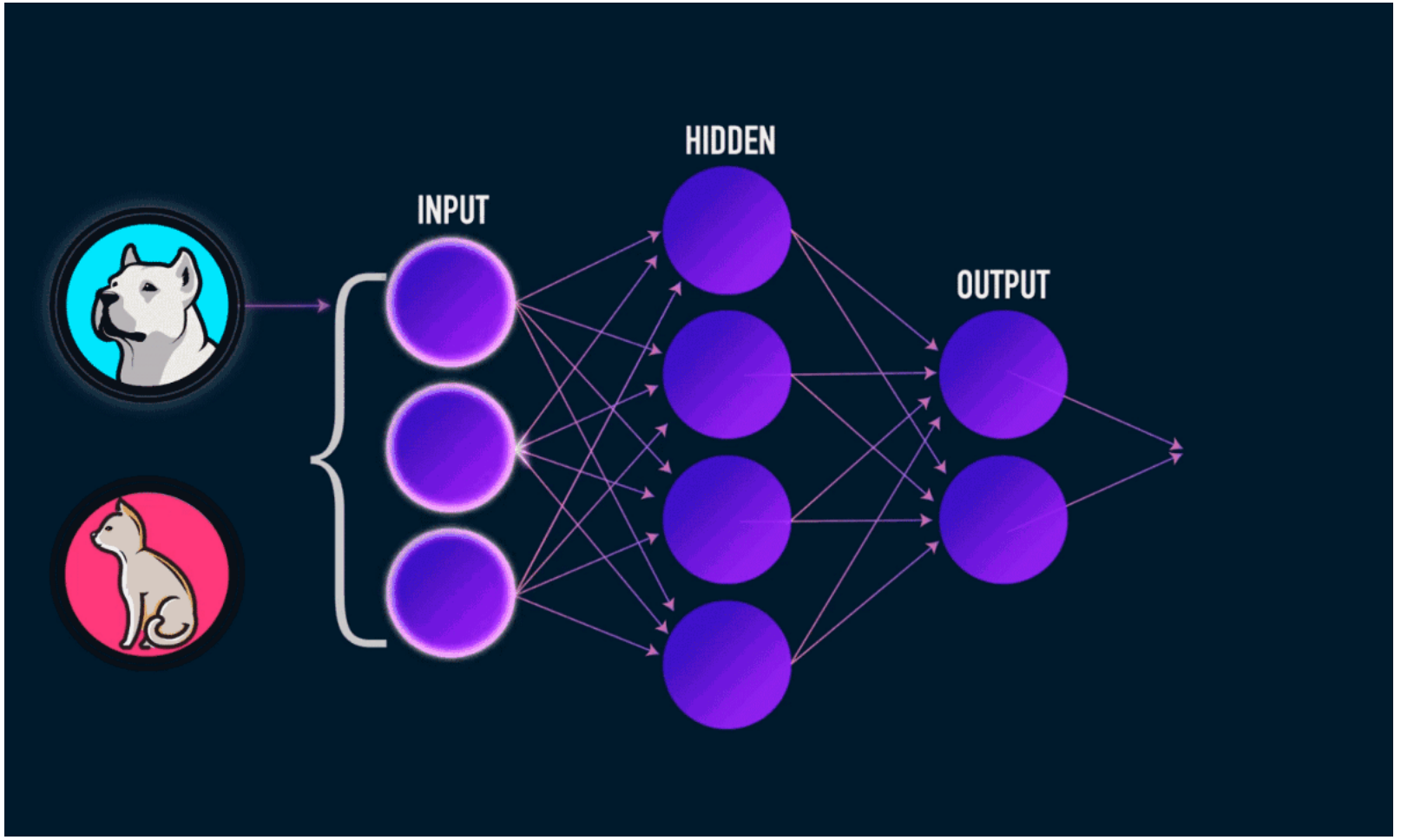
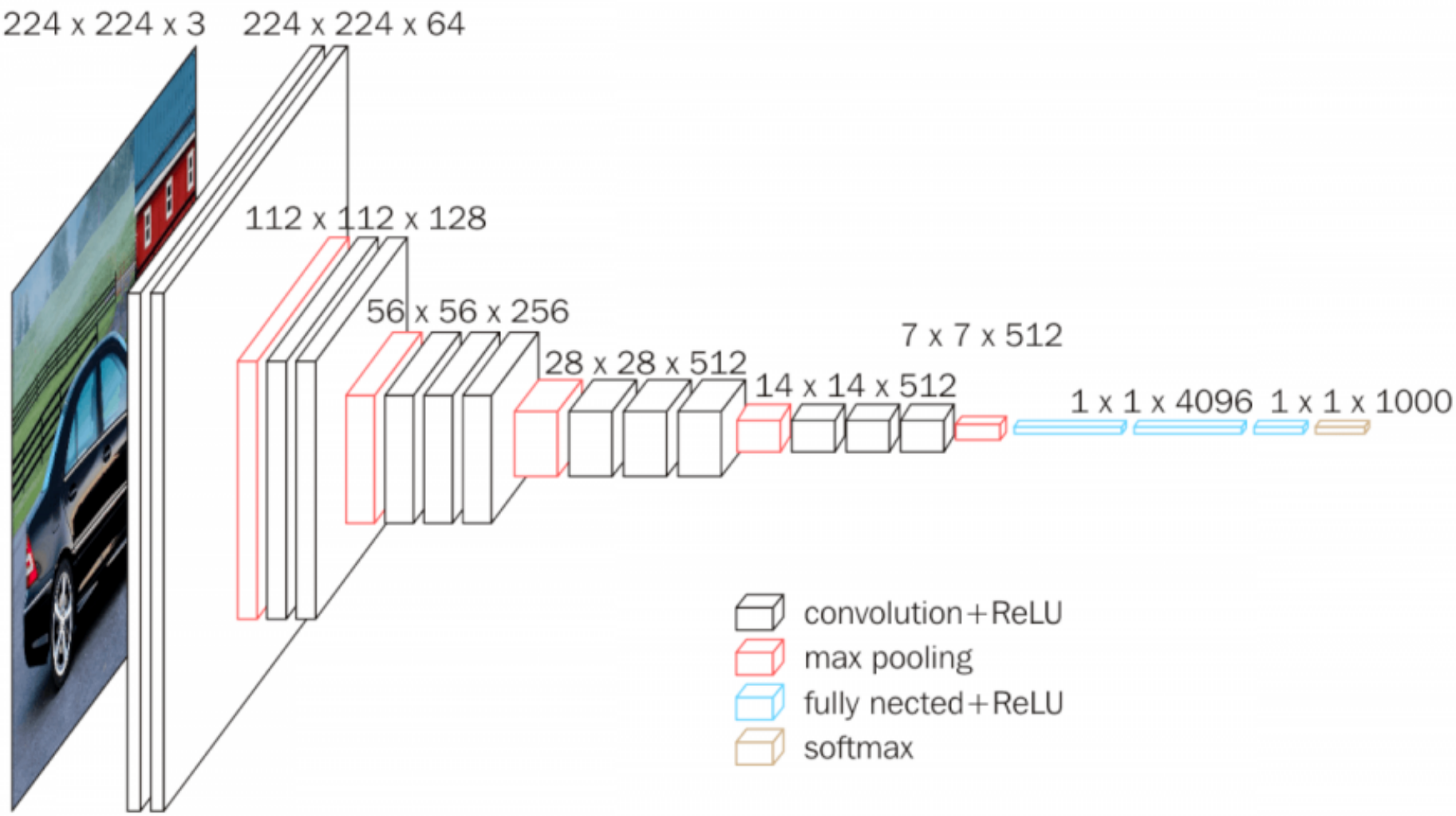
TACC Summer Machine Learning Institute: Deep learning 2

Weijia Xu
Research Scientist, Group Manager
Scalable Computational Intelligence
Texas Advanced Computing Center
University of Texas at Austin

