USC University of Southern California

EE 105: Introduction to Electrical Engineering

Lab 4.2

Camera classifier

Yasser Khan

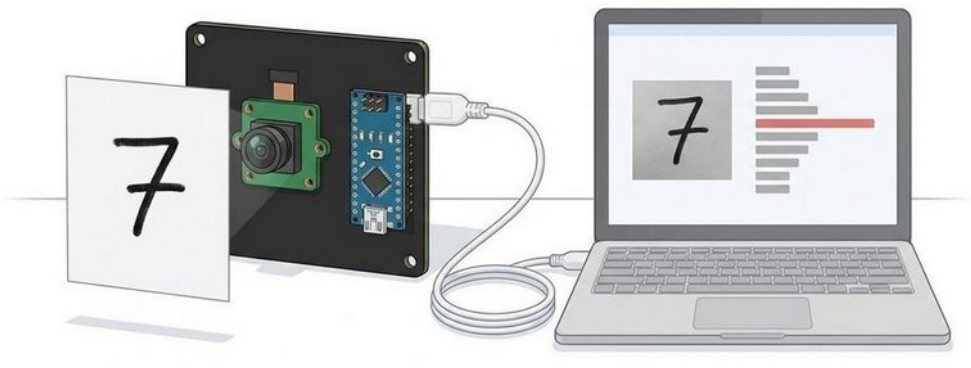
Rehan Kapadia

Today's mission

Show the board's camera a digit you wrote, and let your Lab 4.1 network read it.

By the end of this lab you can:

- Stream **live pictures** from the camera into the notebook
- Turn a picture into a **28 x 28 MNIST-style input**
- Classify your digits live with your **Lab 4.1 CNN**
- Measure the **domain shift** and close part of the gap



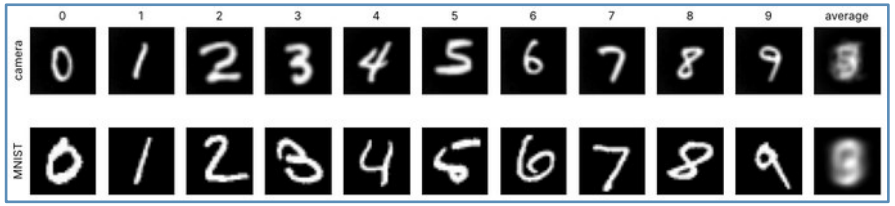
A paper digit, the board's camera, and the network's answer.

A Camera **10 min** → **B** Live view **20 min** → **C** Predict **20 min** → **D** Test set **25 min** → **E** Domain shift **20 min**

Key takeaways

What to remember from this lab, with the numbers that matter.

