

Bonus Homework 03: Digit Recognition

Kaveh Fathian, Email: fathian@ariarobotics.com

Handout: 2025-12-04

Due: 2025-12-10, 11:59pm, on Canvas

General Instructions:

- You should solve this homework and submit your report **individually**. Identical submissions will receive a grade of zero.
- For bonus assignments, getting help from other students is okay and encouraged, but the **TAs** are **not** required to help you.
- Ask any questions on **Ed Discussion** (instead of emailing).
- The deliverables are outlined for each problem, and you should carefully **follow the instructions**. Failing to follow instructions will result in **points being subtracted**.
- Follow the instructions specified by each problem to submit the homework. Upload your **code** or other deliverables as instructed.
- **Late submission:** Late or missed submission will not be accepted. Any excused absence must be documented and disclosed to the instructor (extensions will be granted on a case-by-case basis).

Bonus Homework 03: Digit Recognition

Kaveh Fathian, Email: fathian@ariarobotics.com

Handout: 2025-12-04

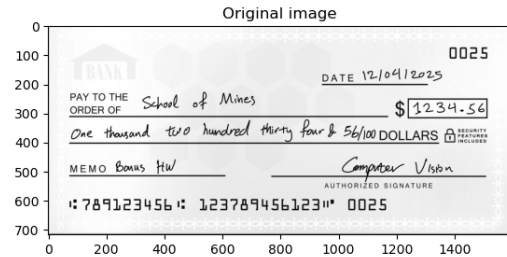
Due: 2025-12-10, 11:59pm, on Canvas

Bonus Problem (4pts) – In this problem, you will write a program that automatically reads the written amount on a scanned, hand-written check. You must use the Jupyter notebook template “bhw03_template.ipynb”, uploaded to the course repository (<https://github.com/ariarobotics/cv/tree/main/code>).

Complete all of the required sections directly in the template.

Your code must run in our “cv” Conda environment. Your

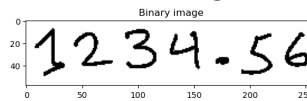
code must run in the “cv” Conda environment we use in this course, and it must load and process the scanned check image located at `data/cv_check.jpg` to extract the numerical amount. Follow the **instructions** below carefully. Deviating from the specified workflow, skipping required steps, or using methods not permitted in the assignment will result in a **deduction** of points.



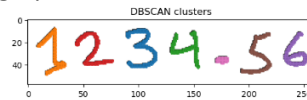
Instructions:

Create a check-scanning system by following the steps outlined below:

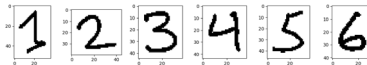
- Extract the number box from the scanned check image and convert it into a binary image.



- Use DBSCAN clustering from the `sklearn` library to identify individual digits in the binary image by clustering neighboring digit pixels.



- Extract an image patch around each digit cluster.



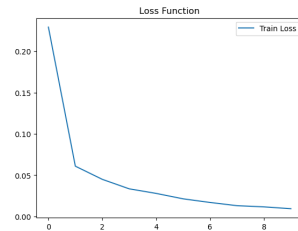
- Train a CNN on the MNIST dataset to classify digits in 28×28 pixel images. You may design your own architecture, but it must satisfy the following constraints:
 - The network must include at least 1 and at most 3 convolutional layers.
 - The network must include at least 1 and at most 3 dense layers.
 - In total, the convolutional and dense layers combined must be between 2 and 6 layers.
 - Each convolutional or dense layer may have no more than 64 filters/neurons. (For example, Dense (128, ...) or Conv2D (128, ...) are not allowed.)
 - You may use any activation function.
 - You may include any additional layers or configuration, such as pooling layer, dropout, strides, batch normalization, etc.
 - You may **not** train your network for more than 20 epochs.
- Plot the loss function vs. epoch for your trained network, demonstrating a good decrease after training.

Bonus Homework 03: Digit Recognition

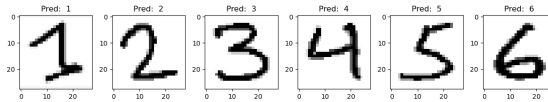
Kaveh Fathian, Email: fathian@ariarobotics.com

Handout: 2025-12-04

Due: 2025-12-10, 11:59pm, on Canvas



- Use the trained CNN to classify each extracted digit image patch.



- Print the final reconstructed number, formed by the digit predictions in left-to-right order. Do not include any decimal points, commas, or other characters; only the digits.

Final number: 123456

Deliverables:

- A single **Jupyter notebook** (.ipynb file) containing your complete implementation.
- All required **outputs** must be visible in the notebook. This means you must run all cells before submitting, so that every plot, image, and printed result appears inline. The template notebook specifies where outputs are expected.

Grading Policy:

- Your notebook will be manually reviewed and executed in the class “cv” Conda environment. If your code fails to run, crashes at any point, or produces errors during execution, you will receive zero points.
- You must follow all instructions stated above exactly as written. For example, you must use DBSCAN for digit segmentation, your neural network must follow the required limits on layer counts and sizes, and you must not substitute other methods or architectures.
- Your code will be tested on a **different** scanned check image with the *same dimensions* but **different handwritten digits**.
- To receive full credit:
 - Your notebook must run start to finish without errors.
 - Your program must correctly extract and classify all digits in the number box.
 - The final printed output must contain only the digits (no decimal point or other formatting).
 - Any misclassified digit will result in a deduction, so ensure your CNN achieves strong accuracy on MNIST and generalizes well.
- You must complete this assignment independently. Students submitting identical or near-identical code will receive a score of zero.