TACC


HIDDEN
INPUT
OUTPUT

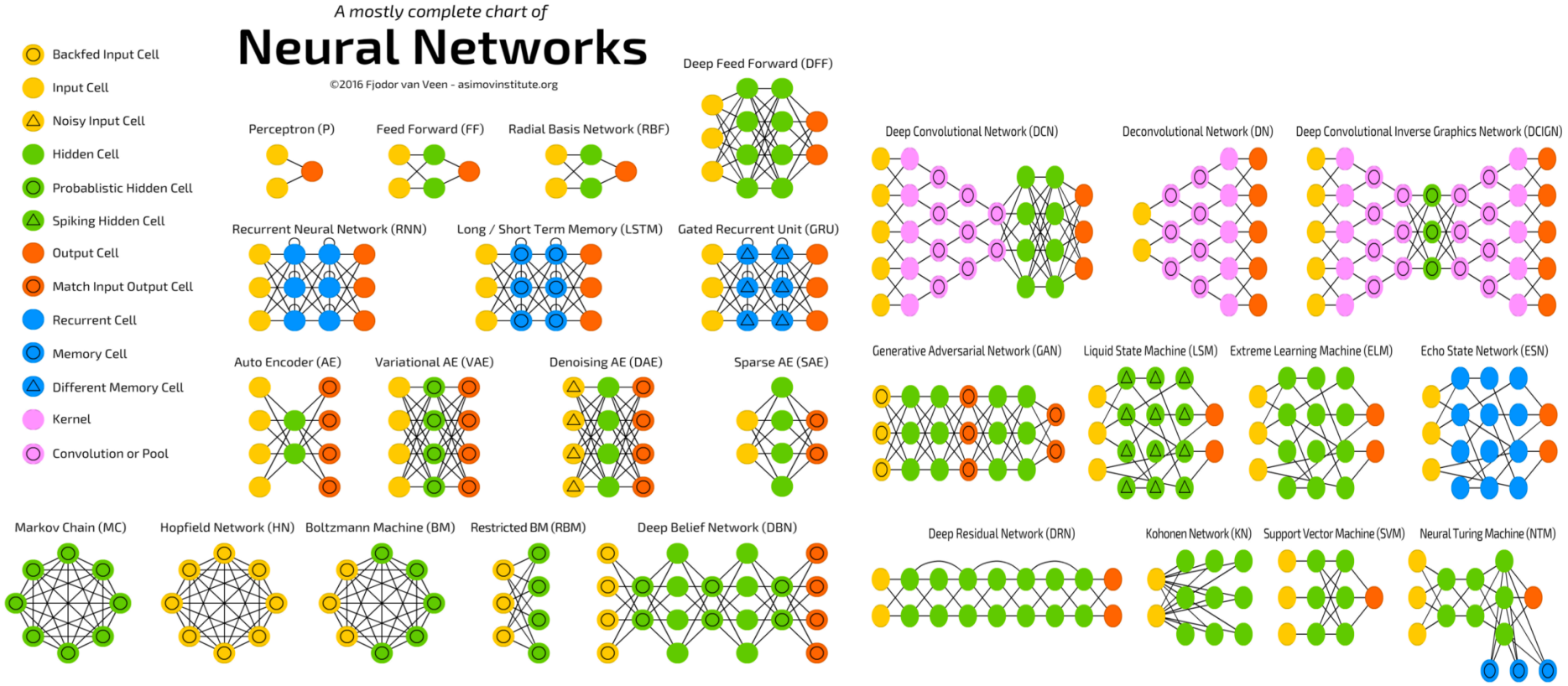
VGG16 model



https://neurohive.io/en/popular-networks/vgg16/

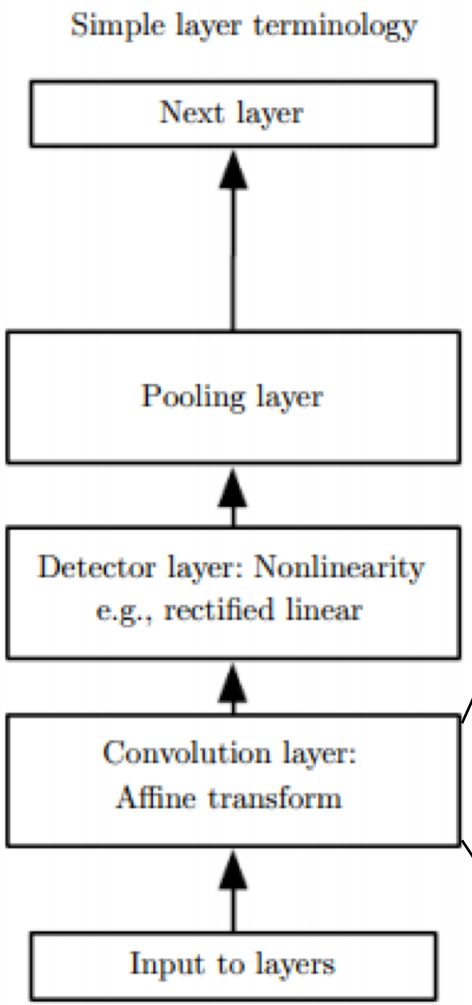
Types Neural Network

- There are many Neural network designs varies on
  - How layers are designed
  - How layers are connected.



https://becominghuman.ai/cheat-sheets-for-ai-neural-networks-machine-learning-deep-learning-big-data-678c51b4b463

Convolutional Neural Network



Convolution operation: A function transformation operation

given x(t), w(t)

Feature Map

$$S(t) = (x * w)(t)$$

Input function

Kernel

Convolution operators

input

output

12	12	17
10	17	19
9	6	14

kernel $\begin{pmatrix} 0 & 1 & 2 \\ 2 & 2 & 0 \\ 0 & 1 & 2 \end{pmatrix}$

Convolutional Layers

- Filter  $\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$

Input Image

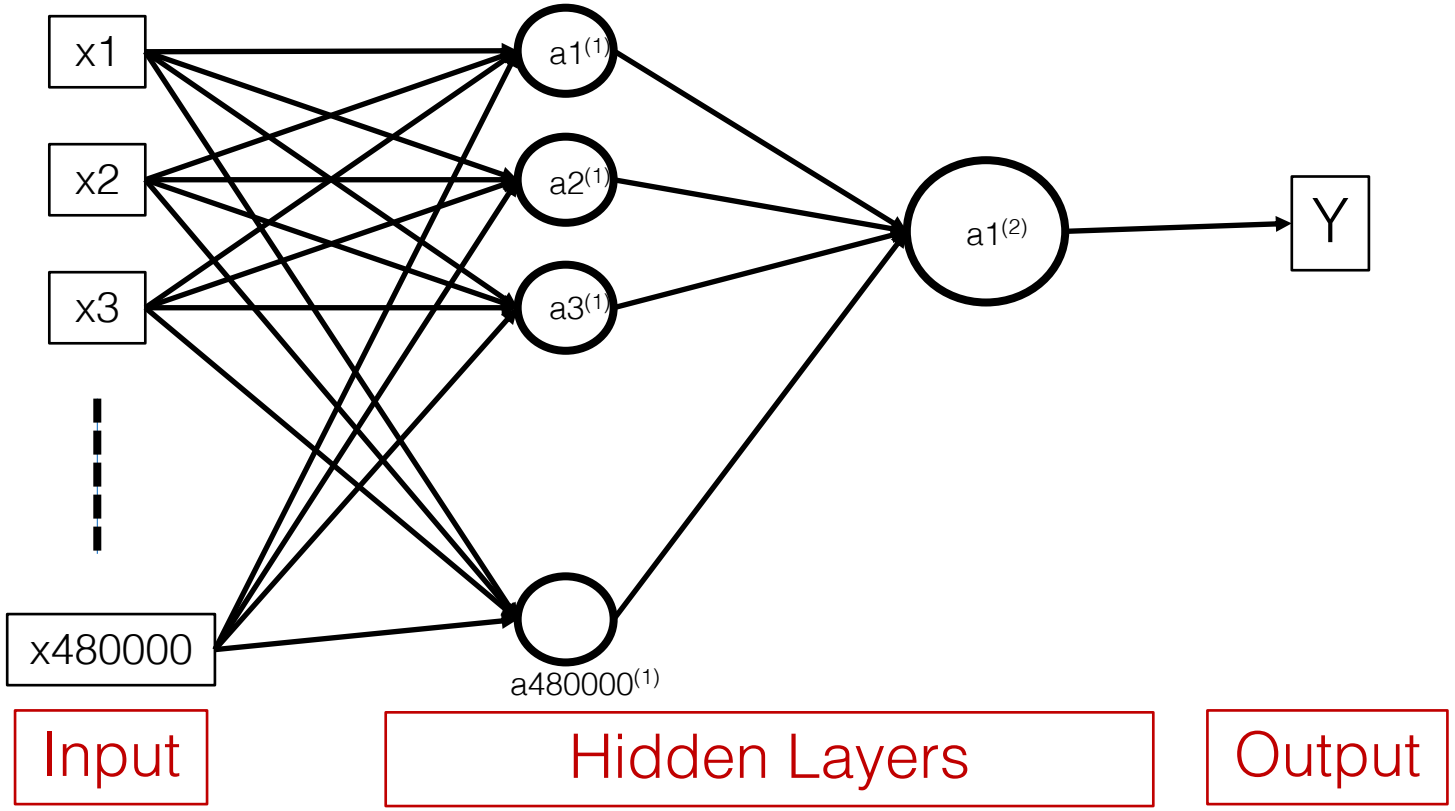
Convoluted Image

- Inspired by the neurophysiological experiments conducted by Hubel and Wiesel 1962.

Convolution vs Fully Connect

- Convolution is also much more efficient

- Assume 400 X 400 X 3, computational cost of convolution is linear.

