Deep Learning for Engineers

John Aynsley

DOULOS

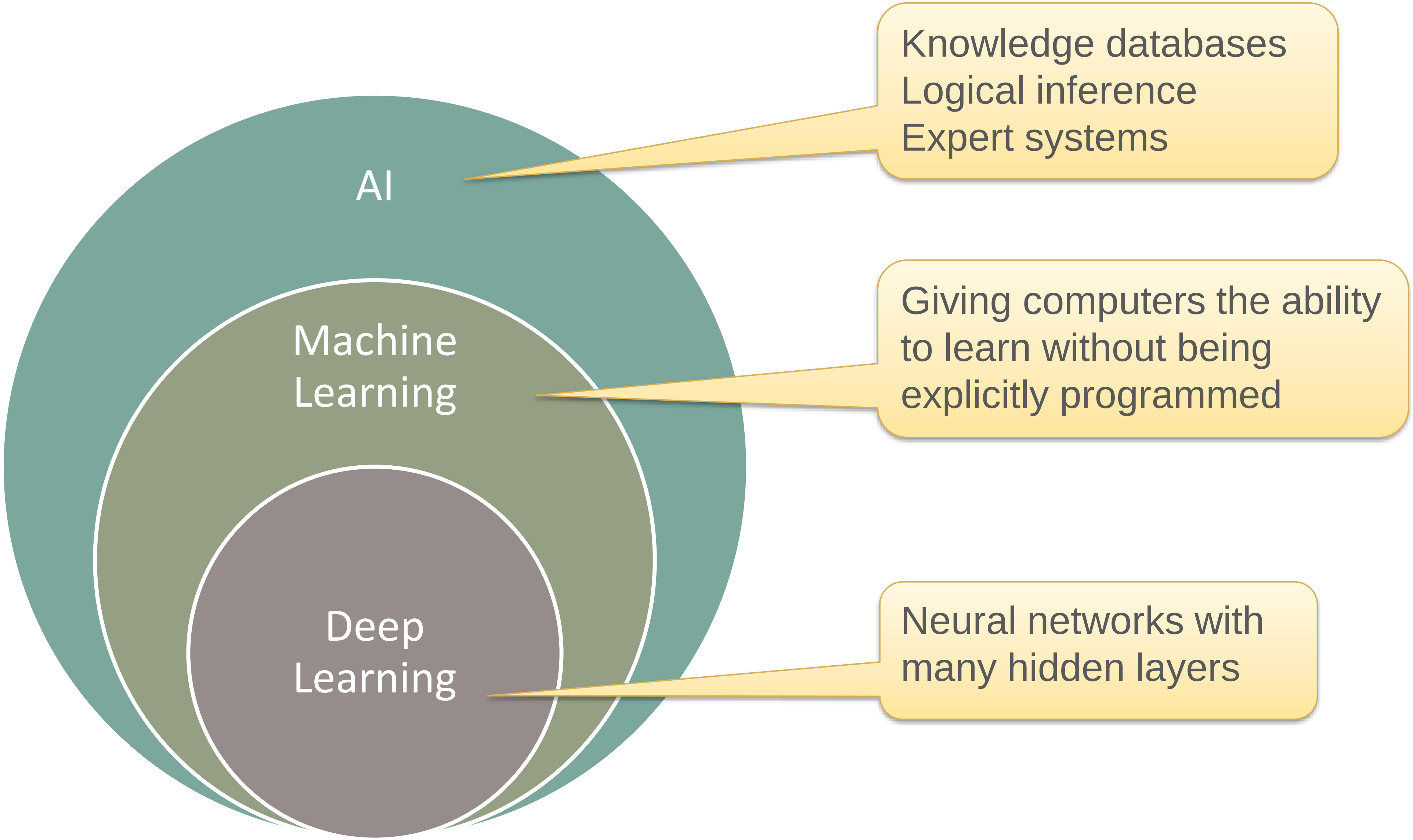
2019 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

accellera SYSTEMS INITIATIVE

Deep Learning for Engineers

- AI, ML, and Deep Learning
- Training a Neural Network
- Deeper Insights
- CNNs and RNNs
- Tool Flow

AI versus ML versus Deep Learning



"Classical" Machine Learning

Tasks	Algorithms
Classification	Support vector machines
Regression	Bayesian statistics
Clustering	Markov models
Anomaly detection	Decision trees
Dimensionality reduction	Random forests
	K-means
	... and many more

Could be all you need!

Appropriate for smaller datasets

Why Deep Learning Now?

2012 – a CNN wins ImageNet Challenge

Bigger datasets

Faster computers

Since 2012

Improved neural network architectures

Neural networks often outperforming previous state-of-the-art

The ImageNet Challenge (ILSVRC)

ImageNet Large Scale Visual Recognition Challenge: 1.2M images in 1000 categories

Year	Network	#Layers	Top-5 Error Rate
2011 winner	(Not a NN)	-	25.8%
2012 winner	AlexNet (CNN)	8	16.4%
2013 winner	ZFNet (CNN)	8	11.7%
2014	VGGNet (CNN)	19	7.3%
2014 winner	GoogLeNet (Inception)	22	6.7%
2015 winner	ResNet (residual)	152	3.6%
2016 winner	CUImage (ensemble)	-	3.0%

← *Dramatic improvement*

← *Human error rate ~ 5%*

← *3% bad labels*

Training typically takes a few weeks on a few GPUs

Cloud Computing versus Edge Computing

Cloud Computing in Data Centers	Edge Computing in Embedded Devices
Massive, scalable compute power	Limited compute power
Unlimited storage	Limited storage
High latency	Low latency (real-time response)
Restricted bandwidth	Unrestricted bandwidth
Low energy efficiency	High energy efficiency
Reliant on internet connection	Can run without internet connection
Data sent over internet (privacy?)	Data kept local
Relatively high cost	Low cost

Cloud versus Edge ML/DL Applications

Cloud

Recommendation engines for websites

Fraud detection on financial transactions

Chat bots

Edge / IoT Sensors

Images, video, voice, temperature, vibration, ...

Edge Applications of Deep Learning

The low-hanging fruit

- Vision
  - Image recognition
  - Object detection
  - Image segmentation
- Speech recognition
- Text analysis
- Anomaly detection

Automotive Applications

ADAS and autonomous vehicles

- Traffic sign recognition
- Lane detection
- Pedestrian detection
- Human pose estimation
- Monitoring for a distracted driver

