Practical Deep Learning in the Classroom

Bridge the gap between domain knowledge and AI

Jianghao Wang, Ph.D.
jianghaw@mathworks.com

Deep learning is a key technology driving the AI megatrend

ARTIFICIAL INTELLIGENCE

Any technique that enables machines to mimic human intelligence

MACHINE LEARNING

Statistical methods that enable machines to "learn" tasks from data without explicitly programming

DEEP LEARNING

Neural networks with many layers that learn representations and tasks "directly" from data

1950s

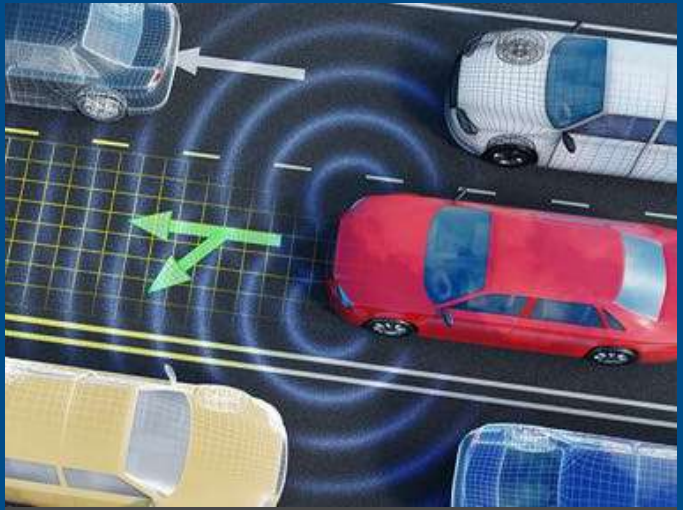
1980s

2010s

Deep learning is part of our everyday lives

