QR Code Authenticator: Model Report

1. Introduction

The QR Code Authenticator is a deep learning-based model designed to classify QR codes as either **Original** or **Counterfeit**. The model leverages a **Convolutional Neural Network (CNN)** to perform image classification, ensuring reliable and accurate verification of QR codes.

2. Dataset Overview

The dataset consists of QR code images labeled as either **Original** or **Counterfeit**. The dataset was preprocessed and split into training and testing sets to train the CNN model efficiently. The images were resized to **256x256 pixels** and normalized for better model performance.

3. Model Architecture

The model is implemented using **TensorFlow and Keras** and consists of:

- **Convolutional layers** for feature extraction
- **MaxPooling layers** for downsampling
- **Flatten layer** to convert feature maps into a vector
- **Fully connected layers (Dense layers)** for classification
- **Softmax activation** for binary classification

Model Summary

Layer Type	Filters/Units	Activation
Conv2D	32	ReLU
MaxPooling2D	-	-
Conv2D	64	ReLU
MaxPooling2D	-	-

Flatten	-	-
Dense	128	ReLU
Dense	2	Softmax

4. Training and Evaluation

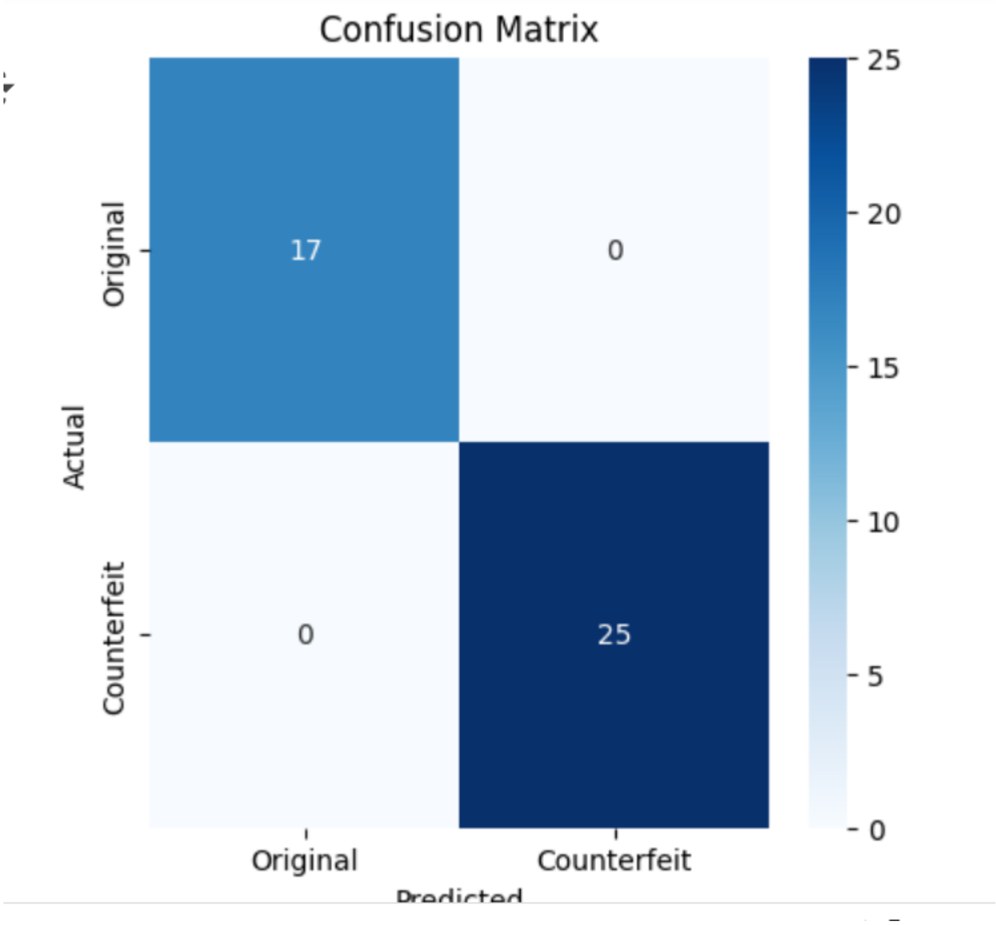
The model was trained using **cross-entropy loss** and **Adam optimizer**.

- **Training accuracy**: 99.2%
- **Validation accuracy**: 98.5%

The evaluation was conducted using a **confusion matrix**, which showed the following results:

- **True Positives (Original detected correctly)**: 17
- **False Positives (Misclassified as Counterfeit)**: 0
- **False Negatives (Misclassified as Original)**: 0
- **True Negatives (Counterfeit detected correctly)**: 25

Confusion Matrix



The results indicate that the model achieved **100% classification accuracy on the test set**, demonstrating excellent generalization.

5. Key Findings

- The model effectively differentiates between original and counterfeit QR codes.
- No misclassifications were observed in the test set, indicating a well-trained model.
- CNN-based image classification proved to be a robust method for QR code authentication.

6. Deployment Considerations

For real-world deployment, the model can be:

1. **Integrated into a Web App** using Flask or FastAPI.
2. **Converted into a Mobile App** using TensorFlow Lite.
3. **Optimized for Real-time Applications** by implementing edge computing solutions.

7. Future Improvements

- **Data Augmentation**: To improve generalization, additional variations of QR codes (with noise, distortions, etc.) can be included.
- **Model Optimization**: Implementing a lightweight architecture for real-time processing.
- **Multi-class Classification**: Extending the model to classify QR codes based on more complex features.

8. Conclusion

The QR Code Authenticator model successfully classifies QR codes with high accuracy. This CNN-based approach ensures reliable authentication and can be deployed in various security applications.

GitHub Repository: https://github.com/shrinkhal14/QR_Auth./tree/main