AI-F19
Assignment 4 (A4)
Machine Learning

General Info
10 Point Individual Assignment Due by 2 pm Friday November 22nd (see syllabus for collaboration & late submission policies).

Learning Objective
This assignment satisfies learning objective 2 (LO2) as specified in the syllabus. You will apply conceptual knowledge of core AI concepts by implementing AI algorithms, analyzing existing intelligent systems (including humans), and using existing AI tools to solve problems.

Getting Help
We want these projects to be rewarding and instructional, not frustrating and demoralizing. But, we don't know when or how to help unless you ask. If you find yourself stuck on something, contact us via Piazza or come by the office hours. If you can't make our office hours, let us know and we will be happy to schedule alternate times.

Submission
You should submit the deliverables on Gradescope under Assignment 4. If you encounter any difficulties during the submission process please contact us via email and also include a copy of your submission files before the deadline.

Deliverables

You will need to submit the following files:

- A4.py
- <Identikey>-TrainingData.csv
- <Identikey>_A4_report.pdf

Instructions - MUST READ BEFORE GETTING STARTED

- You may email us your submission at any time to account for upload difficulties to Gradescope and we can mark your submission “as-is” and timestamped with the email receipt time.
- **It is your responsibility to ensure that your submission passes the autograder.** Here are common autograder failures: (1) code does not run to completion locally; (2) submitting a zipped directory; (3) including any additional libraries other than those specified. **We won’t be resolving errors in submissions that cause the autograder to fail.** If you’re having trouble passing the autograder come to office hours or post to piazza, or email, but this all needs to be done before the deadline.
- For both parts we have provided initial code, please complete all questions within these files. **Do not submit iPython notebooks.**

Prerequisites
For this assignment you will need some additional python libraries. You can install these using pip. You may also need to install the prerequisites for these libraries also. More information can be found on their respective manual pages.

```
pip install matplotlib, scikit-learn, pandas, numpy, pillow
```

Overview

The purpose of this assignment is to use a Hopfield network and a Multilayer Perceptron (MLP) to recognize hand-drawn digits. We will focus on discriminating 5s from 2s. It is specifically designed to give you experience with all the necessary steps in designing such a system starting with generating your own data.

Part 1 - Data Preparation - [1 Pt]

First, you will prepare the data that will be used to train your models. For this, you will use the 5x5 grids provided in Appendix 1 (end of this document). Draw out the digits 5 and 2, four times each (you can draw on paper or digitally draw), with some variation.

The next step is to digitize the data. For each box a digit crosses through, record a 1 in the corresponding box of the 5x5 grid. Otherwise record a 0. This boolean (0 or 1) grid is used as the training data for your network (see Appendix 2 for an example).

The third step is to format the data so it can be inputted into python. The boolean grid should be saved in row major order in a single row of a CSV file (see Appendix 2). You also need to supply a label – “five”, “two” – so each row is associated with a corresponding label of the digit represented by the row.

See the attached file NewInput.csv for an example of how to format your data for python. The first row contains headers and last column specifies the class label. You will use this file in Part 5 (do not use it before Part 5 except as an example for formatting input files).

You can check your data formatting using the provided function **utils.vizualize** which takes an array of length 25, and displays the corresponding image. See skeleton code for an example.

What to submit
- Take a picture of your input grids and paste them in the report.
- Submit <Identikey>-TrainingData.csv.

Part 2 - Hopfield Network - [5 Pts]

You will now implement, train and test a Hopfield network (see 'HopfieldNotes.pdf').

Implementation [4 points]

We have provided an initial class structure for you to use.

- **addSinglePattern -** update your hopfield network with one pattern
- **Fit -** update your hopfield network with a list of patterns
- **retrieve -** takes an input pattern as a parameter and uses your hopfield network to return a retrieved pattern. If necessary you should set your own stopping criteria.
- **Classify -** take an input pattern as a parameter, use your retrieve method, then return a string classification of either "two", "five" or "unknown".
  Hint: You can do this by comparing the retrieved pattern to the 'perfect' patterns provided on lines 6 and 7

Train and Test

Once you have implemented the Hopfield network, you should fit it on the two 'perfect' patterns provided in the code. You should then attempt to classify each of the instances in your <Identikey>-TrainingData.csv file by using them as retrieval cues in the Hopefield network.

Training Data:	The two 'perfect instances provided in the code (lines 6 and 7)
Test Data:	All of <Identikey>-TrainingData.csv
Cross Validation:	None since we have separate training and testing sets

In your report [1 point]

- How accurately did your network classify the digits? *You may use percent accuracy as the metric since the classes are evenly balanced.*
- If it is making errors, analyze the results in order to understand where the Hopfield network is making errors and explain them in your report. If it is not making errors, then discuss why it is classifying perfectly.