Detecting vehicle occupancy for car sharing

Detecting driver identity to store seat settings

Industrial, Medical, Retail, IoT

Touchscreen character recognition

Voice control - keyword spotting

Medical diagnosis from images

Customer counts and demographics from cameras in retail stores

Real-time failure prediction in industrial equipment

Face recognition in smart doorbells

Food classification – allergy advice

Deep Learning for Engineers

AI, ML, and Deep Learning

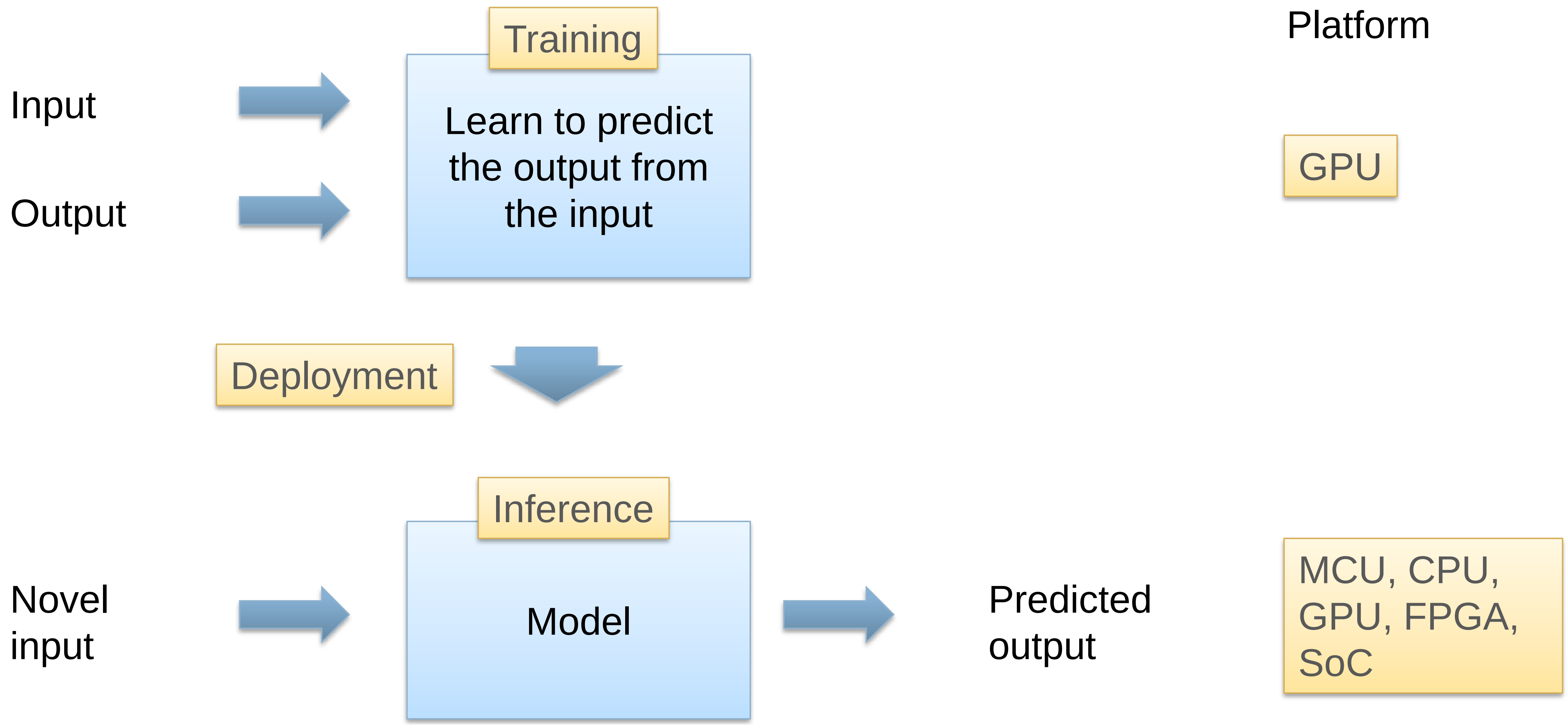
Training a Neural Network

Deeper Insights

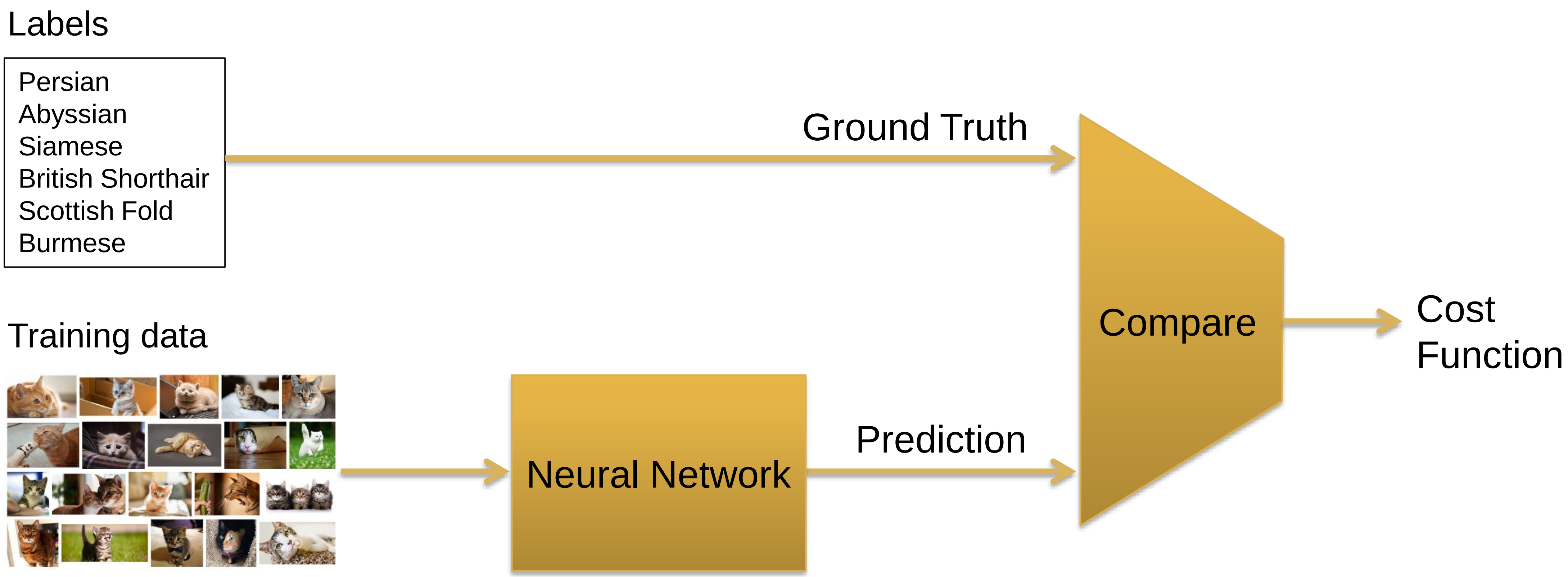
CNNs and RNNs

Tool Flow

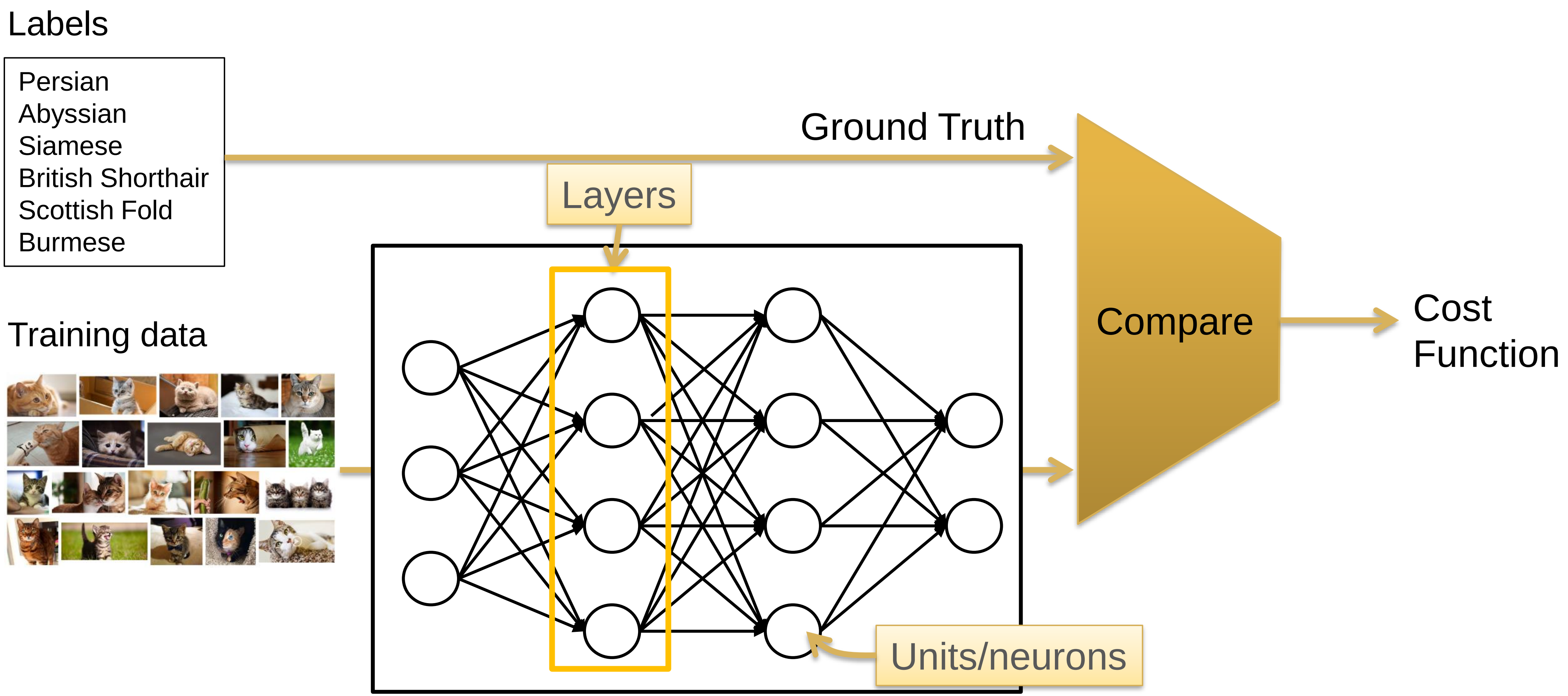
Supervised Learning



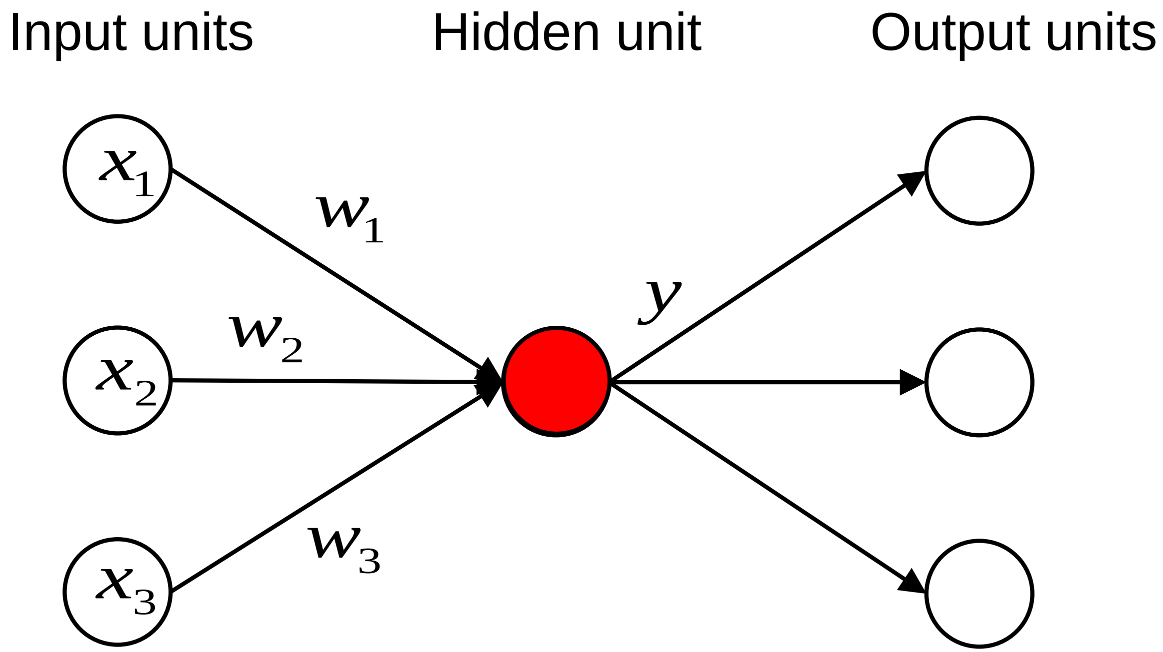
Training a Neural Network



Training a Neural Network



An Artificial Neuron



$$u = \sum_{i=1}^{3} w_i x_i + b$$

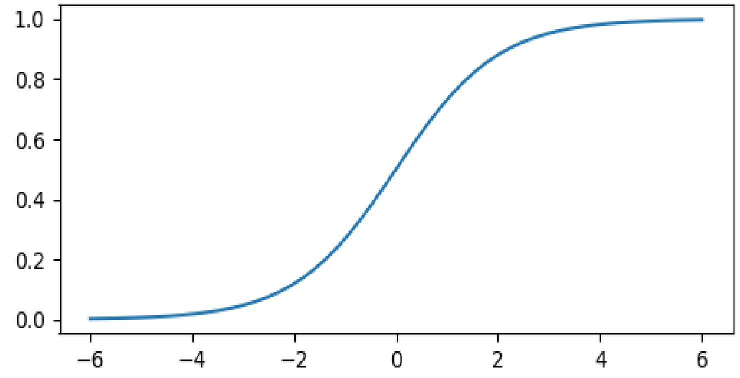
Linear function

$$y = activation(u)$$

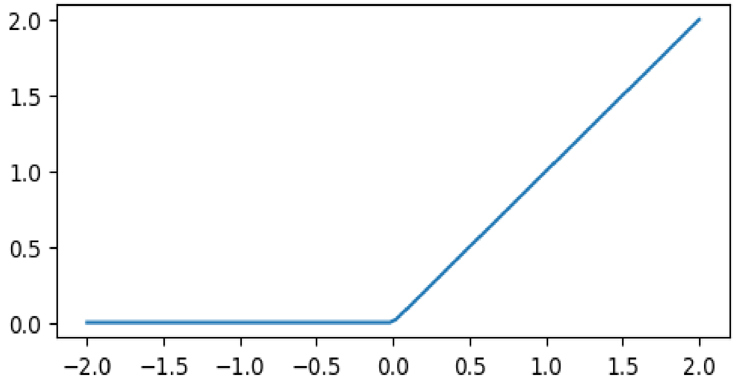
Non-linear function

Common Activation Functions

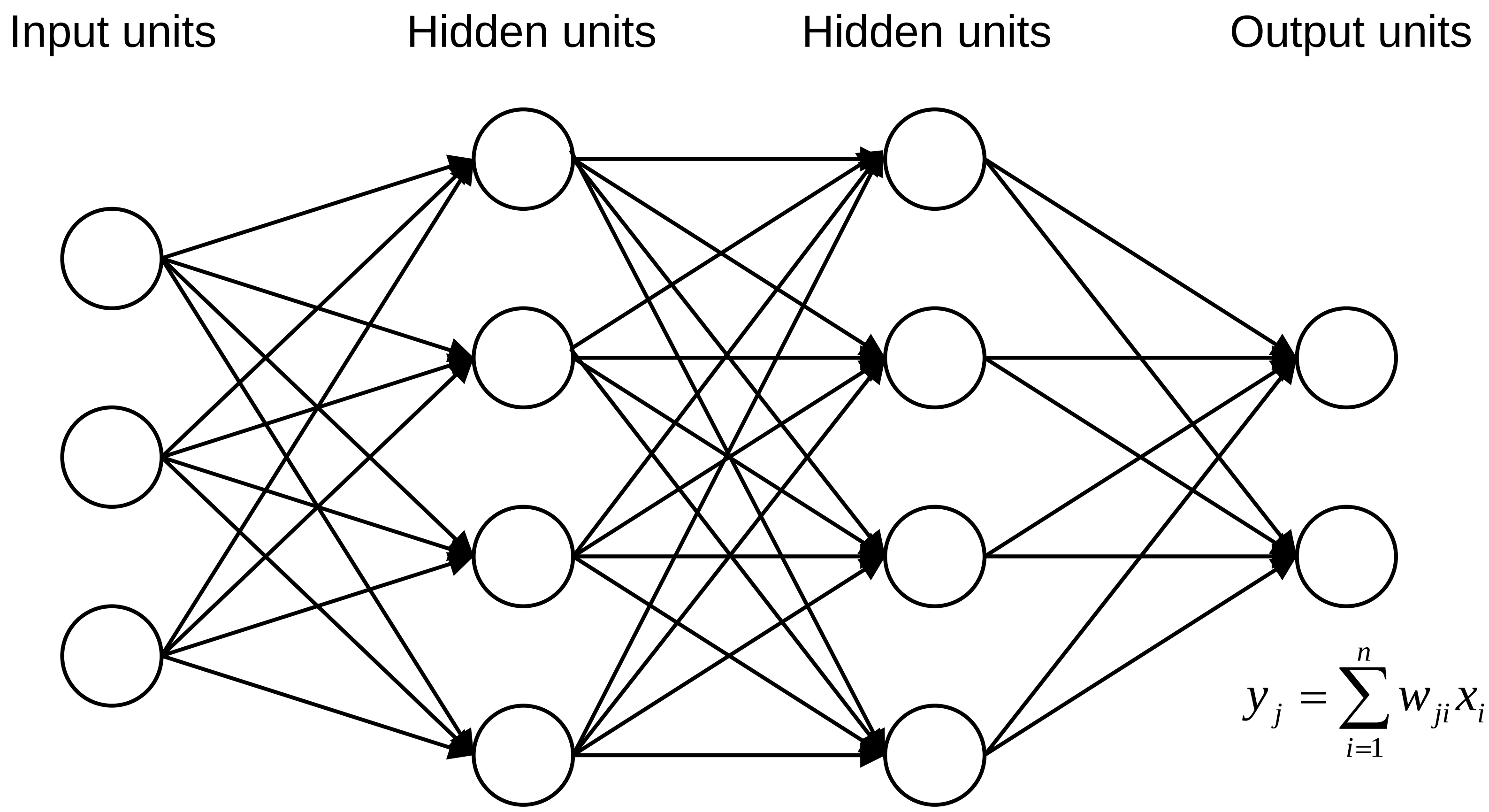
Sigmoid
aka
logistic



ReLU



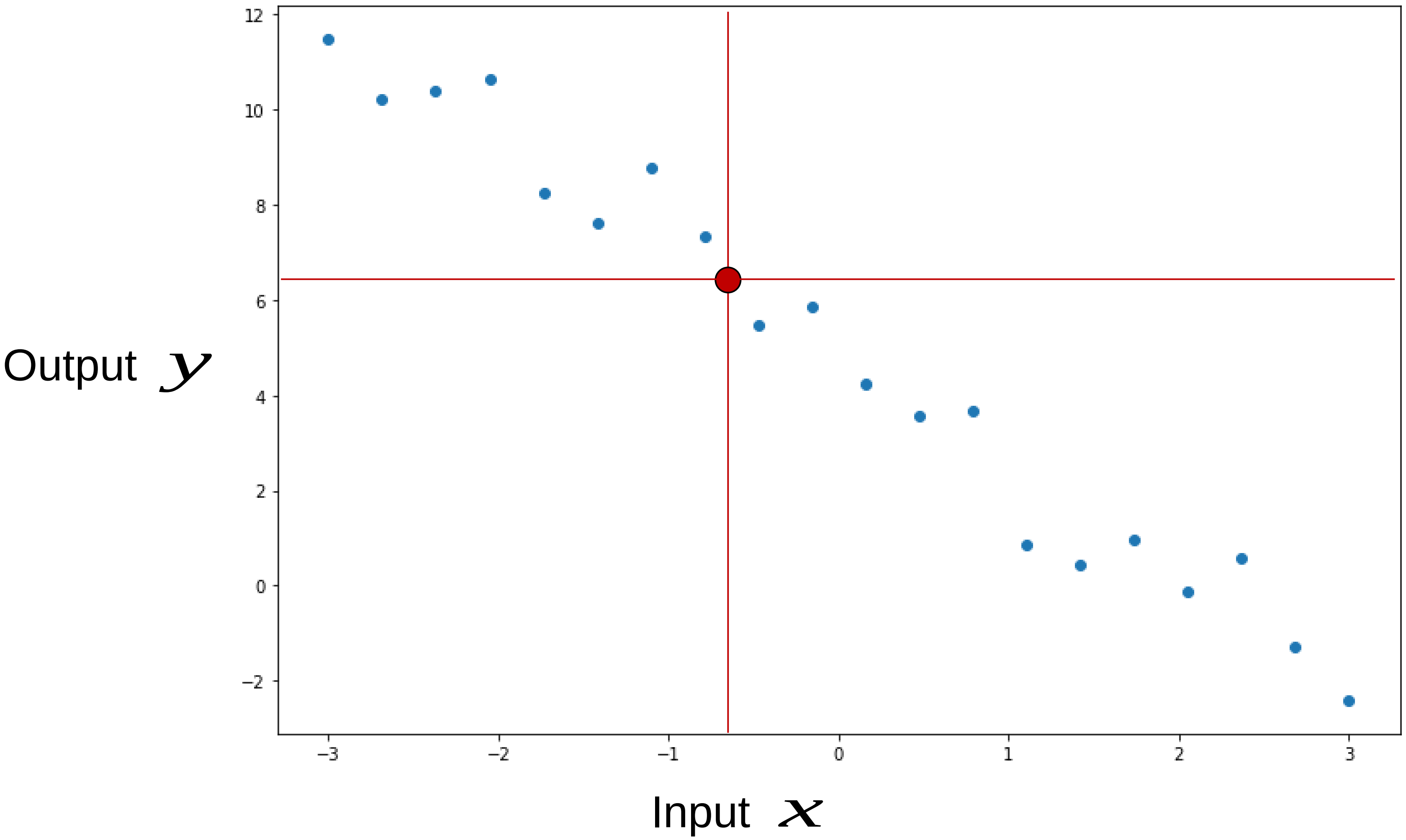
A Deep Neural Network



$$y_j = \sum_{i=1}^{n} w_{ji} x_i + b_j$$

$$y_j = RELU\left(\sum_{i=1}^{3} w_{ji} x_i + b_j\right) \qquad y_j = RELU\left(\sum_{i=1}^{n} w_{ji} x_i + b_j\right)$$

Regression Task



Define a Hypothesis or Model or Network

Trainable parameter (Weight)

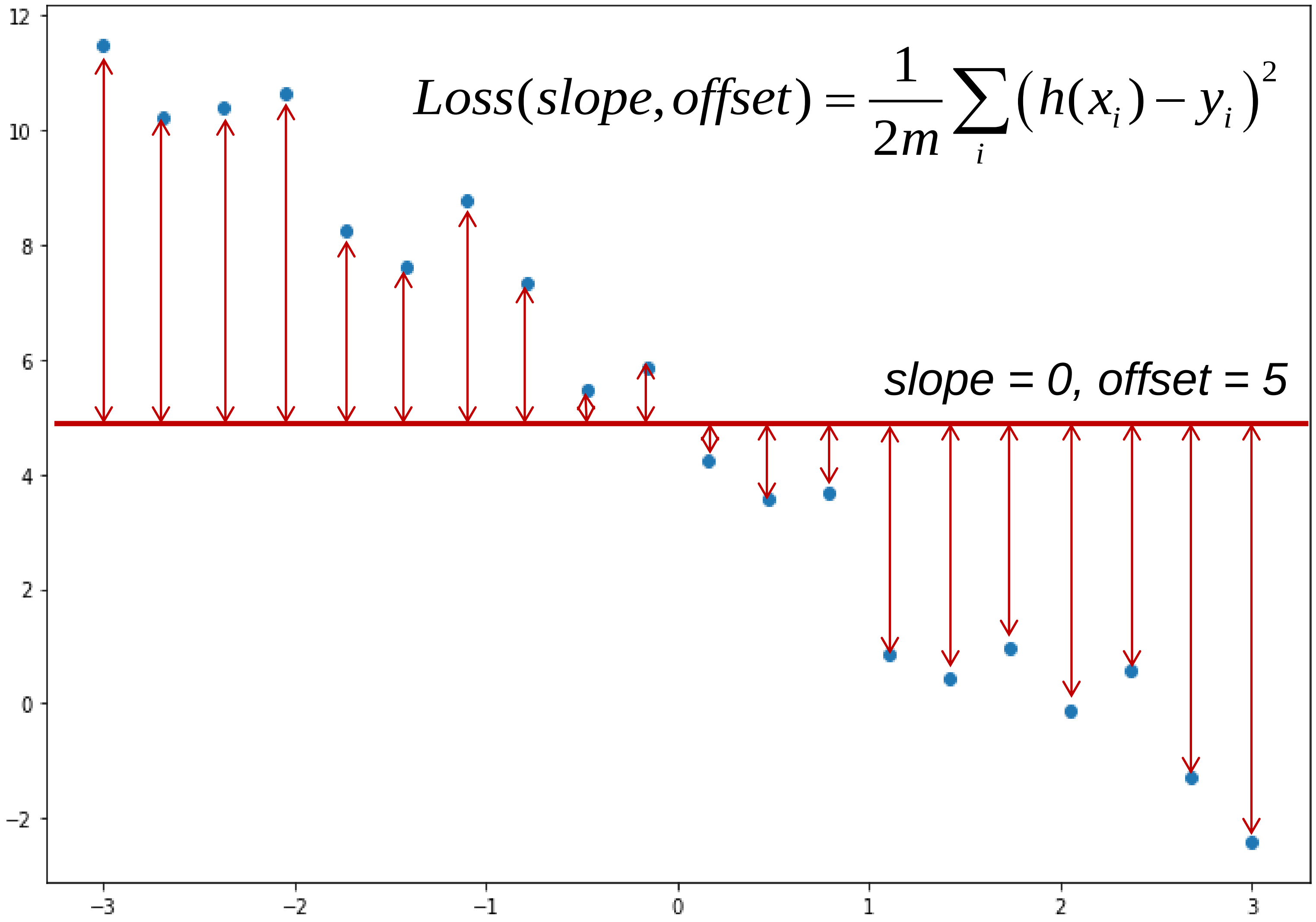
Input vector

Trainable parameter (Bias)

