CSC420: Tutorial 01

by Michael Neumayr

Overview

- this week, we get our hands dirty in the tutorial notebook
- the notebook covers:
  - setting up the environment and jupyter notebook
  - PyTorch basics
  - basics working with images

  - building and analyzing a simple MLP
  - designing appropriate MLPs for example problems

P4: MLP

- given a 2-layer MLP: $f = W_2 max(0\ ,W_1 x + b_1) + b_2$
- input x: quadratic image with side length of 16 pixels
- output y: scalar value
- hidden dim: 512

- what are the dimensions of W_1, b_1, W_2, b_2?

P4: MLP

- given a 2-layer MLP: $f = W_2 max(0, W_1 x + b_1) + b_2$
- input x: quadratic image with side length of 16 pixels
- output y: scalar value
- hidden dim: 512

- what are the dimensions of W_1, b_1, W_2, b_2?
- flatten image to 1D vector: x: [256]
- W_1: [512, 256] b_1: [512] W_2: [1, 512] b_2: [1]

P4: MLP

- given a 2-layer MLP: $f = W_2 max(0\ ,W_1 x + b_1) + b_2$
- input x: quadratic image with side length of 16 pixels
- output y: scalar value
- hidden dim: 512

- how many learnable parameters does this 2-layer MLP have?

P4: MLP

- given a 2-layer MLP: $f = W_2 max(0\ ,W_1 x + b_1) + b_2$
- input x: quadratic image with side length of 16 pixels
- output y: scalar value
- hidden dim: 512

- how many learnable parameters does this 2-layer MLP have?
- 512x256 + 512 + 1x512 + 1 = 132,097 (can quickly reach millions!)

P4: MLP

- given a 2-layer MLP: $f = W_2 max(0\ ,W_1 x + b_1) + b_2$
- input x: quadratic image with side length of 16 pixels
- output y: scalar value
- hidden dim: 512

- what is the FLOP count for one forward pass?

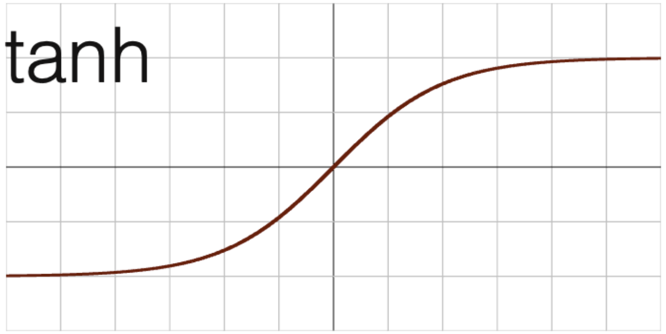
P4: MLP

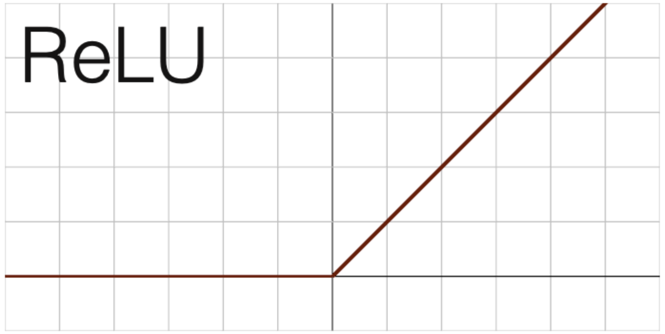
- given a 2-layer MLP: $f = W_2 max(0\ ,W_1 x + b_1) + b_2$
- input x: quadratic image with side length of 16 pixels
- output y: scalar value
- hidden dim: 512

- what is the FLOP count for one forward pass?
- W_1x: 512 x 256 multiplications + 512 x (256 – 1) additions
- + b_1: 512 additions
- max(): 512 max ops (might be ignored in profiling libraries but not in assignment!)
- W_2z: 1 x 512 multiplications + 1 x (512 – 1) additions
- + b_2: 1 addition
- overall: 263,680

