QR Code Authentication: Detecting Original vs. Counterfeit Prints

1 Introduction

QR codes are widely used for authentication and anti-counterfeiting measures. This project aims to classify QR codes as either original "first print" or counterfeit "second print" using machine learning. The classification is based on subtle variations in print quality and copy detection patterns (CDPs) embedded in QR codes.

2 Dataset Description

The dataset consists of two classes:

- **First Prints:** Original QR codes with embedded copy detection patterns.
- **Second Prints:** Counterfeit versions created by scanning and reprinting original QR codes.

Each image is grayscale and resized to **128x128** pixels for consistency. The dataset is split into **80% training** and **20% testing**, ensuring class balance using stratification.

3 Methodology

This project employs two classification approaches:

1. **Traditional Machine Learning Approach (SVM):** Uses handcrafted features extracted from QR code images.
2. **Deep Learning Approach (CNN):** Automatically learns patterns from images using convolutional neural networks.

3.1 Data Preprocessing

- **Grayscale conversion:** Images are converted to grayscale to focus on structural details.

- **Resizing:** All images are resized to **128x128** pixels.
- **Normalization:** Pixel values are scaled between **0 and 1**.
- **Train-test split:** Data is divided into **80% training** and **20% testing** while maintaining class balance.

3.2 Traditional Machine Learning Model: Support Vector Machine (SVM)

- The images are **flattened** into 1D vectors.
- An **SVM with a linear kernel** is trained on the transformed features.
- Model performance is evaluated using **accuracy, precision, recall, and F1-score**.

3.3 Deep Learning Model: Convolutional Neural Network (CNN)

- A CNN architecture with **two convolutional layers** and **max pooling** layers is implemented.
- A **fully connected dense layer** followed by a dropout layer (to prevent overfitting) is added.
- The model is compiled using **binary cross-entropy loss** and **Adam optimizer**.
- It is trained for **10 epochs** with a batch size of **32**.

4 Experimental Results

4.1 SVM Model Performance

- **Accuracy:** 100%
- **Precision:** 100%
- **Recall:** 100%
- **F1-score:** 100%

4.2 CNN Model Performance

- **Accuracy:** 92.5%
- **Precision:** First Print: 100%, Second Print: 87%
- **Recall:** First Print: 85%, Second Print: 100%
- **F1-score:** First Print: 92%, Second Print: 93%

4.3 Comparison of SVM and CNN

Model	Accuracy	Precision	Recall	F1-score
SVM	100%	100%	100%	100%
CNN	92.5%	93.5%	92.5%	92.5%

Table 1: Performance comparison of SVM and CNN models.

5 Conclusion

- The SVM model achieved **100% accuracy**, making it highly effective for this classification task.
- The CNN model, despite a **92.5% accuracy**, still demonstrated strong generalization.
- SVM may be more suitable for this problem due to the high accuracy and lower computational cost.

6 Future Improvements

- **Hyperparameter tuning** to improve CNN performance.
- **Experiment with other deep learning architectures** such as ResNet or EfficientNet.
- **Deployment of a real-time QR code authentication system** using Flask or FastAPI.