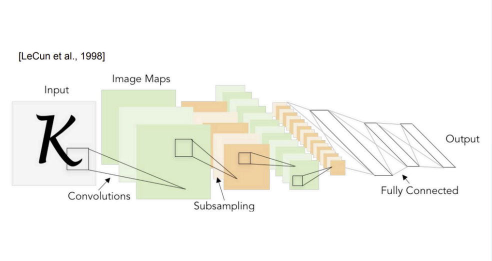
MNIST Digit Recognition: Real-time Handwritten Digit Classification using Convolutional Neural Networks

Aadvait Hirde, Darragh Buckley, Jared Gengnagel, Santiago Diaz

Abstract

This research explores the development of a real-time handwritten digit recognition system using deep learning techniques, specifically focusing on the MNIST dataset. By implementing a neural network with TensorFlow and integrating OpenCV for live video processing, we demonstrate an end-to-end machine learning solution capable of instantaneously classifying handwritten digits with high accuracy. Our approach not only achieves over 99% classification accuracy but also provides a user-interactive interface for real-time digit recognition, bridging theoretical machine learning concepts with practical, interactive applications.

Recent Work

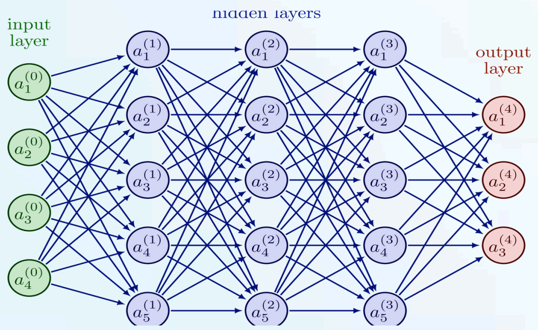


Handwritten digit recognition has progressed through key advancements. LeCun et al. (1998) pioneered CNNs for the MNIST dataset, outperforming traditional methods like SVMs. Simard et al. (2003) improved generalization with elastic distortions, while Wan et al. (2013) introduced dropout to prevent overfitting, achieving great results.

Later research expanded on these foundations with traditional algorithms, shallow neural networks, deep CNNs for hierarchical feature extraction, and ensemble methods for enhanced accuracy. This study builds on these approaches by introducing a novel, real-time interactive system integrating deep learning and computer vision.

Introduction

This study tackles handwritten digit recognition using the MNIST dataset, aiming to build a high-accuracy neural network that generalizes well to real-world handwriting.

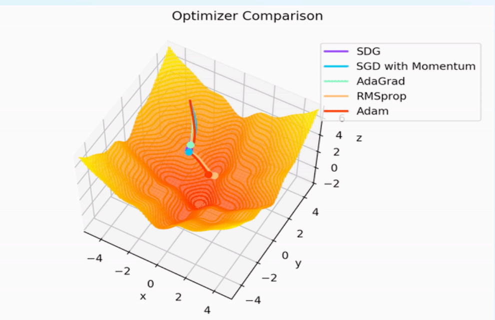


It also develops an interactive, real-time system for dynamic user engagement. Emphasizing performance evaluation through various metrics the research combines deep learning, computer vision, and user-friendly interfaces to create a robust and versatile solution for diverse use cases.

Methodology

The research methodology involves leveraging the MNIST dataset, consisting of 70,000 grayscale images of handwritten digits (0-9), for training and testing a neural network. Images are normalized to enhance consistency in feature extraction. The neural network features a fully connected architecture with 784 input neurons (one per pixel), a hidden layer of 512 neurons using ReLU activation, and a 10-neuron output layer with softmax for digit classification. Training is optimized with the Adam optimizer and Sparse Categorical Crossentropy loss, with early stopping implemented to prevent overfitting.

For real-time digit recognition, OpenCV is integrated to capture webcam input and preprocess images. Key steps include grayscale conversion, noise reduction with Gaussian blurring, thresholding for binarization, and extracting a region of interest. The interactive interface allows users to adjust thresholding and view real-time predictions. The system resizes and pads images to fit the neural network's input requirements, ensuring robust and accurate recognition in dynamic settings.



Conclusion

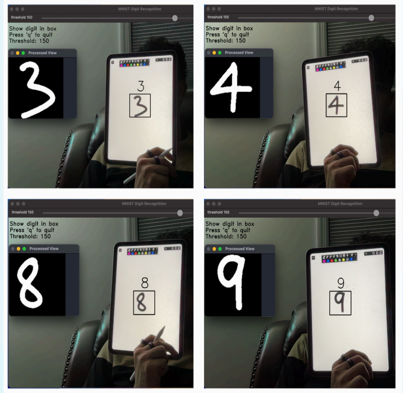
There are extremely high accuracy across all digits (nearly 100% for all classes), and perfect ROC curves (AUC = 1.00) for all digits. Very high precision-recall curves show near-perfect performance. The confusion matrix's strong diagonal pattern indicates excellent classification. Most digits have >950 correct classifications. The model shows very high confidence in its predictions, as the vast majority of predictions made with >90% confidence.

References:
Lecun et al. (1998): Discusses gradient-based learning methods applied to document recognition, particularly using neural networks for image recognition.
Simard et al. (2003): Presents best practices for applying convolutional neural networks (CNNs) in visual document analysis, focusing on performance optimization techniques.
Wan et al. (2013): Introduces DropConnect, a regularization technique for neural networks to prevent overfitting and improve generalization.

Implementation Details

A. Code Structure
The system includes functions to load the MNIST dataset, train the neural network, predict digits from 28x28 images, and handle real-time recognition. The run_opencv_loop() function captures webcam frames, processes them, and passes the image to the model for prediction, which is then displayed via OpenCV.

B. User Interaction Design
The OpenCV interface allows users to start recognition with a click, adjust the threshold with a slider, and view real-time digit predictions overlaid on the video feed. A separate window displays the preprocessed region of interest. The system captures frames, preprocesses them (grayscale, Gaussian blur, and thresholding), extracts the region of interest, resizes it to 28x28 pixels, and predicts the digit. The result is shown in real-time.



Empirical Analysis and Results