Task 1: Documentation for MNIST Image Classification Neural Network

Introduction

This Python script uses the TensorFlow and Keras libraries to create a simple Convolutional Neural Network (CNN) for classifying images from the MNIST dataset. The MNIST dataset is a large database of handwritten digits that is commonly used for training various image processing systems.

Installation

Before running the script, make sure you have the following Python libraries installed:

- TensorFlow
- Keras
- Matplotlib

You can install these libraries using pip:
pip install tensorflow keras matplotlib

Usage

To run the script, simply execute it in a Python environment where TensorFlow and Keras are installed. The script will automatically download the MNIST dataset, preprocess the data, train the model, and evaluate its performance.

Code Explanation

The script starts by importing the necessary libraries and configuring TensorFlow to run on GPU if available.

The MNIST dataset is loaded and preprocessed. The images are reshaped to have a single channel (since MNIST images are grayscale) and normalized so that pixel values are in the range [0, 1]. The labels are one-hot encoded for categorical classification.

A CNN model is built using Keras. The model consists of a 2D convolutional layer with 32 filters, each of size (3, 3), followed by a 2x2 max-pooling layer to reduce spatial dimensions. The output is then flattened to a 1D array for input to a densely connected layer. The final layer is a densely connected layer with 10 neurons (one for each output class) and uses the softmax activation function for multi-class classification.

The model is compiled with the Adam optimizer, categorical crossentropy loss, and accuracy as the metric.

The model is trained on the training data for 10 epochs, with a batch size of 64. The model is validated on 20% of the training data.

Finally, the model is evaluated on the test set, and the test accuracy and loss are printed. The training history (loss and accuracy) is plotted using Matplotlib.

Results

The neural network model was trained on the MNIST dataset for 10 epochs. The model achieved a test accuracy of 98.21%, indicating a high level of performance in classifying the handwritten digits. The test loss, which measures the model's error on the test set, was 0.0574. This low loss value suggests that the model's predictions closely match the actual values.

The training and validation accuracy and loss were plotted against the number of epochs. These plots provide insight into the model's learning process. They can be used to identify whether the model is overfitting (performing well on the training data but poorly on the validation data) or underfitting (not performing well on either the training or validation data).

Conclusion

The results demonstrate the effectiveness of the Convolutional Neural Network model in classifying images from the MNIST dataset. With an accuracy of over 98%, the model shows a strong ability to recognize and classify handwritten digits.

The low loss value indicates that the model's predictions are, on average, very close to the actual values, further demonstrating the model's effectiveness.

These results highlight the power and potential of neural networks in image classification tasks. Future work could explore the use of more complex models, different types of data, or other machine learning tasks.

Task 2: Documentation for Working with Databases

This Python script demonstrates interaction with a MySQL database to perform basic CRUD (Create, Read, Update, Delete) operations on a table named **users**. The script allows users to add new users, retrieve user information, update user details, delete users, and display all users. The MySQL database configuration is provided in the **db_config** dictionary.

Prerequisites

- Python 3.x
- MySQL Server installed locally
- mysql-connector-python library installed (pip install mysql-connector-python)

Installation

- Install Python: Ensure you have Python 3.x installed. If not, download and install it from python.org.
- Install MySQL Server: Install MySQL Server locally. You can download it from MySQL Downloads.
- Install required library: Open a terminal and run the following command to install the mysql-connector-python library:

You can install these libraries using pip:
pip install mysql-connector-python

Configuration

Adjust the db_config dictionary in the script with your MySQL server details:

```
db_config = {
    'host': 'localhost',
    'user': 'root',
    'password': 'your_password',
    'database': 'user_db'
}
```

Usage

1. Run the Script:
    - Open a terminal and navigate to the directory containing the script.
    - Run the script using the following command:

```
python task2.py
```

2. Menu Options:
    - The script presents a menu with options from 1 to 6.
    - Choose an option by entering the corresponding number and pressing Enter.

3. Menu Choices:
    - 1. Create User: Add a new user by providing a username and email.
    - 2. Update User: Update an existing user by entering the user ID and providing new details.
    - 3. Delete User: Delete a user by entering the user ID.
    - 4. Show All Users: Display information about all users in the database.
    - 5. Show User by ID: Display information about a specific user by entering their ID.
    - 6. Exit: Exit the script.

4. Exiting the Script:
    - Press `Ctrl + C` to exit gracefully.

If you encounter any issues, ensure that the MySQL server is running, and the credentials in the db_config dictionary are correct.

Task 3: Documentation for Google Books API Python Script

Description

This Python script uses the Google Books API to fetch information about a book given its ISBN number. The script sends a GET request to the Google Books API and processes the received data to extract and print the book's title, authors, publisher, and published date.

Installation

Before running this script, you need to install the requests library, which is used for making HTTP requests in Python. You can install it using pip:
pip install requests

Usage

To use this script, you simply need to call the fetch_book_info function with the ISBN number of the book you want to fetch information about. Here's an example:

fetch_book_info('9780140449136')

This will print the information of the book with the ISBN number '9780140449136'.

Important Details

- The script uses the 'https://www.googleapis.com/books/v1/volumes?q=isbn:{isbn}' endpoint of the Google Books API to fetch the book information.
- The script handles the HTTP response status codes. If the status code is 200, it means the request was successful and the script will process the response data. If the status code is not 200, the script will print an error message with the status code.
- The script also handles the case where no book is found with the given ISBN number and prints a message accordingly.