizrach (undated) developed financial risk models [23]; and Xing et al. (2006) investigated plasma physics [24].

2. DATA COLLECTION

Facial-recognition data are collected through electronic devices such as cameras. This paper uses 33 students from

a class; each person provides 10 color images captured by a camera using OpenCV + Haar features. Every photo has different expressions, poses, positions, and lighting conditions. They are placed in the same directory and uniformly named name_00.TIF to name_09.TIF.

Figure 1: Facial-image data collection

3. DATA PREPROCESSING

3.1 Face Detection Based on OpenCV

OpenCV is a cross-platform computer-vision library capable of performing a large number of image-processing tasks [1] and achieves high accuracy in face detection.

We use OpenCV's Face Detector to obtain faces from video. For face detection, the Haar-feature classifier can be employed; among these classifiers are those for glasses, head, mouth, nose, and other regions. In this paper, the haarcascade_frontalface_default.xml classifier is adopted.

3.2 Image Resizing and Data Augmentation

3.2.1 Resizing

Haar features are used to uniformly extract the face region from each photo and resize it to 60 × 60 pixels before saving.

3.2.2 Data Augmentation

For some images, the lighting is too dim, causing the face and background to merge, making their pixel values very close; facial deformation itself also hinders computer recognition. Therefore, augmenting the original images can improve task accuracy to some extent. The flip() function in the OpenCV library is used to horizontally flip the original samples, the contrast_brightness_demo() function adjusts brightness, and the dataset is expanded to 1,320 images.

3.3 Labeling and Splitting Training, Validation, and Test Sets

3.3.1 Labeling

Use the imread() function in the OpenCV library to obtain the matrix information of the images, use the get_dummies() function in the pandas library to perform one-hot encoding for labeling, and finally integrate the labels with the image information into DataFrame format for subsequent processing.

3.3.2 Splitting the Training Set

Take 70% of the processed data as the training set, 20% as the test set, and 10% as the validation set. To ensure different samples for each training run and prevent overfitting, use the shuffle function to randomize the training set.

4. A BRIEF DISCUSSION ON CNN

4.1 Introduction to CNN

Convolutional Neural Network (CNN) [2] is a type of neural network that includes convolutional computation and has a deep structure; it is one of the deep learning algorithms.

4.2 Main Structure

4.2.1 Input Layer

In convolutional neural networks for image processing, the input layer is typically a pixel matrix of an image, designed as a three-dimensional structure; for example, 28*28*3,28 is the image size and 3 is the image depth or number of channels—color images usually have R、G、B three channels.

4.2.2 Convolutional Layer

Generally used for feature extraction. A filter, i.e., a convolution kernel or convolutional layer filter, is three-dimensional data whose depth matches that of the input image. By setting a stride, the filter slides across the width and height of the input data, computing the inner product between the filter and any location of the input for each channel. After the convolution kernel's output, an activation function is applied to introduce non-linearity into the network.

4.2.3 Pooling Layer

Reduces matrix size, lowers computational resource consumption, and effectively controls overfitting while improving generalization.

4.2.4 Fully Connected Layer

Connects all local parts through a fully connected layer to perform final classification and obtain the result.

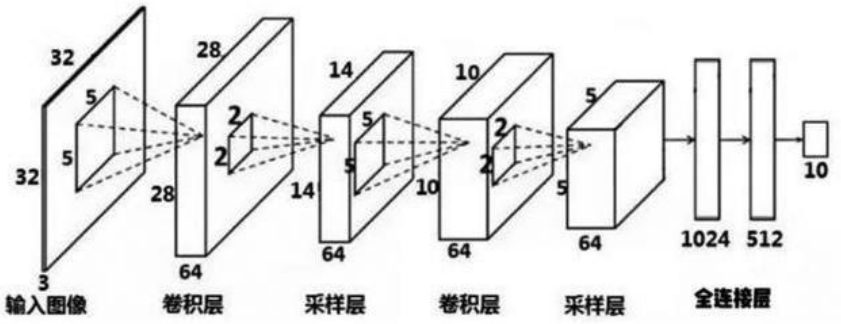

Figure 2: Structure of Convolutional Neural Network

5. BUILDING THE CONVOLUTIONAL NETWORK

Batch value: initial training batch size and the number of samples per batch.

Input layer: x for image information, y for classification information.

Convolutional layer: uses a 3*3 -sized kernel to continuously convolve the image, producing feature maps; from the second layer onward, the filter size becomes the depth of the image after the previous convolution, and the ReLU activation function introduces non-linearity. The stride of the last convolutional layer changes from 1 to 2.

Pooling layer: halves the matrix size.

Fully connected layer: calculates the size and depth of the image with boundary padding, sets multiple neurons in the fully connected layer, feeds in the feature maps obtained from convolution, and the final output layer has 33 neurons.

Softmax layer: uses a softmax classifier to map the output to the 0–1 range, taking the class with the highest score as the correct classification.

After the basic network framework is built and results are obtained, compute the loss, propagate the error backward, update the weights via gradient descent, and iteratively train the network; shuffle the training set in each iteration to enhance model generalization, compare the resulting classification with the validation set to output validation accuracy, and finally output test accuracy on the test set.