Speech Recognition

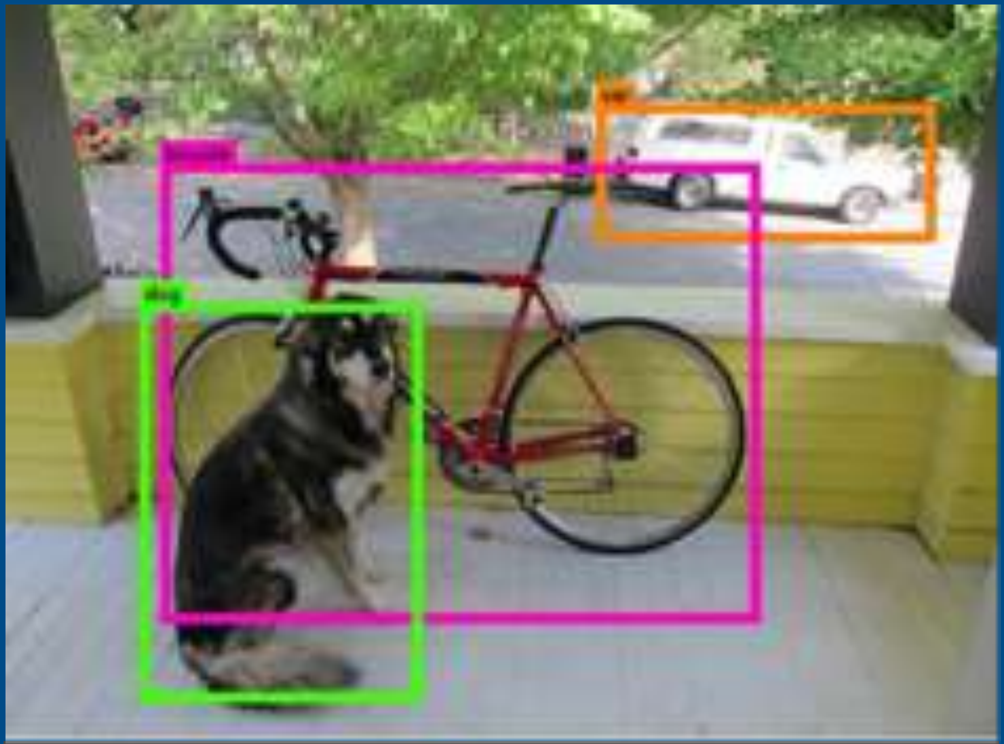
Face Detection



Automated Driving

Deep learning applications: mainstream vs. engineering

Mainstream

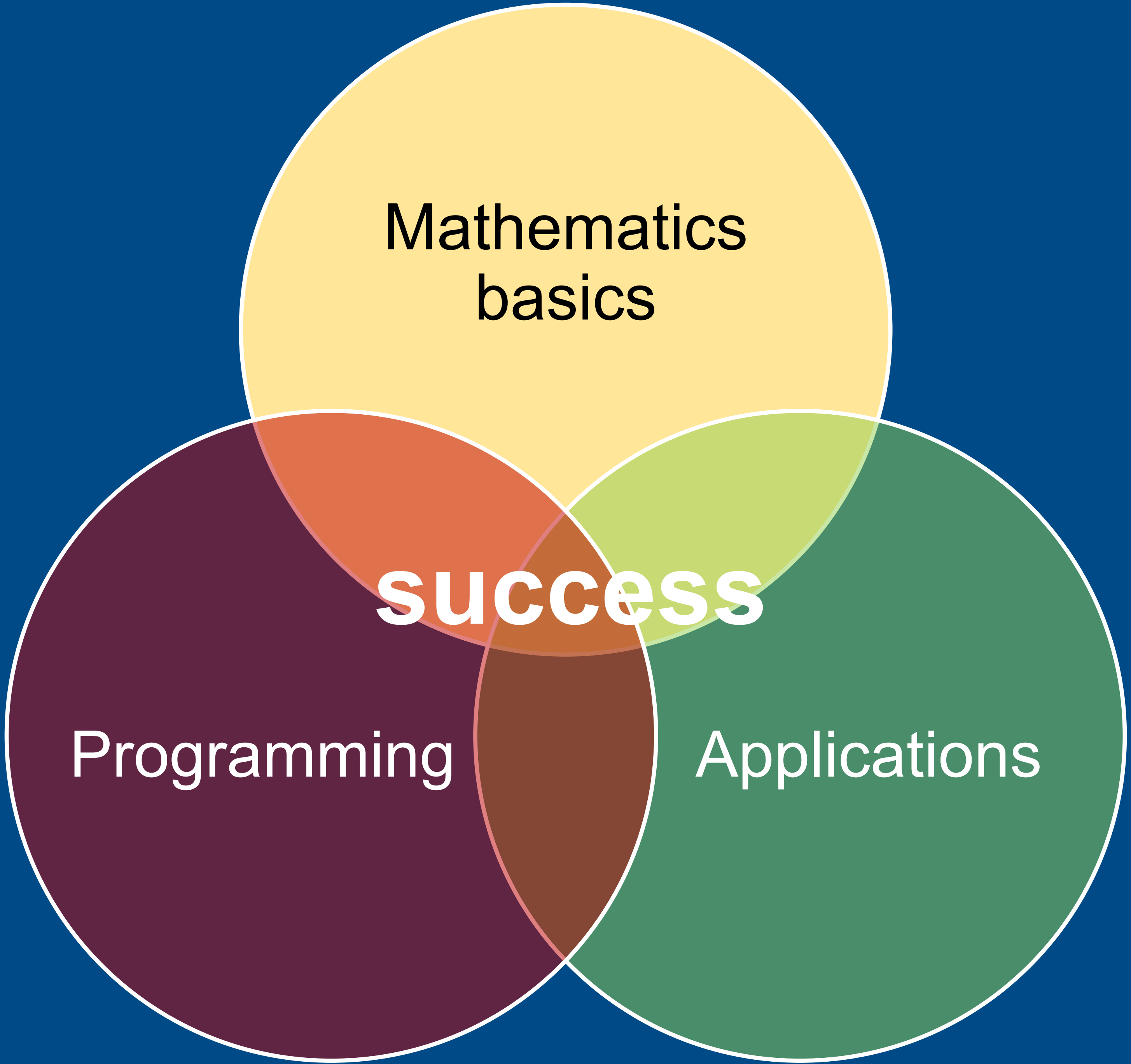


Detecting Objects

Engineering and Science

Identifying Machinery at Shell

Deep Learning Detection


Mathematics
basics
success
Programming
Applications

Mikusa Tunnel
Japan

Mikusa Tunnel
Japan

Traditional Approach

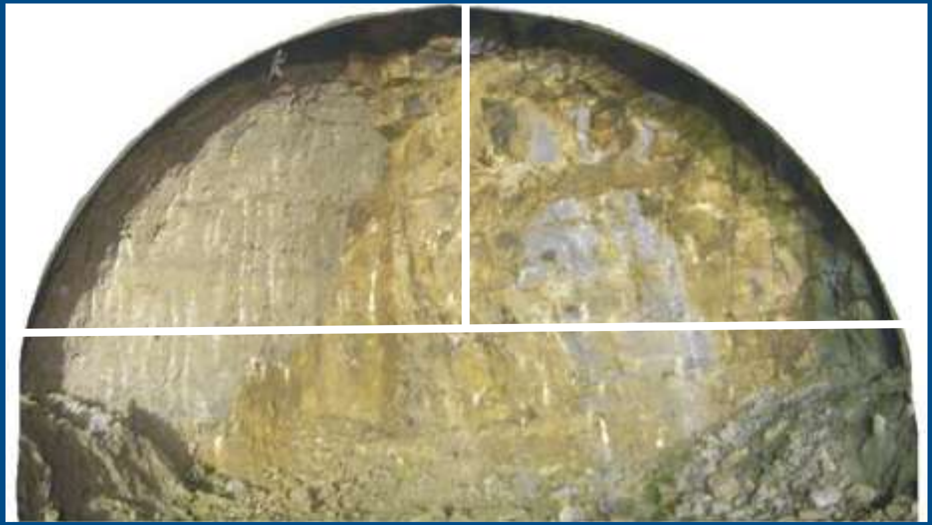
- Geologists assess seven different metrics
- Can take hours to analyze one site
- Critical shortage of geologists