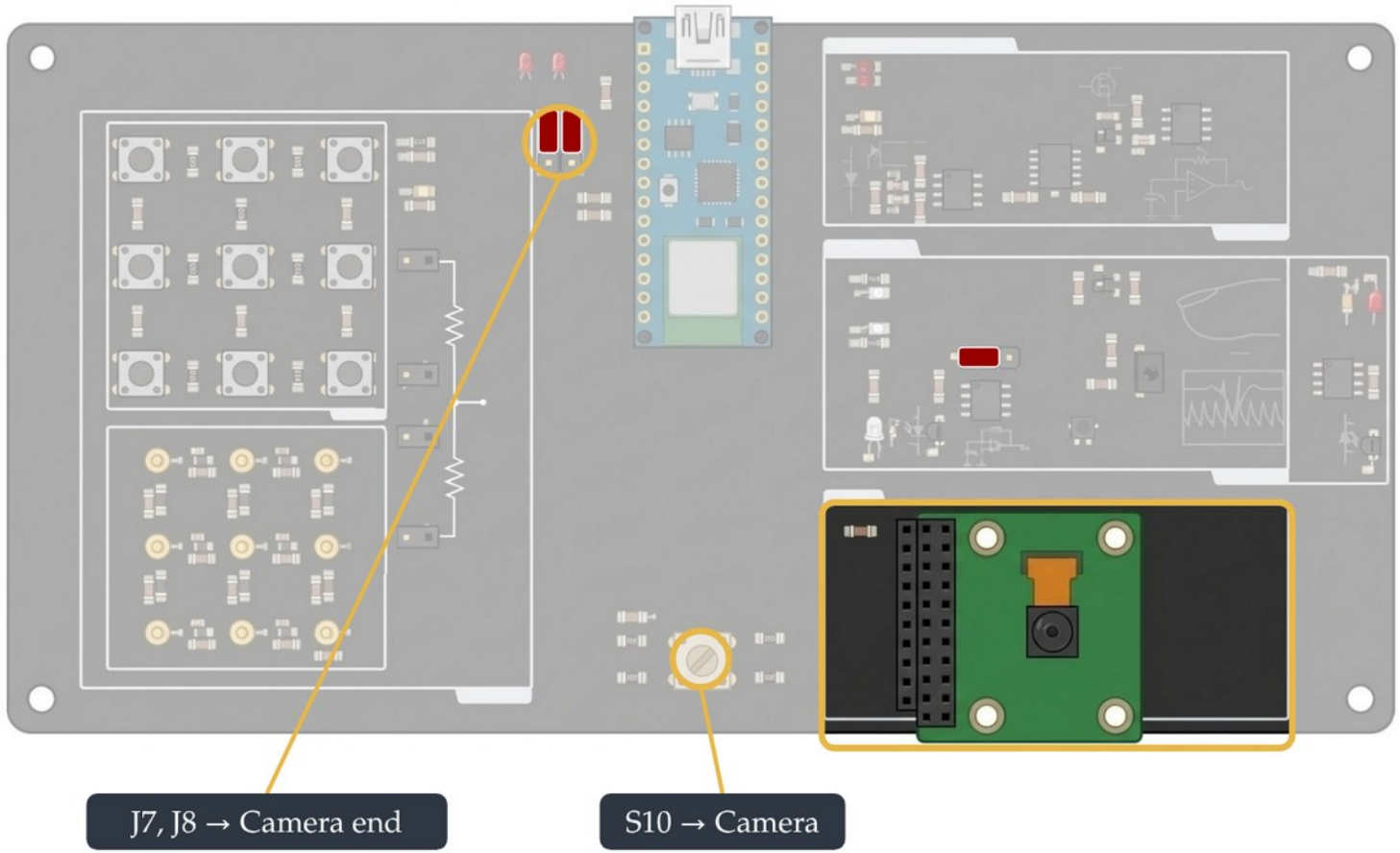
- **25 600 characters per picture.** 2.2 s at 115 200 baud, yet a few hundred ms over the Nano's native USB
- **Match the training data.** 28 x 28, white on black, (x - 0.5)/0.5; without the scaling the accuracy collapses
- **Confidence is the largest softmax output.** 95 % or more for a clear digit, 40 to 50 % for an ambiguous one
- **Domain shift.** 98.5 % on MNIST becomes 60 to 90 % on your camera digits
- **Judge a fix on new pictures.** A crop chosen and scored on the same 20 pictures looks better than it is



E1 on the simulated board: camera digits (top) are smaller and thinner than MNIST (bottom).

SETUP

The camera lab moves the jumper caps. Move them back to Module 1 when you are done.



1. **S10 → Camera**
   Turn the selector to Camera.

2. **J7, J8 → Camera end**
   Both caps on pins 1-2: A0 and A1 go to the camera.

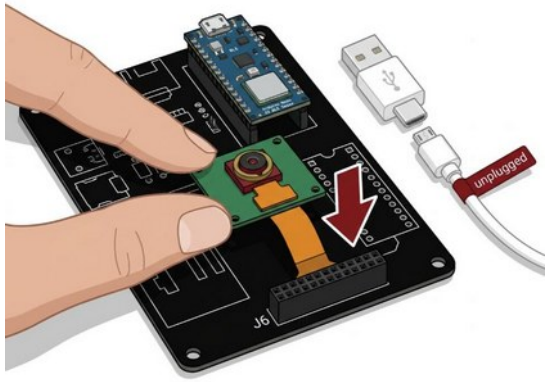
3. **Camera on J6**
   The OV7675, lens facing out. Seat it with the USB cable unplugged.