P5: Practical Classifier

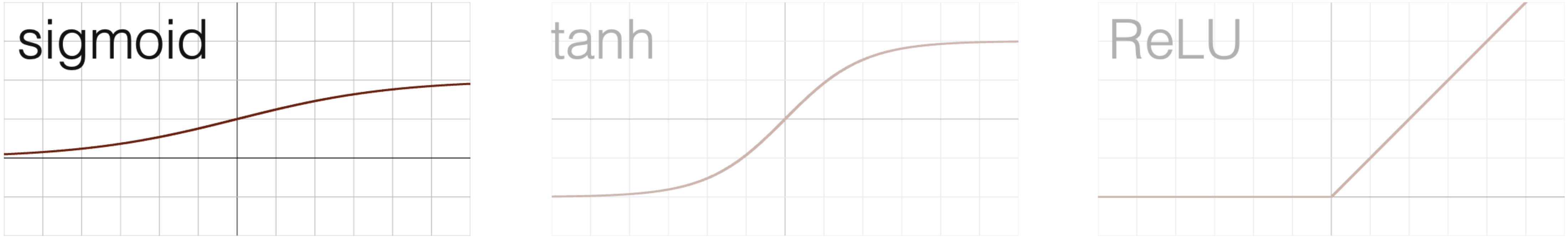
- add activation: $f = activation(W_2 max(0\ , W_1 x + b_1) + b_2)$
- for the following activations, decide:
  - can we use it to train a useful binary classifier? (output: true/false)
  - why or why not?
  - which one do we use?





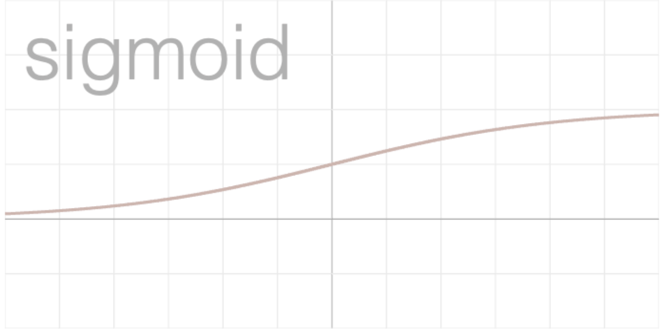
P5: Practical Classifier

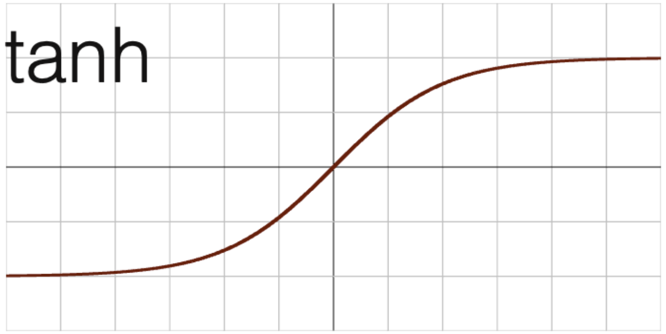
- add activation: $f = sigmoid(W_2 max(0\ ,W_1 x + b_1) + b_2)$
- binary classifier straight-forward: threshold output with 0.5
- directly outputs probability (trained with binary cross-entropy loss)



P5: Practical Classifier

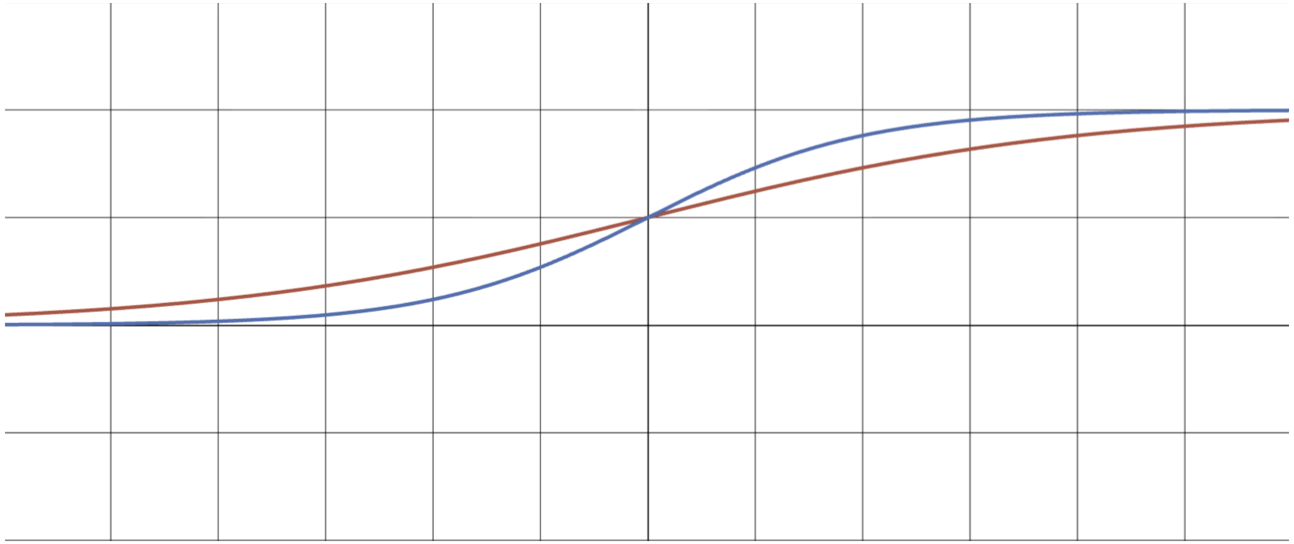
- add activation: $f = tanh(W_2 max(0, W_1 x + b_1) + b_2)$
- yes: convert to probability by (tanh(y) + 1) / 2 -> train like sigmoid