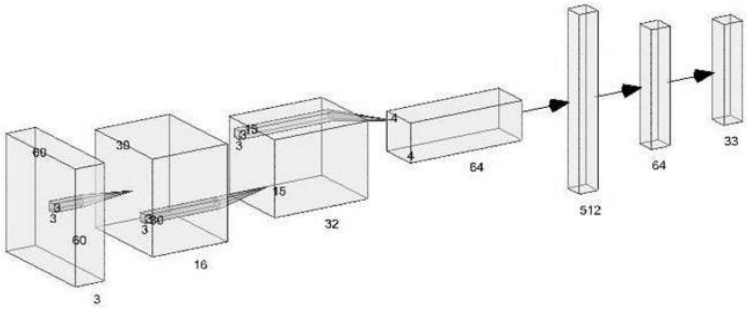

Figure 3: Structure diagram of the convolution process

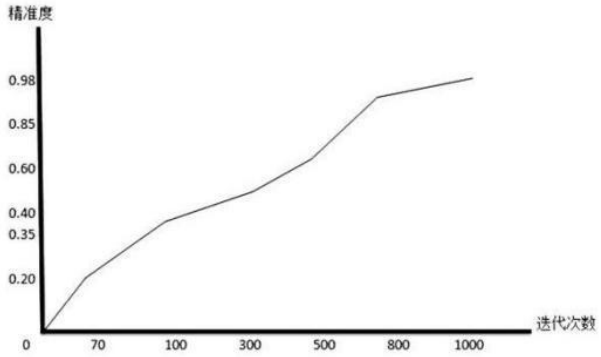

Figure 4: Precision Change Curve

epoch: V30 Valid_acc:0.V0 epoch: 936 valid_acc:0.98 epoch: 938 valid_acc:0.98 epoch: 939 valid_acc:0.98 epoch: 941 valid_acc:0.98 epoch: 943 valid_acc:0.98 epoch: 944 valid_acc:0.98

6. OPTIMIZING THE NETWORK

6.1 Optimizing Training Batches

A small-batch training approach is adopted; the initial batch size is set. Because the training set is not very large, the batch size can be appropriately reduced, yet it should not be too small to avoid excessively slow training. Sizes such as 20, 30, 50, 100, etc., can all be tested to select the optimum. In this paper, the best batch size found through training is 100.

6.2 Optimizing Convolutional Kernel Size, Number, and Layer Count

During convolutional layer training, the height and width of the kernels can be set freely; in this paper, 3×3 is ultimately chosen as the better size. In the first convolutional layer, the kernel depth should match the image channels, i.e., 3, while the number of filters is adjusted as needed. This network uses 16 filters in the first layer, 32 in the second, and 64 in the third. The network can also be tuned by adding or removing convolutional layers; this paper adopts three layers.

6.3 Optimizing the Number of Neurons and Layers in Fully Connected Layers

In the fully connected layers, the number of neurons and the number of layers can be appropriately increased to improve accuracy, but excessively large values will also slow down training. In this network, the first layer is set to 512 neurons and the second layer to 64 neurons.

6.4 Optimizing the Learning Rate

In backpropagation gradient descent, the learning rate—i.e., the step size—is typically set to values like 0.1, 0.01, or 0.001. A value that is too small slows training, while one that is too large may overshoot the optimum; tuning yields the best rate. For this network, the optimal learning rate was found to be 0.001.

6.5 Optimizing the Number of Iterations

In the training network, the accuracy must be computed iteratively multiple times, and the number of iterations must be set manually; too few iterations will yield low accuracy, so a larger number is chosen to optimize the network model. This paper uses 2000 iterations.

7. REAL-TIME FACE RECOGNITION

To achieve real-time face recognition in this project, we also need to use OpenCV library methods to obtain the face data to be recognized in real time.

A convolutional neural network has already been trained to achieve 98% accuracy on the validation set; now this trained model must be used for real-time face prediction. First, OpenCV is used to access the camera and capture the live video stream, then OpenCV's face-detection model is applied within the video window to locate faces and crop the corresponding face images. The cropped faces are pre-processed—for example, resized to the dimensions required by the network. Each processed image is then fed into the network, the model's parameters are used for recognition, yielding a probability distribution over person classes, and finally the name corresponding to the highest probability is returned.

Then use the image-labeling function to display the person's name in the video window.

After multiple tests, the real-time recognition accuracy remains very high, which confirms that the established model has a strong capacity to learn image features and can precisely identify every individual.

8. CONCLUSION

As technology advances ever faster, in this era of big data, facial recognition has brought great convenience to society thanks to its high efficiency and accuracy. Multi-face recognition based on neural networks has strong feature-extraction capability and learning ability: OpenCV is used to detect faces, and the captured face data are then used to train the network, thereby achieving facial recognition. However, because the number of images is too small, the accuracy obtained from the training samples is not high enough; later accuracy will improve as the sample count for each class increases. During network training, overfitting can sometimes occur; the root causes are insufficient data and an overly complex model. Generalization performance can be improved by acquiring more data, reducing the number of network layers, or decreasing the number of neurons to limit the model's capacity to overfit.