4. **Digits on paper**
   0 to 9 in thick black marker, one per 4 cm square.

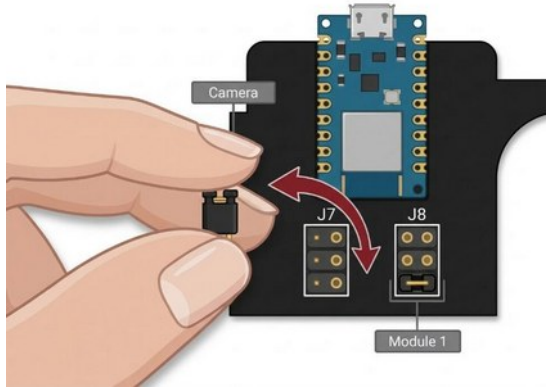
Part A: Connect the camera

Unplug, seat the camera, move the caps, plug back in, run the first cell.

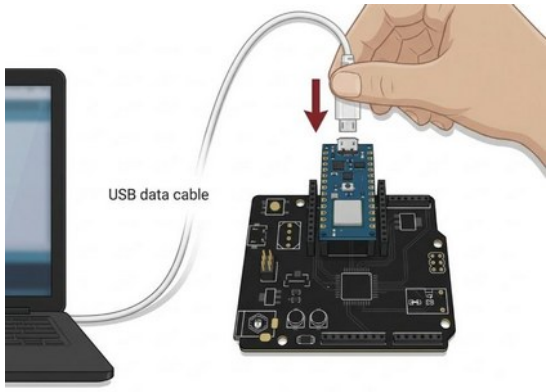
PART A **Goal: A1 · DO** connect the camera and upload its firmware.



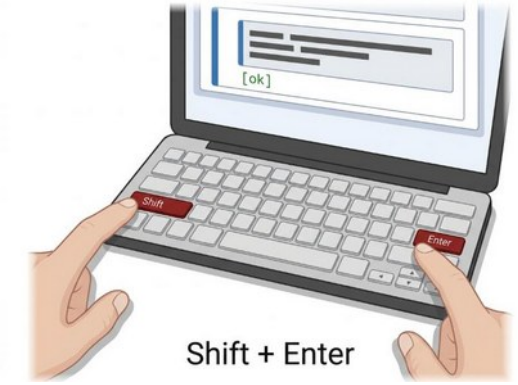
1 USB unplugged: seat the camera on J6



2 J7, J8 (3-pin headers): caps to pins 1-2, the Camera end



3 Plug the USB cable back in



4 Run the first cell with Shift+Enter

You should see: two **[ok]** lines. The first time, **connect_camera** compiles and uploads the camera firmware: about a minute.

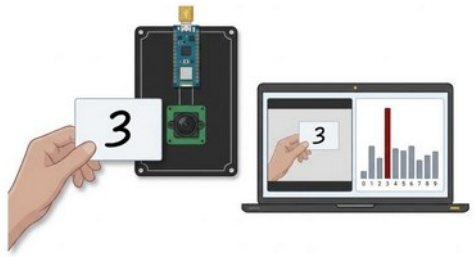
Part B: Look through the camera

The Nano reads a frame, scales it to 160 x 120 and sends it over USB as base64 text.

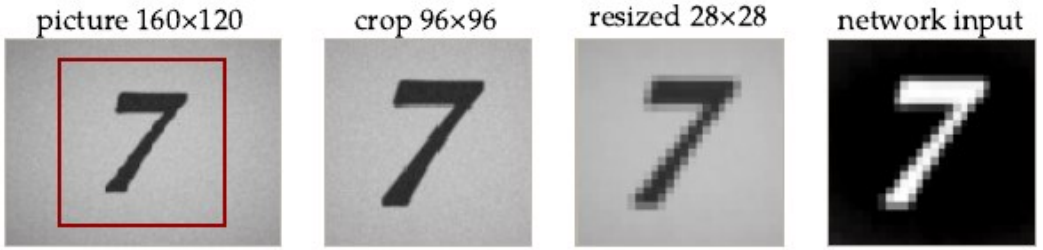
PART B

Goal: look through the camera, time the pictures, make a 28 x 28 input.