$$h_{slope,offset}(x) = slope \cdot x + offset$$

Predicted output

Cost or Loss or Error Function

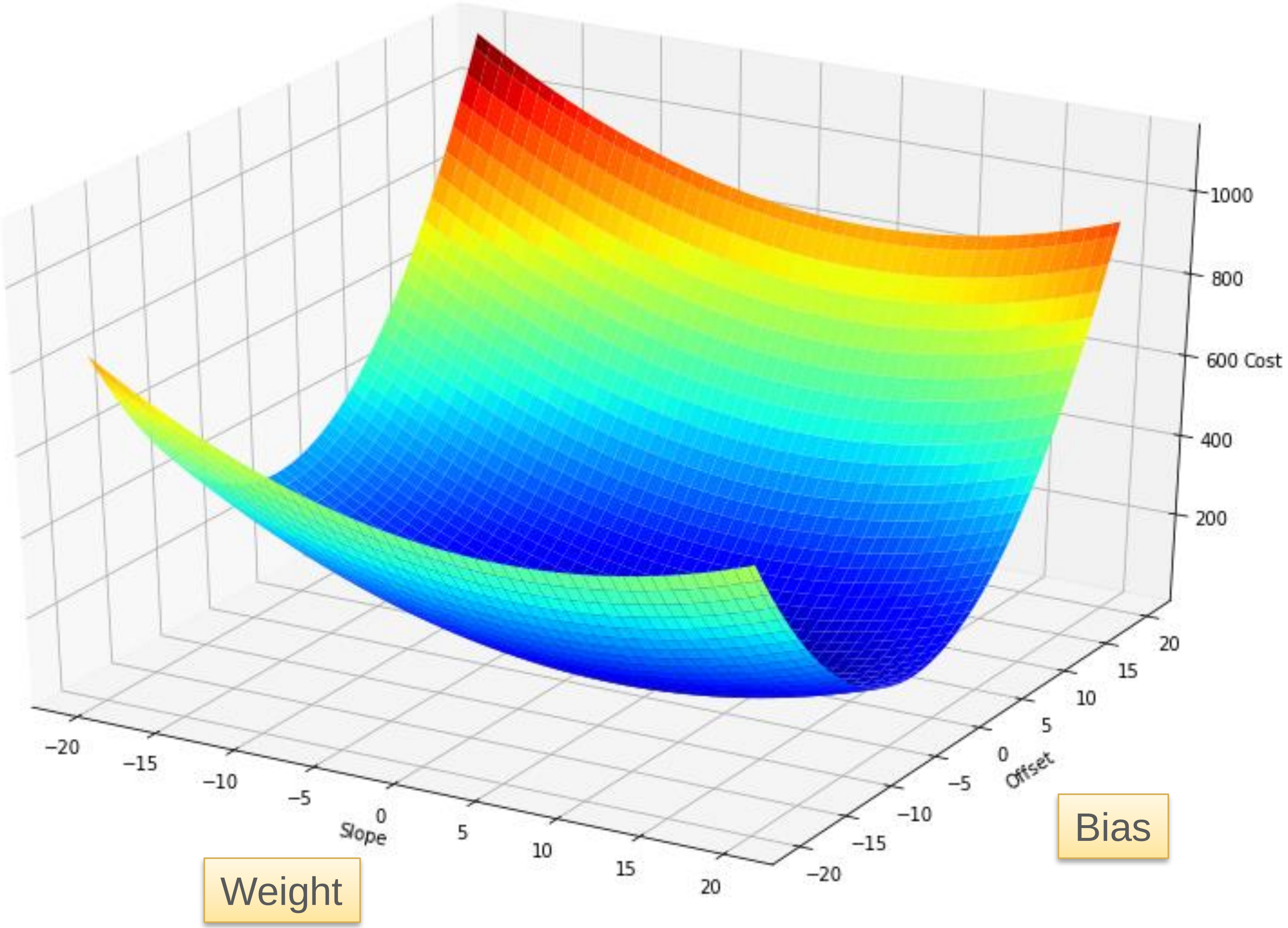


Prediction

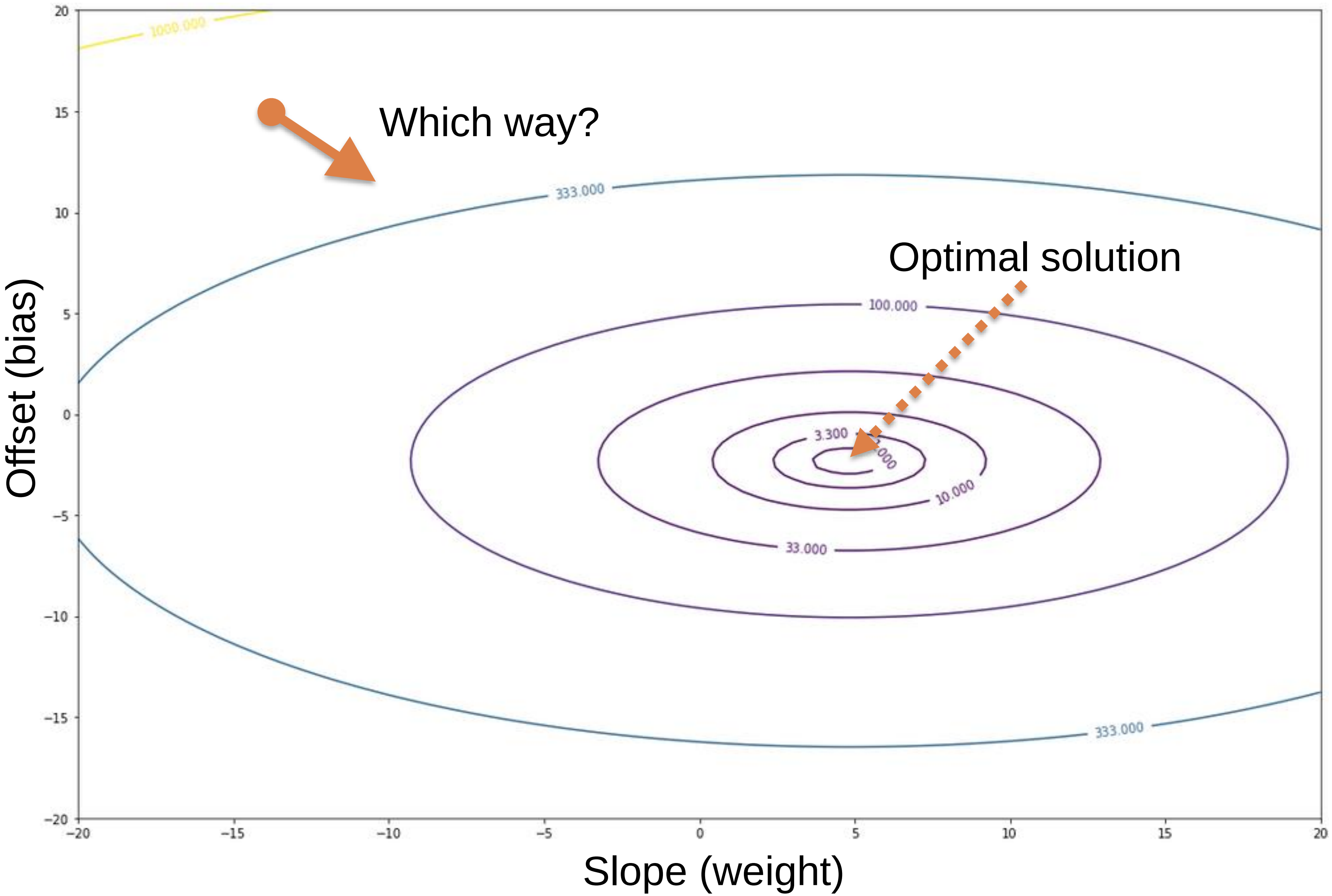
Error

Ground truth

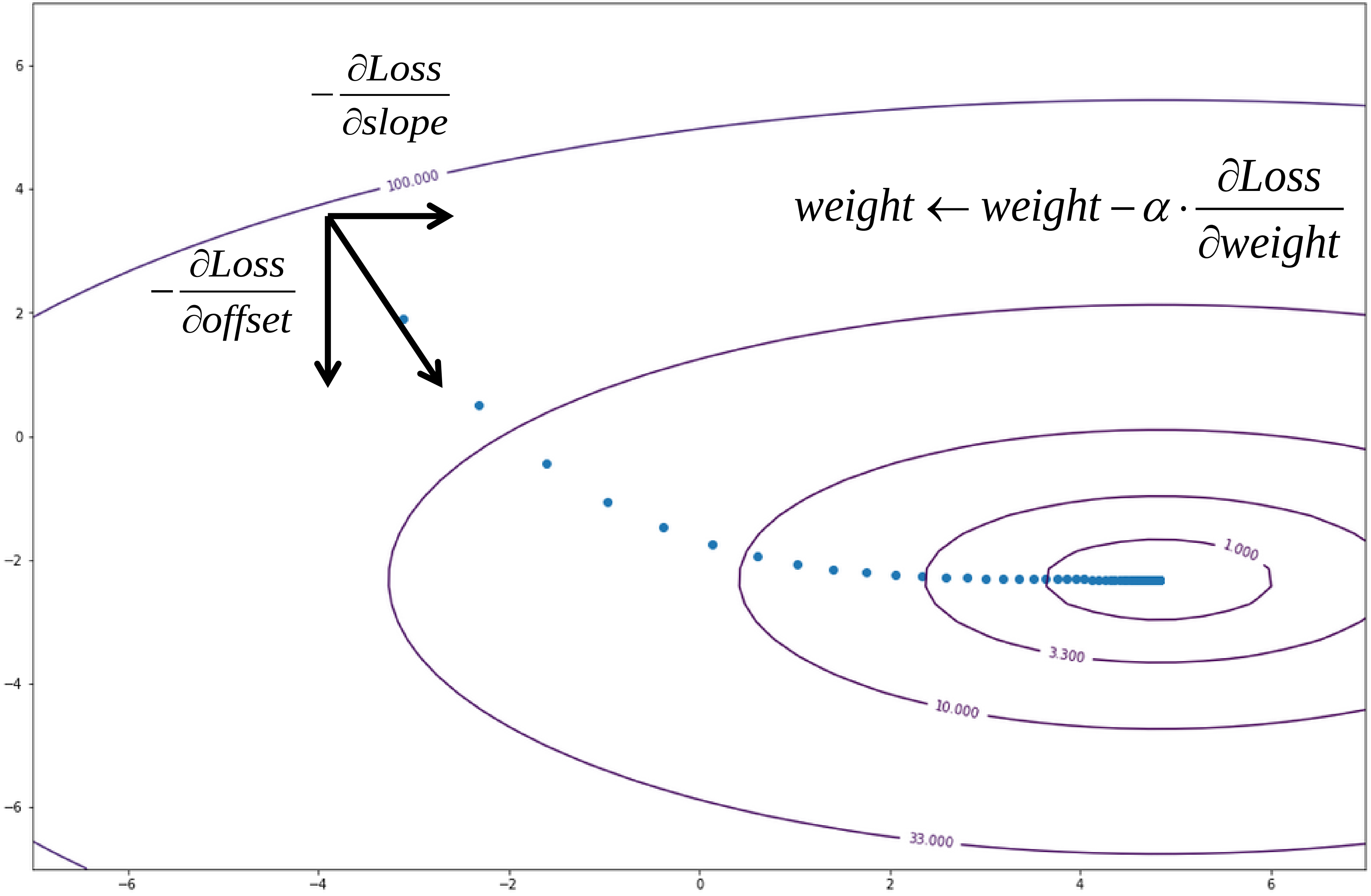
Cost as a Function of Slope and Offset



Contour Plot of Cost Function



Gradient Descent

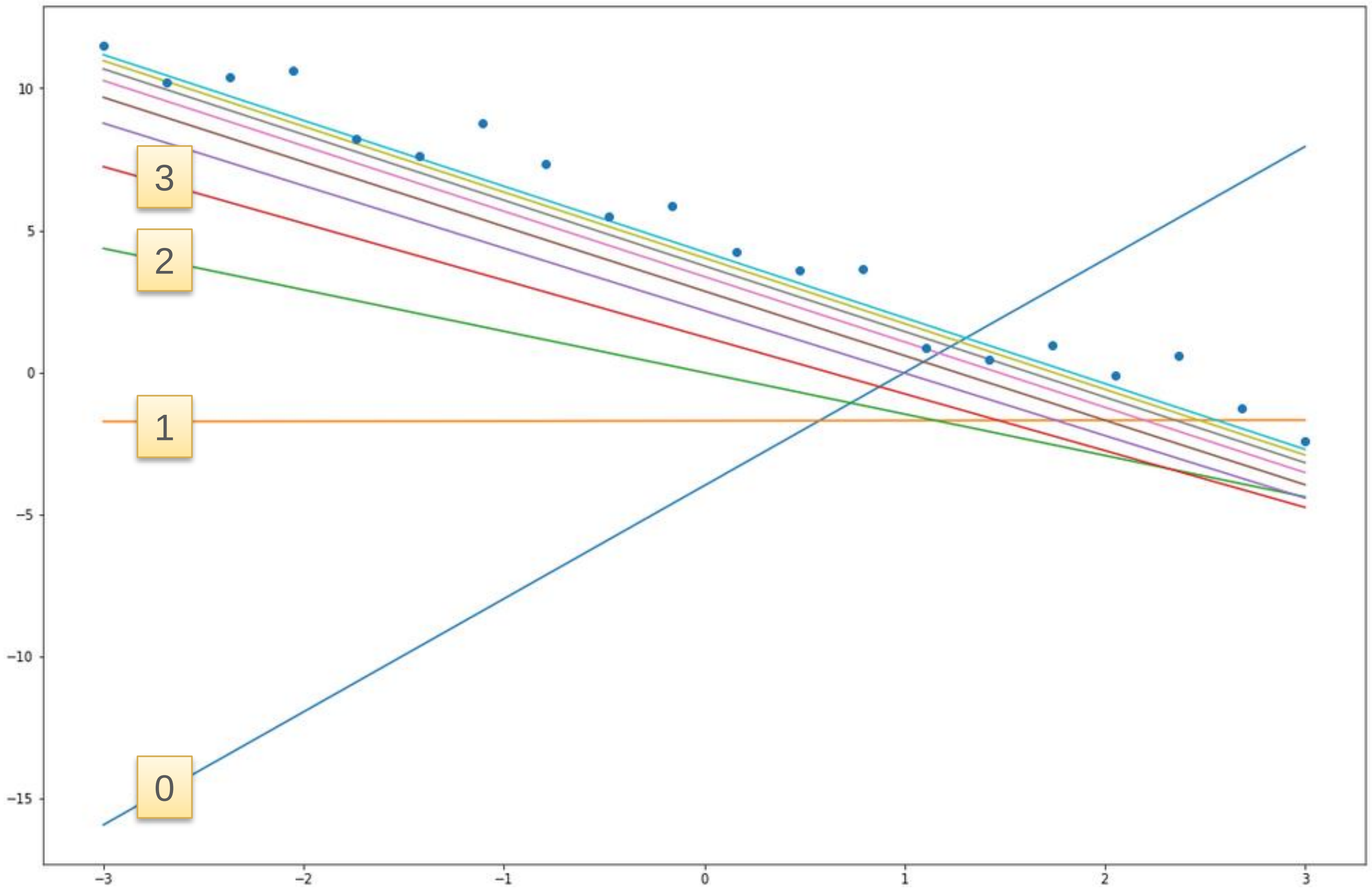


Gradient Descent Algorithm

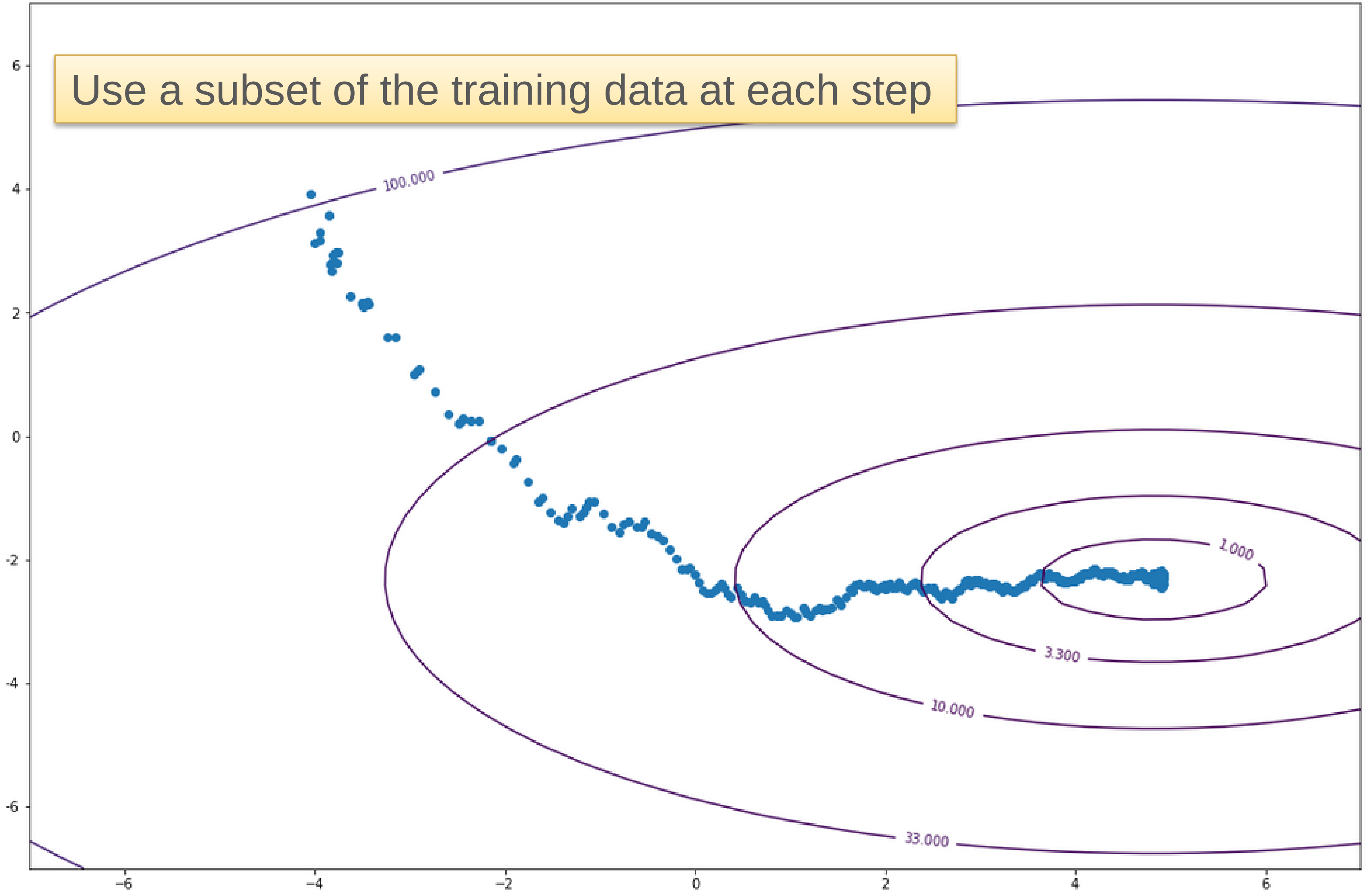
```
Initialize weights (trainable parameters)

for each training step:

    Calculate the gradient with respect to every weight

    for each weight:

        new_weight  = weight - learning_rate * gradient

    for each weight:

        weight = new_weight
```

