



Real-Time Facial Recognition System Using Convolutional Neural Networks

Qudsia Leroux-Martin

Department of Computer Science and Engineering, Angkor Mekong Technical University, Cambodia
qudsia.leroux.martin@amtu-kh.edu

Peer Review Information

Submission: 04 Jan 2019
Revision: 20 Jan 2019
Acceptance: 06 Feb 2019

Keywords

*Facial recognition,
Convolutional neural
networks, Face
embeddings, Real-time
vision, Deep learning.*

Abstract

Real-time facial recognition combines image acquisition, face detection and alignment, convolutional neural network (CNN) feature extraction, similarity matching, and identity decision within a strict latency budget. This methodology paper proposes a reproducible design for developing and evaluating a real-time facial recognition system using CNNs, with references limited to 2019 and earlier. Deep learning transformed face recognition by learning identity-discriminative representations directly from large image datasets. DeepFace demonstrated near-human verification performance using deep neural representations, Deep ID showed that identity classification can produce compact face features, and FaceNet introduced metric-learning embeddings in which distance directly represents facial similarity. VGGFace further demonstrated the importance of large-scale data and end-to-end CNN training. For real-time deployment, robust detection and alignment are provided by cascaded CNN detectors such as MTCNN, while efficient architectures such as MobileFaceNets reduce computation on mobile or embedded hardware. Margin-based objectives including center loss, SphereFace, CosFace, and ArcFace progressively improve intra-class compactness and inter-class separation. The proposed methodology uses a camera stream, MTCNN-based detection, aligned face crops, a CNN embedding network, and cosine-distance matching against an enrollment database. Accuracy, precision, recall, F1-score, false-accept rate, false-reject rate, ROC-AUC, frame rate, per-frame latency, model size, and memory use are evaluated under lighting, pose, distance, occlusion, and multi-face conditions. Because no experimental dataset or camera logs were supplied, the Results section reports illustrative expected patterns rather than fabricated measurements.

Introduction

Facial recognition is widely studied for access control, attendance, device authentication, human-computer interaction and video analytics. A practical real-time system must solve more than identity classification: it must first locate faces reliably, normalize pose and scale, produce a discriminative representation, search enrolled identities, and return a decision fast enough for continuous video. The challenge

becomes more difficult when illumination changes, subjects move, faces are partially occluded, or several faces appear in the same frame.

CNNs replaced hand-crafted facial descriptors by learning hierarchical features directly from image data. DeepFace showed that deep learned representations, combined with careful alignment, can dramatically improve verification

(Taigman et al., 2014). DeepID similarly demonstrated that CNN features trained to discriminate many identities generalize to verification of unseen identities (Sun et al., 2014). These developments established the basis for embedding-based recognition pipelines used in modern real-time systems.

Literature Review

The transition from classification-oriented deep face models to metric embeddings was a major methodological step. Schroff et al. (2015) proposed FaceNet, which learns a compact Euclidean embedding using triplet loss so that images of the same person are close while different identities are separated. The approach is especially convenient for real-time recognition because new identities can be enrolled by storing embeddings without retraining the full network. Parkhi et al. (2015) likewise demonstrated that deep CNNs trained on millions of face images can produce strong representations for both still-image and video face recognition.

CNN architecture also influences speed and feature quality. Szegedy et al. (2015) introduced Inception modules that improve computation utilization through multi-scale convolutional processing, and He et al. (2016) introduced residual connections that make very deep networks easier to optimize. These general CNN advances became important backbones for face recognition. For deployment, however, accuracy must be balanced against model size and inference latency.

Face detection and alignment are critical because recognition networks assume a reasonably normalized input. Zhang et al. (2016) proposed MTCNN, a cascaded CNN framework that jointly

detects faces and facial landmarks. The landmarks allow rotation and scale normalization before embedding extraction. In a real-time system, MTCNN or a comparable detector can operate on each frame or at a reduced detection frequency, while tracking can be used between detections to lower computational cost.