- **B1 · DO cam.live(15)**: a digit **in front of the lens**, 10 to 20 cm away
- **B2 · ANSWER** Upside down or mirrored? **FLIP_UD**, **FLIP_LR**
- **B3 · EDIT** Time the pictures; base64 characters and their time at 115 200 baud
- **B4 · DO** Hold up a digit: crop, resize, invert, stretch



Digit facing the lens, filling about half the picture.



160 x 120 picture, center crop, 28 x 28, network input.

You should see: A 160 x 120 picture is **25,600** characters: **2.2 s** at 115 200 baud, yet it takes only a few hundred ms. B4 ends in a **white digit on black** that fills the 28 x 28 input.

Check (Checkpoint 1): the live view shows your digit, and the preprocessing ends in a centered white digit on black.

Part C: Load the network and predict live

The largest of the ten softmax probabilities is the network's confidence.

PART C

Goal: load your Lab 4.1 CNN, predict live, record three predictions.

- **C1 · RUN** Load **../data/models/mnist_cnn.pt**; if it is missing, C1 trains one (15 s)
- **C2 · DO** Live view with predictions: hold up different digits
- **C3 · DO** Three predictions: hold a digit, press Enter; three digits
- **C4 · EDIT** Softmax by hand from the ten logits z; compare with PyTorch

Must match Lab 4.1 training

- white digit on black, 28 x 28
- digit about 20 px tall, centered
- pixels scaled with (x - 0.5) / 0.5
- the same SmallCNN class

You should see: [ok] loaded your Lab 4.1 CNN, about **98.5 %** on MNIST; a thick, clear digit at **95 % or more**, an ambiguous one (an open 3) near **40 to 50 %**.

Check (Checkpoint 2): the live view predicts your digits, and you have three prediction figures.

Part D: Your own test set

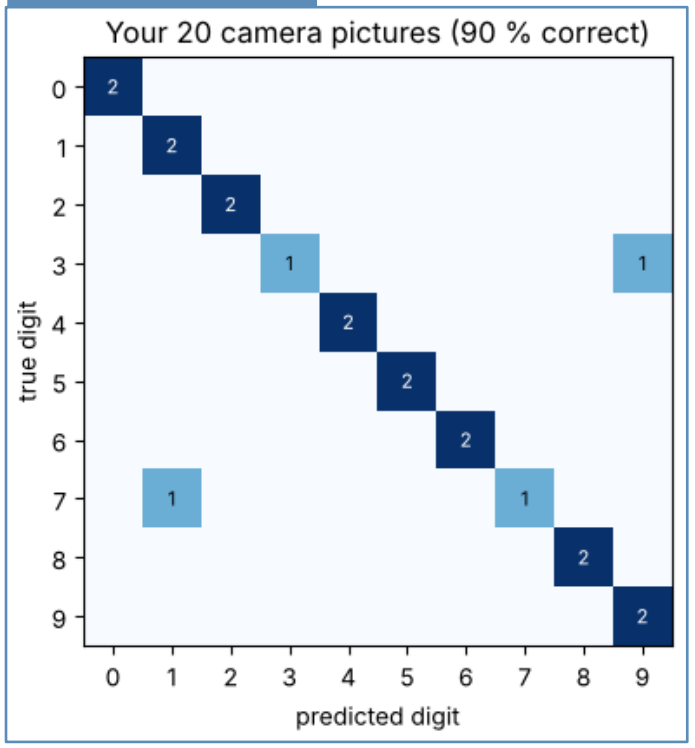
Two pictures of each digit make a 20-picture test set of your handwriting.

PART D

Goal: measure the accuracy and the confusion matrix on your handwriting.

- **D1 · DO** Run **collect**: for each digit, hold it up and press Enter
- Move the paper a little between the two pictures of a digit
- **D2 · RUN** The contact sheet of your 20 inputs
- **D3 · EDIT** Accuracy, and the confusion matrix **cm[true, predicted]**

Expected result · D3



You should see: typically **60 to 90 %** on your 20 pictures, below the **98.5 %** on MNIST. Saved: my_digits.npz and my_digits_results.csv.

