University of Missouri

Neural Networks: Spring 2024

Project 1: Supervised Learning

Goal:

Project 1 asks you to (i) implement a supervised learning-based neural network. You will dive into (ii.a) classification, (ii.b) regression, and (iii) qualitatively (via visual aids like your eyes and plots) and quantitatively (metrics like confusion matrices and F1-score) analyze your various experiment outcomes. Your aim should surpass the mere pursuit of a numerical accuracy like "96.456784." Instead, concentrate on the implementation, experimentation, and exploration aspects. I value your journey and insights more than the ultimate results. Embrace the process and thoroughly investigate!

Details:

You will implement, in whatever language you like (but I will only *code support* (fyi, I don't program, you do, but I MIGHT answer Qs…) if you pick Python), a neural network "library" to tackle the following:

1. Classification
    a. Sanity check and sandbox
        - [Sanity check] The goal here is for you to make sure that your code is working before you try something more complex, e.g., MNIST
        - You can download a dataset, or you can make your own. I recommend the latter! Recommend you pick two dimensions – so you can SEE it.
            - Hint, in the "old days" I opened Paint and made a black image and class 1 was blue and class 2 was red pixels and … Then you write five lines of Python code to read in image and get coordinates for each class (color). You have a dataset and you can draw it 1,000,000,000,… (as much as you want).
        - [Sandbox] This brings us to your second goal. You should not only use this toy data to sanity check your code, i.e., can you learn a solution (decision boundary) to the data you present the NN, you should vary the dataset (e.g., make it harder, add more classes, etc.) and EXPLORE the code you created!
            - E.g., make a linear separable two class problem to start, then do three classes, then try to make them non-linearly separable, etc.

    b. MNIST (digit or fashion) or CiFAR dataset
        - OK, let's try something a little more *real*
        - http://yann.lecun.com/exdb/mnist/
        - (60,000 train, 10,000 test) samples, digits {0, 1, 2, …, 8, 9}
            - Depends on what type of fancy computer you have (or don't have). I am ok if you just end up running on say the first N=small samples from each class and test on whatever size you can handle. Yes, this will bias your outcome, but if you don't have the computing resources, it's what we have… Note, you might want to ensure that you have a near equal sampling of class instances.
        - Input size is 28x28 (so 28*28 = 784)
            - You could consider reducing the input size, e.g., call a Python function to convert the 28x28 images into something like 14x14 via something like

bilinear interpolation or nearest neighbor sampling. Note, while this reduces the dimensionality of your input it comes at the cost of information loss. I have not explored NN performance on MNIST as a function of input size, but I know it can be decreased with not too much error.

- In Jupyter, I provided codes to load this dataset and work with it
    - Note, you can find a LOT of codes on the internet in MANY languages to load the dataset and visualize them (and their results).
- You can build a MLP to tackle this or a CNN (what we are covering now/next!)
    - What questions can you explore? (thanks for asking!)
        a. Can you learn a network that classifies "1"s vs "8"s?
        b. Can you learn a network that classifies "3"s vs "8"s?
        c. Can you learn a network w.r.t. {0, 1, 2, …, 8, 9}?
        d. What is the impact of "hyperparameters": nonlinearities, number of layers, weight init, input scaling, criteria function, learning algorithm, etc.?
        e. Can you solve this using a single convolutional weight layer? What filters were learned (you can visualize them!)? If you add more layers, does it reduce error rate, simplify learning, what?
        f. Do you have a black box? Can you explain how your net works?
        g. Common gang, use your brain, I gave you a visual dataset and there are million fun questions you can explore !!!
- For you over achievers, if you have already looked at MNIST, try something a little harder (still simple in NN land nowadays … 😊) : Fashion MNIST (https://www.kaggle.com/zalando-research/fashionmnist) or CiFAR-10 or CiFAR-100 (https://www.cs.toronto.edu/~kriz/cifar.html)
- For you super over achievers, you can use TSNE, https://danielmuellerkomorowska.com/2021/01/05/introduction-to-t-sne-in-python-with-scikit-learn/, to visualize your input, weights, and/or "projections" (outputs).

2. Regression
    a. Sanity check and sandbox
        - Like above, make a 2D dataset, e.g., pick some linear or non-linear function and sample it!!! That is your dataset, the (x,y) values (sampled locations) and function output value! Have fun. Play with a simple linear to start then make more complicated non-linear functions with more than one output.
    b. Real dataset
        - "Easy"

    - Look at https://archive.ics.uci.edu/ml/index.php (or you can use another one!). Note, real datasets often have missing data, missing features, etc. You might want to find a "clean dataset", or I will allow you to find a dataset already cleaned (you don't have to do that). I just want something with higher dimensions and complexity that did not make and cannot directly see!

