15. Deep Neural Networks: Convolutional Neural Network Part 2

M.A.Z. Chowdhury and M.A. Oehlschlaeger

Department of Mechanical, Aerospace and Nuclear Engineering
Rensselaer Polytechnic Institute
Troy, New York

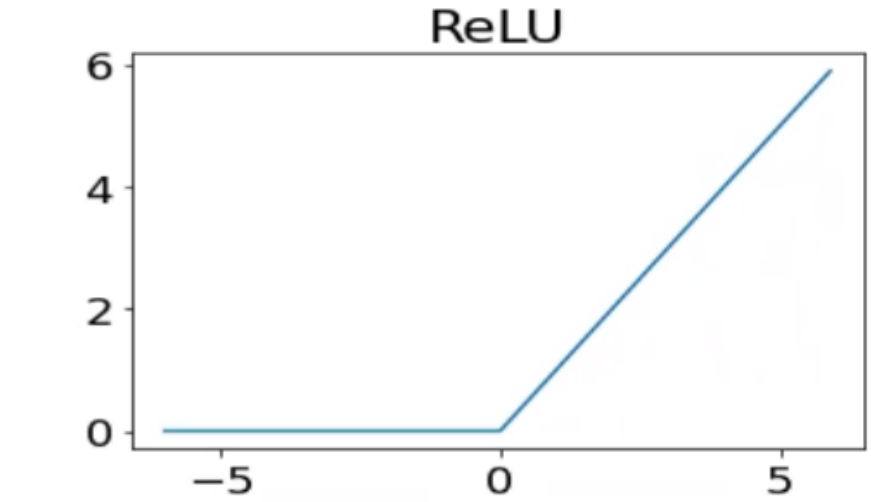
chowdm@rpi.edu
oehlsm@rpi.edu

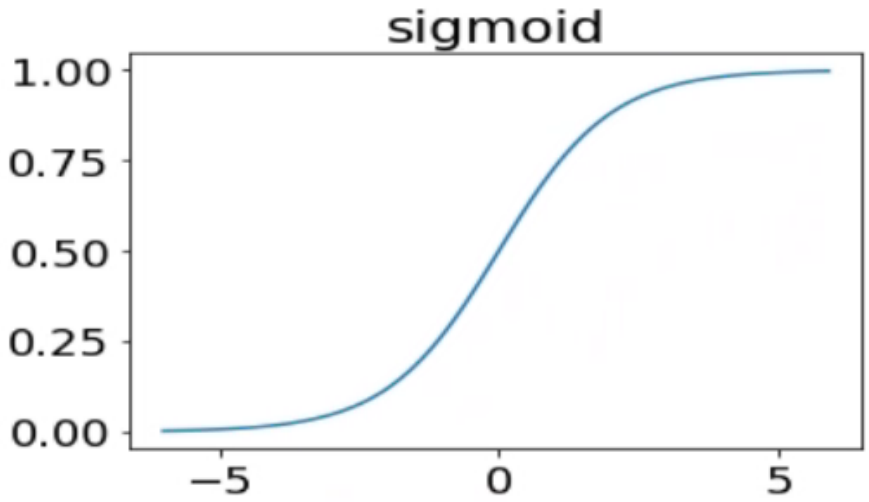
MANE 4962 and 6962

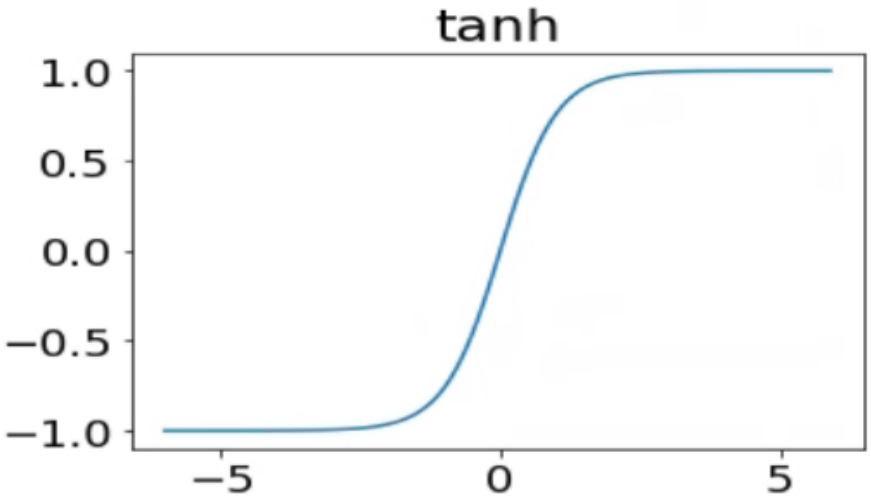
Regular announcement

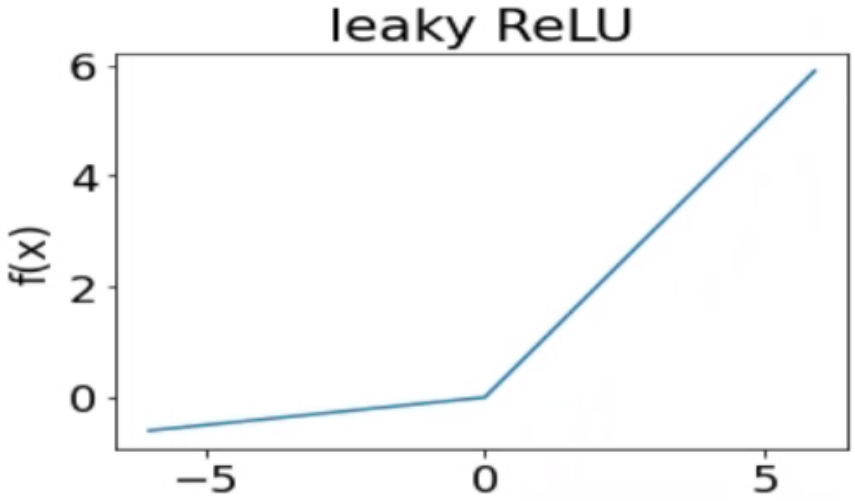
- HW 5 is due March 20, 2023
- Quiz 5 on March 20, 2023
- No office hours on Wednesday, March 15, 2023
- Office hour will be on Friday, March 17, 3-5 PM
- Syllabus
- Final Exam date
- Presentation days
- Project
- HW 5 discussion

Which activation functions to use?