Check (Checkpoint 3): the contact sheet, your accuracy next to MNIST's, and the confusion matrix.

Part E: Domain shift and one improvement

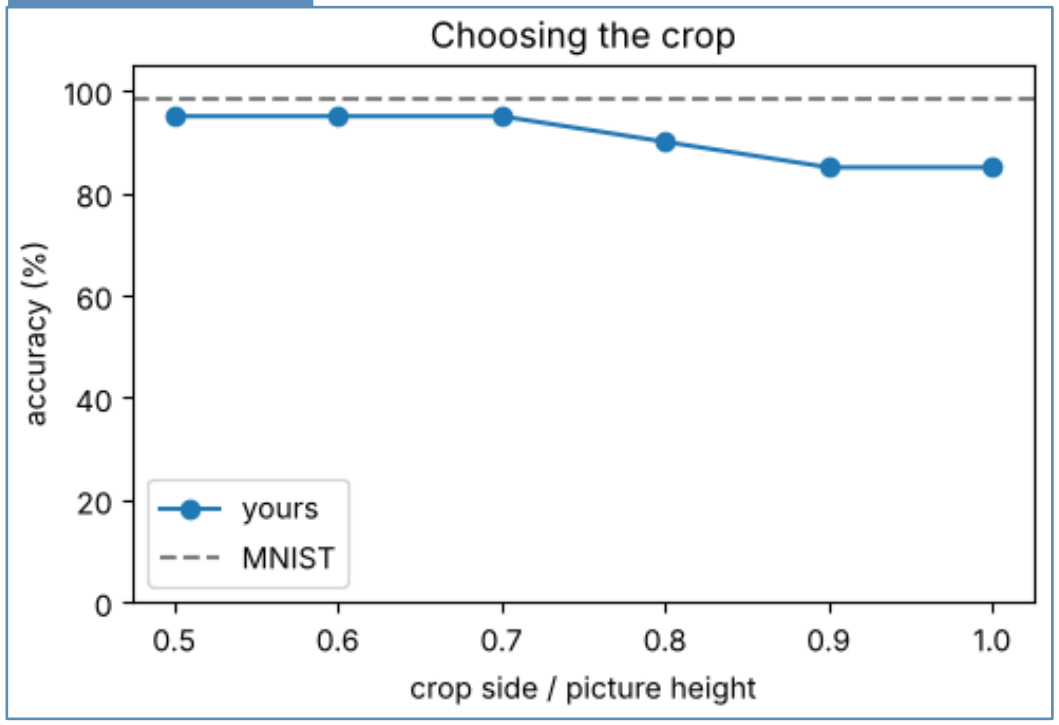
MNIST and your pictures come from different domains. Measure the gap, then narrow it.

PART E

Goal: put numbers on the domain shift and choose a better crop.

- **E1, E3 · RUN** Your inputs next to MNIST; once without the normalization
- **E2 · EDIT** Stroke fraction and digit height, yours against MNIST
- **E4 · EDIT** Crop sweep, 0.5 to 1.0; keep the best crop
- **E5 · DO** A digit written on the spot, read with that crop

Expected result · E4



You should see: camera digits about **14 to 16 px** tall against **20** for MNIST; without normalization the accuracy falls sharply; best crop often **0.5 to 0.7**.

Check (Checkpoint 4): the crop sweep with the best crop, and a live view that reads a digit you just wrote.

Stuck?

Check these first. If the fix does not help, raise your hand.

Camera not found
Unplug USB, reseat the camera on J6 (lens out), plug in, press RESET.

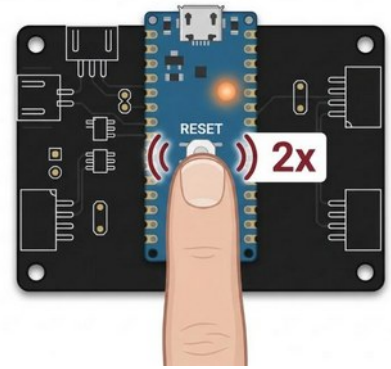
The picture is black or burned out
Change the light or the distance: paper light gray, ink dark.

