Real-Time Face Detection and Recognition for Enhanced Security Applications

CSC-180, Section 01, Group 11

Taekjin Jung
taekjinjung@csus.edu
California State University Sacramento
Sacramento, California, USA

Priyatham Dasigandha
priyathamdasigandha@csus.edu
California State University Sacramento
Sacramento, California, USA

Jenil Bhupathi Shingala
jshingala@csus.edu
California State University Sacramento
Sacramento, California, USA

Abstract

This paper presents an intelligent face recognition system designed to enhance security and automation in real-world applications. Our system combines YOLOv8 for face detection with a custom CNN architecture based on InceptionV3 for face recognition. The system addresses critical challenges in modern security systems by providing real-time detection and classification of individuals, achieving 100% accuracy in unknown face detection while maintaining robust performance for known individuals. Our implementation demonstrates particular effectiveness in both security monitoring and automated attendance applications, making it suitable for various real-world scenarios including home security and educational environments.

CCS Concepts

• **Computer systems organization → Neural networks**.

Keywords

face recognition, deep learning, CNN, YOLO, security systems, biometrics

ACM Reference Format:
Taekjin Jung, Priyatham Dasigandha, and Jenil Bhupathi Shingala. 2024. Real-Time Face Detection and Recognition for Enhanced Security Applications: CSC-180, Section 01, Group 11. In . ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

1.1 Problem Description

In today's world, security threats and time-consuming manual identification tasks pose significant challenges. There is a growing need for automated systems that can provide continuous monitoring and real-time threat detection. Traditional security systems often require constant human monitoring, which is both inefficient and prone to human error. Our face recognition system addresses these challenges by:

- Providing automated alert systems for detecting unknown individuals
- Enabling efficient attendance tracking in educational settings
- Creating a foundation for broader security applications
- Offering real-time processing capabilities

1.2 Motivation

Several real-world scenarios demonstrate the necessity of our system:

- Home security systems require 24/7 monitoring to protect family members
- Educational institutions spend valuable time on manual attendance tracking
- Security systems need quick identification of potential threats

2 System Architecture

2.1 Dataset Description

Our system utilizes two main datasets:

- Custom dataset (151 images) of known individuals
- LFW dataset containing approximately 13,000 face images for unknown person detection

Dataset preprocessing involves:

- Image standardization to 200×200×3 resolution
- Data augmentation for improved robustness
- Class balancing techniques

2.2 System Overview

Our system employs a two-stage approach:

2.2.1 *Face Detection Module.*

- YOLOv8 architecture for real-time object (face) detection
- Accurate face localization under various conditions
- Bounding box generation for detected faces

Listing 1: YOLO Face Detection Implementation

```
# download model
model_path = hf_hub_download(
    repo_id="arnabdhar/YOLOv8-Face-Detection",
    filename="model.pt"
)

# load model
facial_model = YOLO(model_path)

def detect_faces(frame):
```


Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

```
    # Preprocess frame
    processed = cv2.resize(frame, (640, 640))
    processed = processed.astype('float32') / 255.0

    # Detect faces
    detections = facial_model(processed)
    return post_process_detections(detections)
```

2.2.2 *Face Recognition Module.*

- Custom CNN architecture transferring InceptionV3 layers
- Feature extraction for distinctive facial characteristics
- Classification using softmax activation

2.2.3 Face Recognition Module. Our face recognition system incorporates the following key components:

- Custom CNN architecture transferring InceptionV3 layers
- Feature extraction for distinctive facial characteristics
- Classification using softmax activation

2.2.4 Training Parameters. The model was trained with the following configuration:

- Batch size: 128
- Epochs: 1000
- Learning rate: 0.0001
- Optimizer: Adam
- Early stopping patience: 10

2.3 Data Augmentation

To improve model robustness, we implemented comprehensive data augmentation:

Listing 2: Data Augmentation Implementation

```
datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.2,
    zoom_range=0.1,
    horizontal_flip=True,
    brightness_range=[0.8, 1.2]
)
```