Converging on the Minimum

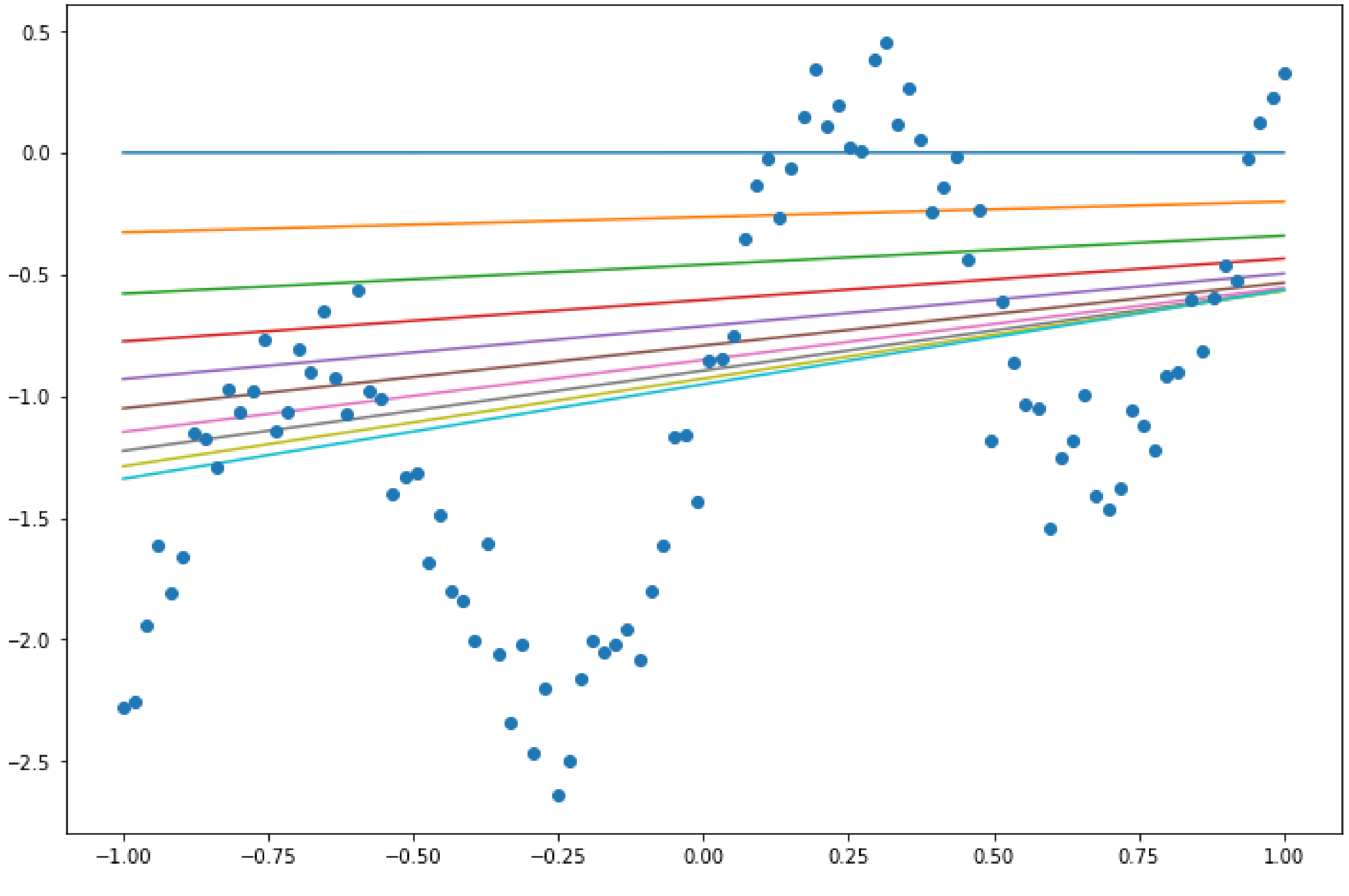
```
Final slope = -2.31499114425  offset = 4.38980555415
```