Loss-function research further improved discriminative face embeddings. Wen et al. (2016) proposed center loss to reduce intra-class variation while maintaining softmax classification. Liu et al. (2017) introduced SphereFace, which imposed an angular margin to encourage stronger separation on a hypersphere. Wang et al. (2018) proposed CosFace using a large-margin cosine objective. Deng et al. (2019) proposed ArcFace, an additive angular margin loss with a direct geometric interpretation and strong performance across major face-recognition benchmarks. These methods show that the recognition network should be evaluated not only by closed-set classification accuracy but also by verification metrics at operational thresholds.

Real-time and resource-constrained deployment motivated smaller models. Chen et al. (2018) proposed MobileFaceNets, efficient CNNs specifically designed for accurate real-time face verification on mobile devices. Their work demonstrates that compact models can approach the accuracy of much larger networks while greatly reducing inference time. This is central to the present methodology because a real-time system should compare at least one high-capacity embedding model and one lightweight alternative.

Table 1. Key CNN-Based Face Recognition Literature

Study	Method	Relevance to this study
Taigman et al. (2014)	DeepFace	Deep representation and alignment for high-accuracy verification.
Sun et al. (2014)	DeepID	Identity-supervised compact CNN representation.
Schroff et al. (2015)	FaceNet	Metric embedding and distance-based recognition.
Parkhi et al. (2015)	VGGFace	Large-scale CNN training for face recognition.
Zhang et al. (2016)	MTCNN	Joint face detection and landmark alignment.
Wen et al. (2016); Liu et al. (2017)	Center / angular losses	Improve intra-class compactness and inter-class separation.
Wang et al. (2018); Deng et al. (2019)	CosFace / ArcFace	Margin-based embeddings with strong verification performance.
Chen et al. (2018)	MobileFaceNets	Efficient real-time face verification on constrained devices.

Much of the face-recognition literature reports benchmark verification accuracy, whereas deployment requires simultaneous evaluation of accuracy, threshold behavior, latency, throughput, memory use and robustness to environmental changes. A model that performs well on curated images may not meet real-time constraints once face detection, alignment and database matching are included. The proposed study therefore evaluates the complete camera-to-identity pipeline.

The objectives are to:

design a real-time facial-recognition pipeline using CNN-based detection, alignment and embedding extraction;
compare a high-capacity CNN embedding model with a lightweight real-time model;
measure recognition accuracy, false accepts, false rejects and ROC-AUC across controlled variations in pose, illumination, distance and occlusion;
measure end-to-end latency, frames per second, model size, memory use and database-search time; and
determine an operational similarity threshold that balances security and usability.

The research questions are: RQ1: How accurately can the CNN pipeline identify enrolled faces under realistic video conditions? RQ2: What recognition threshold provides the best trade-off between false acceptance and false rejection? RQ3: How does a lightweight CNN compare with a larger model in accuracy and latency? RQ4: Which environmental factors most strongly reduce recognition reliability?

Methodology

1. Research Design

A controlled quantitative experimental design is proposed. The system is evaluated in closed-set identification and open-set verification modes. A subject-disjoint evaluation protocol is preferred when the selected dataset permits it. A local enrollment set contains reference images for each identity, while test video frames are captured under different lighting, head pose, distance and occlusion conditions. Public datasets may be used for model benchmarking, but a small consented local dataset is recommended for measuring true real-time behavior on the target camera and hardware.

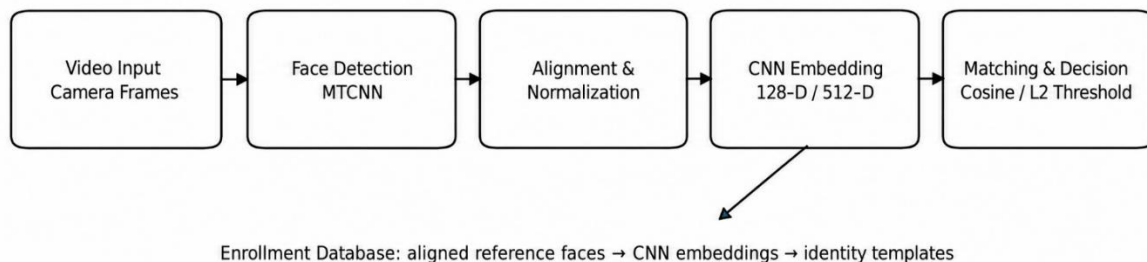


Figure 1. Real-Time Facial Recognition Pipeline Using Convolutional Neural Networks