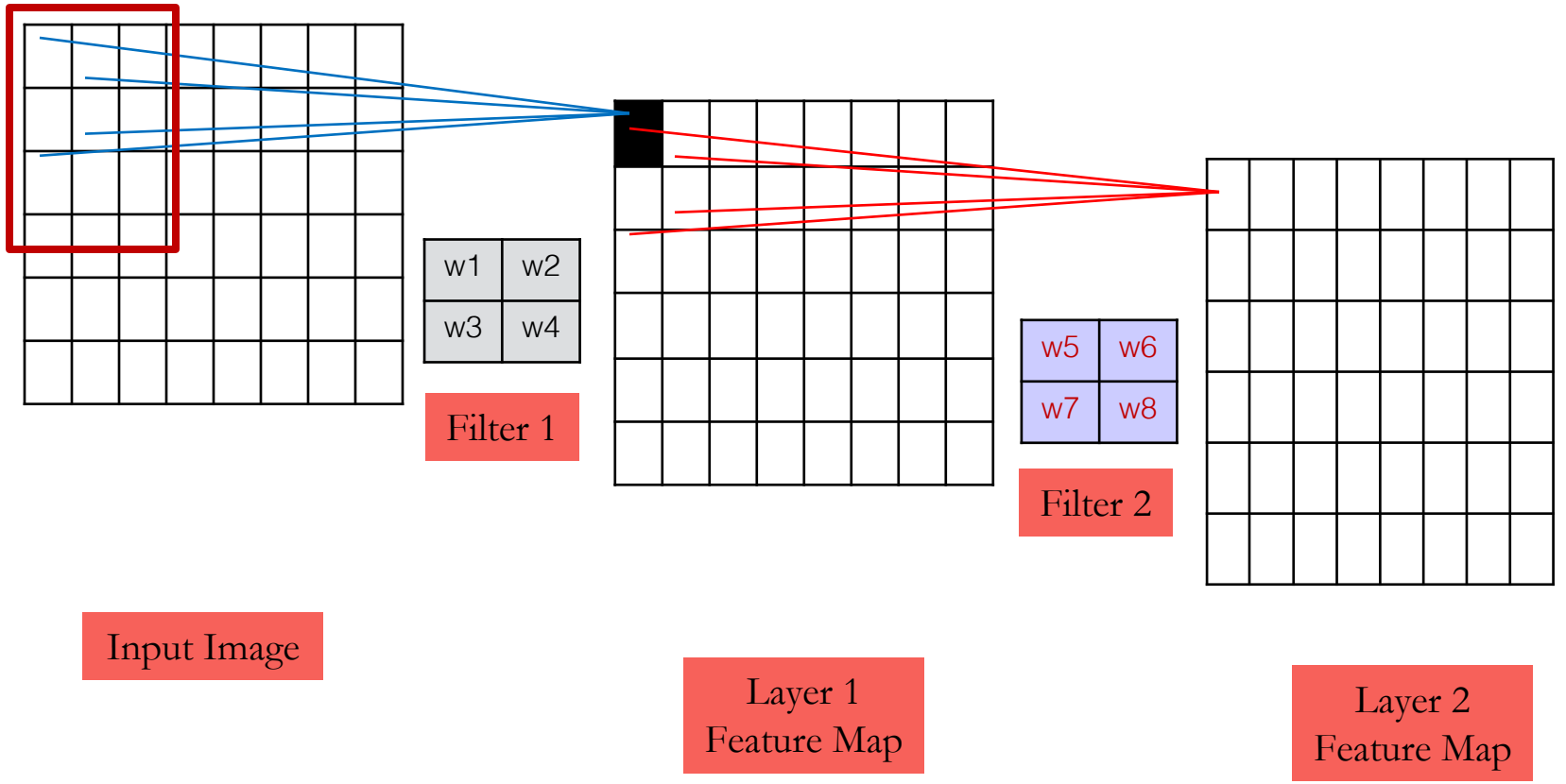


Number of Parameters

480000*480000 + 480000 +1 = **approximately 230 Billion !!!**

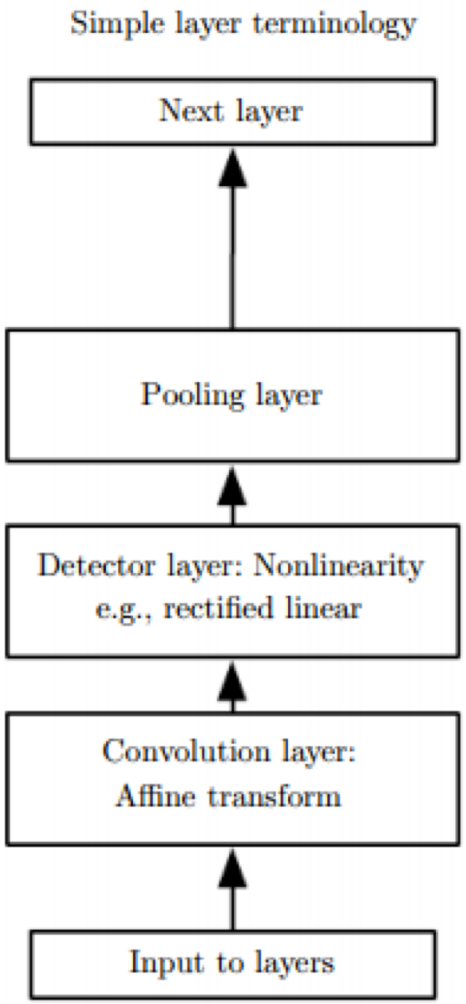
480000*1000 + 1000 +1 = approximately **480 million !!!**

Lower Level to More Complex Features

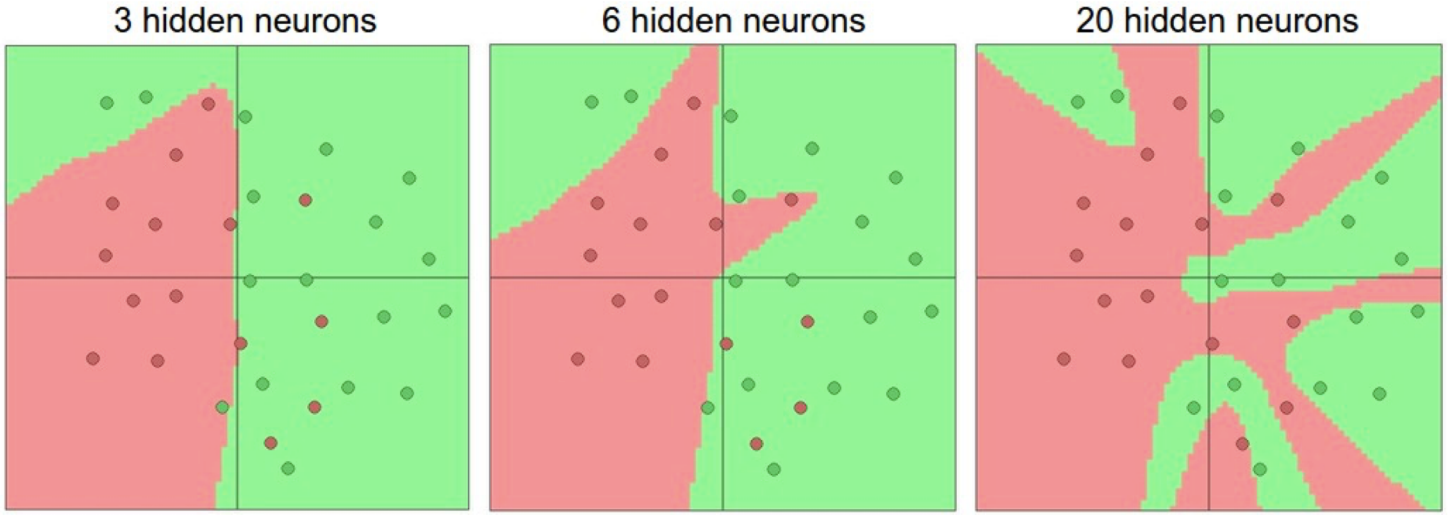


- In Convolutional neural networks, hidden units are only connected to local receptive field.

Activation Functions



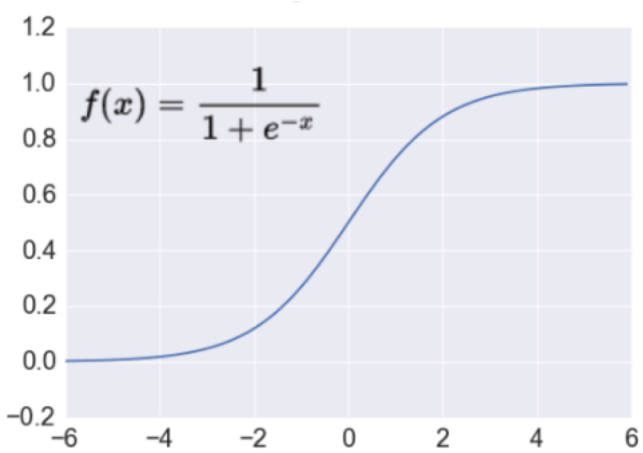
- Activation functions add non-linearities.
- Non-linearities needed to learn complex (non-linear) representations of data, otherwise the NN would be just a linear function mapping.



More layers and neurons can approximate more complex functions

http://cs231n.github.io/assets/nn1/layer_sizes.jpeg

Activation: Sigmoid



http://adilmoujahid.com/images/activation.png

Takes a real-valued number and “squashes” it into range between 0 and 1.

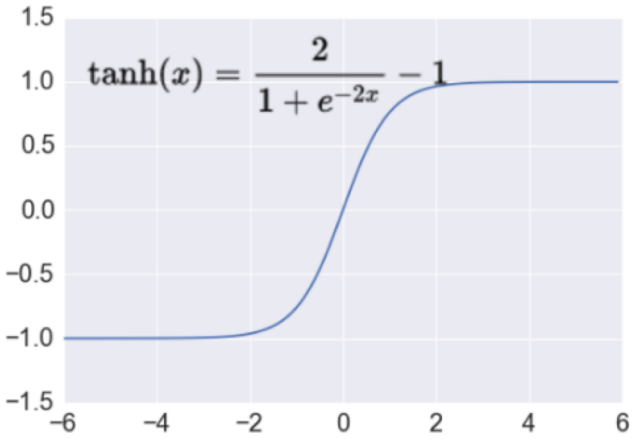
$R^n \rightarrow [0, 1]$

+ Nice interpretation as the firing rate of a neuron
  - 0 = not firing at all
  - 1 = fully firing

- Sigmoid neurons saturate (just 0 and 1) and kill gradients, thus NN will barely learn

Other Activation Functions

Tanh

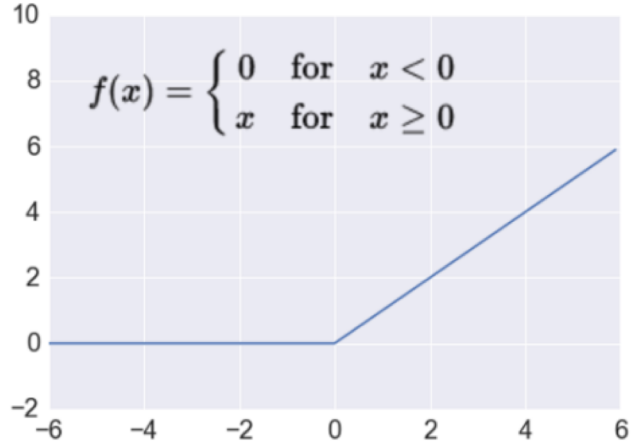


Takes a real-valued number and "squashes" it into -1 and 1.