Part 3 - Train a MLP - [1 Pt]

Using your generated dataset implement a MLP classifier using the function provided in scikitLearn and the default parameters [0.5 points]

Training Data: The two 'perfect instances provided in the code (lines 6 and 7)
Test Data: All of <indentikey>-TrainingData.py
Cross Validation: None since we have separate training and testing sets

In your report [0.5 points]

- How accurately did your network classify the digits?
- If it is making errors, analyze the results in order to understand where the MLP is making errors and explain them in your report. If it is not making errors, then discuss why it is classifying the data perfectly.

Part 4 - Distortion - [2 Pts]

Perform an experiment to test the effect of distortion your Hopfield Network (from part 2) and MLP (from part 3) when multiple levels of noise is introduced.

Distorting Your Input
We have provided a skeleton function to perform the distortion on each instance, complete this function first.

distort_input takes as parameters one instance and a distortion rate, which should be a float between 0 and 1. This is similar to mutation rate in an earlier assignment. For each bit in the input array, if distortion rate is 0.1, there is a 0.1 probability that the bit will be flipped (1 changes to 0 and 0 to 1).

Experiments
Once you have implemented the functions, you will experiment with how distortion rate impacts your classifier accuracy. You will distort the instances from **<indentikey>-TrainingData.py** using distortion rates ranging from 0 to 0.5, in increments of 0.01. For each distortion rate, train your hopfield network (from part 2), and MLP (from part 3) on **undistorted 'perfect instances'** (from parts 2 and 3) and then attempt to classify the distorted instances. You should then calculate the accuracy for each classifier at each distortion rate.

As you increase the distortion rate at each step of your testing make sure you are passing undistorted data to your distortion function and are only changing the distortion rate. If you pass previously distorted data then you will quickly end up with data that is all noise.

Training Data: The two 'perfect instances provided in the code (lines 6 and 7)
Test Data: Distorted instances

Cross Validation: None since we have separate training and testing sets

In your report

- Produce a line plot, with classifier accuracy on the y axis, and distortion rate on the x axis. Your line plot should have two lines, one for your hopfield network and one for your MLP.
- Provide a brief (1 to 2 sentences) commentary on your graph. What does this data tell you about the two methods robustness to distortion.

Part 5 - Experimenting with number of hidden layers [1 Pt]

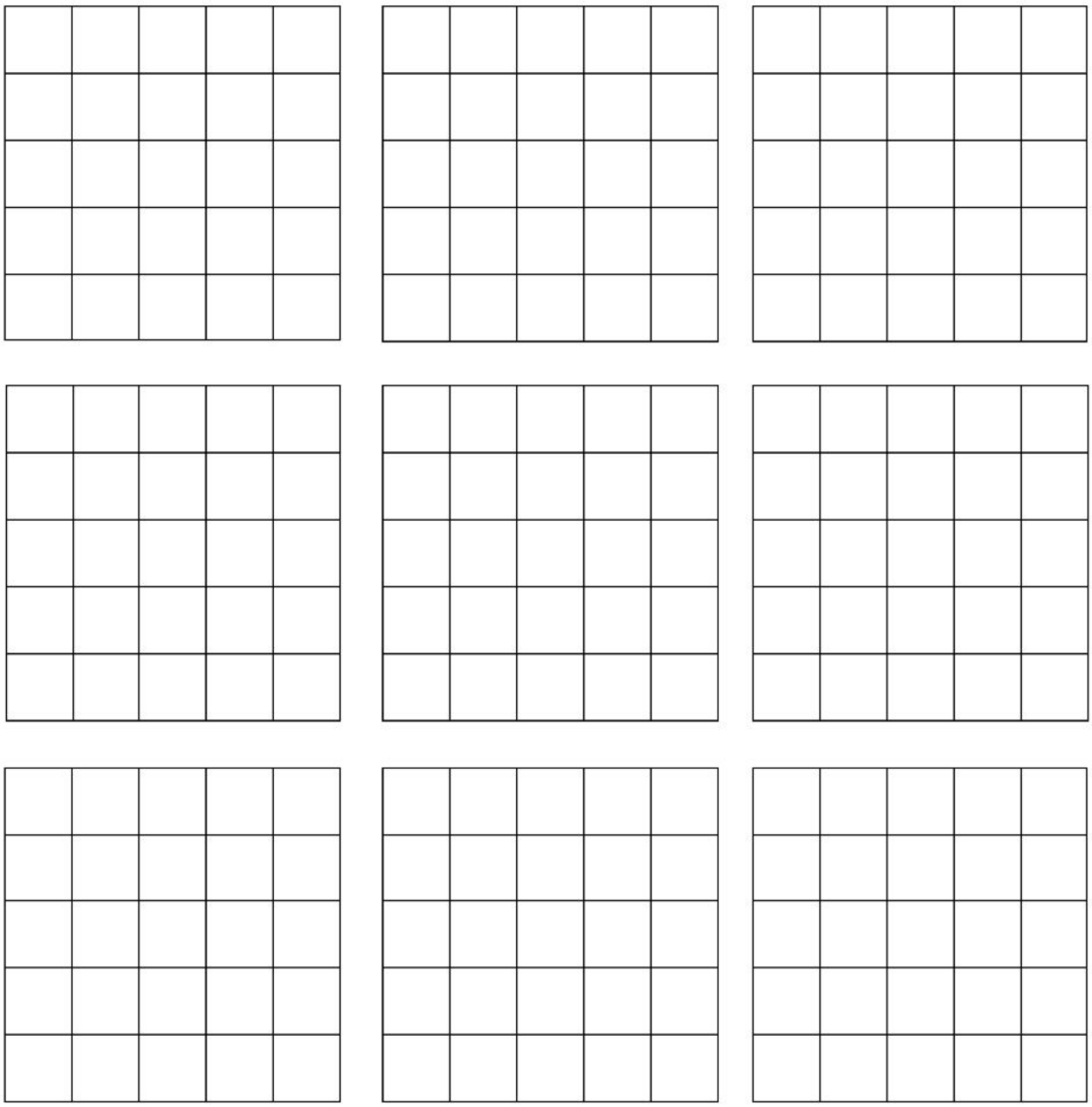
We have provided some additional data points in NewInput.csv. You should combine this with your <indentikey>-TrainingData.py and build a MLP where you vary the number of layers and assess whether this improves performance on the distorted data.