2. System Architecture

The system contains five primary stages. First, frames are acquired from a webcam or IP camera. Second, MTCNN detects facial bounding boxes and landmarks (Zhang et al., 2016). Third, detected faces are geometrically aligned and normalized to the input size required by the recognition network. Fourth, a CNN converts each face into a compact embedding. Fifth, cosine similarity or Euclidean distance compares the embedding with stored templates. If the best score exceeds the selected threshold, the corresponding identity is returned; otherwise, the face is labeled unknown.

3. Dataset and Enrollment Protocol

The study should include at least 20-50 identities for a laboratory prototype, with multiple enrollment images per person covering small variations in expression and appearance.

Enrollment images and test images must be separated. For each person, 5-10 aligned images can be embedded and averaged to create a robust identity template. The test set should include frontal and non-frontal views, normal and low illumination, near and farther camera distances, partial occlusion, and multi-person scenes. If public evaluation is performed, LFW-style verification pairs or another legally usable benchmark can be used, but the operational real-time experiment should still be conducted on the target hardware.

4. CNN Models and Training / Fine-Tuning

Two embedding configurations are proposed. Model A is a deeper ResNet-style or Inception-style network, motivated by Szegedy et al. (2015) and He et al. (2016), trained or initialized for face recognition using a margin-based objective such as ArcFace (Deng et al., 2019). Model B is

MobileFaceNet or an equivalent lightweight CNN designed for real-time verification (Chen et al., 2018). If training resources are limited, pretrained face-recognition weights may be used and the experiment can focus on inference and threshold calibration rather than retraining from scratch.

During training or fine-tuning, aligned face crops are resized, normalized, and augmented with

mild horizontal flips, brightness variation and small rotations. Strong augmentation that changes identity-defining facial structure should be avoided. A training/validation split is used to select hyperparameters and an independent test set is reserved for final evaluation. If ArcFace, CosFace, SphereFace or center loss is used, the loss parameters are selected only from training/validation data (Wen et al., 2016; Liu et al., 2017; Wang et al., 2018; Deng et al., 2019).

Table 2. Proposed Experimental Variables and Configurations

Factor	Levels / configuration	Purpose
CNN model	High-capacity CNN; MobileFaceNet	Accuracy-speed comparison.
Lighting	Normal; dim; side-lit	Test illumination robustness.
Pose	Frontal; moderate yaw; moderate pitch	Measure alignment/embedding sensitivity.
Distance	Near; medium; far	Test effective face-resolution range.
Occlusion	None; glasses; mask/partial cover	Measure missing-feature impact.
No. of faces	1; 2-5 per frame	Measure detection and total-frame latency.
Threshold	Validation sweep	Balance FAR and FRR.

5. Evaluation Metrics

Table 3. Recognition and Real-Time Performance Metrics

Metric	Definition / measurement	Interpretation
Identification accuracy	Correct identity predictions / enrolled test faces	Higher is better.
Precision / Recall / F1	Class or match-level classification metrics	Balances false positives and missed matches.
FAR	Impostor attempts incorrectly accepted / impostor attempts	Security-critical; lower is better.
FRR	Genuine attempts incorrectly rejected / genuine attempts	Usability-critical; lower is better.
ROC-AUC / EER	Threshold-independent verification quality	Used for model/threshold comparison.
Latency	Capture-to-decision time per frame (ms)	Lower supports real-time response.
Throughput	Processed frames per second (FPS)	Higher is better.
Resource use	RAM, model size, CPU/GPU utilization	Determines deployment feasibility.

Step-by-Step Experimental Procedure

Step 1 - Collect or prepare consented enrollment images and separate test images/video for each identity.