Stochastic Gradient Descent



Non-Linear Regression and Classification



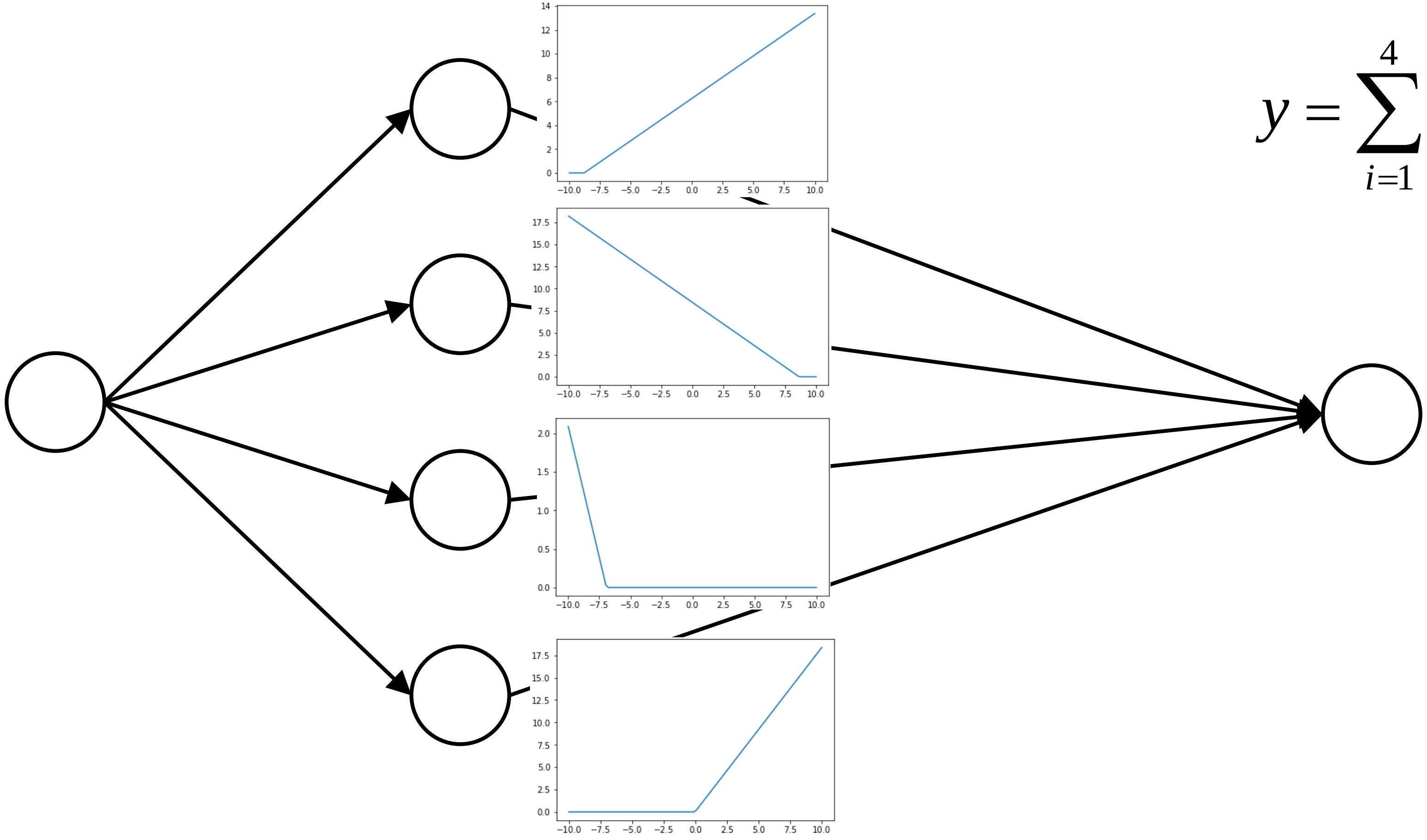
Piecewise Linear Approximation

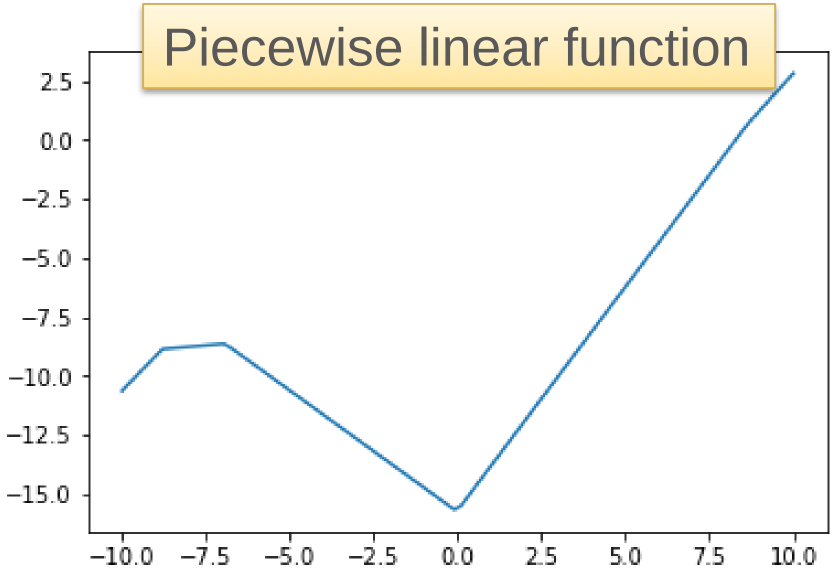
Input unit

Hidden units

Output unit

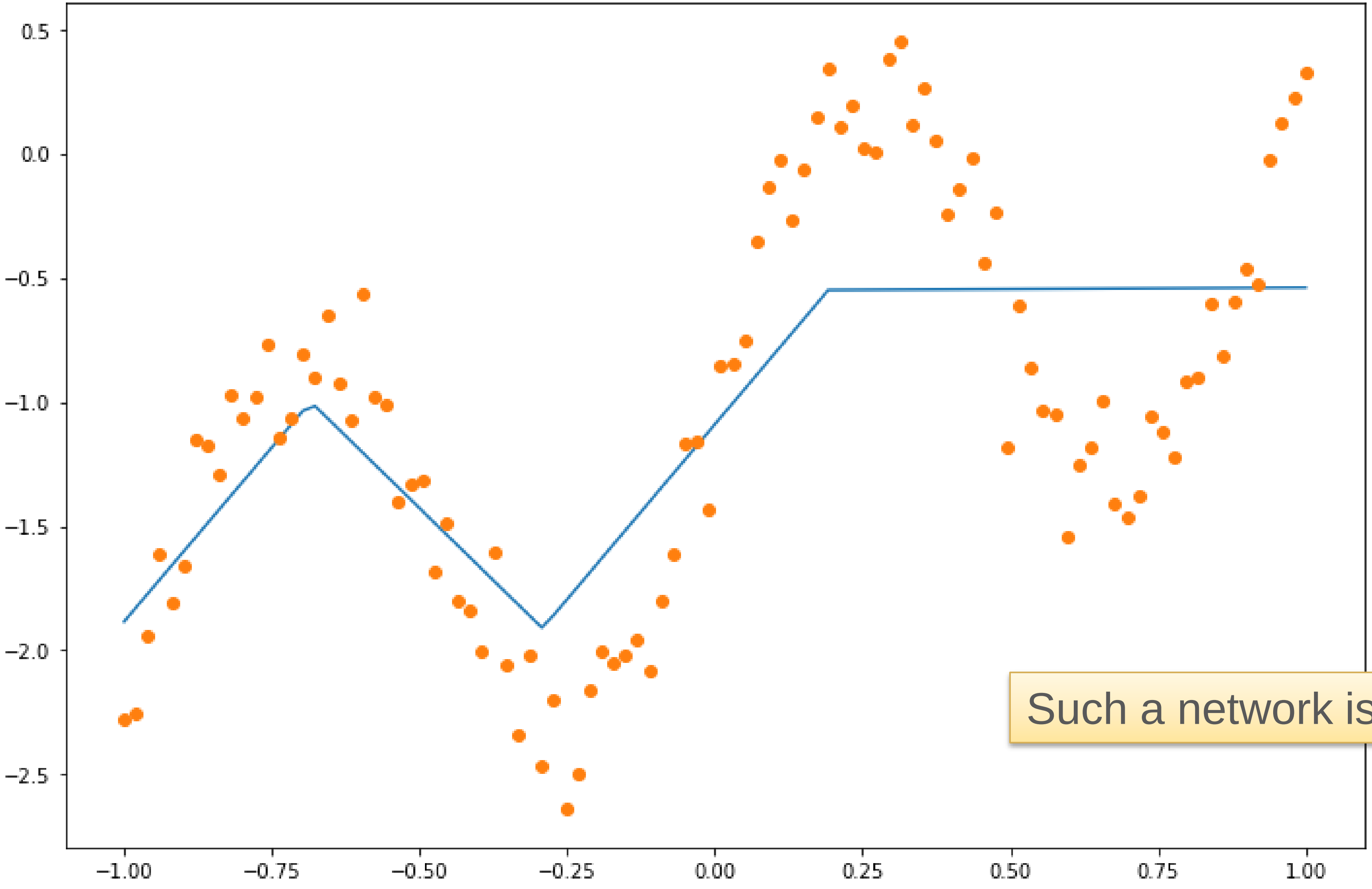
$$y = \sum_{i=1}^{4} w_i x_i + b$$





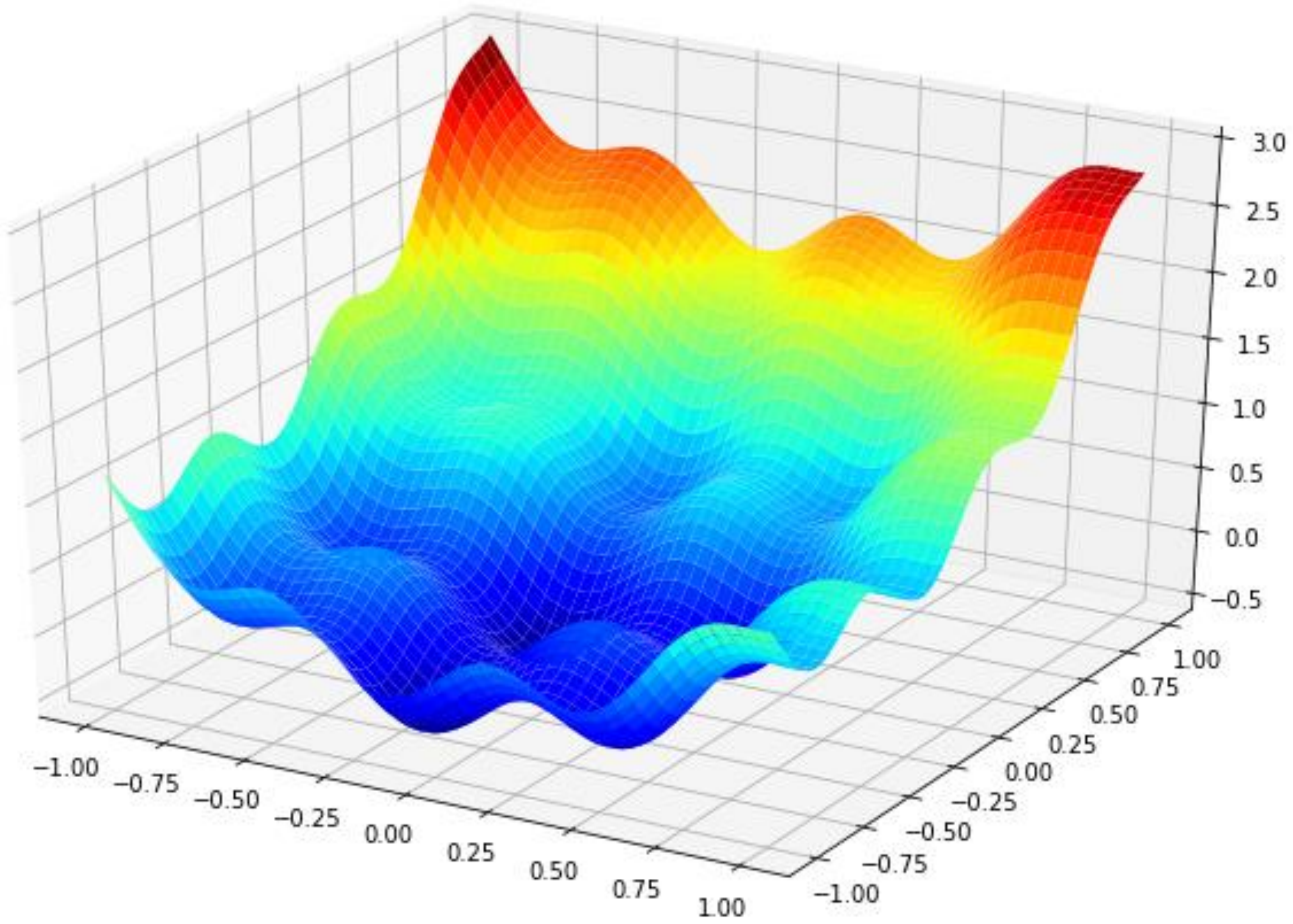
$$y_j = RELU(w_j x + b_j)$$

The Predicted Output



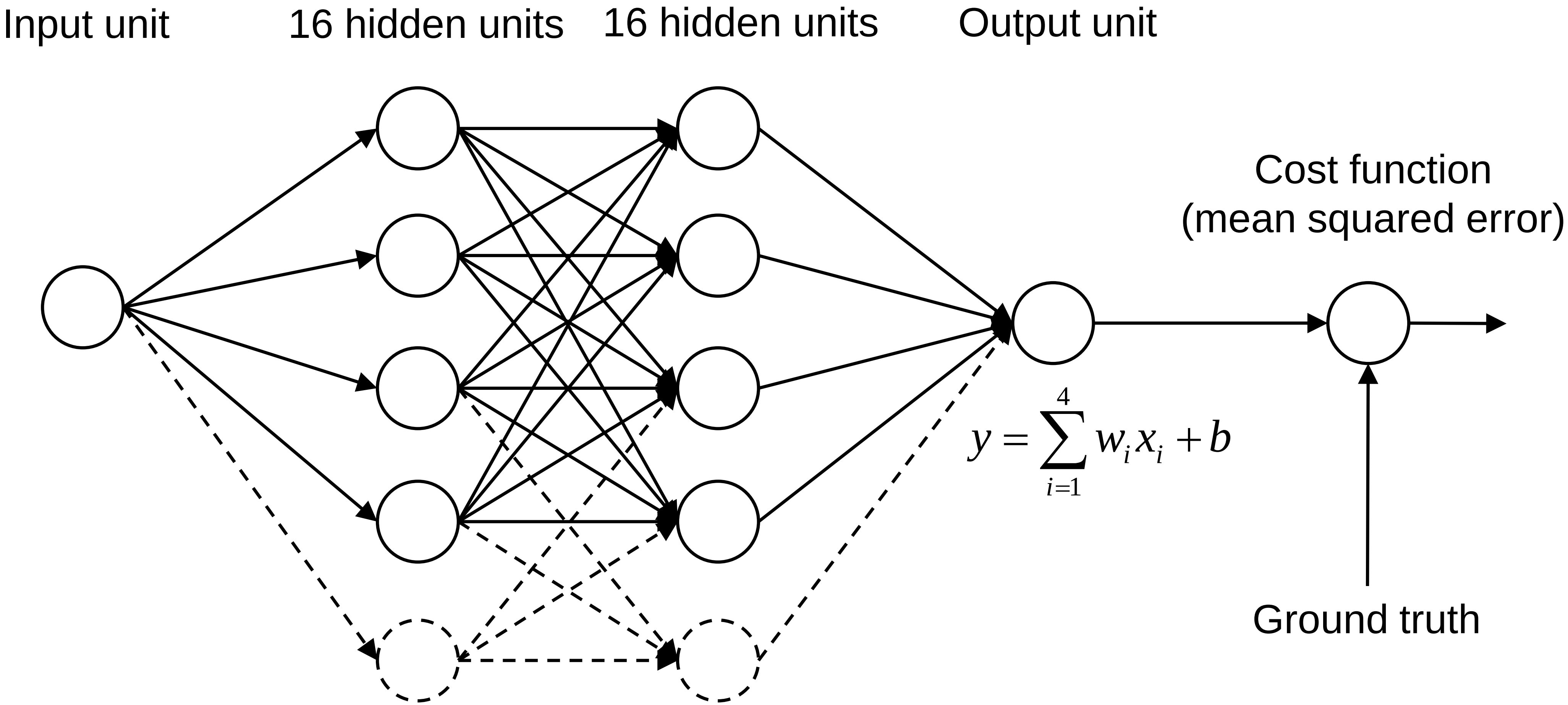
Such a network is hard to train

The Landscape of the Cost/Loss Function



Local and global minima and saddle points

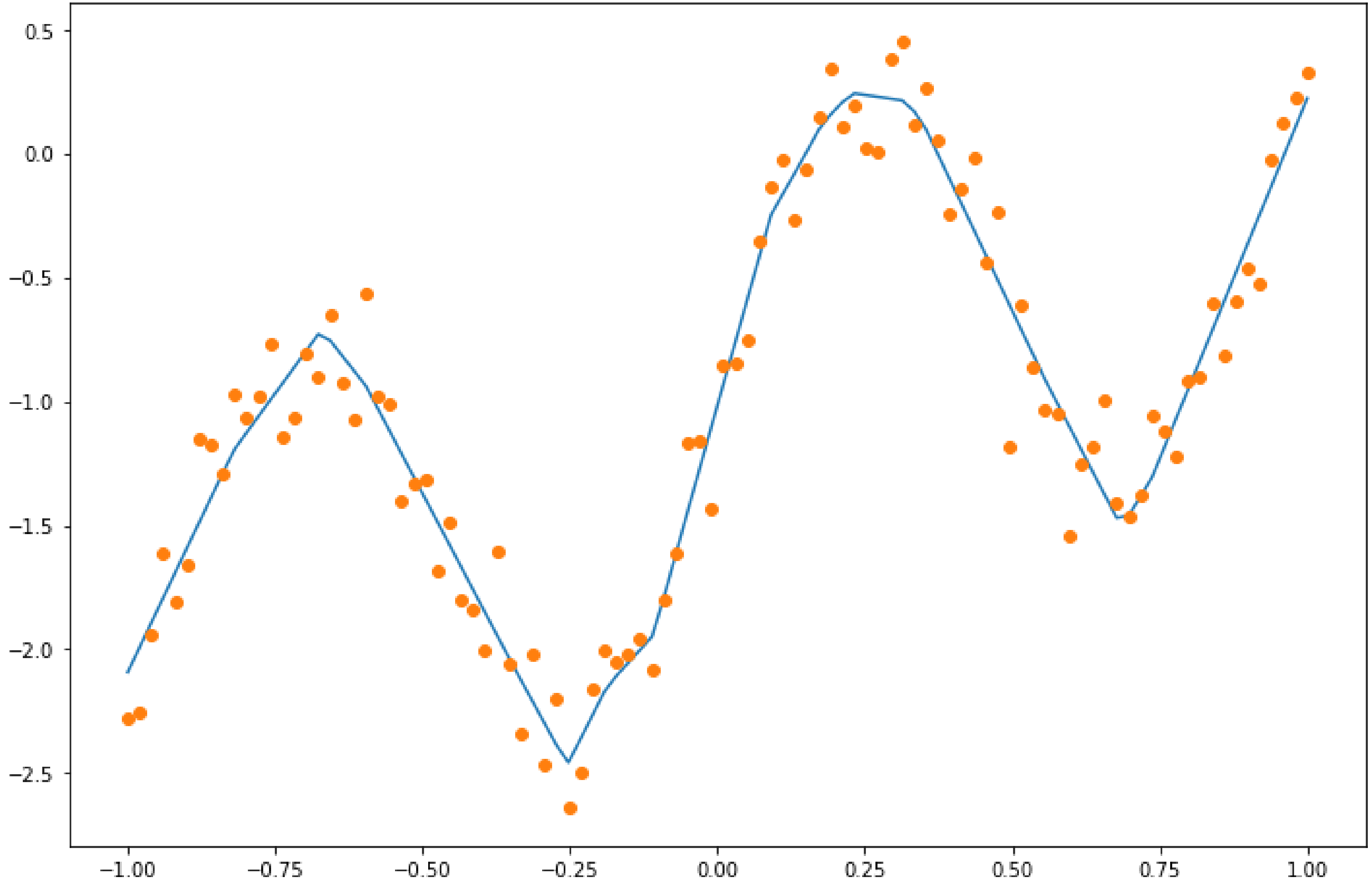
A Deep Neural Network



$$y = \sum_{i=1}^{4} w_i x_i + b$$

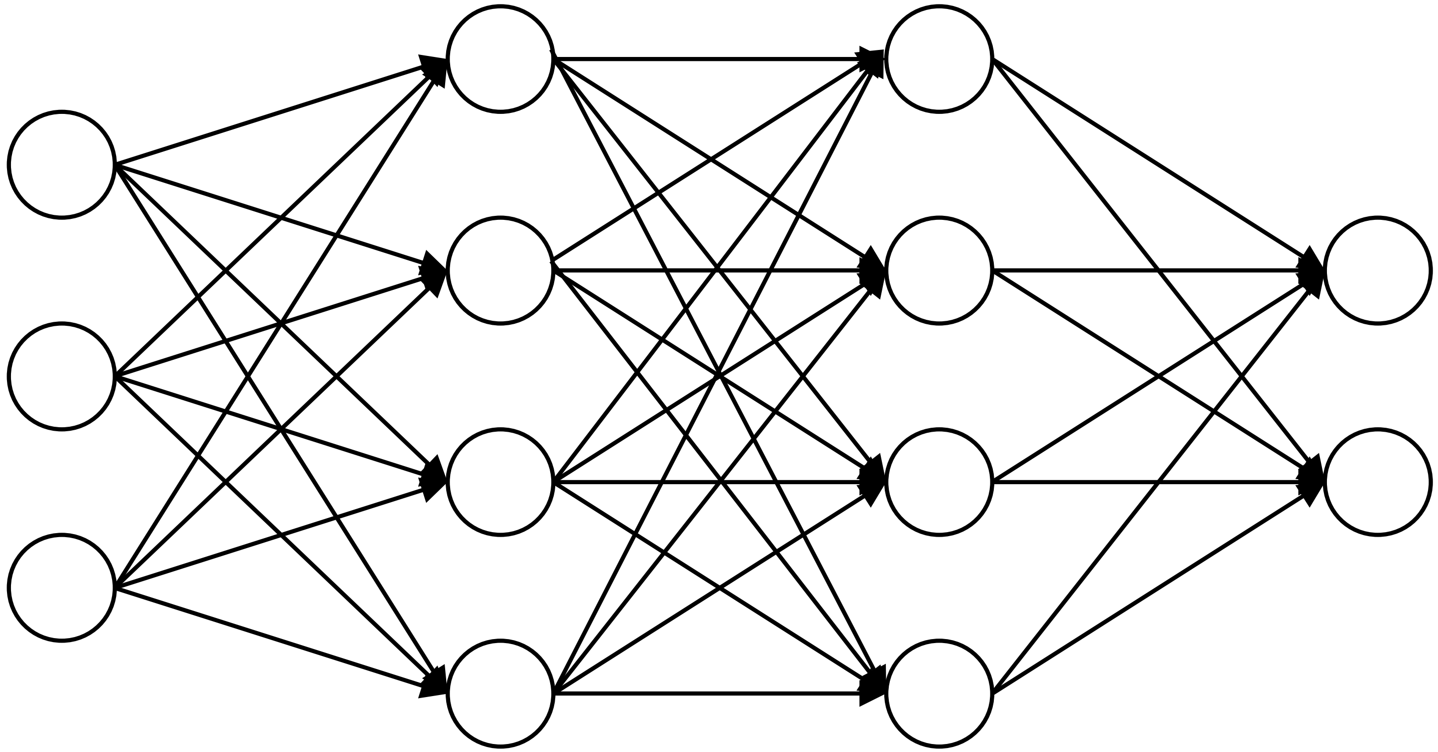
$$y_j = RELU\left(\sum_{i=1}^{n} w_{ji} x_i + b_j\right)$$

