bytewise.

PROJECT PROPOSAL

Convolutional Neural Network for Image Classification using (MNIST Dataset)

Submitted By: Hashir Aziz

Track Name: Machine Learning / Deep learning

Track Lead: Zayan Rashid Rana

Bytewise Fellowship 2024

TABLE OF CONTENTS

1. Introduction

This project is focused on developing a Convolutional Neural Network (CNN) for image classification, specifically using the MNIST dataset. The MNIST dataset consists of **70,000 images** of handwritten **digits (0-9)** and serves as an excellent benchmark for machine learning models. The primary objective is to create a robust model that can accurately classify digits, leveraging CNN's capability to learn spatial hierarchies in image data.

2. Existing System/Related Works

Several machine learning approaches have been employed to tackle digit classification problems using the MNIST dataset. Classical methods like Support Vector Machines (SVM) and K-Nearest Neighbours (KNN) have been widely used, but they lack the deep hierarchical feature extraction capabilities of CNNs. Modern advancements have demonstrated that CNNs outperform traditional methods due to their inherent ability to automatically learn relevant features from images. Our project builds upon these foundations to further enhance accuracy and scalability.

3. Proposed System

3.1. Scope

The proposed system aims to develop a CNN-based image classifier that processes the MNIST dataset, trains on a subset of images, and achieves high accuracy in identifying handwritten digits. The system will consist of various modules, including image preprocessing, CNN model development, and performance visualization, making it adaptable to similar image classification tasks.

3.2. Proposed System Modules

3.2.1. Image Preprocessing

This module handles the transformation of the MNIST images into a format suitable for CNN input. Key tasks include:

- Rescaling images to standard dimensions.
- Normalization of pixel values to enhance model convergence.
- Augmentation techniques such as rotation and scaling to increase dataset variability and improve generalization.

3.2.2. CNN Architecture

The CNN will consist of multiple convolutional layers followed by pooling and fully connected layers. The architecture will be carefully designed to capture the spatial relationships between pixels:

- Convolutional Layers to extract image features.
- Pooling Layers to down sample feature maps.
- Fully Connected Layers for final classification.

3.2.3. Training and Evaluation

This module involves training the CNN on the MNIST dataset using:

- Optimization techniques such as Adam or Stochastic Gradient Descent (SGD).
- Loss functions like Categorical Cross-Entropy.
- Evaluation metrics such as accuracy, precision, recall, and the confusion matrix to measure model performance.

3.2.4. Visualization and User Interface

To enhance user interaction and interpretability, a graphical user interface will be developed for:

- Visualizing performance through accuracy/loss curves.
- Displaying confusion matrices and other relevant statistics to give insights into model predictions.
- Enabling users to upload custom images for prediction and model testing.

3.3. Advantages of the Proposed System

- **High Accuracy**: Leveraging deep CNN layers ensures high accuracy in recognizing handwritten digits.
- **Automatic Feature Extraction**: The CNN's ability to automatically detect and learn features reduces manual feature engineering.
- **Adaptability**: The system can be extended to more complex datasets beyond MNIST, making it flexible for other image classification tasks.

3.4. Limitations/Disadvantages

- **Hardware Intensive**: CNNs require substantial GPU power for efficient training, which can be costly for larger datasets.
- **Training Time**: Training deep networks, especially for more complex tasks, can take significant time depending on the computational resources.

3.5. Application

- **Digit Recognition**: Useful in financial sectors for automatic processing of handwritten checks, and postal services for digitizing handwritten addresses.
- **Broader Image Classification**: The techniques developed can be adapted for facial recognition, object detection, and medical imaging analysis.

3.6. SDG Goals

This project aligns with the United Nations' Sustainable Development Goal (SDG) 9: Industry, Innovation, and Infrastructure. By automating image classification tasks, we contribute to technological innovation and infrastructure improvement across various industries.

4. Hardware/Software Requirements

Hardware Requirements

- **GPU**: NVIDIA GTX/RTX series or Tesla V100 for efficient model training.
- **CPU**: Intel i7/i9 or AMD Ryzen 7/9 for preprocessing and handling large datasets.
- **RAM**: Minimum 16GB, 32GB recommended for smooth training and evaluation processes.
- **Storage**: SSD of at least 256GB to store the dataset, model checkpoints, and outputs.

Software Requirements

- **Operating System**: Linux (Ubuntu preferred) or Windows with GPU support.
- **Programming Language**: Python 3.7+.
- **Libraries/Frameworks:**
    - TensorFlow or PyTorch for building and training the CNN.
    - NumPy and Pandas for data manipulation.
    - OpenCV for image preprocessing.
    - Matplotlib and Seaborn for data visualization.
- **Cloud Resources**: AWS EC2, Google Collab, or Azure for scalable computation when required.