New Approach

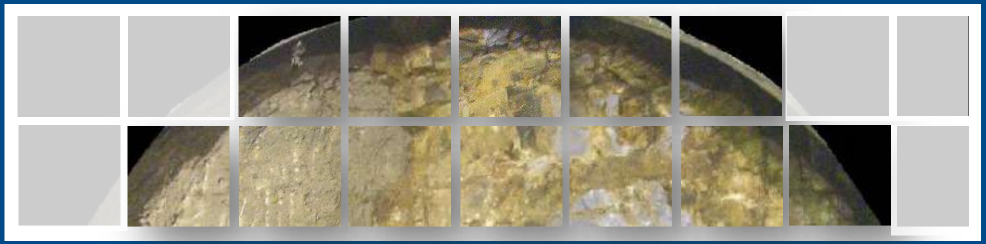
- Use deep learning to automatically recognize metrics based on images
- On-site evaluators decide with support from deep learning

Efficient tunnel drilling with deep learning

Obayashi Corporation



Split into sub-images

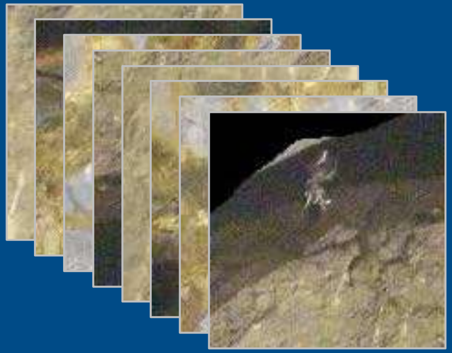


Label each sub-image

Done by geologists

Image	Weathering Alteration (1-4)	Fracture Spacing (1-5)	Fracture State (1-5)
	3	3	2
	4	1	1
	2	3	2
	3	3	2
⋮	⋮	⋮	⋮

Efficient tunnel drilling with deep learning
Obayashi Corporation



Transfer learning

AlexNet
PRETRAINED MODEL

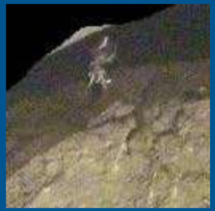

Ice cream


Teapot


Goose

Custom Network



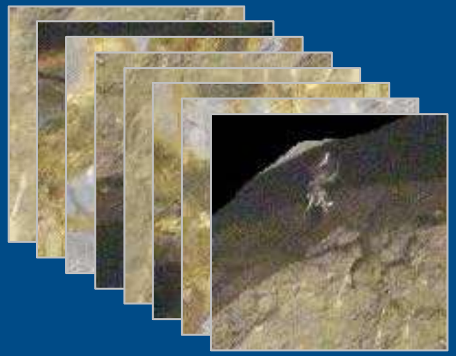


Weathering alteration: 4

Fracture spacing: 3

Fracture state: 2

Efficient tunnel drilling with deep learning
Obayashi Corporation



Transfer learning

MATLAB Production Server

AlexNet
PRETRAINED MODEL

Ice cream

Teapot

Goose

Custom Network

Weathering alteration: 4

Fracture spacing: 3

Fracture state: 2

Bring human insights into AI

- We are the domain experts
- Shortage of data scientists
- We need the right knowledge & tools

A typical deep learning course looks like…

- Introduction to deep learning
  - Historical context, reason of success, etc.
- Theoretical foundations
  - Mathematics basics
  - Neural networks
- Visualization and debugging of neural networks
- Pretrained models, advanced architectures
- Applications

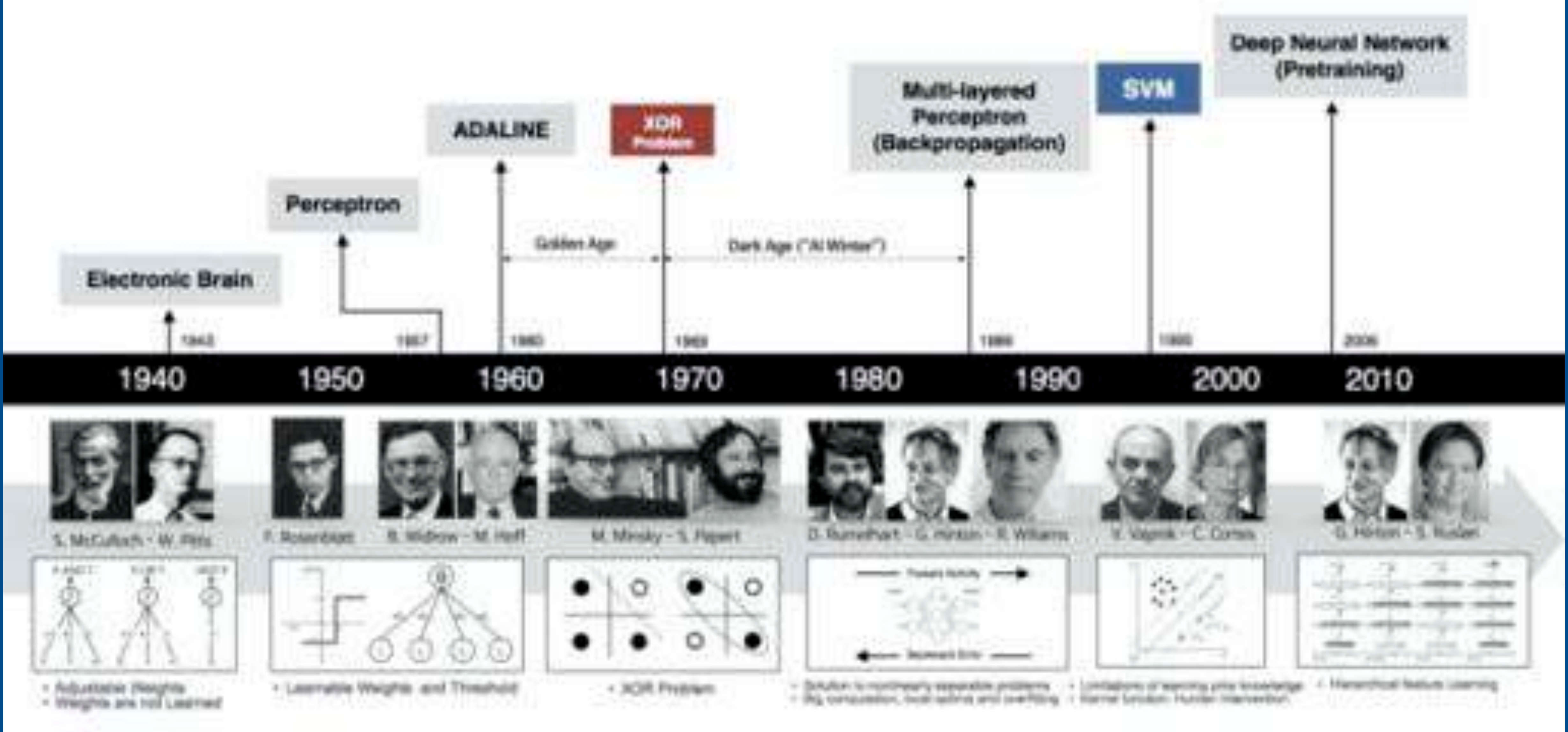
Reference:
- Stanford CS230, 231n
- UMD CMSC 828L
- Dartmouth CS078/178

