MNIST CNN Project – Command Line Usage Guide

This document lists the PowerShell commands used to train and test the MNIST CNN models. All commands are copy–paste friendly.

1. Environment Setup

Activate the virtual environment (required in every new PowerShell):

```
.\.venv\Scripts\Activate.ps1
```

Verify PyTorch and GPU availability (optional):

```
python -c "import torch; print(torch.__version__); print(\'cuda:\',
torch.cuda.is_available())"
```

2. Training Commands

Train all 10 models, save weights, loss CSV, and plots:

```
python main.py
```

3. Prediction Commands

Predict a single image with one model:

```
python main.py predict --model 3 --image .\Different_test_data\8_example.png
```

Predict all images in a folder with one model:

```
python main.py predict --model 3 --folder .\Different_test_data\dataset1
```

Predict all images in a folder recursively:

```
python main.py predict --model 3 --folder .\Different_test_data --recursive
```

Compare all 10 models on a single image:

```
python main.py predict --models all --image .\Different_test_data\8_example.png
```

Compare all 10 models on a dataset folder:

```
python main.py predict --models all --folder .\Different_test_data\dataset1
```

Save predictions to CSV:

```
python main.py predict --model 3 --folder .\Different_test_data\dataset1 --out
results_model3.csv
```

Save predictions for all models to CSV:

```
python main.py predict --models all --folder .\Different_test_data\dataset1 --out
results_all_models.csv
```

4. Dataset Naming Rules

For automatic accuracy calculation, image filenames should start with the true digit (0–9).

Examples:

```
0_sample.png
7_phone_photo.jpg
8_mywriting_01.png
```

Recommended folder structure:

```
Different_test_data/ dataset1_clean/ dataset2_phone_photos/ dataset3_my_handwriting/
```