REFERENCES

[1] Ding, C.; Wu, C. Self-Supervised Learning for Biomedical Signal Processing: A Systematic Review on ECG and PPG Signals. medRxiv 2024.
[2] D. Restrepo, C. Wu, S.A. Cajas, L.F. Nakayama, L.A. Celi, D.M. López. Multimodal deep learning for low-resource settings: A vector embedding alignment approach for healthcare applications. (2024), 10.1101/2024.06.03.24308401
[3] Xie, Minhui, and Shujian Chen. "InVis: Interactive Neural Visualization System for Human-Centered Data Interpretation." Authorea Preprints (2025).
[4] Zhu, Bingxin. "RAID: Reliability Automation through Intelligent Detection in Large-Scale Ad Systems." (2025).
[5] Zhang, Yuhan. "InfraMLForge: Developer Tooling for Rapid LLM Development and Scalable Deployment." (2025).
[6] Hu, Xiao. "GenPlayAds: Procedural Playable 3D Ad Creation via Generative Model." (2025).

[7] Qin, Haoshen, et al. "Optimizing deep learning models to combat amyotrophic lateral sclerosis (ALS) disease progression." Digital health 11 (2025): 20552076251349719.
[8] Weng, Yijie, et al. "SafeGen-X: A Comprehensive Framework for Enhancing Security, Compliance, and Robustness in Large Language Models." 2025 8th International Conference on Advanced Algorithms and Control Engineering (ICAACE). IEEE, 2025.
[9] Zhao, Shihao, et al. "KET-GPT: A Modular Framework for Precision Knowledge Updates in Pretrained Language Models." 2025 IEEE 6th International Seminar on Artificial Intelligence, Networking and Information Technology (AINIT). IEEE, 2025.
[10] Li, Xuan, et al. "MLIF-Net: Multimodal Fusion of Vision Transformers and Large Language Models for AI Image Detection." 2025 8th International Conference on Advanced Algorithms and Control Engineering (ICAACE). IEEE, 2025.
[11] Chen, Rensi. "The application of data mining in data analysis." International Conference on Mathematics, Modeling, and Computer Science (MMCS2022). Vol. 12625. SPIE, 2023.
[12] Chen, Yinda, et al. "Bimcv-r: A landmark dataset for 3d ct text-image retrieval." International Conference on Medical Image Computing and Computer-Assisted Intervention. Cham: Springer Nature Switzerland, 2024.
[13] Sun, N., Yu, Z., Jiang, N., & Wang, Y. (2025). Construction of Automated Machine Learning (AutoML) Framework Based on Large LanguageModels.
[14] Pal, P. et al. 2025. AI-Based Credit Risk Assessment and Intelligent Matching Mechanism in Supply Chain Finance. Journal of Theory and Practice in Economics and Management. 2, 3 (May 2025), 1–9.
[15] Koutrintzes, D. , Spyrou, E. , Mathe, E. , & Mylonas, P. . (2022). A multimodal fusion approach for human activity recognition. International journal of neural systems, 2350002.
[16] Plexousakis, P. D. . (2005). Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents. Engineering Applications of Artificial Intelligence.
[17] Schwegler, E. , & Challacombe, M. . (1996). Linear scaling computation of the hartree–fock exchange matrix. Journal of Chemical Physics, 105(7), 2726-2734.
[18] Brink, P. J. V. D. , Wijngaarden, R. P. A. V. , Lucassen, W. G. H. , Brock, T. C. M. , & Leeuwangh, P. . (1996). Effects of the insecticide durban 4 e (a.i. chlorpyrifos) in outdoor experimental ditches: ii. community responses and recovery. Environmental Toxicology & Chemistry, 15(7), 1143-1153.
[19] Karp, P. D. , & Paley, S. . (1996). Integrated access to metabolic and genomic data. Journal of Computational Biology, 3(1), 191-212.
[20] Theologos, K. N. , Nikou, I. D. , Lygeros, A. I. , & Markatos, N. C. . (1997). Simulation and design of fluid-catalytic cracking riser-type reactors. AIChE Journal, 43(2), 486-494.
[21] Brazier, F. M. T. , Jonker, C. M. , & Treur, J. . (1996). Formalization of a cooperation model based on joint intentions. Springer, Berlin, Heidelberg.
[22] Martin, W. A. , Church, K. W. , & Patil, R. S. . (1987). Preliminary analysis of a breadth-first parsing algorithm: theoretical and experimental results. Natural Language Parsing Systems, 267-328.
[23] Feng, Y. , & Mizrach, B. . Estimation of Value-at-Risk and Expected Shortfall based on Nonlinear Models of Return Dynamics and Extreme Value Theory.
[24] Xing, Q. , Wang, D. , Huang, F. , & Deng, J. . (2006). Two-dimensional theoretical analysis of the dominant frequency in the inward-emitting coaxial vircator. IEEE Transactions on Plasma Science, 34(3), 584-589.