A typical deep learning course looks like…

- **Introduction to deep learning**
  - **Historical context, reason of success, etc.**
- Theoretical foundations
  - Mathematics basics
  - Neural networks
- Visualization and debugging of neural networks
- Pretrained models, advanced architectures
- Applications

Reference:
- Stanford CS230, 231n
- UMD CMSC 828L
- Dartmouth CS078/178

Introduction to deep learning



Reference: https://beamandrew.github.io/deeplearning/2017/02/23/deep_learning_101_part1.html

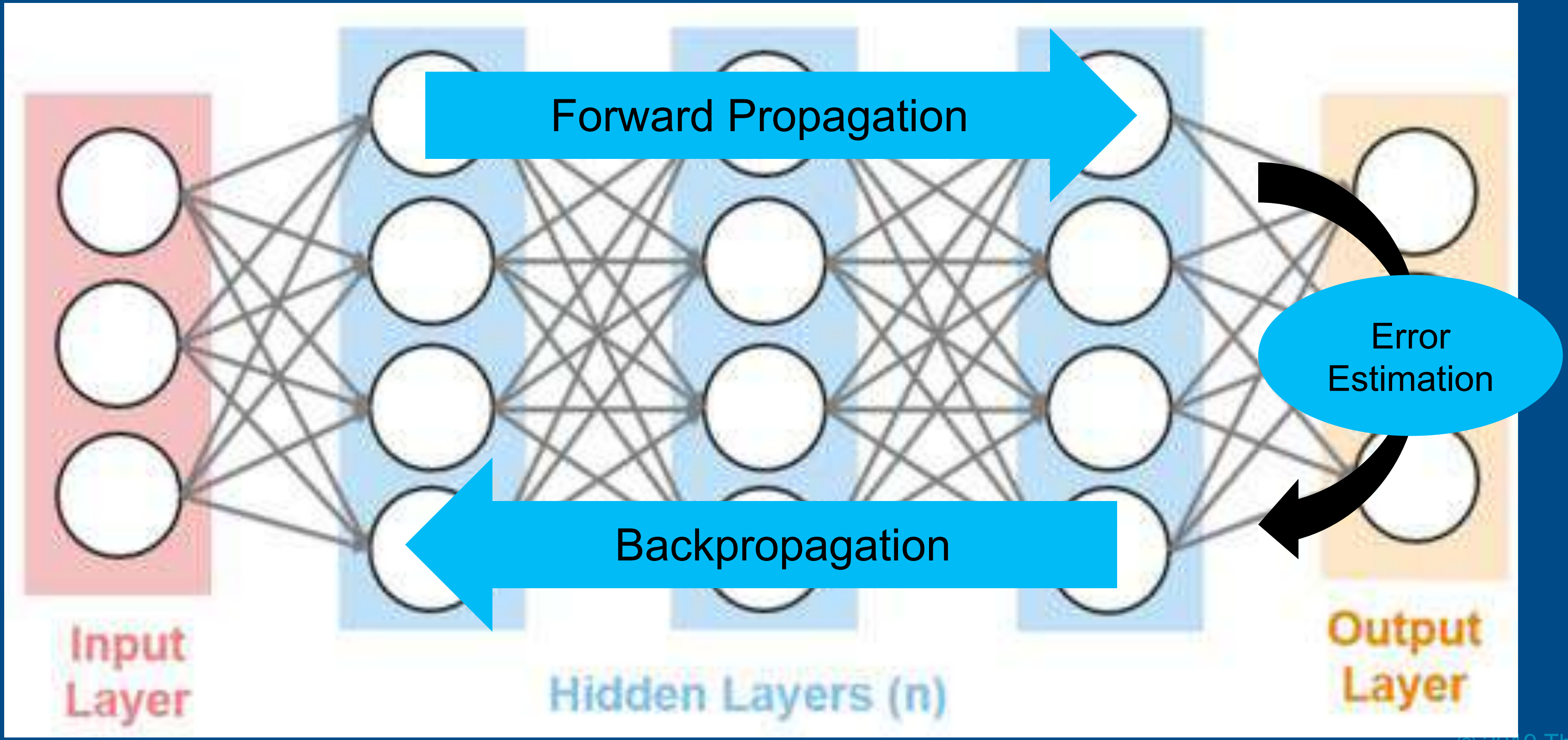
A typical deep learning course looks like…