The digit is upside down or mirrored
Set FLIP_UD or FLIP_LR to True in B2 and rerun it.

[!!] mnist_cnn.pt not found
Finish Lab 4.1 (G3), or let C1 train one now (15 s).

Poor accuracy on your digits
Thick marker, digit centered and half the height, no glare; try the crop sweep.

A [!!] line about the selector
Only a warning. Check S10 on Camera and both caps on the Camera end.



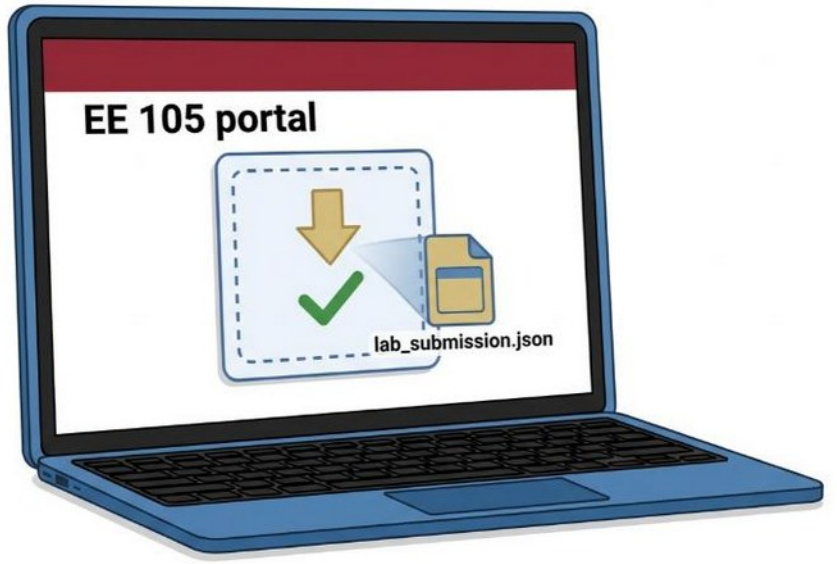
Upload stuck? Double-tap RESET, then rerun.

Submit your results

The last cell saves your results in one file. Upload it on the course portal.

`lab4.2_submission.json`

1. **Save · RUN** writes **lab4.2_submission.json** next to the notebook
2. It holds your numbers, the labels and predictions of your 20 pictures, and their 28 x 28 inputs
3. Log in at **ee105.org/portal** with your student ID and portal PIN; open Lab 4.2
4. Upload the file. It is graded automatically; you can upload again after a fix



ee105.org/portal

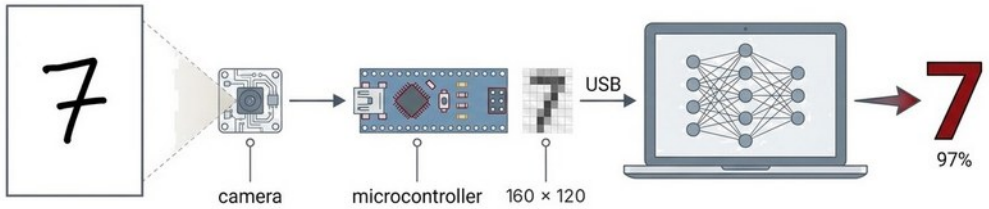
Check: before you leave, move both caps of J7 and J8 back to the **Module 1** end (pins 2-3).

Lab complete

You built a camera that reads handwriting with your own neural network.

You built:

- A live camera view inside the notebook
- One preprocessing for photos and camera pictures
- A 20-picture test set and its confusion matrix
- A crop study of the domain shift



Camera, microcontroller, laptop: 97 % on a 7.

Did you know?

Running machine learning on a microcontroller is called TinyML. The Nano has 256 KB of RAM; your CNN needs 808 kB in float32, so it would first be cut to 8-bit numbers (Chapter 4).

Next

Lab 5.1, LiFi: send text from one board to another with light. Caps of J7 and J8 back on the Module 1 end now.