$R^n \rightarrow [-1, 1]$

- Like sigmoid, tanh neurons saturate
- Unlike sigmoid, output is zero-centered
- Tanh is a scaled sigmoid:

ReLU



Takes a real-valued number and thresholds it at zero

$$f(x) = \max(0, x)$$

- Trains much faster
- Less expensive operations,
- Implemented by simply thresholding a matrix at zero

Pooling Layer

- Max pooling: reports the maximum output within a rectangular neighborhood.
- Average pooling: reports the average output of a rectangular neighborhood.

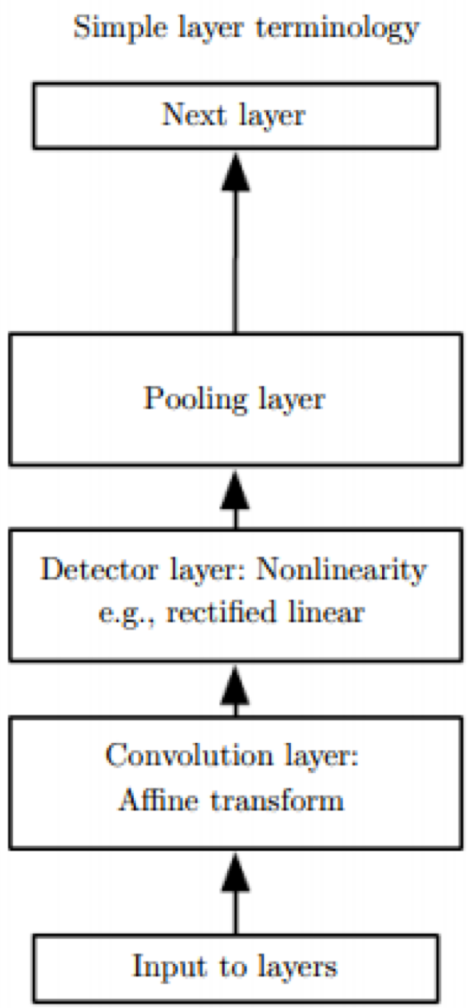
1	3	5	3
4	2	3	1
3	1	1	3
0	1	0	4

Input Matrix

MaxPool with 2X2 filter with stride of 2

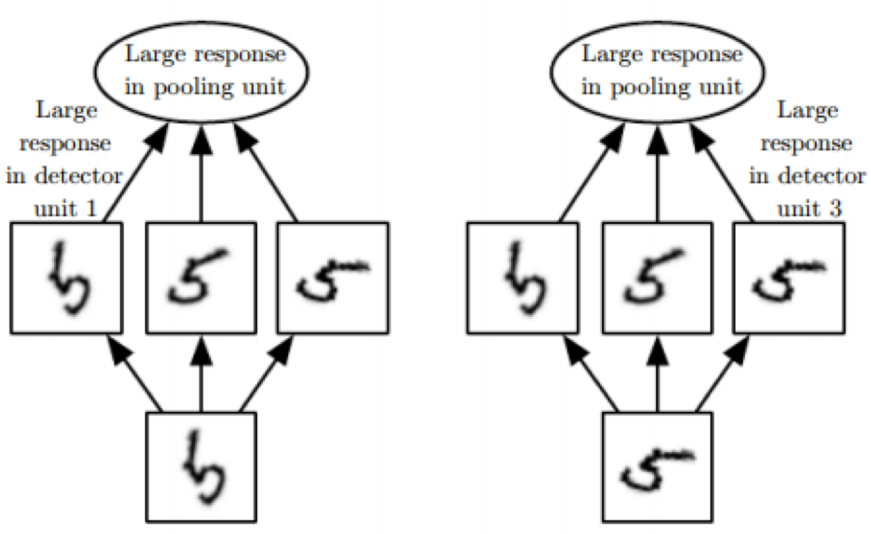
4	5
3	4

Output Matrix



Why Pooling?

- Reduce complexity
- Invariance to local translation



Example – LeNet-5 – MNIST Classification

- How many parameters to be trained in total?

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

TABLE I

EACH COLUMN INDICATES WHICH FEATURE MAP IN S2 ARE COMBINED BY THE UNITS IN A PARTICULAR FEATURE MAP OF C3.

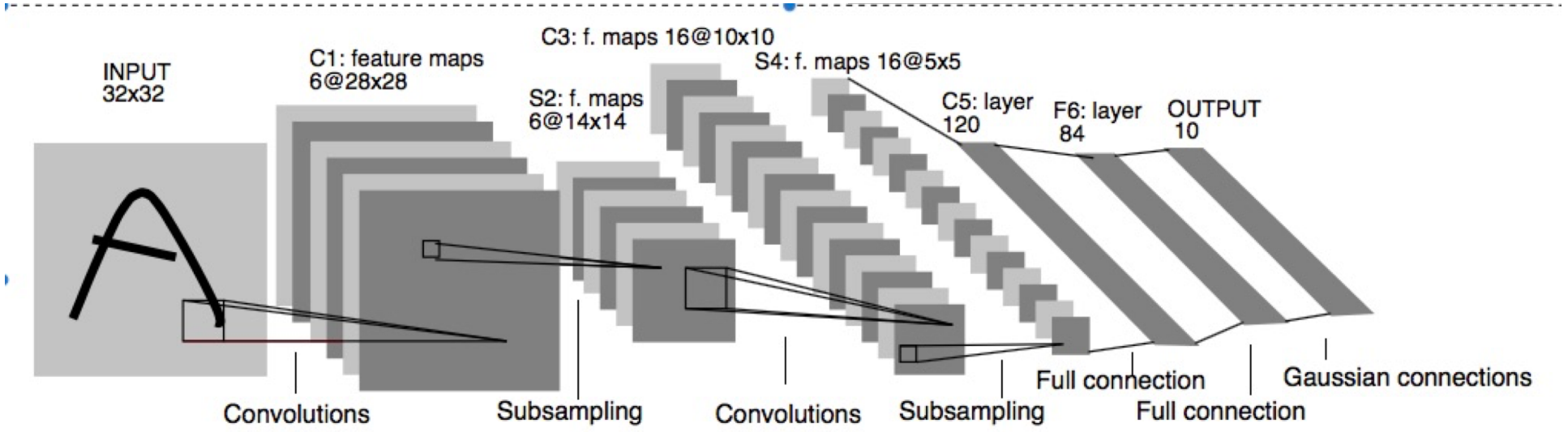


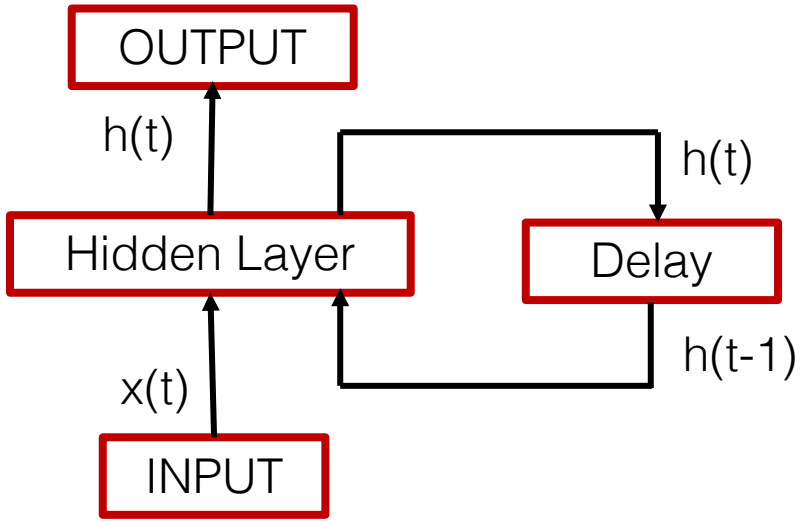
Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

5x5 2x1 5x5 2x1 Fully Connected

Recurrent Neural Network (RNN)

- Recurrent neural networks are connectionist models with the ability to selectively pass information across sequence steps, while processing sequential data one element at a time.

- Allows a memory of the previous inputs to persist in the model's internal state and influence the outcome.

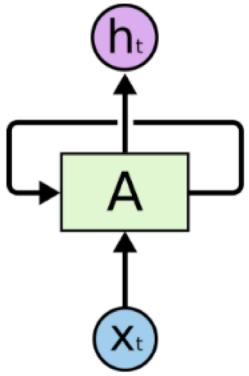


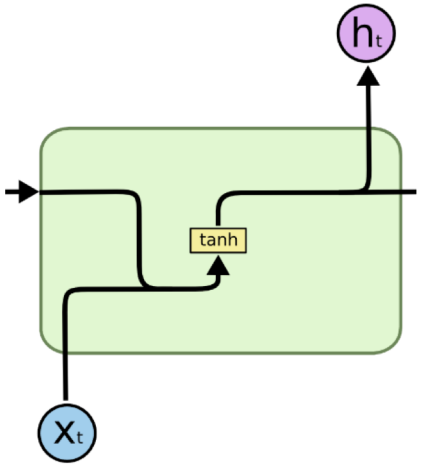
Why Recurrent Neuron Network ?

- The limitations of the Convolutional Neural Networks
  - Take fixed length vectors as input and produce fixed length vectors as output.
  - Allow fixed amount of computational steps.