- Harder

    - Note, if you realllly want to go challenging, ☺, you can use the KITTI or synthetic KITTI data set. The first is an input image with sparse depth ground truth from lidar. The latter is a dense depth acquired using simulation. The goal is for the network to basically learn features then map and regress to depth. So, you have a feature extraction then real-valued depth regression task. But, like I said, it's not easy ☺

    - Note, there are 'fun' datasets out there, e.g., material science, which I can point you to, that are core regression problems (variable prediction tasks).

Do What???:

If you took Intro to Computational Intelligence, do either 1 or 2:

1. Do (sanity check, sandbox) classification and digit MNIST (or Fashion MNIST, CiFAR).
2. Do (sanity check, sandbox) classification and hard/challenging regression (e.g., KITTI).

If you did **NOT** take Intro to CI, do either 3 or 4:

3. Do (i) (sanity check, sandbox) classification and (ii) (sanity check, sandbox) regression and (iii) one real (easy or hard) classification or regression dataset.
4. ORRRR, do 1 OR 2.

Expectations:

I am happy if you complete the above. Really. However, I will grade you based on your report (see below). Your goal is not to implement and run the above and be done. Your job is to vary "params" (net architecture, learning algorithm, etc.) and evaluate your results in different ways, e.g., summarized F1-score, confusion matrix analysis, show weights, … get creative! *At the end of the day, I will be looking for how well you understand these tools, are able to apply them, and can understand what is going on (to the best of your ability) based on analysis of (input, parameters, output)!!!!*

Code:

You must program this yourself. Here are DOs and DONTs:

NOT ALLOWED:

- **No** working with other humans, robots, aliens, internet, ChatGPT … this is a YOU project!
    - NO code generation from ChatGPT, "sorry"
- **NO** neural network libraries (beyond the PyTorch (or equivalent) discussed in class)
    - NOTE, if you want an exception, e.g., to do a specific task like plotting network weights or confusion matrices, ask now!!! If I approve anything, it will be for the **entire** class.
- **NO** codes from online. What you use as code YOU need to write. Note, you can look online and learn about programming and neural network, but you **cannot** simply copy and paste nor leverage other codes as your own.
    - Example? Its ok to go online and learn about autograd, or nn.optimize. Its ok to look at how to format your data to pass to net.eval(). It's NOT ok to take a code example online that loads MNIST, does mini batch, and plots the result. This is you getting the entire program from online. I do want you to be able to look online and learn about specific things, e.g., what does torch.variable do, but you cannot get your program from online.

ALLOWED:

- You CAN use existing libraries to load data (e.g., load MNIST) and analyze results (e.g., plot).
    - These are tedious things that we find important, but I don't care that you code up.
- You CAN use my "BASIC PYTORCH" examples.
    - E.g., you can use and extend my MlpInPyTorch.ipynb and SimpleCNN.ipynb examples.
    - If you pick another language, you are allowed similar functionality

Why ?!?! Well, I use to make you code everything 100%... It took forever (was a nice exercise) and y'all did not get very far as a result ... Many of you also were unable to produce numerically stable codes (crashed all the time, e.g., Inf's or Nan's coming out of nonlinearities). So, I embraced our new NN programming era. But it's a tightrope. I want you to use as basic of functionality as possible and still program up a NN. I just want to provide you more stable codes and simplify some things (e.g., loading datasets). What is your best bet? Start from scratch or start from my examples and go from there. ***AND IF I did not cover a question that you have, ASK!!! I will not penalize for asking. But, if you cross the line and ask for forgiveness, that is another thing… Overall, gang I am trying to work with you and help you.***

Final Report:

<u>Remember, this is what your **grade comes from** !!!!!</u>

Turn in a report (double spaced, one column, Times New Roman 12-point font) in the following format:

1) Title Page (1 page)

- name, pawprint, date

2) Technical Description (5 pages minimum and 10 pages max)

- high-level details (e.g., text description and figures)
- low-level details (e.g., equations and algorithms)
- use text, equations, figures, etc.
- end of day, do I believe you understand this stuff?!?!

3) Experiments and Results (8 pages minimum and 15 pages max)

- what did you run?
- why did you run it?
- discussion, tables, plots, etc.
- co-"show/plot" as much as possible (comparison)
- do NOT DUMP data/results, find the STORY and report those parts
- **think then write!**

4) Reflection (2-4 pages)

- quick summary paragraph
- did everything turn out like you thought it would? surprises? etc.
- discuss shortcomings
- discuss what you would do in the future if you had more time