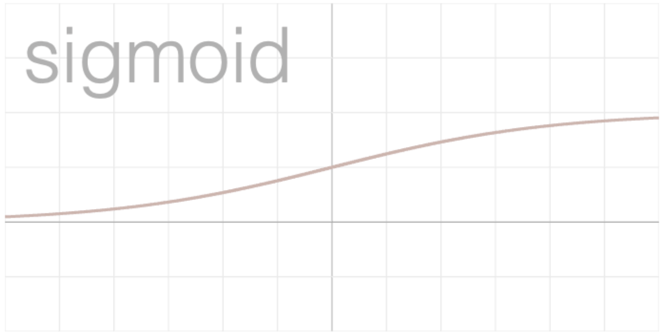
P5: Practical Classifier

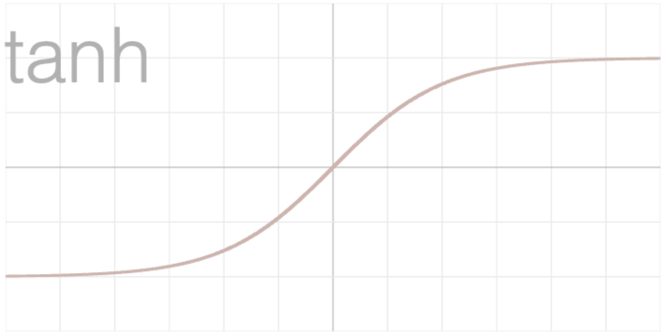
- add activation: $f = tanh(W_2 max(0, W_1 x + b_1) + b_2)$
- yes: convert to probability by (tanh(y) + 1) / 2 -> train like sigmoid
- but: gradients saturate faster (shifted and scaled tanh in blue)

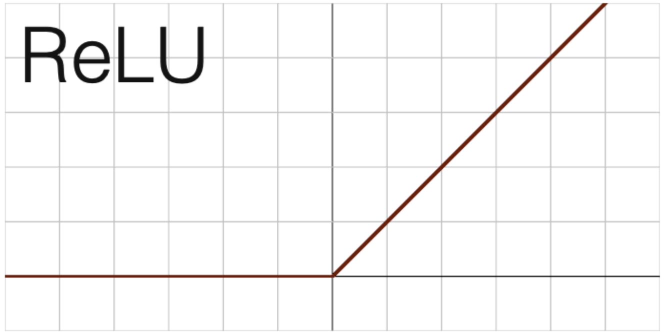


P5: Practical Classifier

- add activation: $f = ReLU(W_2 max(0, W_1 x + b_1) + b_2)$
- not useful: can in principle force a binary decision
- but in training: when positive sample is <0, the gradient through ReLU is 0 → training stalls







P5: Neural Field

- what is the input dimension when using sinusoidal positional with L=8 frequencies for a neural field that fits a 3D shape?

P5: Neural Field

- What is the input dimension when encoding 3D coords with L = 8-frequency sinusoidal positional encoding in a neural field?

- original coordinates: 3 frequency bands: 8
- pe_input = [[x, y, z],
  [sin(x * 2**0 * π), sin(y * 2**0 * π), sin(z * 2**0 * π)],
  [cos(x * 2**0 * π),cos(y * 2**0 * π),cos(z * 2**0 * π)],
  [sin(x * 2**1 * π), sin(y * 2**1 * π), sin(z * 2**1 * π)],
  [..., ..., ...],
  [cos(x * 2**7 * π),cos(y * 2**7 * π),cos(z * 2**7 * π)]]
- overall: 3 + 2 * 8 * 3 = 51