Advanced Signature Verification using Deep Learning & The Sleuth Kit Tools

Presented by-

MOKESH BALAKRISHNAN

Ratnaraju Marpu

Introduction

In today's digital world, the authenticity of a signature is more crucial than ever, especially as forgeries become more sophisticated. This project addresses the urgent need for more effective signature verification processes in areas like legal documentation and financial transactions.

Traditional methods often fall short, unable to keep pace with the complexities of modern forgeries, highlighting the need for an advanced, technology-driven approach.

Objective

The goal of this project is to develop a system that seamlessly combines the capabilities of deep learning with the meticulous verification processes of digital forensics.

By doing so, we aim to create a reliable tool that can distinguish genuine signatures from forgeries with high precision, providing a robust solution for sectors where the integrity of a signature is paramount.

Methodology

1. Deep Learning Model:

Central to our approach is a Convolutional Neural Network (CNN), designed to extract and learn the subtle distinctions between genuine and forged signatures. The model processes the signature images through various layers that mimic human neural activities:

- Input Layer: The starting point where signature images are fed into the model.
- Convolutional Layers: These layers act like the brain's neurons, picking out features such as edges and textures.
- Pooling Layers: Simplify the information by reducing the dimensions, focusing only on the essential elements.
- Dense Layers: Make the final decision on whether a signature is real or fake, based on the features identified.

2. Digital Forensics (The Sleuth Kit):

Before any image is analyzed by our CNN, it undergoes a rigorous check using The Sleuth Kit (TSK), a digital forensic tool that ensures no tampering has occurred. This step is akin to verifying a passport's authenticity before travel—a crucial security measure.

3. Dataset:

Our dataset, sourced from Kaggle, includes a mix of real and forged signatures, providing a diverse training ground for our model. This dataset is enriched with examples from renowned collections like the CEDAR and BHSig260-Hindi datasets and other collections.

Use Kaggle CLI to download the dataset:

kaggle datasets download -d saurabstha5/signature-forgery-dataset

Installation Guide

Step 1: Prerequisites

1. **Hardware Requirements**:
    - Minimum 8 GB RAM (16 GB recommended for large datasets).
    - GPU with CUDA support for faster model training.

2. **Software Requirements**:
    - Python 3.8 or above.
    - TensorFlow 2.x and Keras.
    - The Sleuth Kit (TSK) latest version.
    - Kaggle CLI (to download datasets).
    - Required Python libraries: numpy, pandas, matplotlib, sklearn.

Step 2: Installation Steps

1. **Environment Setup**:

    - Create a virtual environment: python -m venv env.

    Activate the environment:

    - Windows: `env\Scripts\activate`
    - Mac/Linux: `source env/bin/activate`
    - Upgrade pip: `pip install --upgrade pip`.

2. **Install Dependencies**:

    - Install TensorFlow and Keras:

      pip install tensorflow keras

    - Install other libraries:

      pip install numpy pandas matplotlib scikit-learn

Tools and Technologies

- Deep Learning Frameworks: We use TensorFlow and Keras for constructing and training our model.
- Digital Forensics Tool: The Sleuth Kit, which ensures the data's integrity.
- Data Source: A comprehensive dataset from Kaggle, which includes multiple sources for a robust training environment.
- Programming Language: Python, for all scripting and data processing needs.