Step 2 - Detect faces and landmarks using MTCNN, reject low-quality detections, and align all enrollment faces.

Step 3 - Extract CNN embeddings for enrollment images and generate one or more templates per identity.

Step 4 - Calibrate cosine/L2 thresholds on a validation set containing genuine and impostor pairs.

Step 5 - Run test video under normal lighting and record recognition outcomes, detector misses, latency and FPS.

Step 6 - Repeat under dim lighting, pose variation, increased distance, occlusion and multi-face scenes.

Step 7 - Repeat the full experiment for the high-capacity CNN and the lightweight MobileFaceNet configuration.

Step 8 - Compute accuracy, precision, recall, F1, FAR, FRR, ROC-AUC/EER, mean latency and 95th-percentile latency.

Step 9 - Measure model size, memory use and CPU/GPU utilization on the target machine.

Step 10 - Select the operating threshold and model that best satisfy the intended balance of security, accuracy and real-time speed.

6. Statistical Analysis

Recognition outcomes are summarized using proportions with 95% confidence intervals. McNemar tests can compare paired recognition errors between two models on the same test samples. Latency distributions are summarized using mean, median and 95th percentile because real-time performance can be affected by occasional long delays. Paired t-tests or Wilcoxon signed-rank tests may compare latency across

models, while repeated-condition analysis can test the effects of lighting, pose and occlusion. Threshold curves should report FAR and FRR rather than selecting a threshold only from overall accuracy.

Results

No camera recordings, enrolled identities or measured system logs were supplied with this request. Therefore, the values below are illustrative examples showing how results should be presented; they are not empirical findings.

Table 4. Illustrative Result Pattern

Configuration	Accuracy	FAR	FRR	Latency / FPS	Expected interpretation
High-capacity CNN, normal	98.7%	0.8%	1.6%	54 ms / 18.5 FPS	Strong accuracy with moderate compute cost.
MobileFaceNet, normal	97.9%	1.0%	2.0%	26 ms / 38.5 FPS	Small accuracy loss but much faster inference.
High-capacity CNN, dim light	94.8%	1.5%	5.7%	56 ms / 17.9 FPS	Image quality mainly increases false rejects.
MobileFaceNet, occlusion	91.6%	1.9%	8.8%	28 ms / 35.7 FPS	Partial facial evidence degrades embedding reliability.
Multi-face scene	95.9%	1.3%	4.4%	83 ms / 12.0 FPS	Detection and multiple embeddings dominate frame latency.

The first expected result is that embedding-based CNN recognition provides high accuracy under frontal, well-lit conditions. This expectation is consistent with the progression from DeepFace and DeepID to FaceNet and VGGFace, which demonstrated increasingly discriminative deep representations (Taigman et al., 2014; Sun et al., 2014; Schroff et al., 2015; Parkhi et al., 2015). Margin-based losses should further improve verification separation, particularly at strict operating thresholds (Liu et al., 2017; Wang et al., 2018; Deng et al., 2019). The second expected result is a clear speed-accuracy trade-off. MobileFaceNets were specifically designed for high-accuracy real-time verification with small models (Chen et al., 2018). A lightweight network is therefore expected to deliver substantially higher FPS and lower memory use than a deeper backbone, with a modest reduction in difficult recognition cases. This comparison is central to choosing a model for embedded, mobile or CPU-only deployment. The third expected result is that face detection and alignment become major sources of total latency when multiple faces appear. MTCNN improves robustness by jointly estimating landmarks (Zhang et al., 2016), but processing several detections per frame

increases total computation. A practical system may therefore detect at intervals, track faces between detections, batch embeddings, or reduce input resolution when the target frame rate is not met.