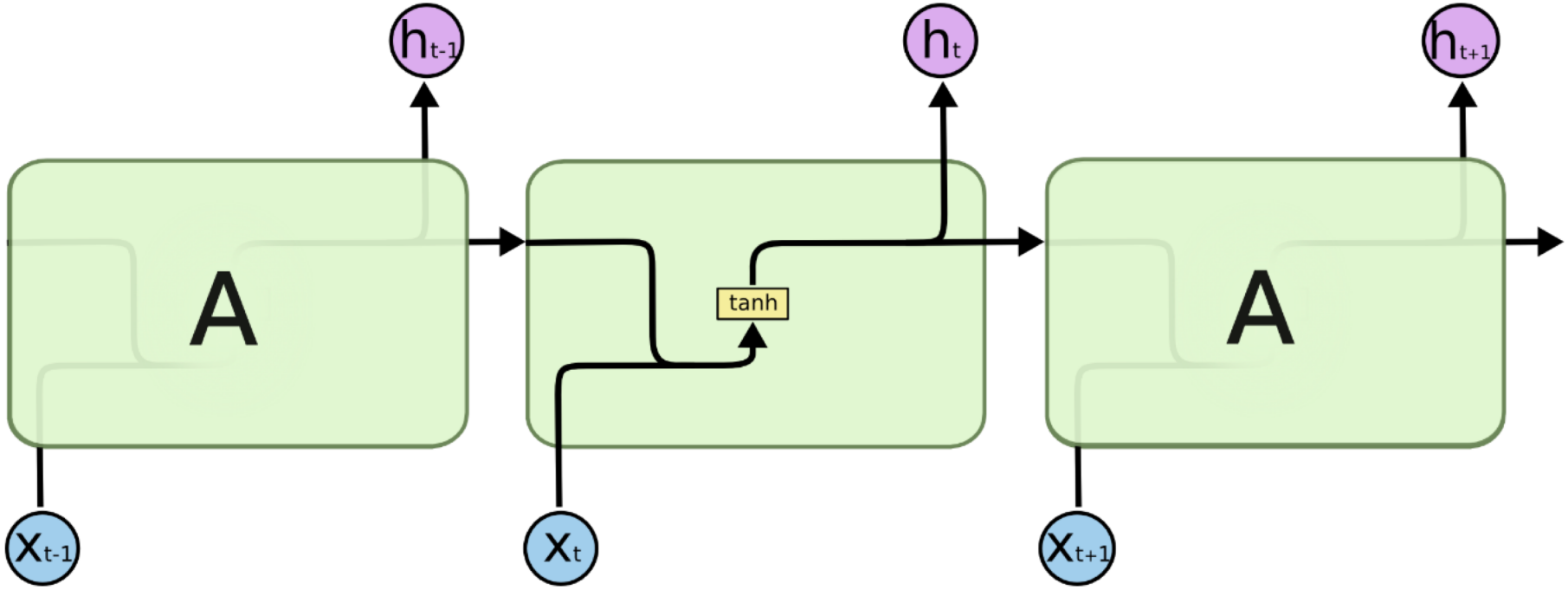
- We need to model the data with temporal or sequential structures and varying length of inputs and outputs

RNN Rolled Over Time





Simple RNN

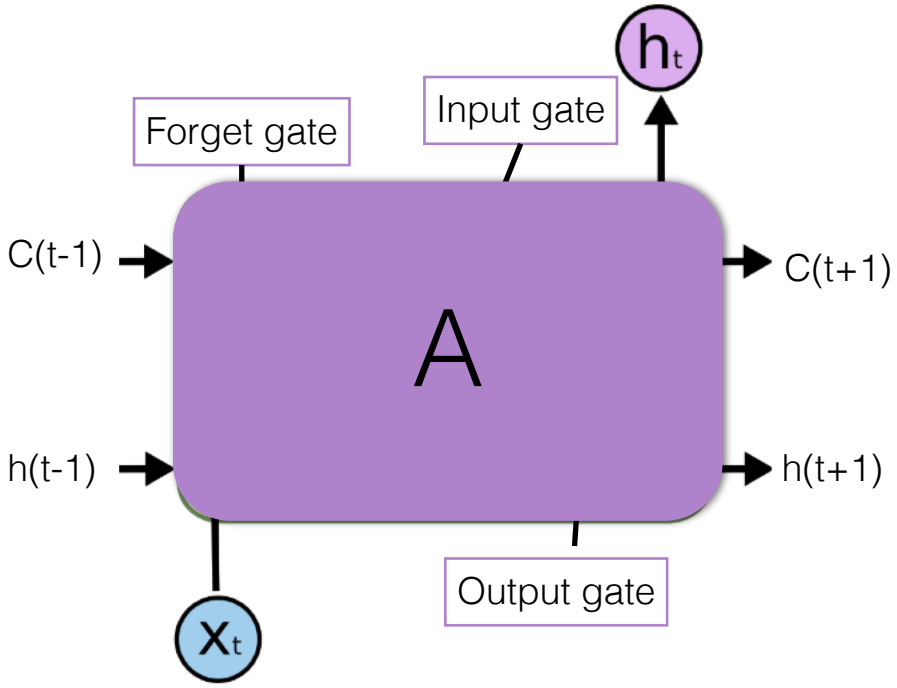


Long Short Term Memory (LSTM)

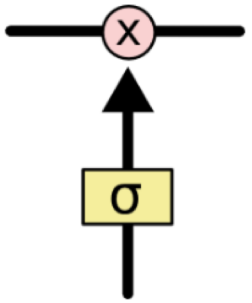
- Memory Cell, which is updated at each step in the sequence.

- Three Gates (control the flow of information to and from the Memory cell
  - Input Gate: protect the current step from irrelevant inputs
  - Output Gate: prevents current step from passing irrelevant information to later steps.
  - Forget Gate: limits information passed from one cell to the next.

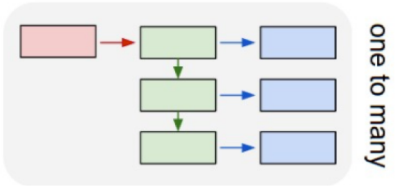
LSTM Structure



- Four layers
  - Sigmoid layer
  - tanh Layer
- Three gates
  - Sigmoid layer with pointwise multiplication operation.

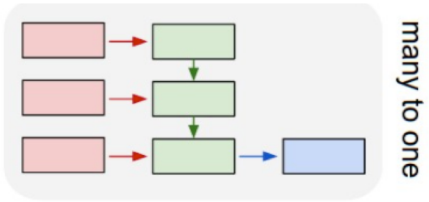


Applications of RNN



A person riding a motorbike on dirt road

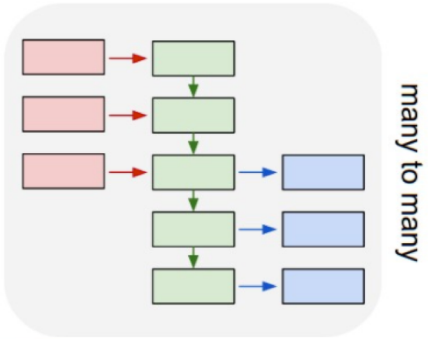
Image Captioning



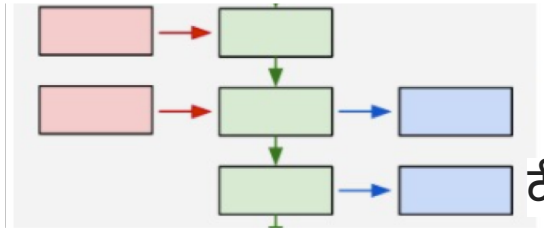
Awesome tutorial.