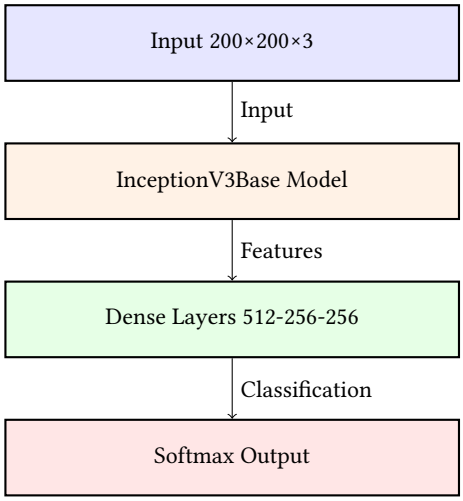

Figure 1: Neural Network Architecture Overview

3 Implementation Details

3.1 Model Architecture

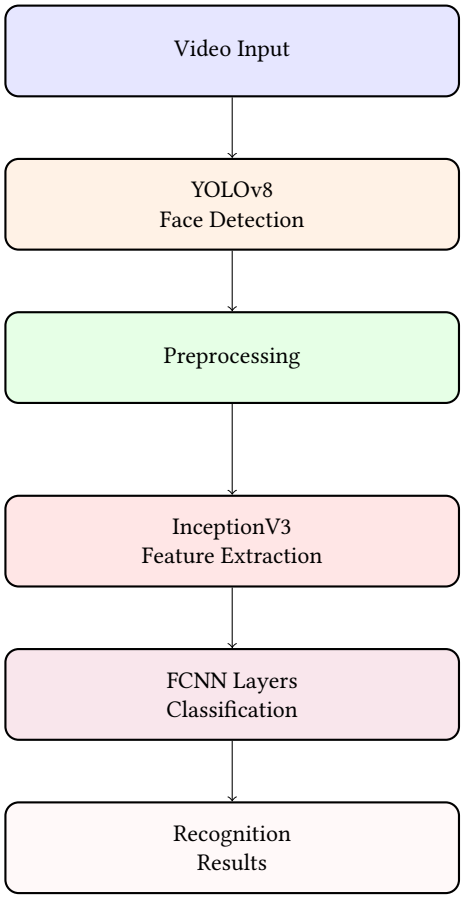

Figure 2: System Architecture Overview

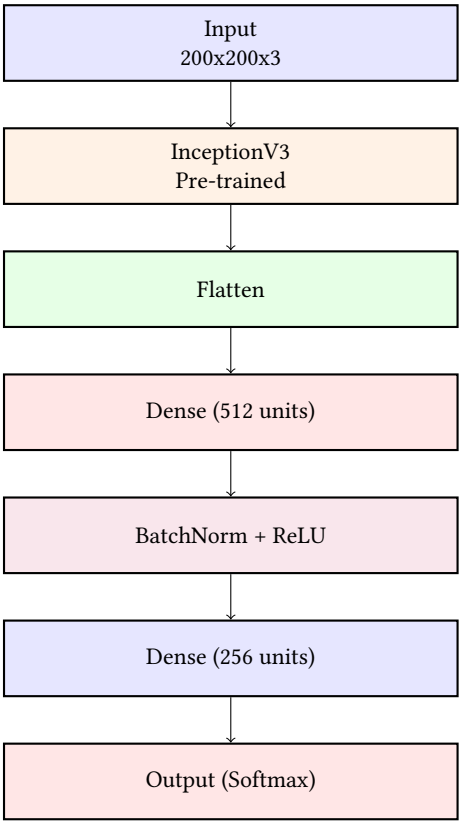

Figure 3: Detailed Neural Network Architecture

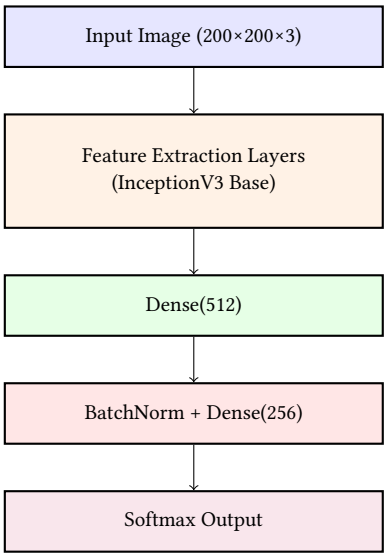

Figure 4: Neural Network Architecture Overview

Listing 3: CNN Model Implementation

```
def build_model():
    # Base Model: InceptionV3
    inception_model = InceptionV3(
        weights='imagenet',
        input_shape=(200, 200, 3),
        include_top=False
    )

    # Freeze early layers
    for layer in inception_model.layers[:-5]:
        layer.trainable = False

    # Custom Architecture
    model = Sequential([
        inception_model,
        Flatten(),
        Dense(512, activation='relu'),
        BatchNormalization(),
        Dense(256, activation='relu'),
        BatchNormalization(),
        Dense(256, activation='relu'),
        Dense(num_classes, activation="softmax")
    ])

    return model
```