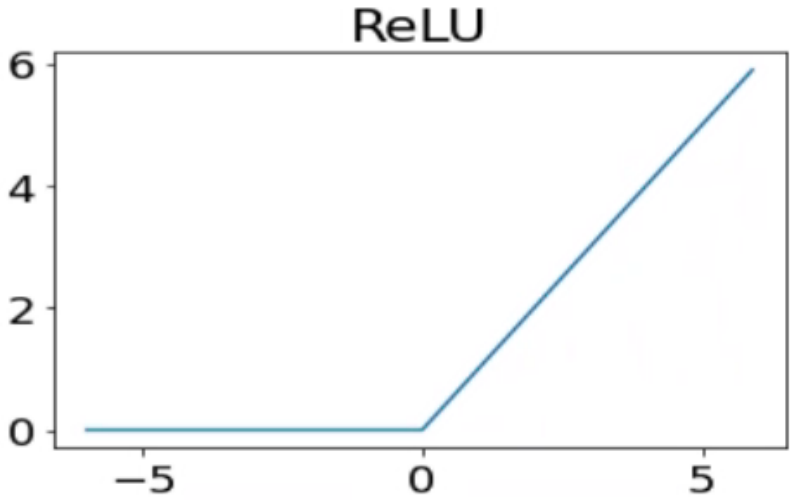


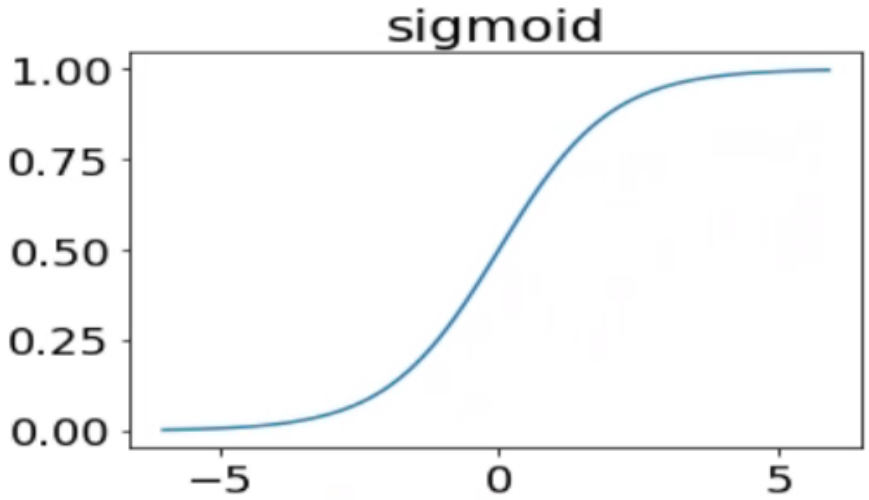


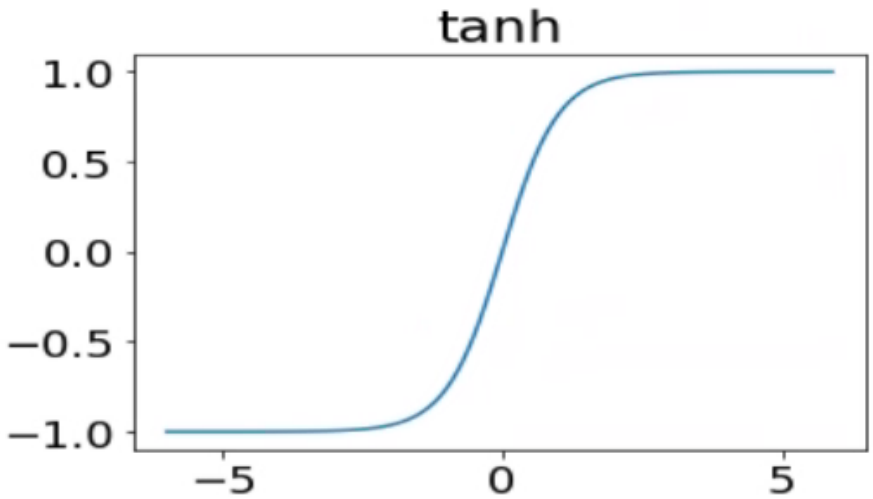


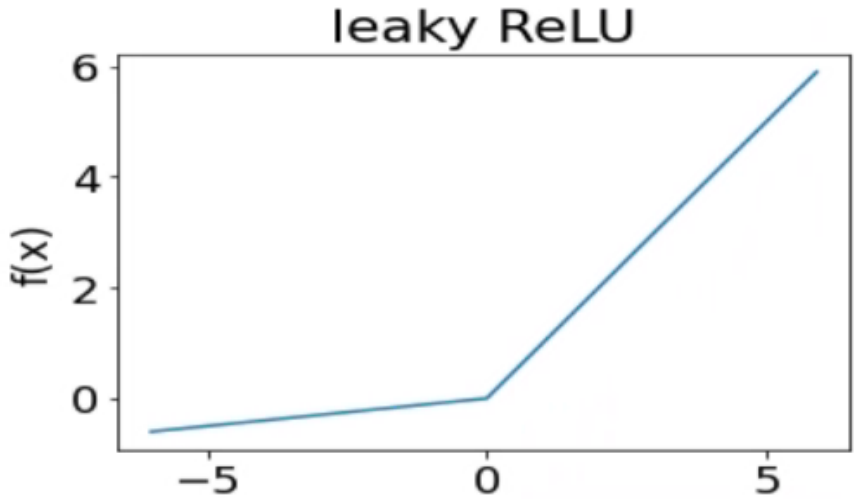
What about the rest of the network after convolutions and pooling?

Which activation functions to use?









Parametric ReLU $\sigma_{PReLU}(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \alpha x, & \text{otherwise} \end{cases}$ $\alpha = 0.01, \quad (LeakyReLU)$

Characteristics of the activation functions

- How quickly they saturate, i.e., gradient of the activation function becomes zero w.r.t. input?
- Both sigmoid and tanh suffers from saturating too quickly. Both also change very rapidly for small positive inputs.
- ReLU will ignore neurons with negative values.
- PReLU or LeakyRelu both solves the problem near zero input neurons.

Which optimizers to use?

- BGD, mBGD, SGD
- Momentum
- Nesterov Accelerated Gradient
- AdaGrad
- AdaDelta .
- RMSProp
- Adam
- AdaMax
- Nadam
- and more

Data Preprocessing

- Normalization
- Noise reduction
- Data augmentation : Can artificially create image data which will increase the size of the training data and improve learning process.