- Introduction to deep learning
  - Historical context, reason of success, etc.
- **Theoretical foundations**
  - **Mathematics basics**
  - **Neural networks**
- Visualization and debugging of neural networks
- Pretrained models, advanced architectures
- Applications

Reference:
- Stanford CS230, 231n
- UMD CMSC 828L
- Dartmouth CS078/178

Theoretical foundations
Understand the architecture

Deep learning is usually implemented using a **neural network architecture** with many **hidden layers.**



Theoretical foundations
Stochastic gradient descent

- Standard gradient descent: $x_{k+1} = x_k - s_k \nabla L(x_k)$
  - Not successful in deep learning due to large data volume

- Stochastic gradient descent
  - Uses a "**minibatch**" (B) of the training data at each step
    - Computing ∇l_i by **backpropagation** on B samples is much faster
    - Producing **weights** x^* that generalize on unseen test data
  - Needs to determine a proper step size: **learning rate** and a good **minibatch** size

Theoretical foundations
Choose the right hyperparameters

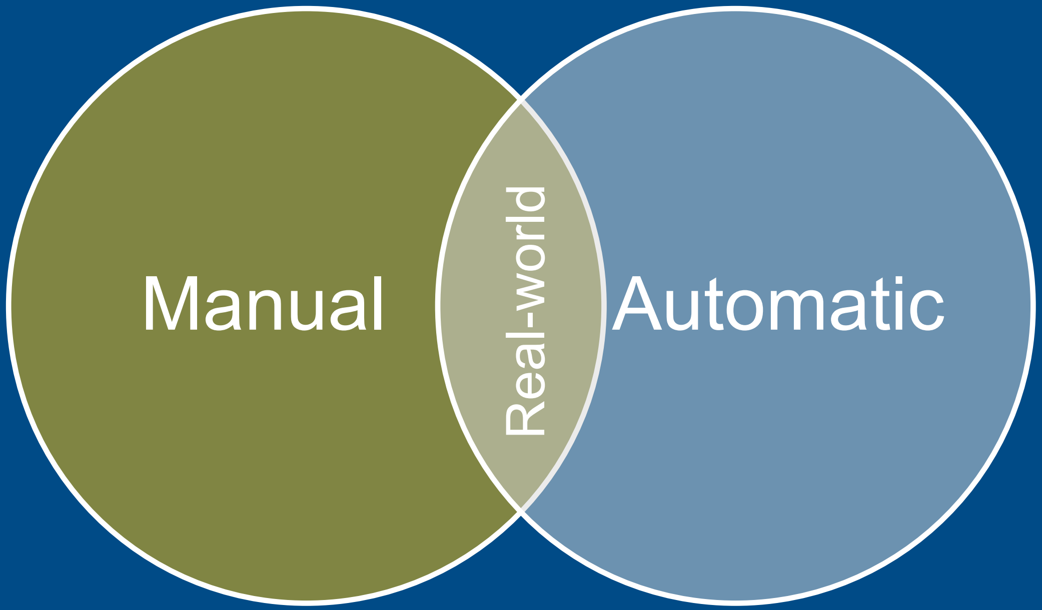
A key hyperparameters: **learning rate**

– Controls how much weights are adjusted with respect to loss
  - Too large: overshoot the minimum, may not converge or even diverge
  - Too low: slow to converge

– Need to be optimized, but can be expensive
  - **Manual**: trial & error (aka graduate student descent)
  - **Automatic**: grid search, random, Bayesian, etc
  - **Real-world**: a combination of both

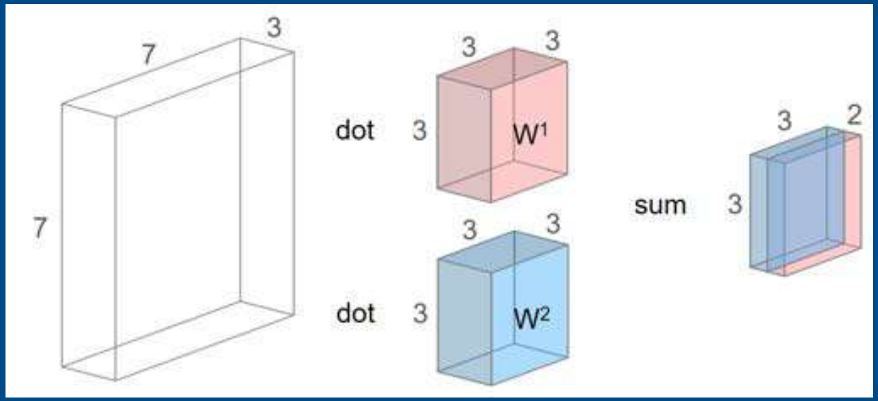
    (Educational) Use manual to understand theory