Train: <indentikey>-TrainingData.py + NewInput.csv
Test: Distorted instances
Cross Validation: None since we have separate training and testing sets

In your report [1 point]

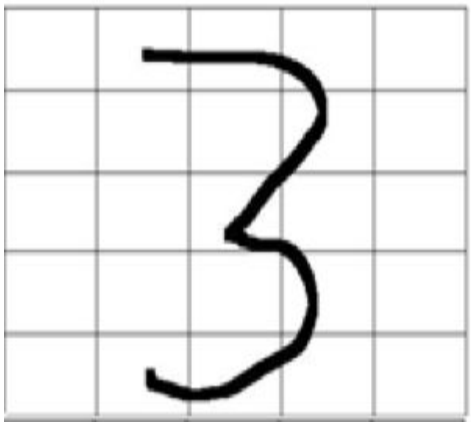
- Report the results from your experiments with number of layers. Specifically, reproduce the graph from Part 4, but now with an additional line for different versions of your MLP (You still need to include the graph from Part 4 separately in your report). Briefly discuss your findings.

APPENDIX 1 : Data Collection Grids



APPENDIX 2 : How to prepare Data Grids

1. Write the digit in the grid



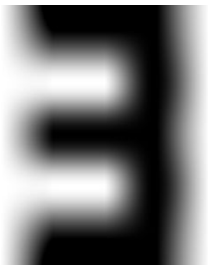
2. Record the state of the cells (1 if black, 0 if white) in a grid

0	1	1	1	0
0	0	0	1	0
0	0	1	1	0
0	0	0	1	0
0	1	1	1	0

3. Convert the grid to row major form and save as a CSV file.

0	1	1	1	0	0	0	0	1	0	0	0	1	1	0	0	0	0	1	0	0	1	1	1	0

4. Check in visualizer



APPENDIX 3 : Scoring Rubric

The scoring rubric **for your report** is based on the Kentucky General Scoring Rubric from the Kentucky Department of Education (KDE).

Score	Description
Category 4 (Score 90%-100%)	● The student completes all important components of the task and communicates ideas clearly. ● The student demonstrates in-depth understanding of the relevant concepts and/or process. ● Where appropriate, the student chooses more efficient and/or sophisticated processes. ● Where appropriate, the student offers insightful interpretations or extensions (generalizations, applications, analogies).
Category 3 (Score 70%-90%)	● The student completes most important components of the task and communicates clearly. ● The student demonstrates an understanding of major concepts even though he/she overlooks or misunderstands some less important ideas or details.
Category 2 (Score 60%-70%)	● The student completes some important components of the task and communicates those clearly. ● The student demonstrates that there are gaps in his/her conceptual understanding.
Category 1 (Score 10%-60%)	● The student shows minimal understanding. ● The student addresses only a small portion of the required task(s).
Category 0 (Score 0)	● Response is totally incorrect or irrelevant.
Blank (Score 0)	● No response.