How to regularize the model?

- L1, L2, L1+L2, Early stopping
- Dropout regularization: For a deep network arbitrarily setting some weights in a layer to zero is beneficial.
- Trains multiple smaller networks in an **ensemble**.

Convolutional and pooling layers in Tensorflow

```
model.add(Conv2D(filters=3, kernel_size=(3, 3),
strides=(1, 1), padding='valid',
activation='relu', input_shape=(28, 28, 1)))

model.add(MaxPooling2D(pool_size=(2, 2),  strides=(2,2)))
```

This is for convolution and pooling on 2D images.

Classifying MNIST digits

```
# CNN model
model = Sequential()
model.add(Conv2D(filters=3, kernel_size=(3, 3),strides=(1, 1), padding='valid',
activation='relu', input_shape=(28, 28, 1)))
model.add(MaxPooling2D(pool_size=(2, 2),  strides=(2,2)))
model.add(Conv2D(filters=3, kernel_size=(3, 3),strides=(1, 1), padding='valid',
activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2),  strides=(2,2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(10, activation='softmax'))

# Compile model
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

Check the notebook

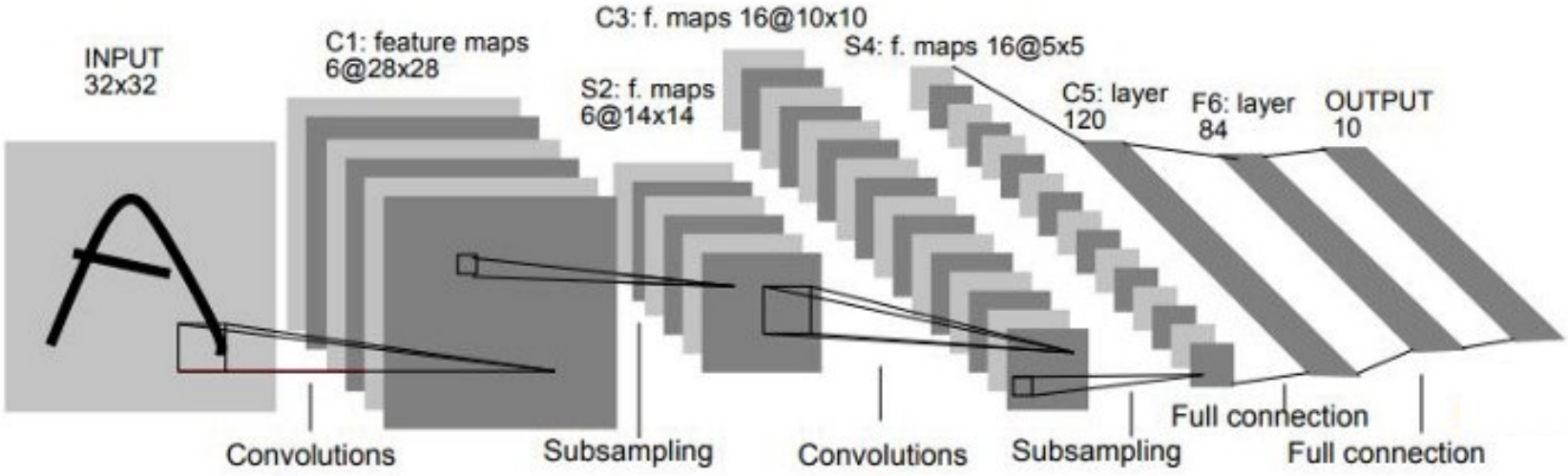
Batch normalization

- Normalize the output after linear combination and before adding non-linearity. (Ioffe and Szegedy, 2015)
- By normalizing the inputs, batch normalization helps to stabilize the training process, improve the gradient flow, and increase the speed of convergence.
- tf.keras.layers.BatchNormalization()

Initialization

- Must be non zero
- Randomized