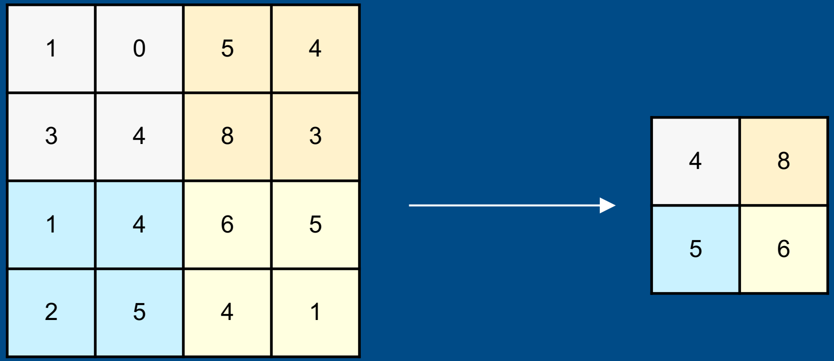
    (Practical) Then use automatic to speed things up



Theoretical foundations
Understand the architecture

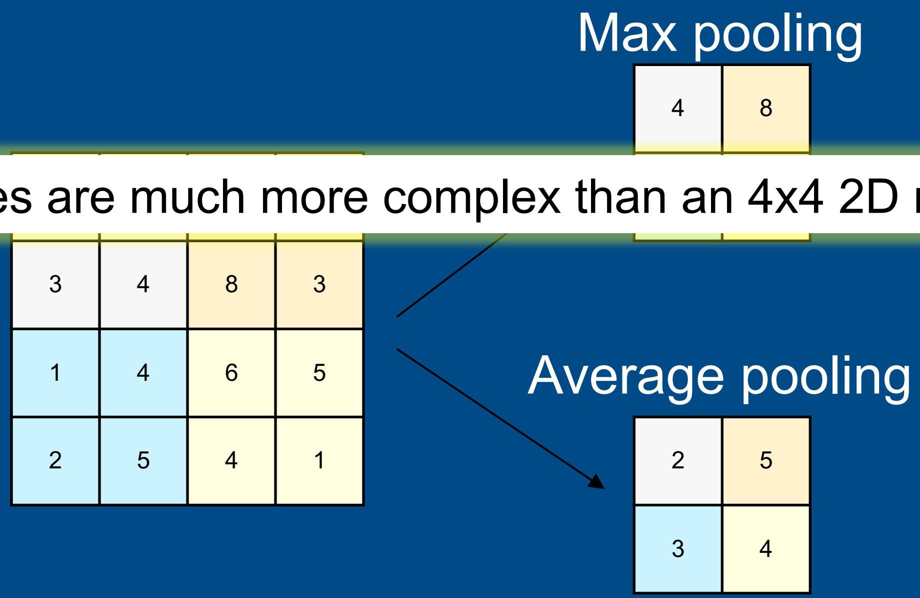
- Layer
  - Building blocks of a neural network
  - Many different types:
    - Convolution Layers
    - Sequence Layers
    - Activation layers
    - Pooling Layers
    - Fully Connected Layers
    - …
  - Different combinations are for different purposes