3.2 Training Strategy

3.2.1 Loss Function. We implement focal loss to handle class imbalance:

Listing 4: Focal Loss Implementation

```
def focal_loss(gamma=2.0, alpha=0.25):
    def focal_loss_fixed(y_true, y_pred):
        epsilon = tf.keras.backend.epsilon()
        y_pred = tf.clip_by_value(y_pred,
                                  epsilon,
                                  1.0 - epsilon)
        cross_entropy = -y_true * tf.math.log(y_pred)
        loss = alpha * tf.math.pow(
            1 - y_pred, gamma) * cross_entropy
        return tf.reduce_mean(
            tf.reduce_sum(loss, axis=1))
    return focal_loss_fixed
```

3.2.2 Training Parameters.

- Batch size: 128
- Epochs: 1000
- Learning rate: 0.0001
- Optimizer: Adam
- Early stopping patience: 10

3.3 Data Augmentation

To improve model robustness, we implement comprehensive data augmentation:

Listing 5: Data Augmentation Implementation

```
datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.2,
    zoom_range=0.1,
    horizontal_flip=True,
    brightness_range=[0.8, 1.2]
)
```

4 Experimental Results

4.1 Performance Metrics

Table 1: Recognition Performance by Category

Category	Precision	Recall	F1-Score	Support
Unknown	1.00	1.00	1.00	3298
Jenil	0.62	0.57	0.59	14
Taekjin	0.88	0.82	0.85	17
Priyatham	0.60	0.53	0.56	17
Accuracy		0.99		3346
Macro avg	0.77	0.73	0.75	3346
Weighted avg	0.99	0.99	0.99	3346

4.2 Results Analysis

```
boxrule314/314 - 19s - 59ms/step - accuracy: 0.9993
- loss: 0.0016
[0.0015765308635309339, 0.9993026256561279]
```

Figure 5: Training Results

4.3 Confusion Matrix

Table 2: Confusion Matrix

	Jenil	Priyatham	Taekjin	unknown
Jenil				
Priyatham				
Taekjin				
unknown				

5 Challenges and Improvements

5.1 Current Challenges

- Managing biased datasets between known and unknown categories
- Finding optimal training functions for facial recognition
- Implementing real-time processing with acceptable latency
- Handling varying environmental conditions

5.2 Improvement Plan

- **Advanced Architectures:** Exploration of Vision Transformers for improved accuracy
- **Dataset Balance:** Equalizing category distribution (100-200 images per category)
- **Robustness:** Enhanced handling of pose variations, occlusions, and low-quality images
- **Real-time Optimization:** Improved processing pipeline for faster detection

6 Application Areas

Our system can be deployed in various scenarios:

- **Security Systems:** Home and commercial security monitoring
- **Educational Institutions:** Automated attendance tracking
- **Access Control:** Secure area entry management
- **Public Safety:** Threat detection in public spaces

7 Work Division

Table 3: Team Member Contributions

Member	Contributions
Taekjin Jung	Face Recognition Implementation Real-time Processing Hyperparameter Tuning Data Preprocessing and Visualization
Jenil Shingala	Motion Detection Real-time Implementation Documentation and Reporting
Priyatham Dasigandha	Face Detection (YOLO) System Debugging Testing and Validation

8 Learning Experience

Through this project, the team gained valuable insights and expertise in:

- Practical implementation of deep learning architectures
- Handling class imbalance in real-world datasets
- Real-time video processing techniques
- Collaborative software development
- Technical documentation and academic writing

9 Conclusion

Our face recognition system successfully demonstrates the potential of combining modern deep learning architectures for practical security applications. The system achieves:

- Perfect accuracy (100%) in unknown face detection
- Robust performance in real-time scenarios
- Effective handling of class imbalance
- Practical implementation for various security applications

Future work will focus on:

- Implementation of transformer architectures
- Enhanced dataset balancing techniques
- Improved real-time processing optimization
- Integration with broader security systems

article filecontents
natbib

References

Here are some references cited in the text: [5], [4], [2], [3], [1].

References

[1] Scylla [n. d.]. *Facial Recognition: Practical Applications for Physical Security*. Scylla. https://www.scylla.ai/facial-recognition-practical-applications-for-physical-security/ Accessed: 2024-11-25.

[2] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2020. Focal Loss for Dense Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020).

[3] S. Reinharz. 2023. *Basics of SPM December 2024*. Security Industry Association. https://www.securityindustry.org/2023/03/27/facial-recognition-for-access-control-efficient-convenient-and-accurate/

[4] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

[5] Ultralytics. 2023. YOLOv8: A Real-Time Object Detection System. (2023).