The Predicted Output



Forward and Back-Propagation

Forward propagation calculates weighted sums and activation function



Back propagation calculates gradients and adjusts weights

Deep Learning for Engineers

AI, ML, and Deep Learning

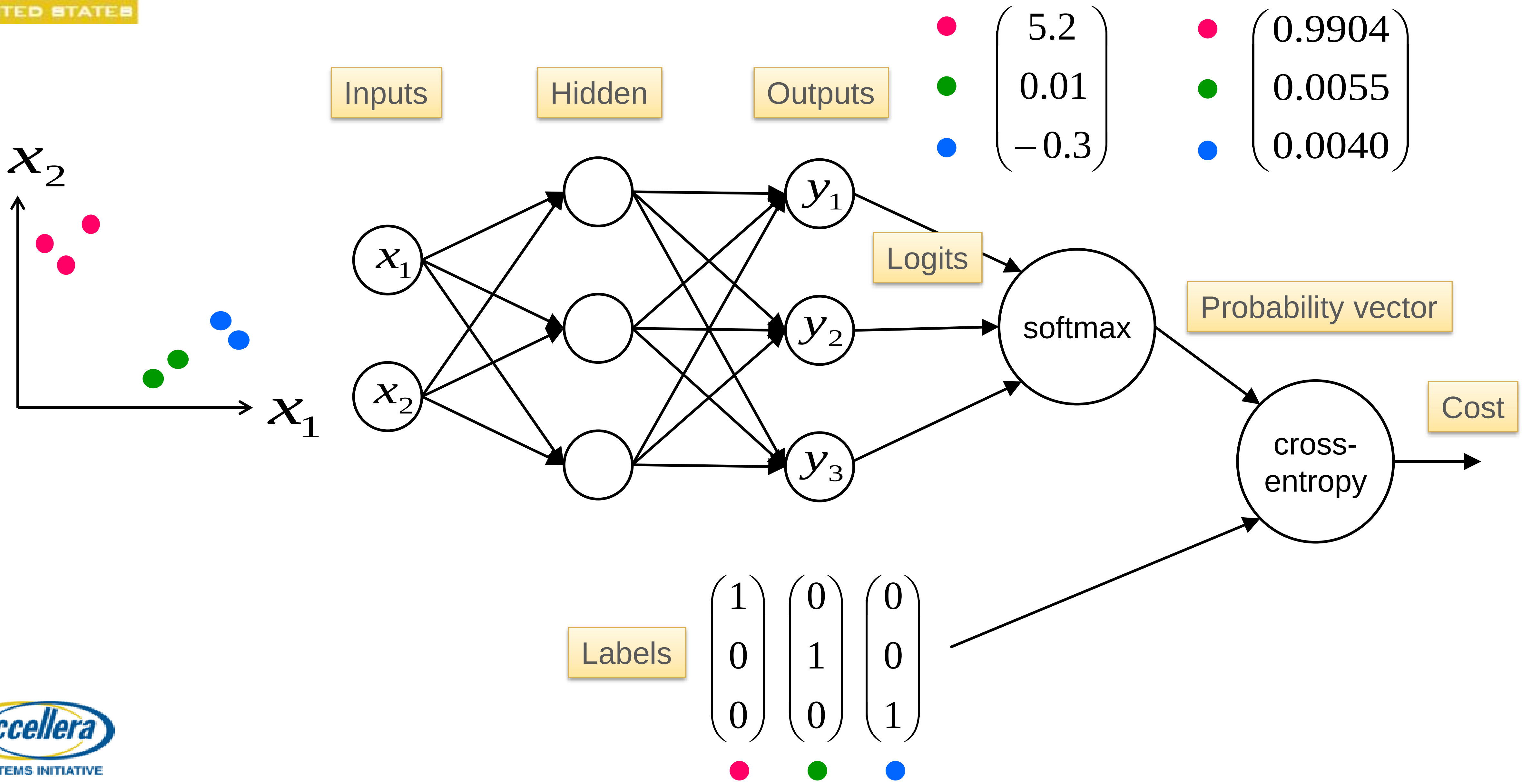
Training a Neural Network

Deeper Insights

CNNs and RNNs

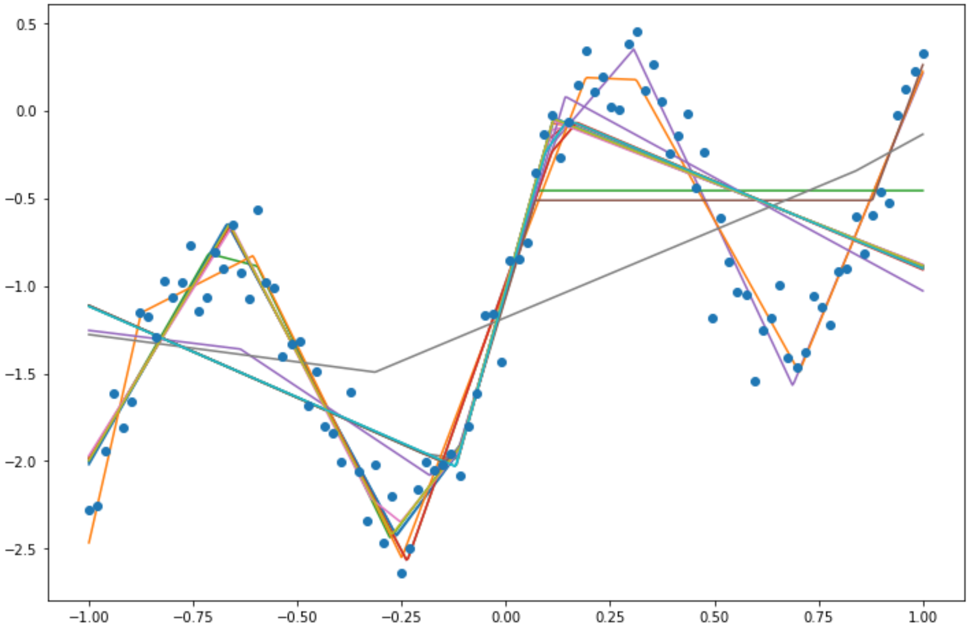
Tool Flow

Classification

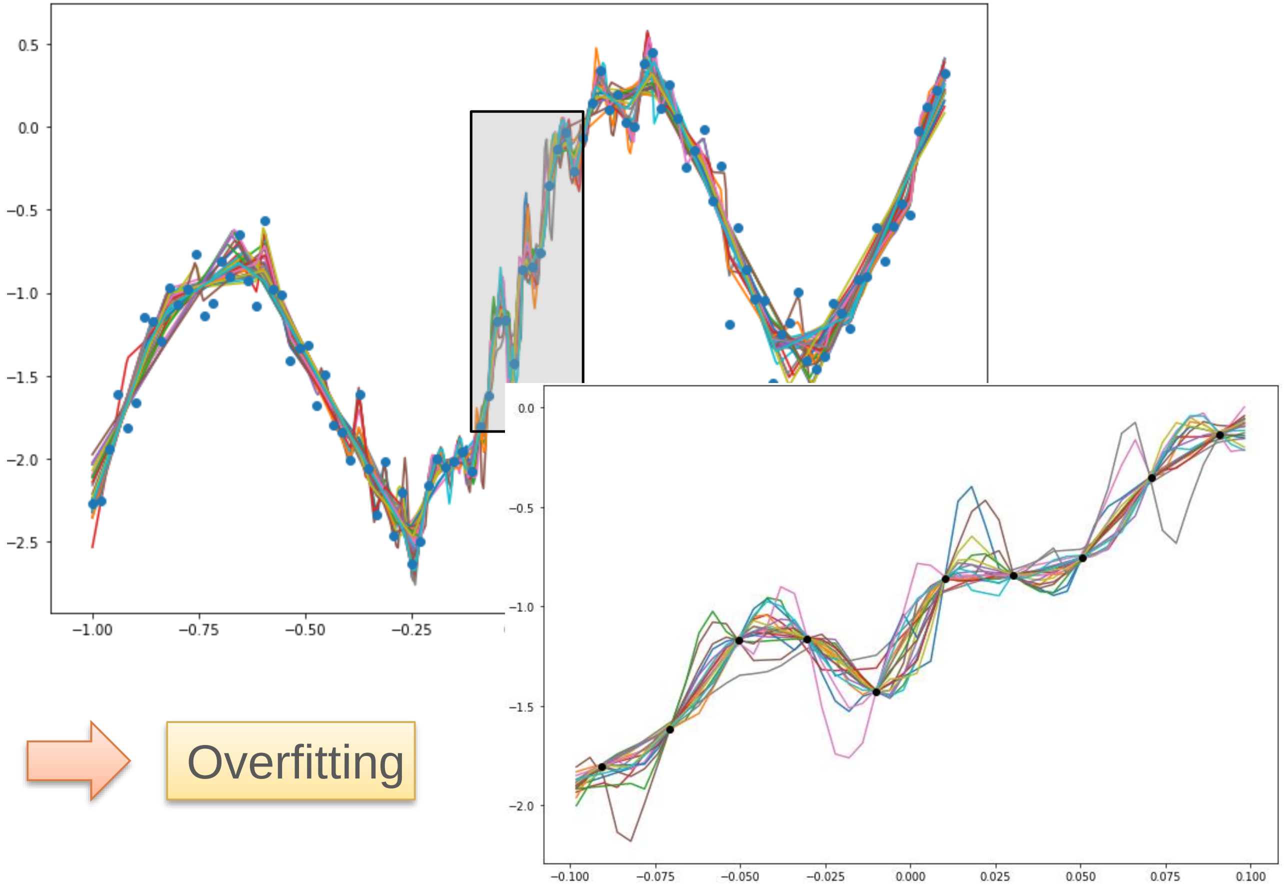


Underfitting and Overfitting

1 hidden layer of 6 neurons, 20 runs

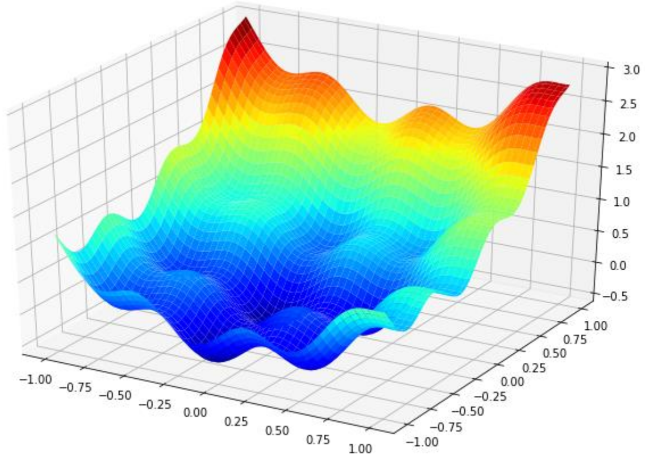


4 hidden layers of 16 neurons, 20 runs



Underfitting → Go Deeper! → Overfitting

Regularization



$$Loss(\vec{w}) = \frac{1}{2m}\sum_i \left(h(x_i) - y_i\right)^2$$

Naked cost function

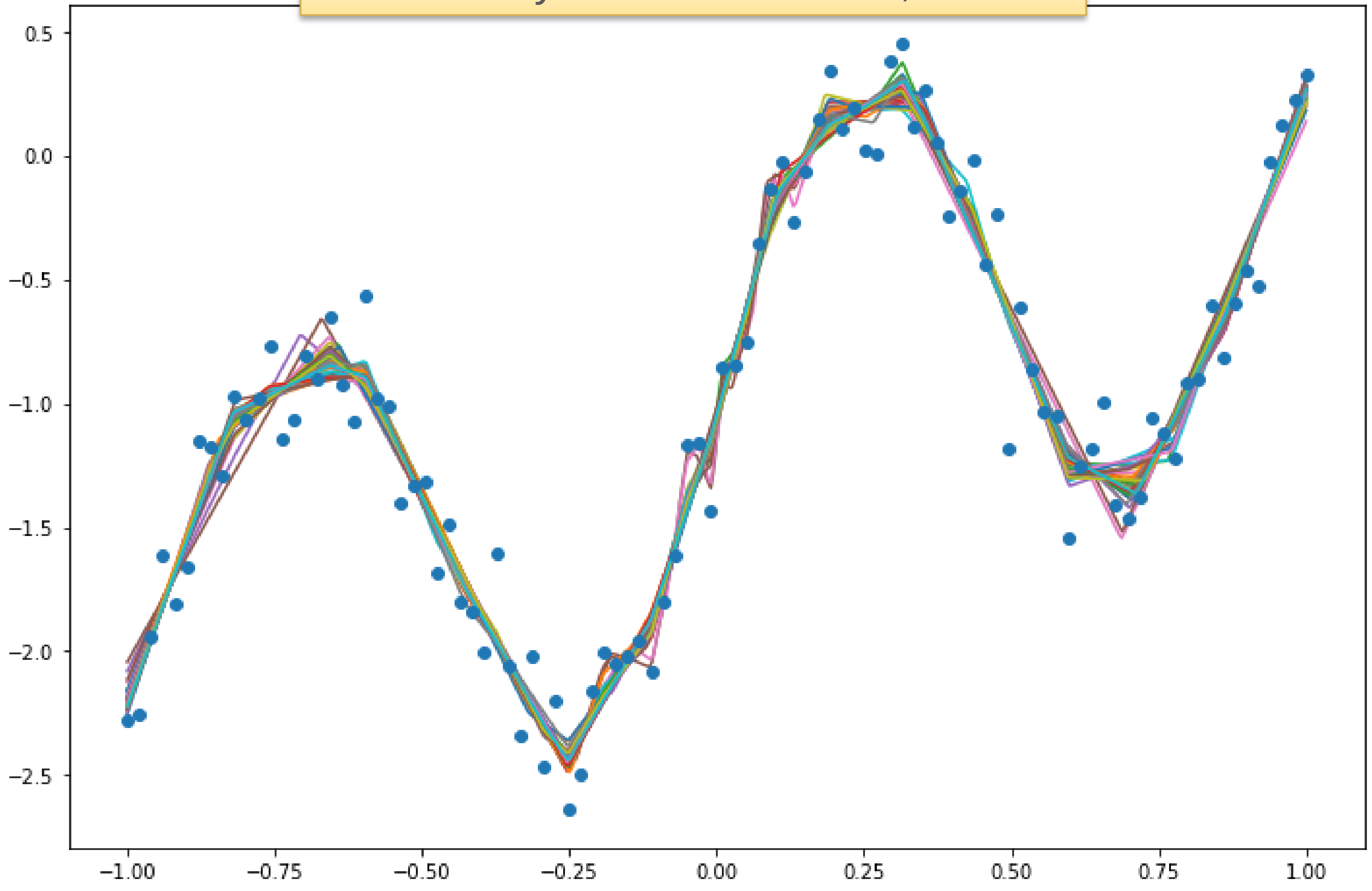
Regularization term

Mean square error for regression

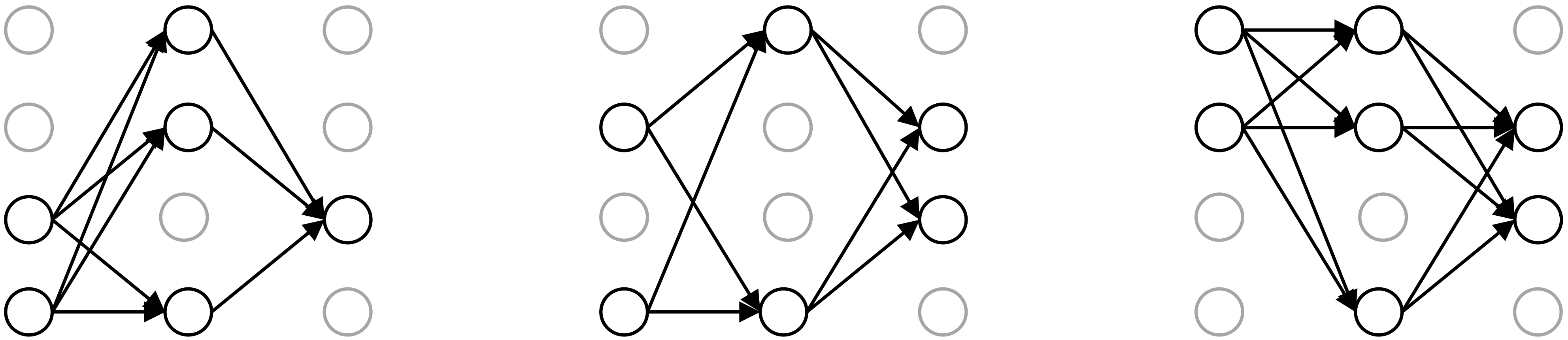
Softmax + cross-entropy for classification

With L2 Regularization

4 hidden layers of 16 neurons, 40 runs



Dropout



Training: Drop half the hidden units for each training step

Testing: Keep all the hidden units, divide activations by 2

Hyperparameters

Number of hidden layers

Number of neurons in each layer

Sigmoid or ReLU activation

Choice of cost function

Choice of gradient descent algorithm

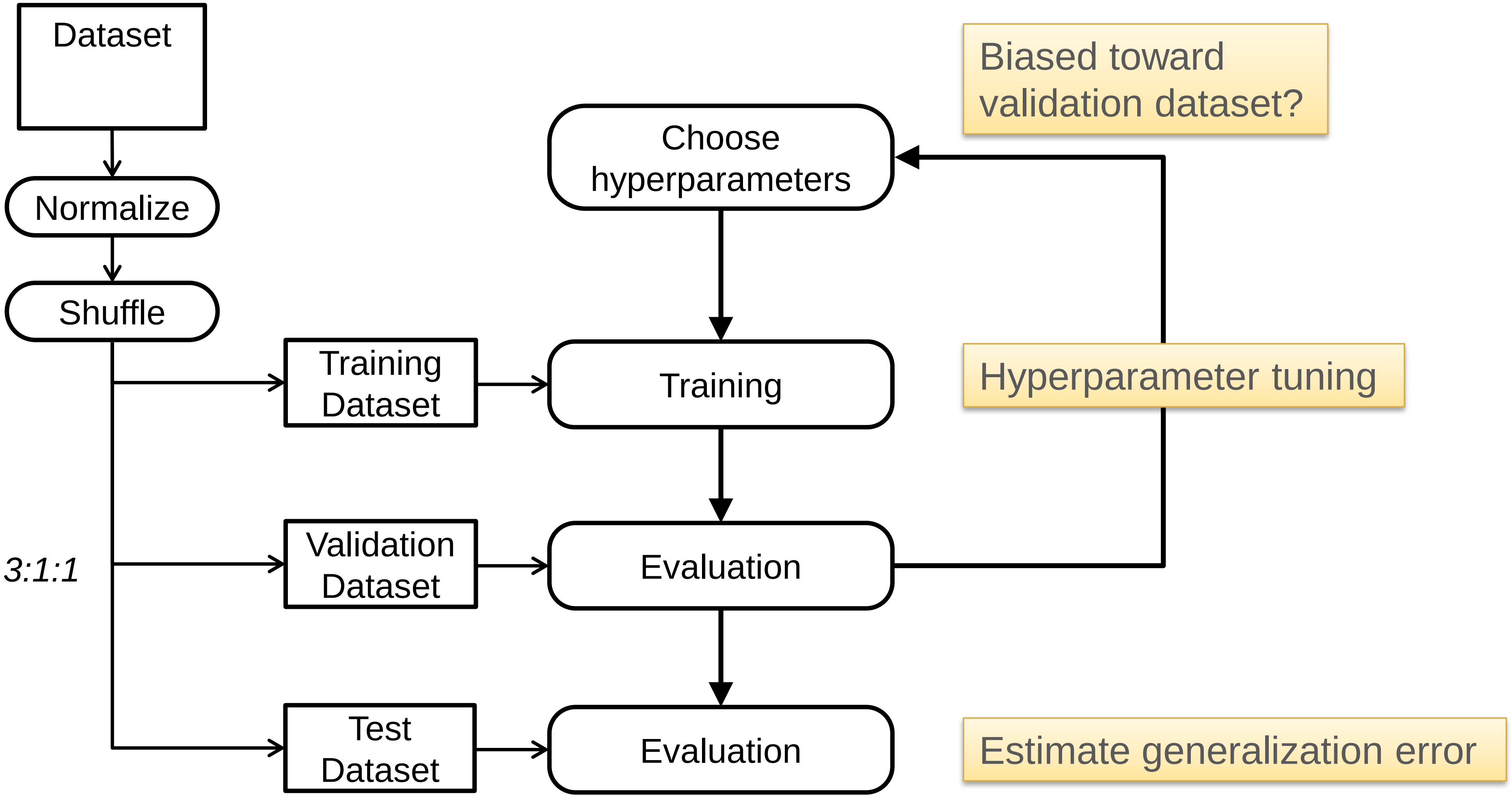
Learning rate

L2 regularization factor

Amount of dropout

(Many more ...)

Training, Validation, and Test Datasets



Training:validation:test ~ 3:1:1

Deep Learning for Engineers

AI, ML, and Deep Learning

Training a Neural Network

Deeper Insights

CNNs and RNNs

Tool Flow

Kinds of Neural Network

ANN – Artificial Neural Network

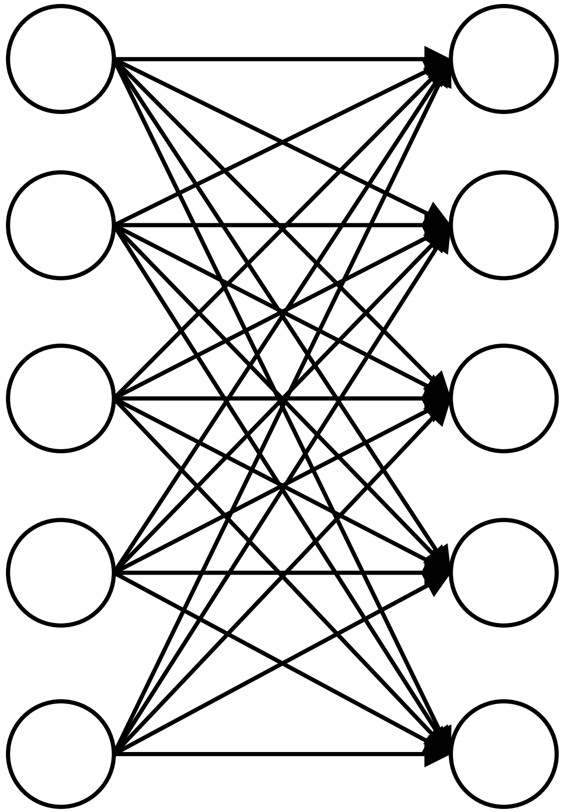
CNN – Convolutional Neural Network (e.g. object recognition)

R-CNN – Regional CNN (image segmentation)

RNN – Recurrent Neural Network (e.g. speech & text processing)

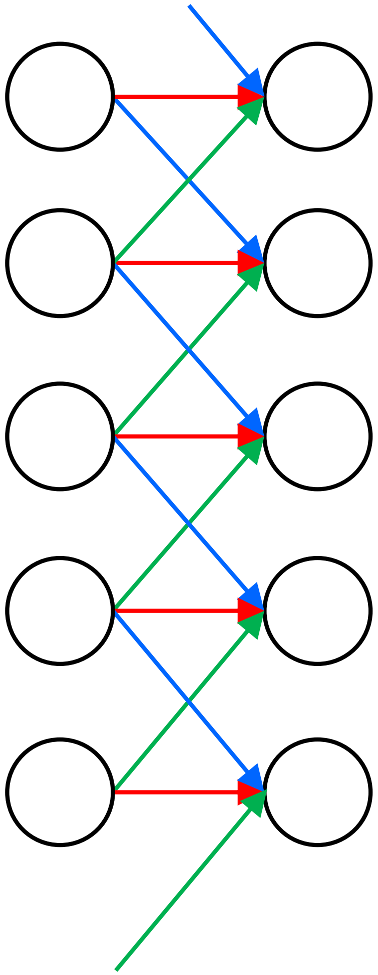
Convolutional Neural Network

Fully-connected

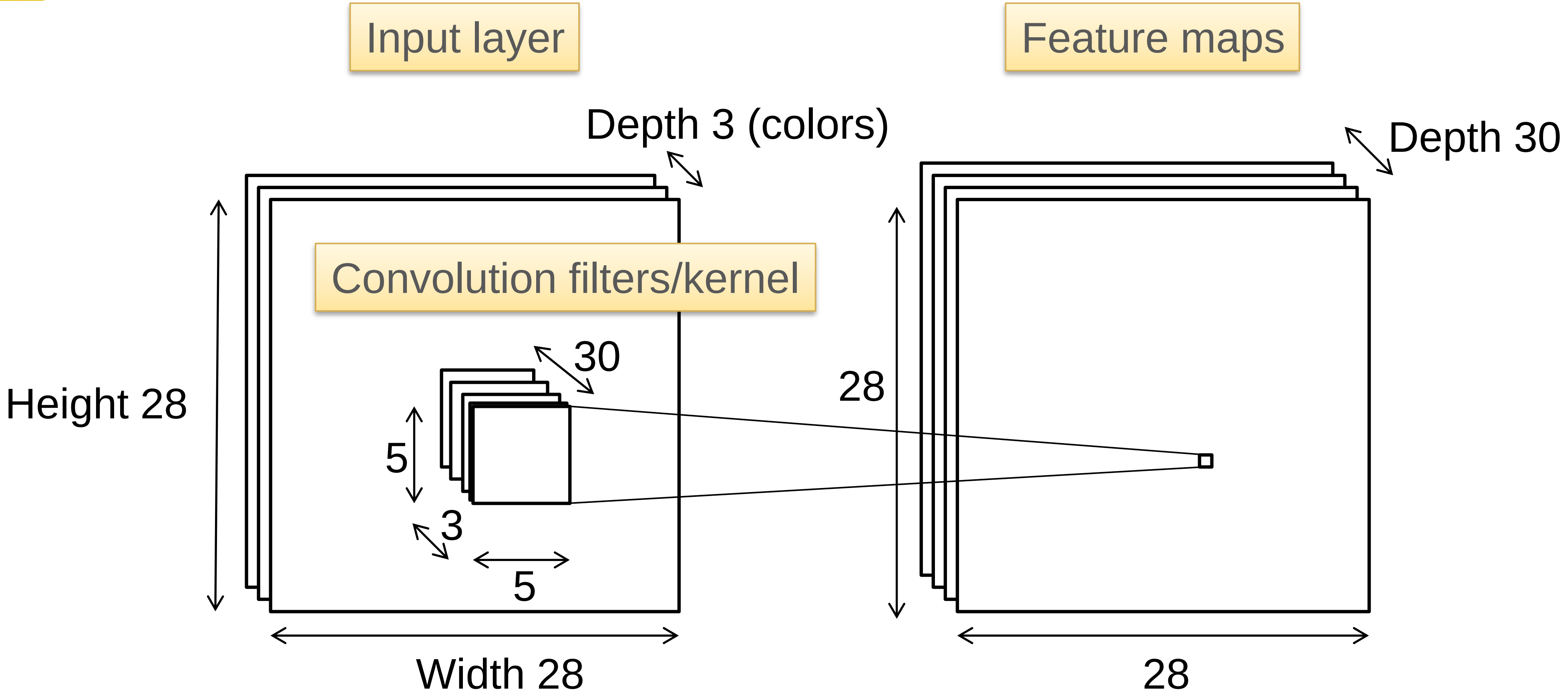


$$y_j = RELU\left(\sum_{i=1}^{5} w_{ji}x_i + b_j\right)$$

3x convolution



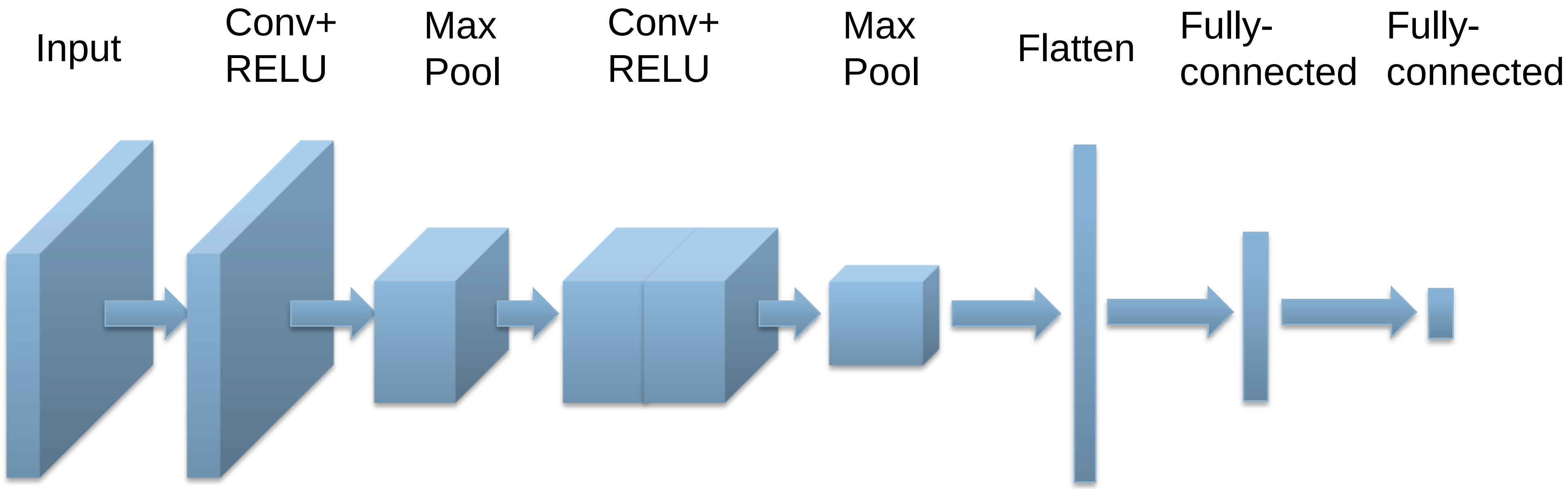
Multiple Filters and Feature Maps



$$y_n = \sum_{i=1}^{5} \sum_{j=1}^{5} \sum_{k=1}^{3} w_{ijkn} x_{ijk} + b_n$$

5 x 5 x 3 x 30 weights, 30 biases

The Classical CNN Architecture



Shape	32x32x3	32x32x16	16x16x16	16x16x32	8x8x32	2048	128	10
# Parameters		416		832			262,272	1,290
# Values	3,072	16,384	4,096	8,192	2,048	2,048	128	10