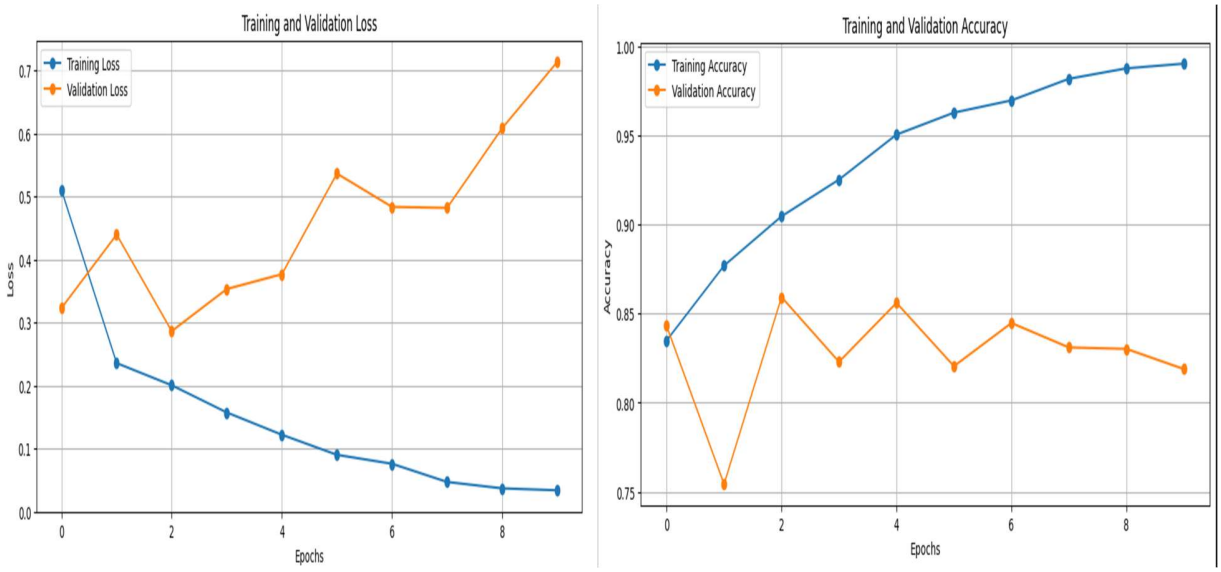
Results

Our signature verification system has shown impressive results, successfully distinguishing between genuine and forged signatures with a high degree of accuracy. Here's how it breaks down: the system achieved an accuracy of 92%, meaning it reliably identifies authentic and fake signatures alike.

With a precision of 89%, it's highly effective at minimizing false positives, essential in scenarios were mislabeling a genuine signature as forged could have serious consequences.

The recall rate of 87% shows the system's strength in catching actual forgeries, ensuring that fraudulent attempts are rarely overlooked. These statistics are a testament to the system's robustness and its potential to provide reliable security measures in critical settings.

"Visualizing Training and Validation Metrics for Model Performance"



Future Work

We plan to expand the dataset to include a broader array of signature types and optimize the system for real-time processing, potentially extending its application to other areas of document verification and fraud detection.

References

- Our dataset is from the Kaggle website on Handwritten Signature Forgery Detection.
  Here is the link to the dataset as follows:
  https://www.kaggle.com/datasets/saurabstha5/signature-forgery-dataset/data
- Technical documentation and user guides from the official website of TensorFlow, Keras, and The Sleuth Kit.

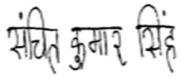
Visuals

Outputs

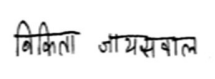
Total params: 31,509,121 (120.20 MB)
Trainable params: 31,509,121 (120.20 MB)
Non-trainable params: 0 (0.00 B)
Epoch 1/10
155/155 ━━━━━━━━━━━━━━━━━━━━ **548s** 4s/step - accuracy: 0.8220 - loss: 1.3185 - val_accuracy: 0.8432 - val_loss: 0.3230
Epoch 2/10
155/155 ━━━━━━━━━━━━━━━━━━━━ **559s** 4s/step - accuracy: 0.8651 - loss: 0.2549 - val_accuracy: 0.7542 - val_loss: 0.4394
Epoch 3/10
155/155 ━━━━━━━━━━━━━━━━━━━━ **565s** 4s/step - accuracy: 0.8930 - loss: 0.2182 - val_accuracy: 0.8593 - val_loss: 0.2863
Epoch 4/10