Positive

Sentiment Analysis



Happy Diwali



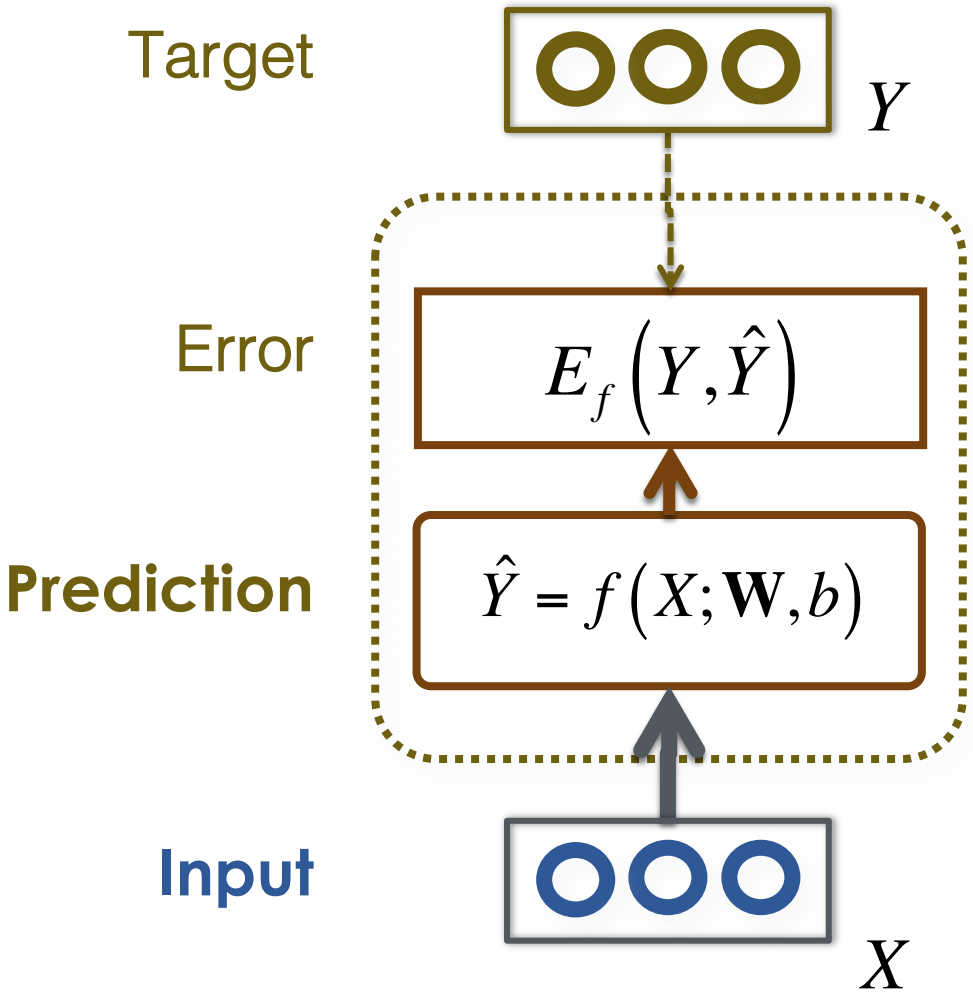
शुभ दीपावली

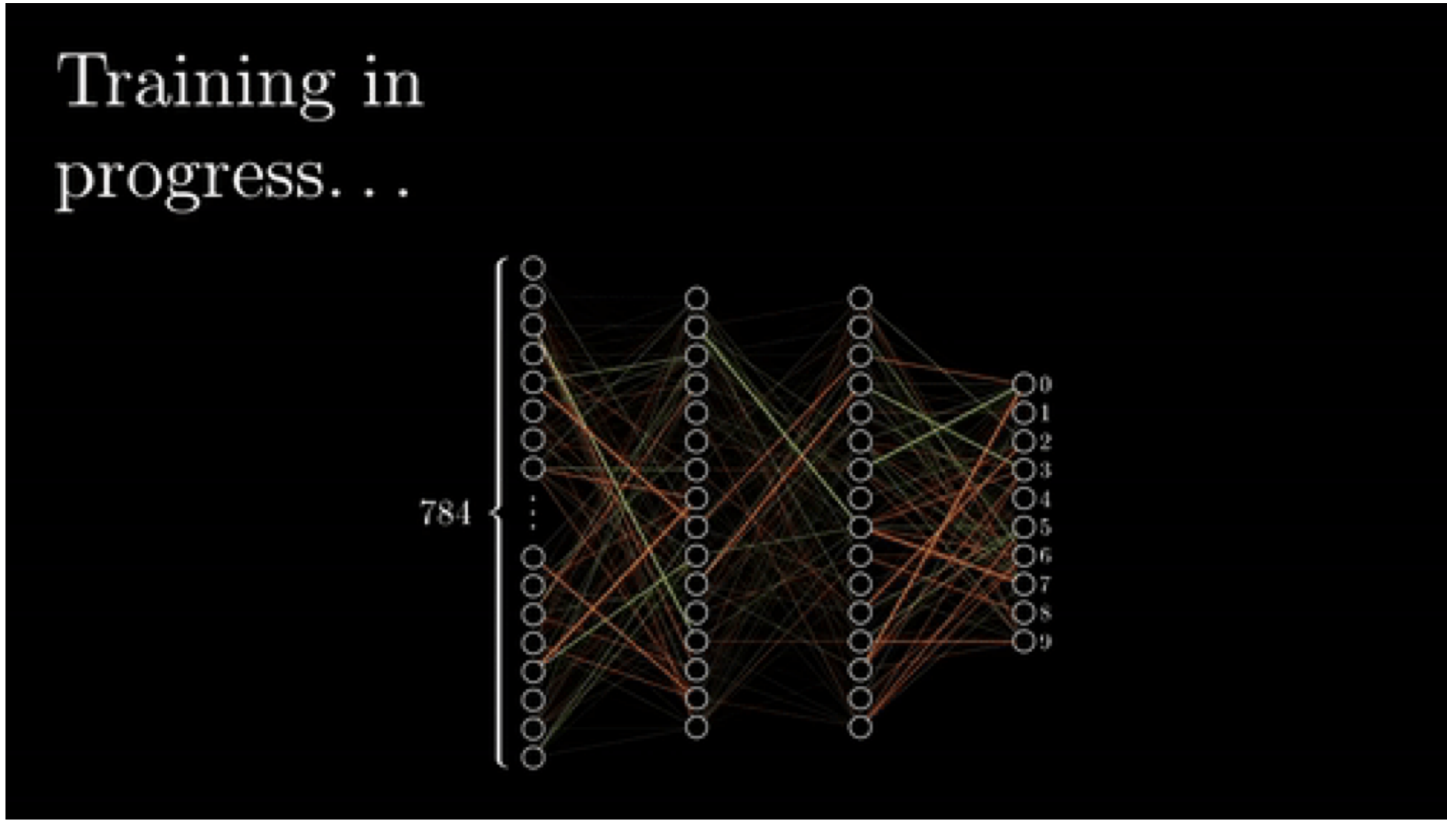
Machine Translation

Autoencoder

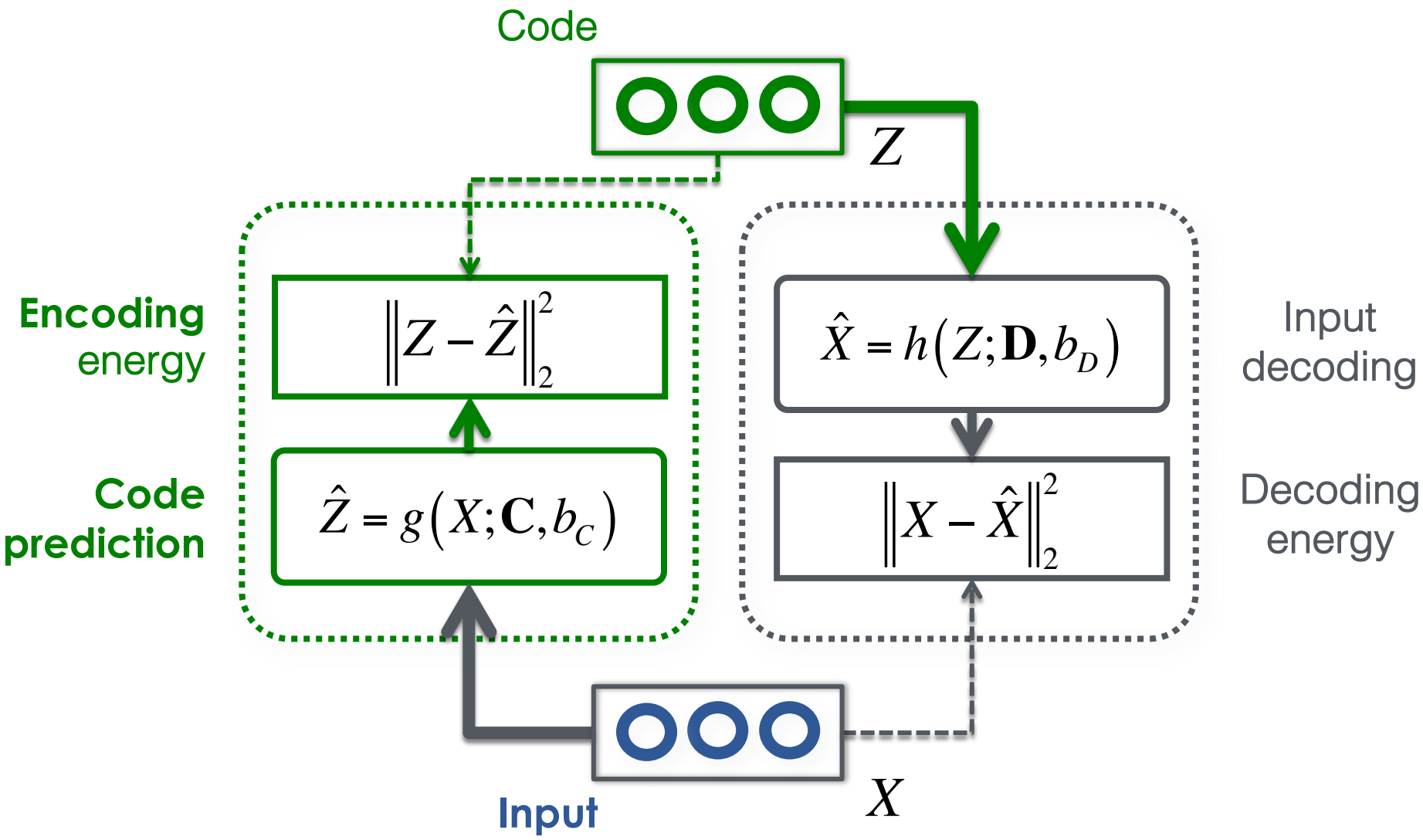
- Given a set of data points *X*, map *X* to *Z, where Z* has a lower dimensionality than X.
  - PCA?
- It doesn't necessarily contains special layer designs or network connections.
- A system to find lower dimension representation of the data.
- Unsupervised ! ?