Features Detection

Classifier

Evolution of CNN Architectures

Traditional CNN:

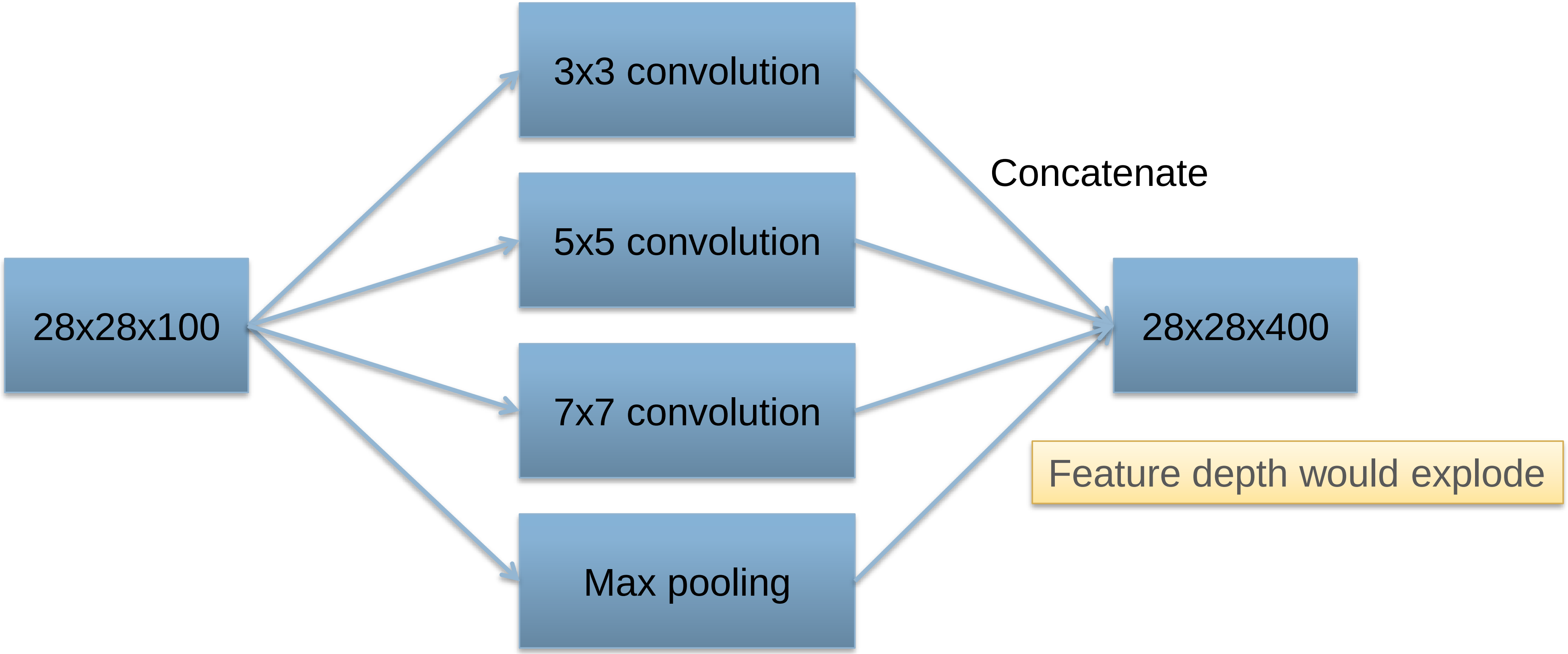
$$\underbrace{\text{Convolution} \rightarrow \text{pool} \rightarrow \text{convolution} \rightarrow \text{pool}}_{\text{Hierarchical feature detectors}} \rightarrow \underbrace{\text{full} \rightarrow \text{full} \rightarrow \text{full}}_{\text{Classifier}} \rightarrow \text{output}$$

Sledgehammer!

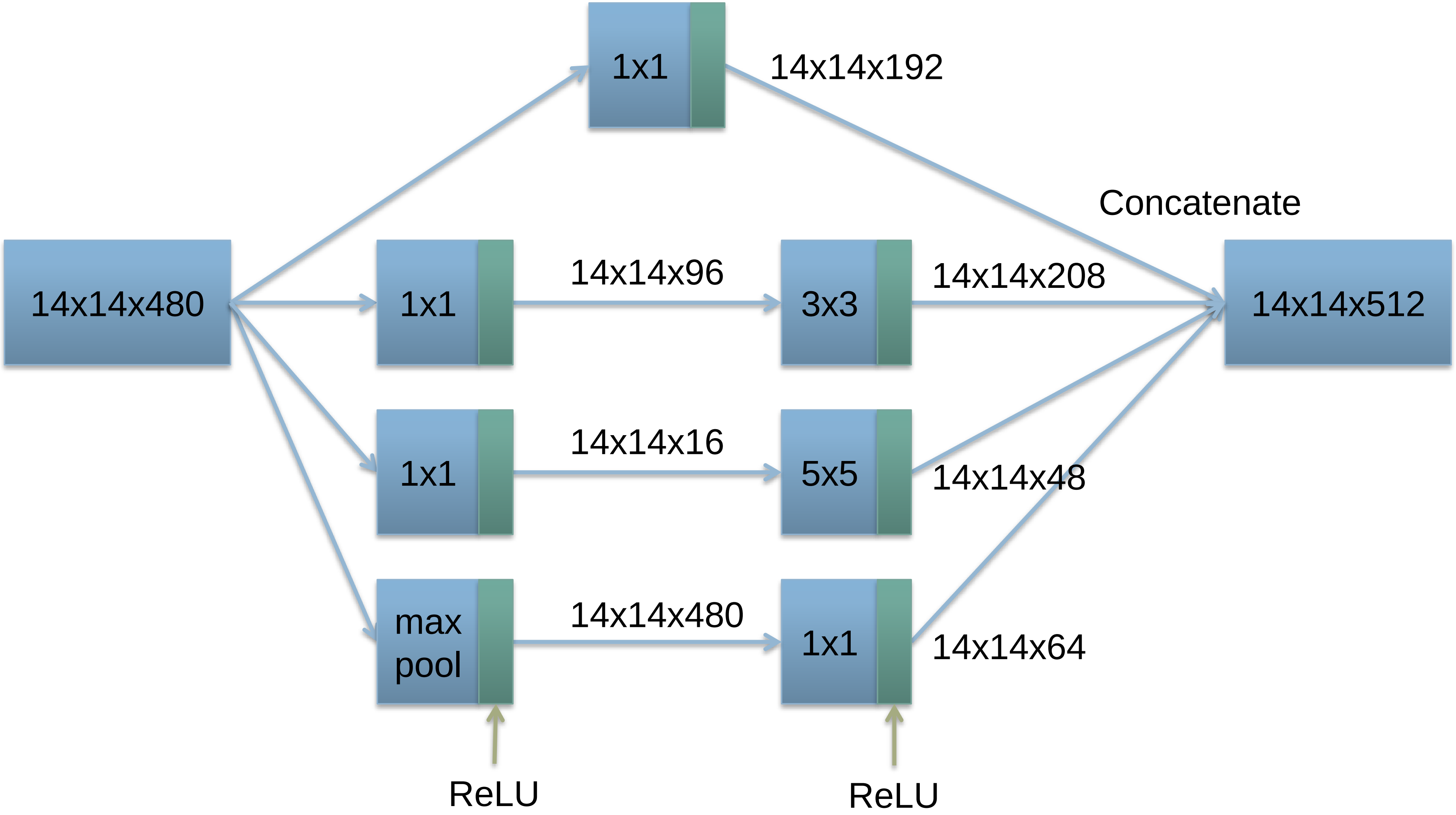
Replace with convolutions – Fully Convolutional Network

Naïve Inception Module

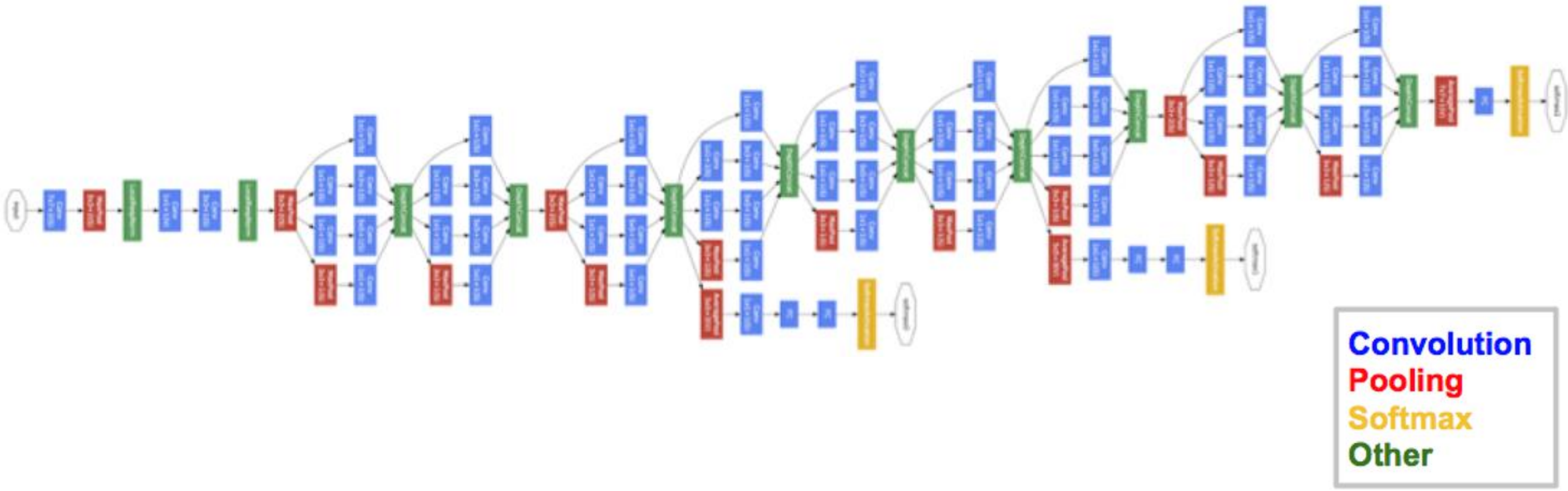
Don't know what the optimal sparse structure is, so hedge our bets:



Example GoogLeNet Inception Module

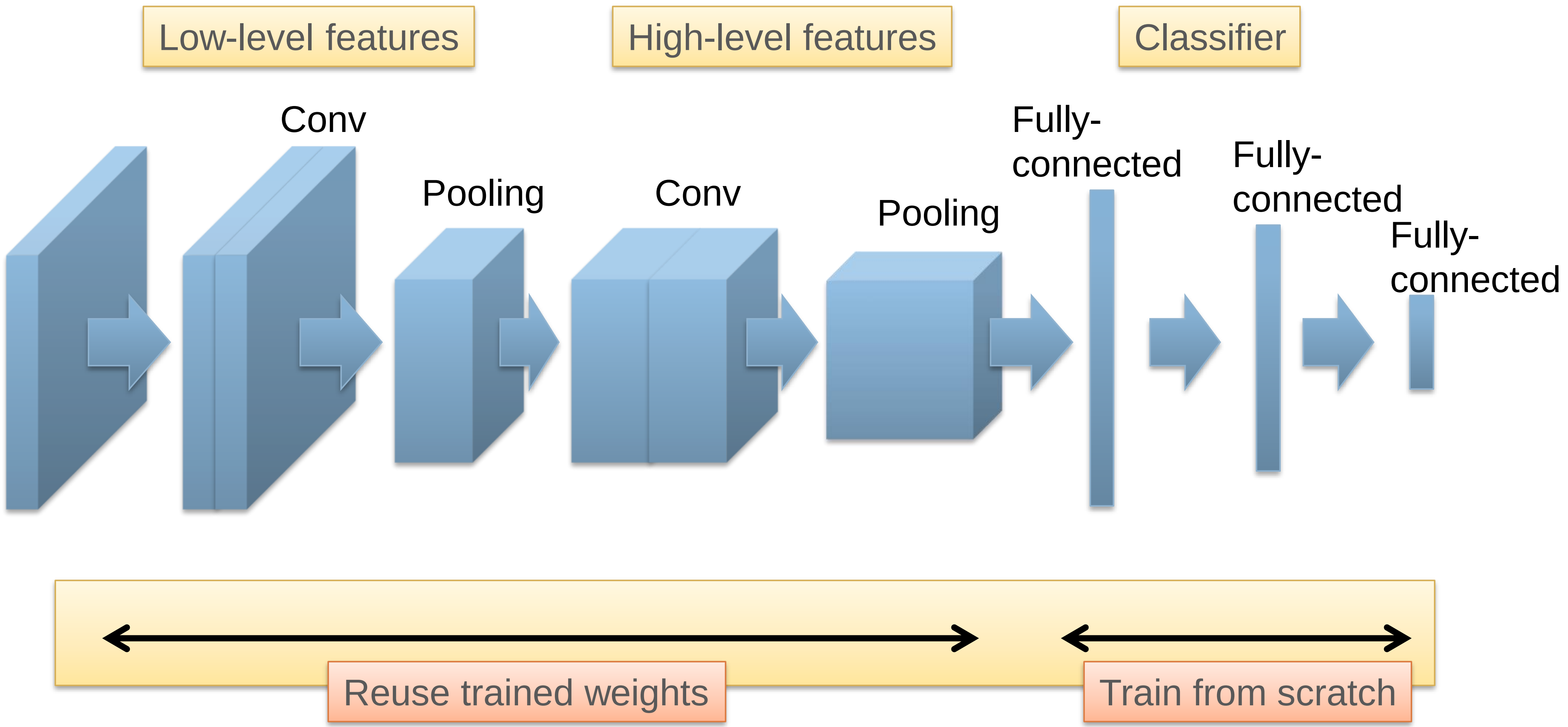


GoogLeNet

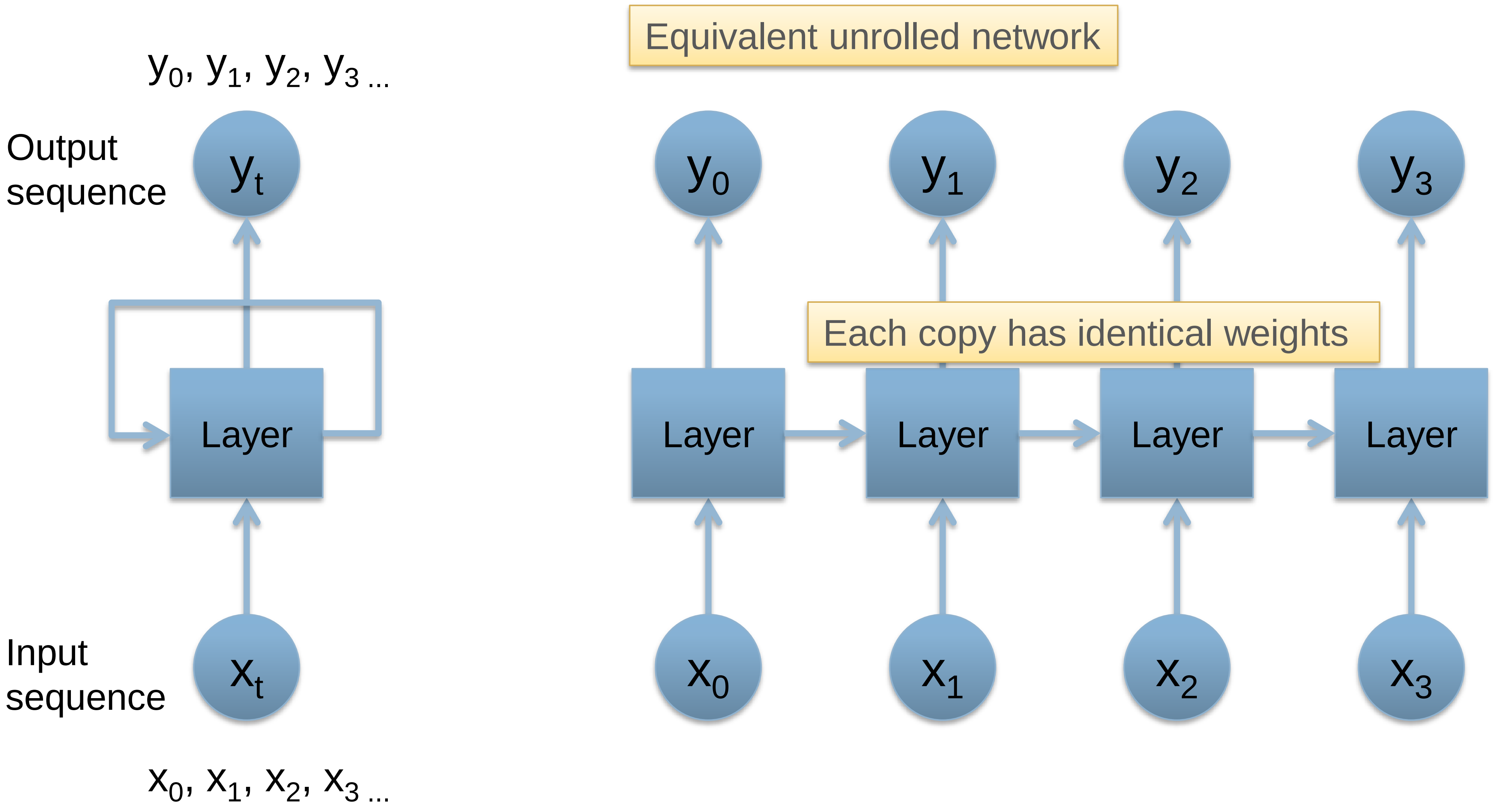


C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich (2014) *Going Deeper with Convolutions.* https://arxiv.org/abs/1409.4842