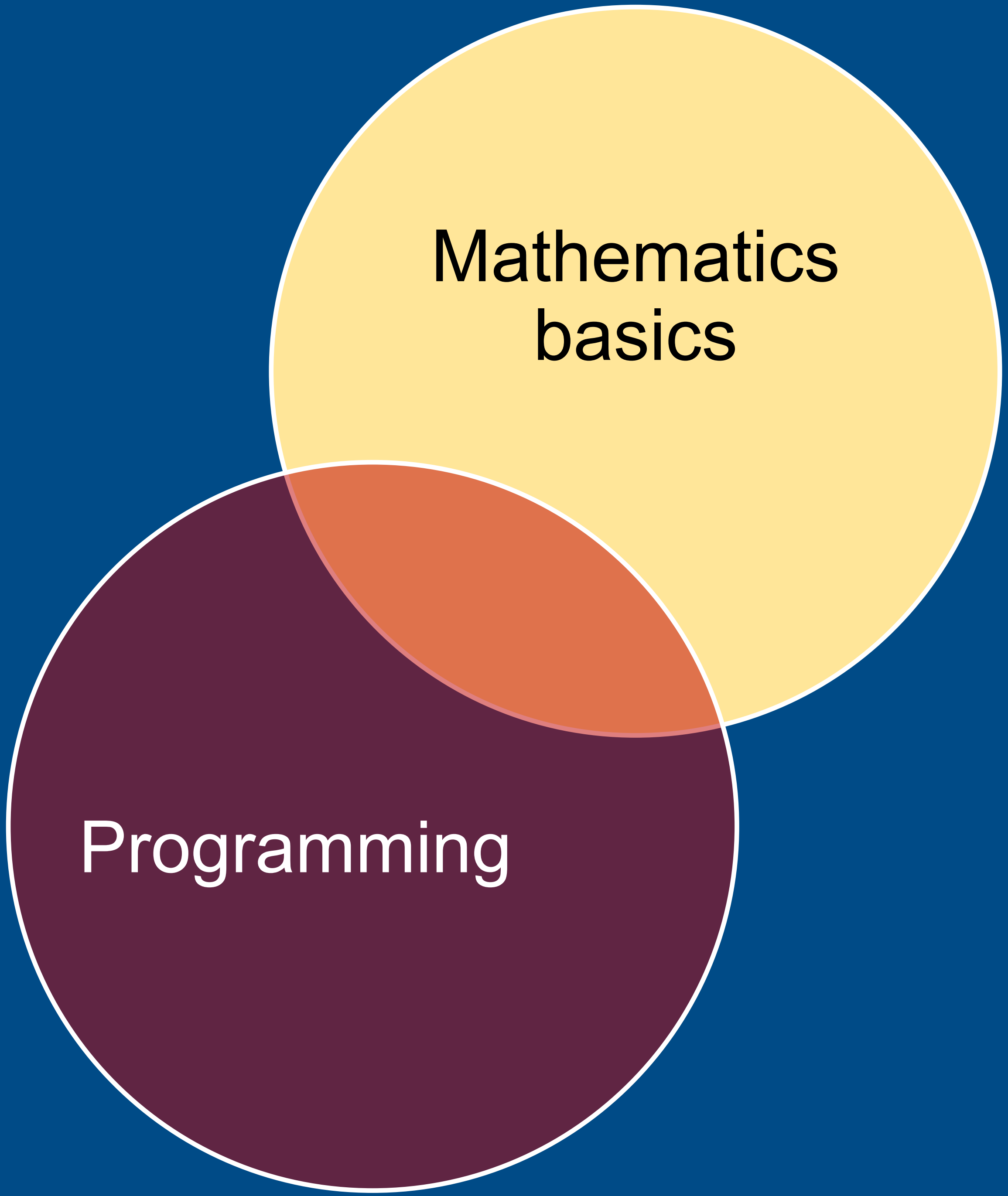


Pooling Layer

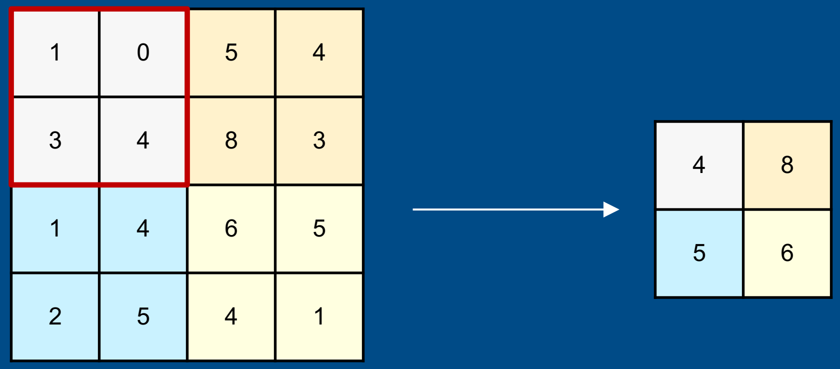
- Perform a **downsampling** operation across the spatial dimensions
- Goal: progressively decrease the size of the layers
- Max pooling and average pooling methods
- Popular choice: Max pooling with 2x2 filters, Stride = 2



Features are much more complex than an 4x4 2D matrix…


Mathematics basics
Programming

```
function B = maxpool(A, filtersize,stridelength)

% Get dimension of input matrix
[wA,hA] = size(A);

% Calculate output matrix dimension
wB = floor((wA - filtersize)/stridelength) + 1;
hB = floor((hA - filtersize)/ stridelength) + 1;

% Initialize output matrix
B = nan(wB,hB);

% pooling starts
iStart = 1;
for i = 1:wB
    iEnd= iStart + filtersize - 1;
    jStart = 1;
    for j = 1:hB
        jEnd = jStart + filtersize - 1;

        % regions of pooling
        subregionA = A(iStart:iEnd, jStart:jEnd);

        % this can be other functions too (e.g., mean)
        B(i,j) = max(subregionA(:));
        jStart = jStart + stridelength;
    end
    iStart = iStart + stridelength;
end
```



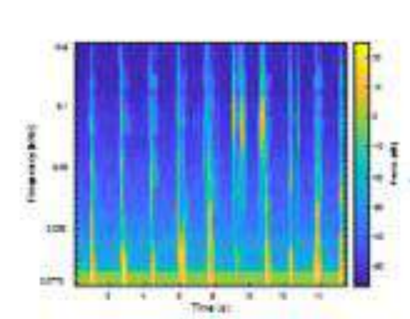
maxPooling2dLayer.m
maxPooling3dLayer.m

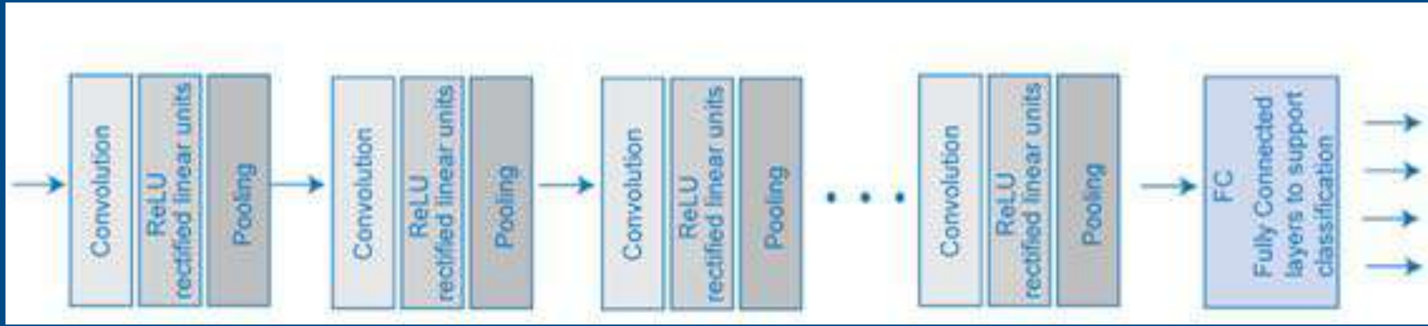
Theoretical foundations
Design the architecture