Processing Flow with Target (Supervised Learning)

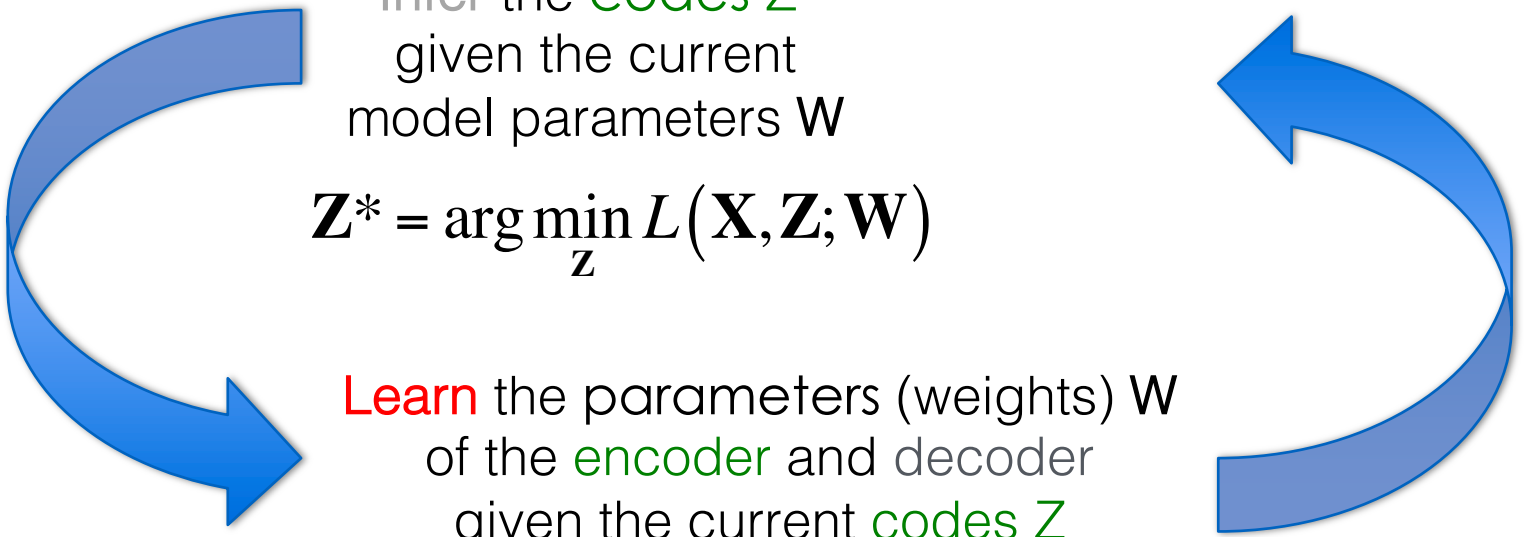




Autoencoder Processing Flow



Learning and inference in auto-encoders



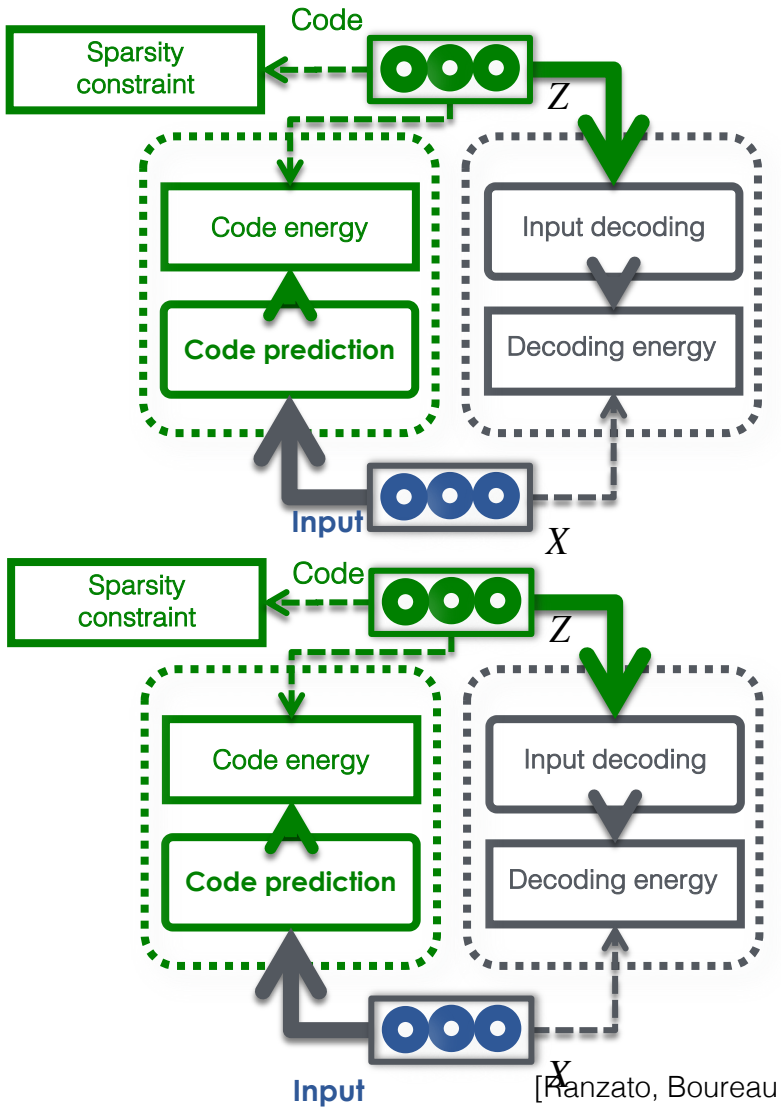
Single Dense Layer Example

- Using Mnist data set.
  - 28x28 image of digit
  - 60K training and 10K test images
  - Labels are available but not used.

- This example maps input image to a 32d vector.
- ~ 24.5 times compression from 784d vector.
- Sparsity constraint can be introduced through activity regularizer

