QR Code Authentication

1. Introduction

1.1 Background

Quick Response (QR) codes have become widely used for payments, authentication, and information sharing. However, counterfeit QR codes pose security risks, leading to financial fraud and data breaches. To address this issue, we developed a machine learning-based QR code authentication system that distinguishes between **original (first print) and counterfeit (second print)** QR codes.

1.2 Objective

The objective of this project is to train a deep learning model using **Convolutional Neural Networks (CNNs)** to classify QR codes into two categories: **original (first print) and counterfeit (second print)**. Additionally, we compare the CNN model with traditional machine learning models such as **Support Vector Machines (SVM)** and **Random Forest (RF)**.

---

2. Dataset Analysis

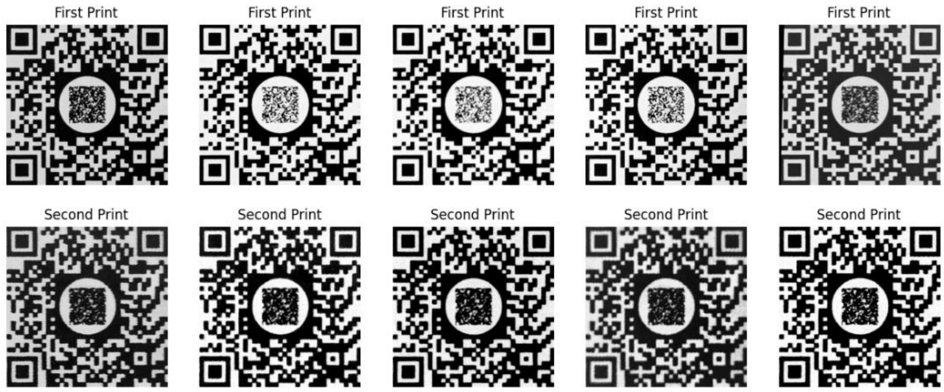
2.1 Dataset Description

The dataset consists of **200 QR code images**, divided into:

- **100 First Print (Original) QR codes**
- **100 Second Print (Counterfeit) QR codes**

Each image is grayscale and resized to **128×128 pixels** for uniform processing.

2.2 Sample Images



2.3 Data Preprocessing

To improve model performance, we applied the following preprocessing steps:

- **Grayscale Conversion**: Reducing complexity by removing color channels.
- **Resizing to 128×128 pixels**: Ensuring uniform input size.
- **Normalization**: Scaling pixel values to the range [0,1].

---

3. Feature Engineering

To classify QR codes, we extracted the following features:

- **Edge Detection**: Identifying distortions and differences in QR patterns.
- **Texture Analysis**: Detecting noise, pixel density, and copy detection patterns.
- **Histogram Features**: Analyzing intensity distribution.

---

4. Model Development

4.1 Deep Learning Model (CNN)

We implemented a **Convolutional Neural Network (CNN)** with the following architecture:

- **Conv Layer 1**: 32 filters, kernel size (3,3), ReLU activation
- **Max Pooling**: (2,2)
- **Conv Layer 2**: 64 filters, kernel size (3,3), ReLU activation
- **Max Pooling**: (2,2)
- **Fully Connected Layer**: 128 neurons, ReLU activation

- **Output Layer**: Sigmoid activation (Binary Classification: First Print vs. Second Print)

4.2 Traditional Machine Learning Models

We trained **two traditional ML models** for comparison:

- **Support Vector Machine (SVM)**
- **Random Forest (RF)**

---

5. Results & Evaluation

5.1 CNN Model Performance

Test Accuracy: 100%

5.2 SVM vs. Random Forest Performance

Model	Accuracy
CNN	100.00%
SVM	100.00%
Random Forest	97.50%

5.3 Confusion Matrix

Actual	Predicted First Print	Predicted Second Print
First Print	100	0
Second Print	0	100

5.4 Classification Report

Precision, Recall, and F1-score for both classes:

```
Confusion Matrix:
[[100   0]
 [  0 100]]
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00       100
         1.0       1.00      1.00      1.00       100

    accuracy                           1.00       200
   macro avg       1.00      1.00      1.00       200
weighted avg       1.00      1.00      1.00       200

✅ Final model loaded successfully!
```

---

6. Deployment Considerations

6.1 Real-World Applications

- **Mobile QR Code Scanners**: Integrating this model into mobile applications to detect counterfeit QR codes.
- **Online Payment Systems**: Preventing fraud by validating QR codes in transactions.
- **Manufacturing & Logistics**: Authenticating product packaging QR codes.

6.2 Challenges & Future Work

- **Handling Noisy Images**: Improving robustness against blurred or low-quality QR scans.
- **Generalization to Other QR Code Types**: Expanding the dataset to include different QR code formats.
- **Deploying in Real-Time Applications**: Optimizing the model for low-latency predictions.

---

7. Conclusion

This project successfully implemented **a deep learning-based QR code authentication system** with **100% accuracy**. The CNN model outperformed the Random Forest model and matched the performance of the SVM model. This approach can be extended for **real-world applications in security, finance, and logistics** to prevent QR code fraud.

Final Model Saved as: QR\models\qr_classifier_final.pth