- Glorot/Xavier initialziation: aims to set the initial weights from an uniform distribution for the the network so the variance of the activations is preserved across each layer.
- If the variance is too high or too low, it can lead to vanishing or exploding gradients during training, making it difficult for the network to learn.
- He: He initialization, is specifically designed for rectified linear units (ReLU).

LeNet-5



Source: Yann LeCun

LeNet-5 Summary

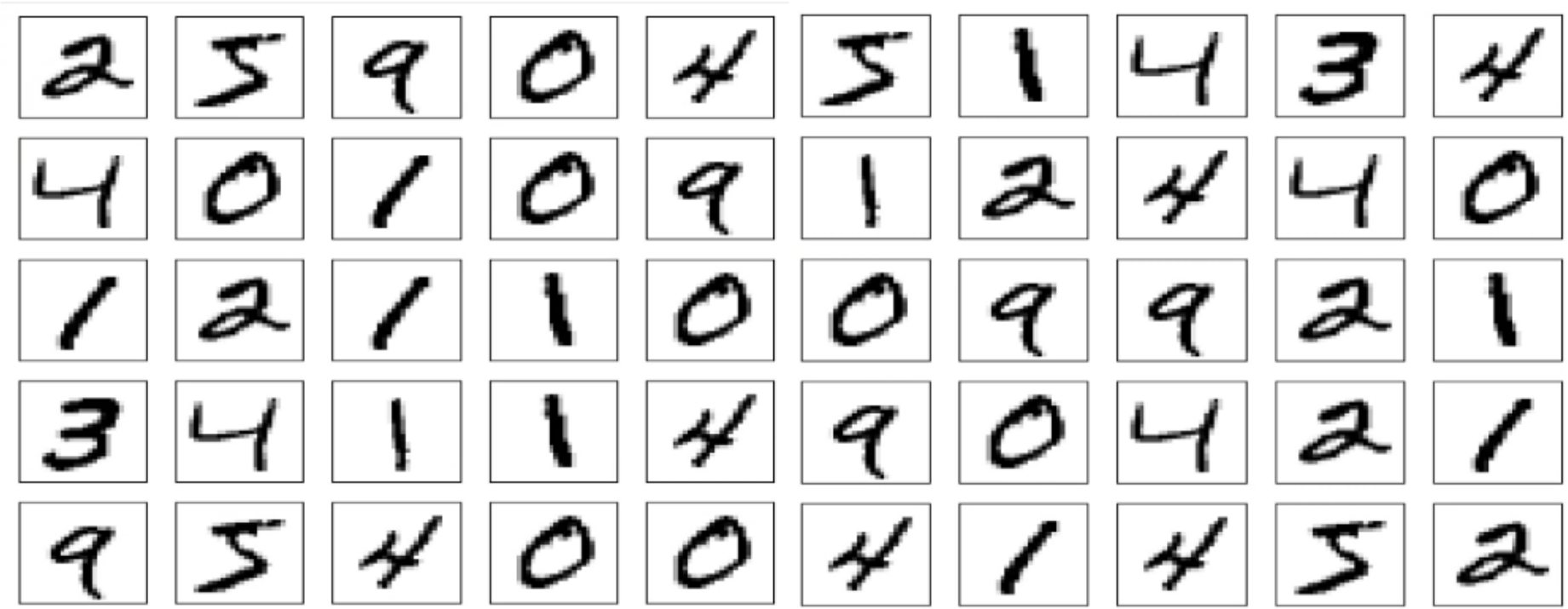
Layer (Type)	Output Shape
Input (32x32x1)	(32, 32, 1)
Convolution (6x5x5 filters)	(28, 28, 6)
Max Pooling (2x2)	(14, 14, 6)
Convolution (16x5x5 filters)	(10, 10, 16)
Max Pooling (2x2)	(5, 5, 16)
Flatten	(400)
Fully Connected (120 units)	(120)
Fully Connected (84 units)	(84)
Output (10 units)	(10)

Pretrained convolutional neural networks

- Resnet
- VGG,
- AlexNet
- GoogLeNet

```
import tensorflow as tf
from tensorflow import keras
from keras import applications
```

A key advantage of CNNs: Parameter Sharing



NN successes

- Accelerating analysis and scientific discovery
- Deep learning scales well with the size of the training data
- Engineers features
- Performs very well with signal/image/video or high-dimensional data. Voice/Face recognition. Video segmentation, speech/video to text conversion are some of the crowning successes.

NN weaknesses

- Requires a lot of training data
- Lots of model parameters
- Computational resources
- Understanding "bad" or "wrong" predictions