Transfer Learning



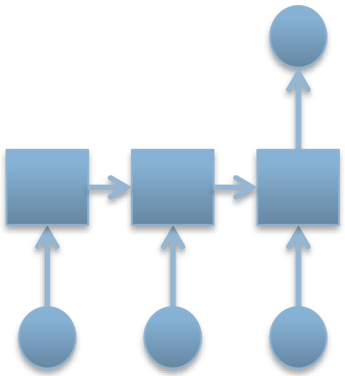
Recurrent Neural Network (RNN)



RNN Applications

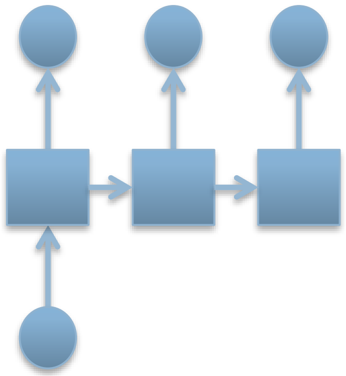
Natural Language Processing

Category / next word
Output is final state



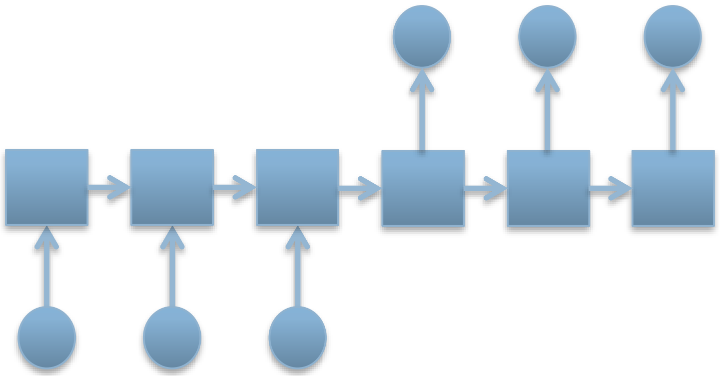
Sequence of words

Sequence of words
Output is whole sequence



Image

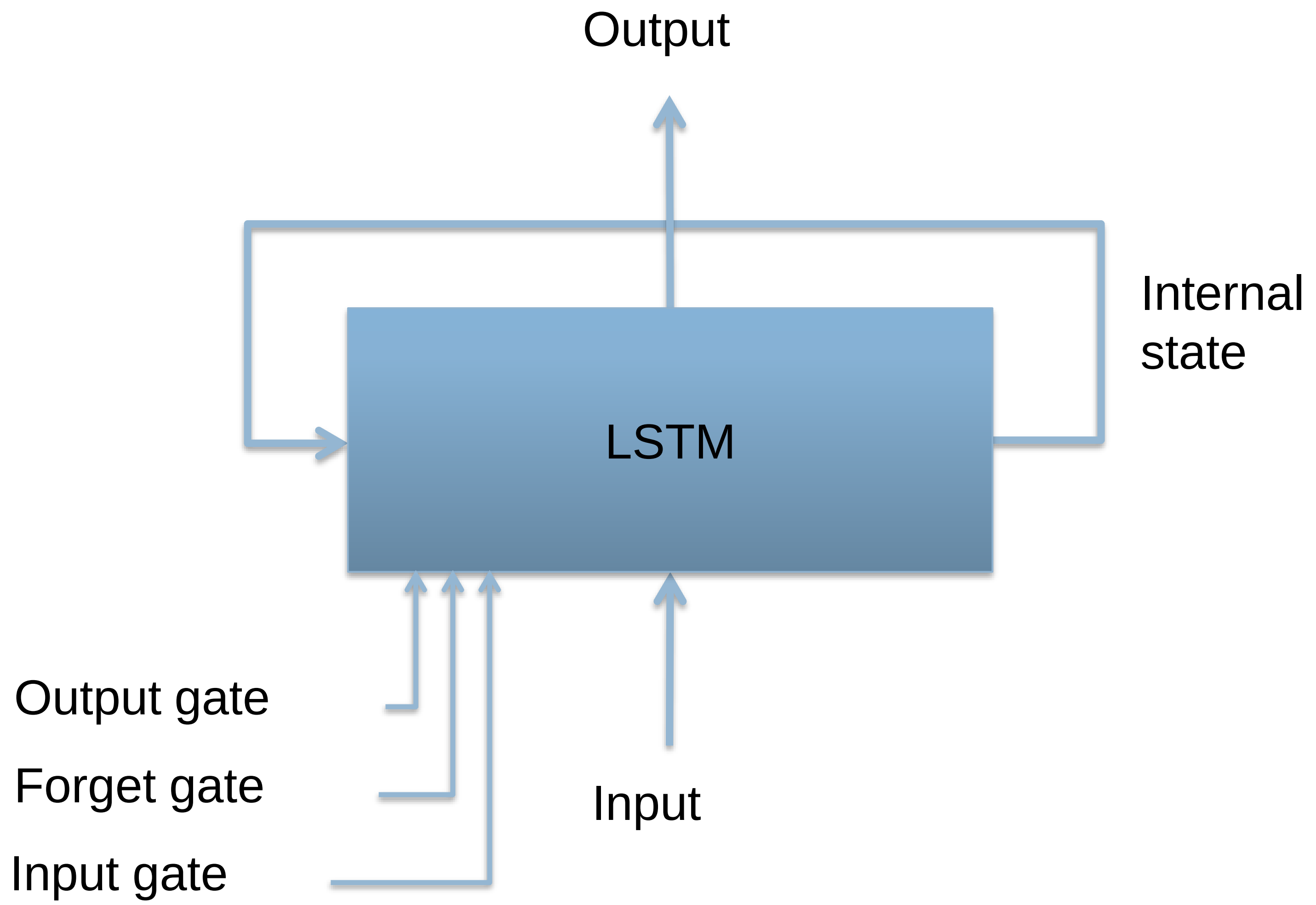
French sentence



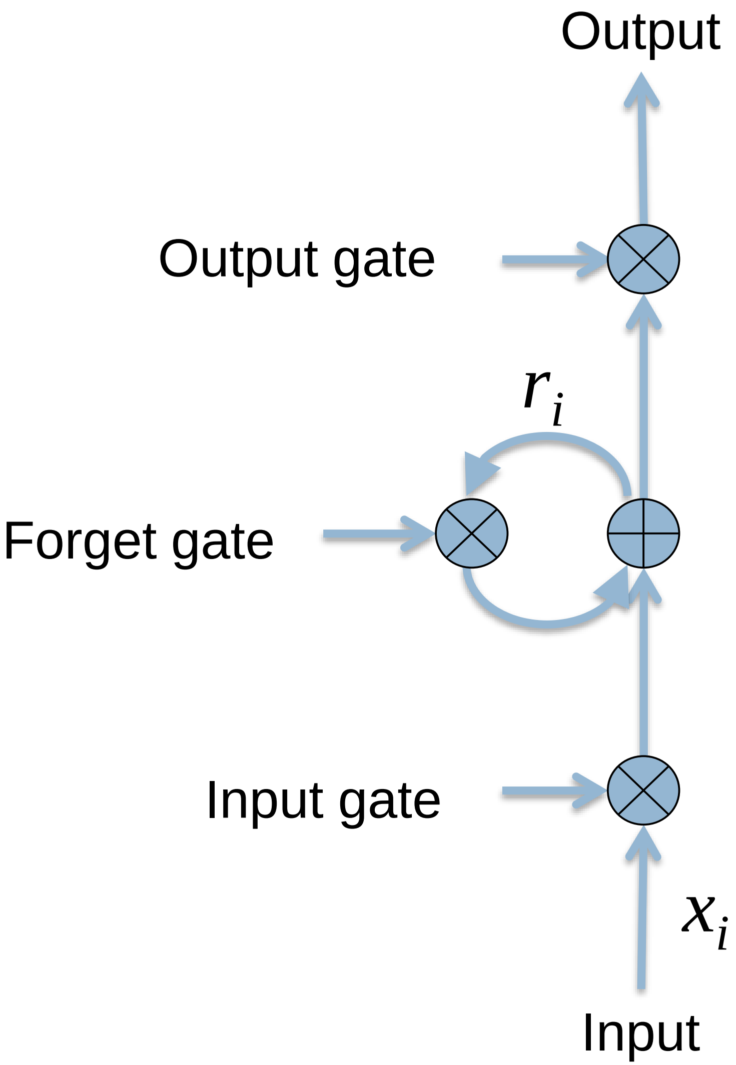
English sentence

Both input and output sequences can be variable length
Very powerful and effective, but training can be tricky

Long Short Term Memory – LSTM



LSTM Gates



$$Gate = \sigma\left(\sum_i w_i x_i + \sum_i u_i r_i\right)$$

LSTM Trained on Linux Source Code

Generated by a 3-layer LSTM trained on the entire Linux source code ...

```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
  int error;
  if (fd == MARN_EPT) {
    /*
     * The kernel blank will coeld it to userspace.
     */
    if (ss->segment < mem_total)
      unblock_graph_and_set_blocked();
    else
      ret = 1;
    goto bail;
  }
```

Andrej Karpathy (2015). *The Unreasonable Effectiveness of Recurrent Neural Networks*. [Online] http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Deep Learning for Engineers

AI, ML, and Deep Learning

Training a Neural Network

Deeper Insights

CNNs and RNNs

Tool Flow

Training versus Inference

Training	Inference
Desktop or cloud computing	Cloud or edge computing
Large dataset	One sample at a time
Forward and backward passes through neural network	Forward pass only (simpler network)
Minutes-weeks on GPU	Milliseconds-seconds/sample on edge device
Weights are computed	Weights are known and can be compressed

Open Source Training Frameworks

Caffe

Berkeley Artificial Intelligence Research (BAIR)

Berkeley Vision

Caffe2

facebook

TensorFlow™

Google

Keras

Microsoft | Cognitive Toolkit

torch

mxnet

theano

Cloud Platforms for Training and Inference

aws Amazon SageMaker

Google Cloud

Microsoft Azure

IBM

IBM Cloud

IBM AI OpenScale

and many others

The Cloud can Provide

Just the hardware (VM)

VM with pre-installed, pre-configured ML software

MLaaS – Machine Learning as a Service

Specific ML services (language translation, chat bots, ...)

ML / DL Tool Flow for Edge Computing

ONNX Open Neural Network Exchange Format

Training

- Caffe
- TensorFlow ™

→ Format conversion

ML / DL Tool Flow for Edge Computing

Training

Caffe

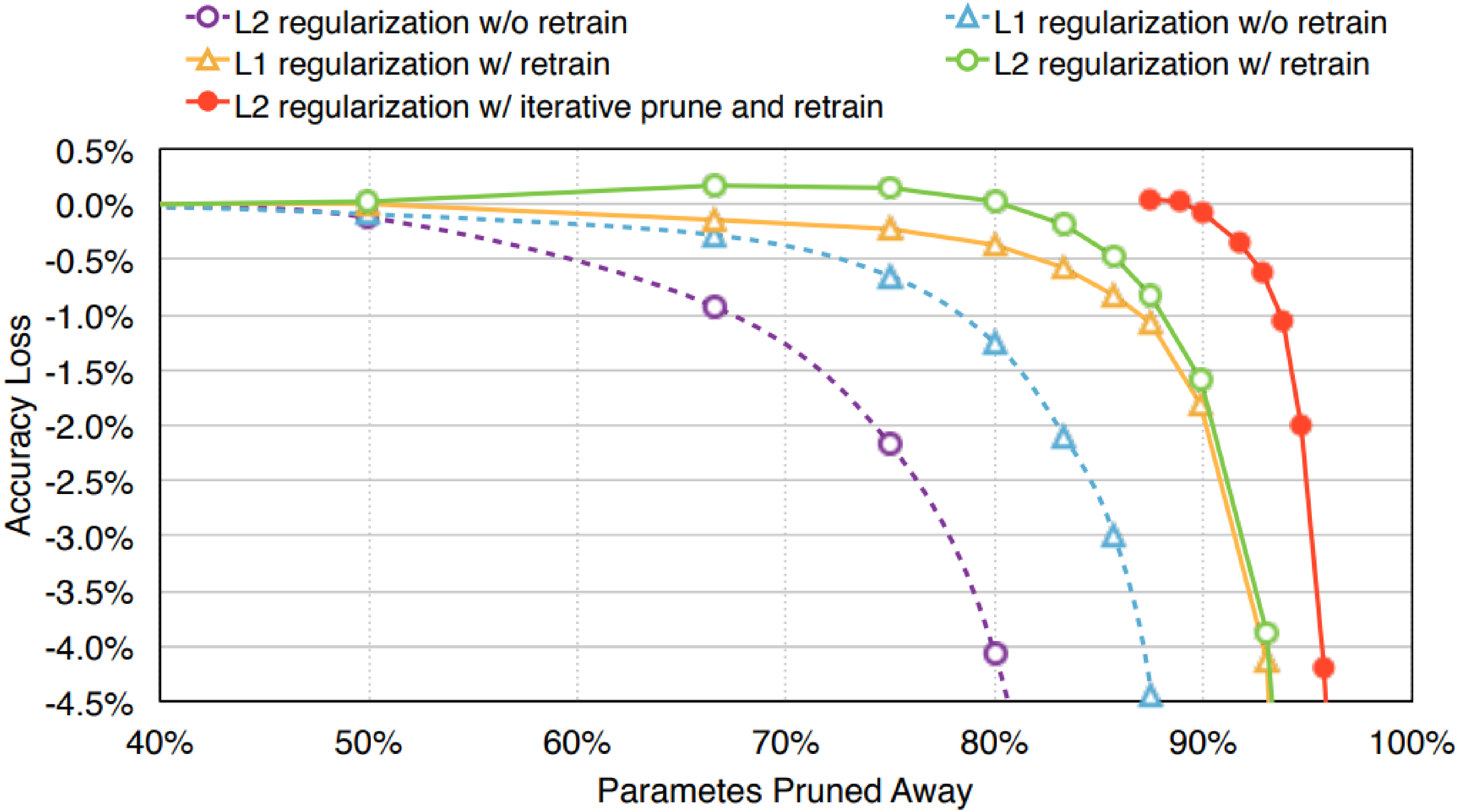
TensorFlow ™

Format conversion

Pruning, quantization, compression

Pruning – setting near-zero weights to zero

Trade-Off Curve



Song Han, Jeff Pool, John Tran, William J. Dally (2015). *Learning both Weights and Connections for Efficient Neural Networks*. https://arxiv.org/abs/1506.02626

ML / DL Tool Flow for Edge Computing

Training

Caffe

TensorFlow™

→ Format conversion → Pruning, quantization, compression

Pruning – setting near-zero weights to zero

Re-training – with sparse weight matrix (iterative)

Quantization – replacing 32-bit floats with 8-bit integers (typical)

Compression – storing sparse weight matrix in compressed format

10 X reduction in weight memory

ML / DL Tool Flow for Edge Computing

Training

Caffe

TensorFlow ™

→ Format conversion → Pruning, quantization, compression → Engine-specific optimizations

Using DSP or VLIW or GPU processor instructions

Operation fusing – using specialist neural network instructions

E.g. Arm CMSIS-NN uses DSP instructions of Cortex-M4

E.g. Arm Compute Library uses NEON acceleration for Cortex-A

Benchmark CNNs

ImageNet Challenge Winners	Mobile
AlexNet	SqueezeNet
VGG	MobileNet
Inception	DenseNet
Resnet	SSD
	YOLO

Pre-trained networks and transfer learning

MobileNet V1 and V2

CNN for image recognition and object detection on mobile

Hyperparameters:

- Scale down the size of the feature maps
- Scale down the number of features

MobileNet v2 available as 22 pre-built, pre-trained models:

- From 6M to 1.6M parameters, from 75% to 45% Top-1 accuracy

Deploying Mobilenet

Feature engineering

Collect and curate training, validation, test datasets

Select hyperparameters

Select a classifier (recognition) or detector (object detection)

Train classifier/detector, measure overfitting and generalisation error

For More Information

5-Day Training Course: **Practical Deep Learning**

www.doulos.com

john.aynsley@doulos.com