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_3 (InputLayer)         (None, 784)               0
_________________________________________________________________
dense_5 (Dense)              (None, 32)                25120
_________________________________________________________________
dense_6 (Dense)              (None, 784)               25872
=================================================================
Total params: 50,992
Trainable params: 50,992
Non-trainable params: 0
```

Deep Autoencoders



- Instead of just a single layer, we can stacking several layers together to form deep structure.

[Ranzato, Boureau & LeCun, "Sparse Feature Learning for Deep Belief Networks ", *NIPS*, 2007]

Deep Autoencoders

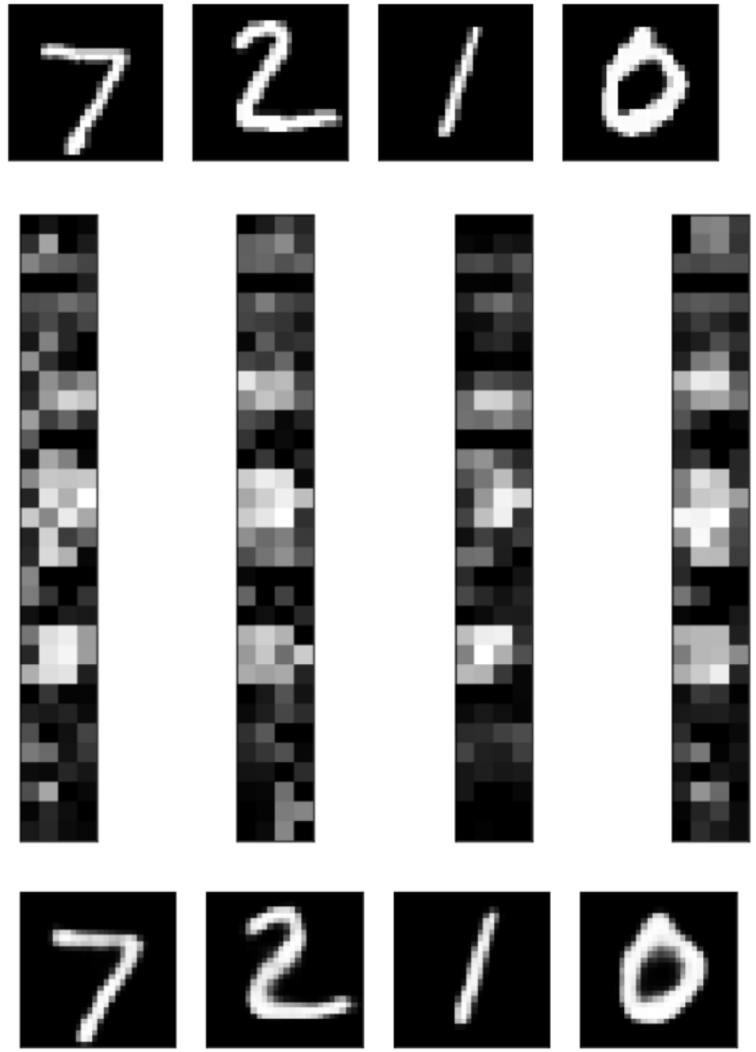
Layer (type)	Output Shape	Param #
input_11 (InputLayer)	(None, 784)	0
dense_13 (Dense)	(None, 128)	100480
dense_14 (Dense)	(None, 64)	8256
dense_15 (Dense)	(None, 32)	2080
dense_16 (Dense)	(None, 64)	2112
dense_17 (Dense)	(None, 128)	8320
dense_18 (Dense)	(None, 784)	101136

Total params: 222,384
Trainable params: 222,384
Non-trainable params: 0

- What's pros and cons?
- When this might be useful?

AE with Convolutional Layers

- Original
- Encoding
  - 4x4 * 8
- Reconstructed



AE with Convolutional Layers

Layer (type)	Output Shape	Param #
input_12 (InputLayer)	(None, 28, 28, 1)	0
conv2d_1 (Conv2D)	(None, 28, 28, 16)	160
max_pooling2d_1 (MaxPooling2	(None, 14, 14, 16)	0
conv2d_2 (Conv2D)	(None, 14, 14, 8)	1160
max_pooling2d_2 (MaxPooling2	(None, 7, 7, 8)	0
conv2d_3 (Conv2D)	(None, 7, 7, 8)	584
max_pooling2d_3 (MaxPooling2	(None, 4, 4, 8)	0
conv2d_4 (Conv2D)	(None, 4, 4, 8)	584
up_sampling2d_1 (UpSampling2	(None, 8, 8, 8)	0
conv2d_5 (Conv2D)	(None, 8, 8, 8)	584
up_sampling2d_2 (UpSampling2	(None, 16, 16, 8)	0
conv2d_6 (Conv2D)	(None, 14, 14, 16)	1168
up_sampling2d_3 (UpSampling2	(None, 28, 28, 16)	0
conv2d_7 (Conv2D)	(None, 28, 28, 1)	145

Total params: 4,385
Trainable params: 4,385
Non-trainable params: 0

AE for Denoising

- A common usage of AE is for denoising

- Model is trained using noisy data and clean data
  - Matching input noisy data to clean output data.

- The noisy data can be generated from clean data by adding artificial noise.

AE for Denoising

Layer (type)	Output Shape	Param #
input_13 (InputLayer)	(None, 28, 28, 1)	0
conv2d_8 (Conv2D)	(None, 28, 28, 32)	320
max_pooling2d_4 (MaxPooling2	(None, 14, 14, 32)	0
conv2d_9 (Conv2D)	(None, 14, 14, 32)	9248
max_pooling2d_5 (MaxPooling2	(None, 7, 7, 32)	0
conv2d_10 (Conv2D)	(None, 7, 7, 32)	9248
up_sampling2d_4 (UpSampling2	(None, 14, 14, 32)	0
conv2d_11 (Conv2D)	(None, 14, 14, 32)	9248
up_sampling2d_5 (UpSampling2	(None, 28, 28, 32)	0
conv2d_12 (Conv2D)	(None, 28, 28, 1)	289

Total params: 28,353
Trainable params: 28,353
Non-trainable params: 0

Hands on session

1. Log on to the Visualization Portal with your TACC account.
2. Launch a Jupyter Notebook job from the Visualization Portal.
   - Use reservation "ML_Institute_day4"
   - Choose "rtx" queue
   - Specify "02:00:00" for job time.

Wait for Jupyter notebook to start

3. Open a new terminal from your Jupyter session.

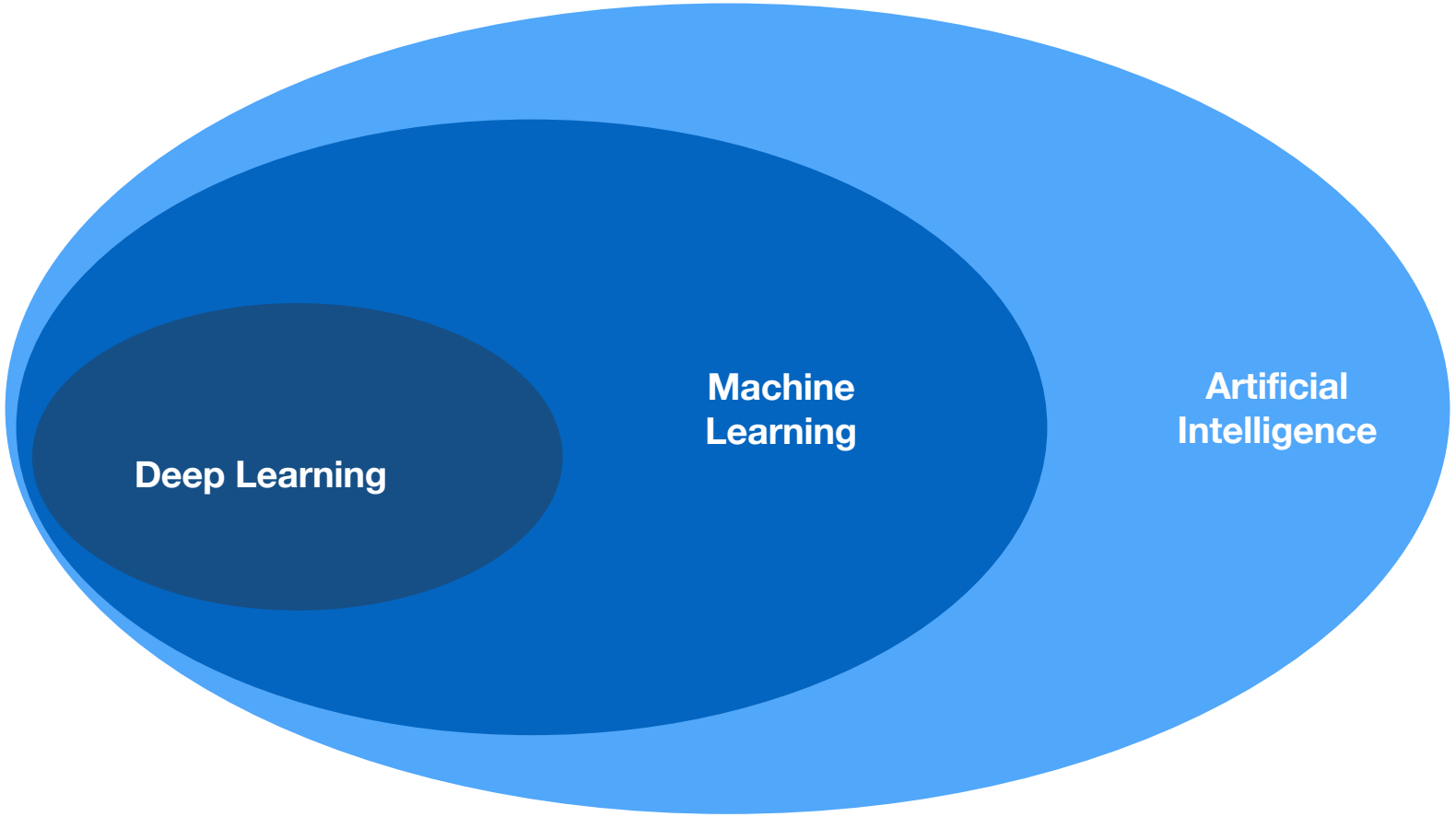
4. Copy example notebook over:
   cp /work/00791/xwj/DMS/TACC_SSI_2021/DL.ipynb .

5. Open 'DL.ipyb'

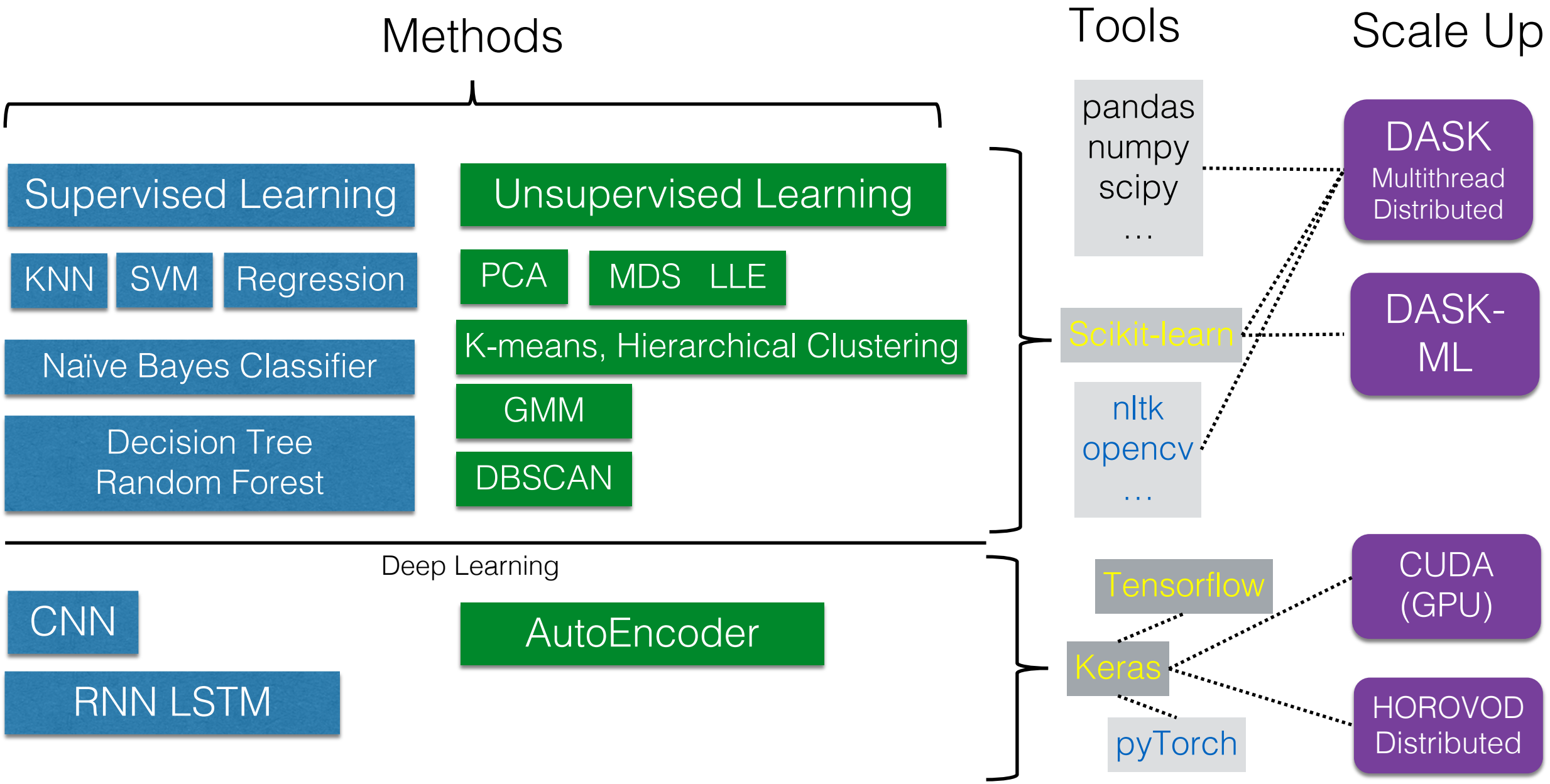
Note: the full exercise from morning sessions also available at

/work/00791/xwj/DMS/TACC_SSI_2021/dl_tutorial/

Deep learning, Machine Learning, and Artificial Intelligence



Summary


Methods
Tools
Scale Up
Supervised Learning
Unsupervised Learning
KNN
SVM
Regression
PCA
MDS
LLE
Naïve Bayes Classifier
K-means, Hierarchical Clustering
Decision Tree
Random Forest
GMM
DBSCAN
Deep Learning
CNN
AutoEncoder
RNN LSTM
pandas
numpy
scipy
…
Scikit-learn
nltk
opencv
…
DASK
Multithread
Distributed
DASK-
ML
Tensorflow
Keras
pyTorch
CUDA
(GPU)
HOROVOD
Distributed