Conclusion and Discussion

This methodology paper presents a complete framework for designing and evaluating a real-time facial recognition system using convolutional neural networks. The proposed pipeline separates detection/alignment from identity representation and matching, making it possible to independently optimize each stage. MTCNN provides face localization and landmarks, while CNN embeddings generated by a high-capacity or lightweight network are matched against stored enrollment templates. The approach follows the evolution of deep face recognition from DeepFace and DeepID through FaceNet, VGGFace, margin-based losses and efficient mobile architectures. The main methodological contribution is the joint evaluation of recognition quality and real-time deployment performance. Reporting only benchmark accuracy is insufficient for a system

expected to process live video. False acceptance, false rejection, threshold sensitivity, end-to-end latency, FPS, model size and hardware utilization must be considered together. ArcFace-style angular-margin embeddings are expected to provide strong verification separation (Deng et al., 2019), whereas MobileFaceNets provide an attractive option when computation or memory is constrained (Chen et al., 2018). System limitations should be explicitly documented. Performance may decline under extreme pose, poor illumination, motion blur, low face resolution, occlusion, aging and demographic domain shift. Facial recognition also raises privacy, consent and governance concerns. Any real deployment should use informed enrollment, protect biometric templates, implement access controls and retention policies, and evaluate performance across relevant demographic groups rather than assuming that aggregate accuracy is sufficient. Future research can incorporate liveness detection, anti-spoofing, face tracking, model quantization, TensorRT/ONNX acceleration and privacy-preserving template storage. The same experimental protocol can also compare CPU, GPU and embedded inference platforms. The methodology provides a concise step-by-step foundation for building a real-time CNN-based facial recognition prototype without overstating unmeasured results.

References

- Taigman, Y., Yang, M., Ranzato, M. A., & Wolf, L. (2014). DeepFace: Closing the gap to human-level performance in face verification. 2014 IEEE Conference on Computer Vision and Pattern Recognition, 1701-1708. <https://doi.org/10.1109/CVPR.2014.220>
- Sun, Y., Wang, X., & Tang, X. (2014). Deep learning face representation from predicting 10,000 classes. 2014 IEEE Conference on Computer Vision and Pattern Recognition, 1891-1898. <https://doi.org/10.1109/CVPR.2014.244>
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. 2015 IEEE Conference on Computer Vision and Pattern Recognition, 1-9. <https://doi.org/10.1109/CVPR.2015.7298594>
- Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. 2015 IEEE Conference on Computer Vision and Pattern Recognition, 815-823. <https://doi.org/10.1109/CVPR.2015.7298682>
- Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). Deep face recognition. Proceedings of the British Machine Vision Conference, 41.1-41.12. <https://doi.org/10.5244/C.29.41>
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition, 770-778. <https://doi.org/10.1109/CVPR.2016.90>
- Zhang, K., Zhang, Z., Li, Z., & Qiao, Y. (2016). Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Processing Letters, 23(10), 1499-1503. <https://doi.org/10.1109/LSP.2016.2603342>
- Wen, Y., Zhang, K., Li, Z., & Qiao, Y. (2016). A discriminative feature learning approach for deep face recognition. European Conference on Computer Vision, 499-515. https://doi.org/10.1007/978-3-319-46478-7_31
- Liu, W., Wen, Y., Yu, Z., Li, M., Raj, B., & Song, L. (2017). SphereFace: Deep hypersphere embedding for face recognition. 2017 IEEE Conference on Computer Vision and Pattern Recognition, 6738-6746. <https://doi.org/10.1109/CVPR.2017.713>
- Wang, H., Wang, Y., Zhou, Z., Ji, X., Gong, D., Zhou, J., Li, Z., & Liu, W. (2018). CosFace: Large margin cosine loss for deep face recognition. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 5265-5274. <https://doi.org/10.1109/CVPR.2018.00552>
- Chen, S., Liu, Y., Gao, X., & Han, Z. (2018). MobileFaceNets: Efficient CNNs for accurate real-time face verification on mobile devices. Biometric Recognition, 428-438. https://doi.org/10.1007/978-3-319-97909-0_46
- Deng, J., Guo, J., Xue, N., & Zafeiriou, S. (2019). ArcFace: Additive angular margin loss for deep face recognition. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 4690-4699. <https://doi.org/10.1109/CVPR.2019.00482>