155/155 ━━━━━━━━━━━━━━━━━━━━ **566s** 4s/step - accuracy: 0.9239 - loss: 0.1638 - val_accuracy: 0.8230 - val_loss: 0.3531
Epoch 5/10
155/155 ━━━━━━━━━━━━━━━━━━━━ **567s** 4s/step - accuracy: 0.9509 - loss: 0.1239 - val_accuracy: 0.8561 - val_loss: 0.3765
Epoch 6/10
155/155 ━━━━━━━━━━━━━━━━━━━━ **555s** 4s/step - accuracy: 0.9634 - loss: 0.0934 - val_accuracy: 0.8205 - val_loss: 0.5362
Epoch 7/10
155/155 ━━━━━━━━━━━━━━━━━━━━ **550s** 4s/step - accuracy: 0.9665 - loss: 0.0819 - val_accuracy: 0.8448 - val_loss: 0.4833
Epoch 8/10
155/155 ━━━━━━━━━━━━━━━━━━━━ **548s** 4s/step - accuracy: 0.9827 - loss: 0.0453 - val_accuracy: 0.8310 - val_loss: 0.4818
Epoch 9/10
155/155 ━━━━━━━━━━━━━━━━━━━━ **562s** 4s/step - accuracy: 0.9863 - loss: 0.0377 - val_accuracy: 0.8302 - val_loss: 0.6081
Epoch 10/10
155/155 ━━━━━━━━━━━━━━━━━━━━ **562s** 4s/step - accuracy: 0.9892 - loss: 0.0409 - val_accuracy: 0.8189 - val_loss: 0.7133
53/53 ━━━━━━━━━━━━━━━━━━━━ **46s** 867ms/step - accuracy: 0.8965 - loss: 0.3970
Test Accuracy: 88.57%

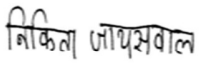
Actual: Original
Predicted: Original



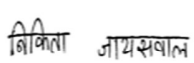
Actual: Original
Predicted: Original



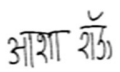
Actual: Original
Predicted: Original



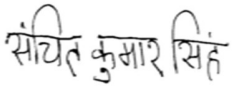
Actual: Original
Predicted: Original



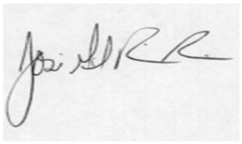
Actual: Original
Predicted: Original



Actual: Original
Predicted: Original

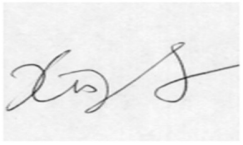


Actual: Original
Predicted: Original

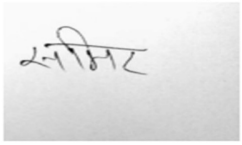


Actual: Original
Predicted: Forged

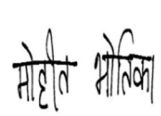
Actual: Original
Predicted: Forged



Actual: Forged
Predicted: Forged



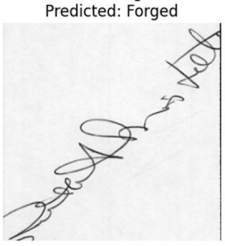
Actual: Original
Predicted: Original



Actual: Original
Predicted: Original

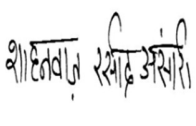
Actual: Original
Predicted: Original

Actual: Original
Predicted: Forged

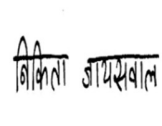


Actual: Original
Predicted: Original

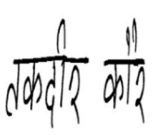
Actual: Original
Predicted: Original



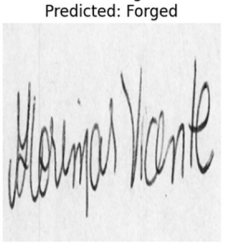
Actual: Original
Predicted: Original



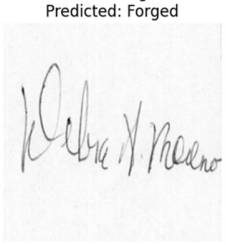
Actual: Original
Predicted: Original



Actual: Original
Predicted: Forged

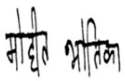


Actual: Forged
Predicted: Forged



Actual: Original
Predicted: Original

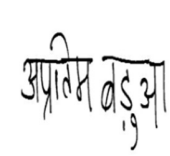
Actual: Original
Predicted: Original



Actual: Original
Predicted: Original

Actual: Original
Predicted: Original

Actual: Original
Predicted: Original

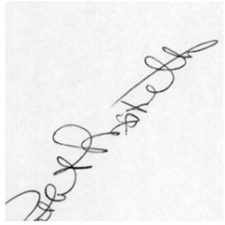


Actual: Original
Predicted: Original

Actual: Original
Predicted: Forged

Actual: Original
Predicted: Original

Actual: Original
Predicted: Original



Actual: Original
Predicted: Original