- Define layer graph
  - Choose different types of layers
  - Stack layers in the right order

Theoretical foundations
Common network architecture

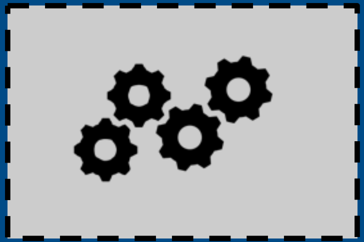
Convolutional Neural Networks (**CNN**)

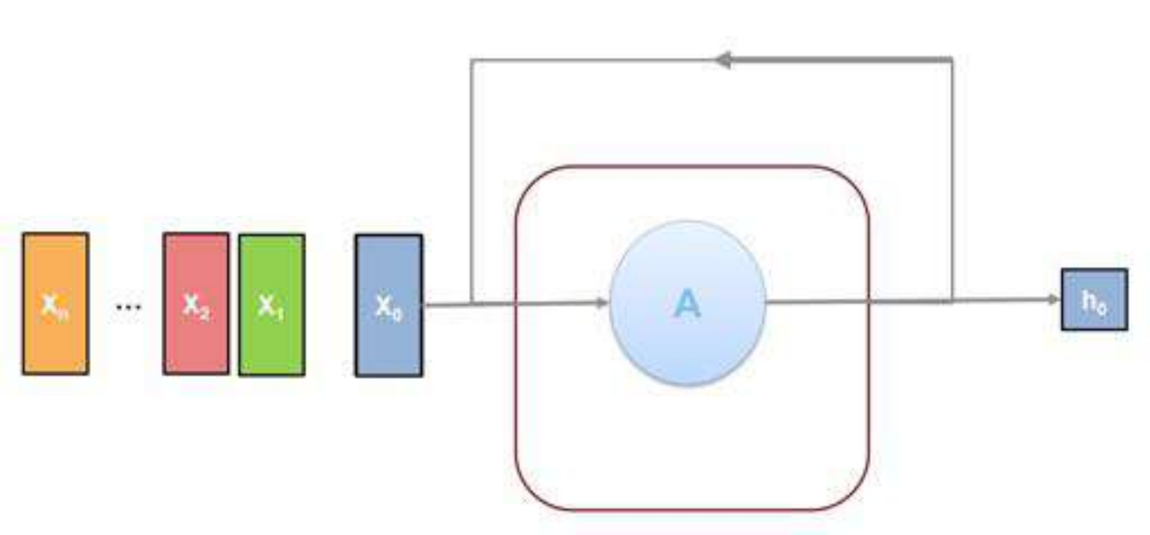




Time-Frequency Transformation

Long Short Term Memory (**LSTM**) Networks

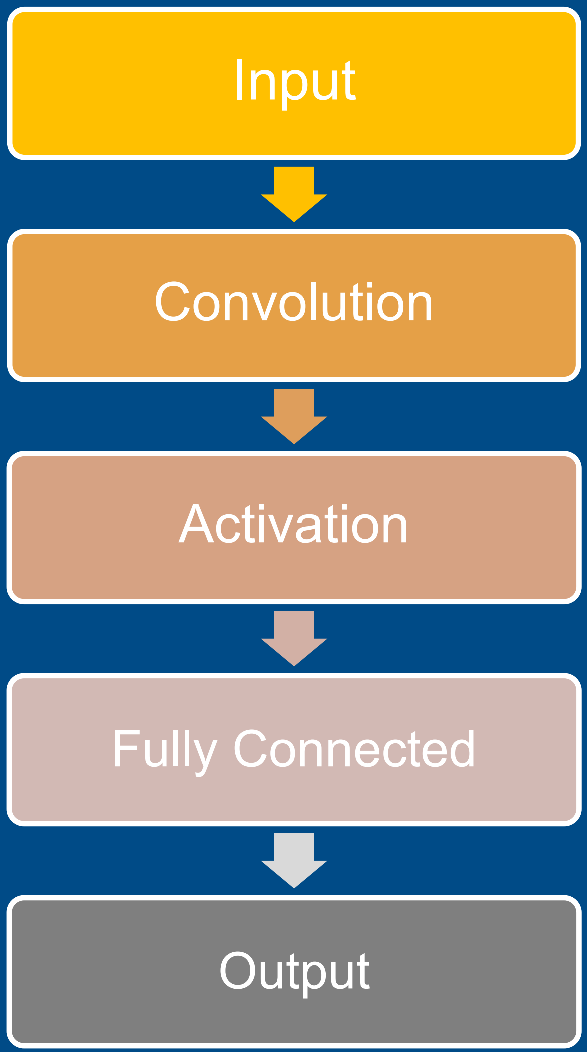


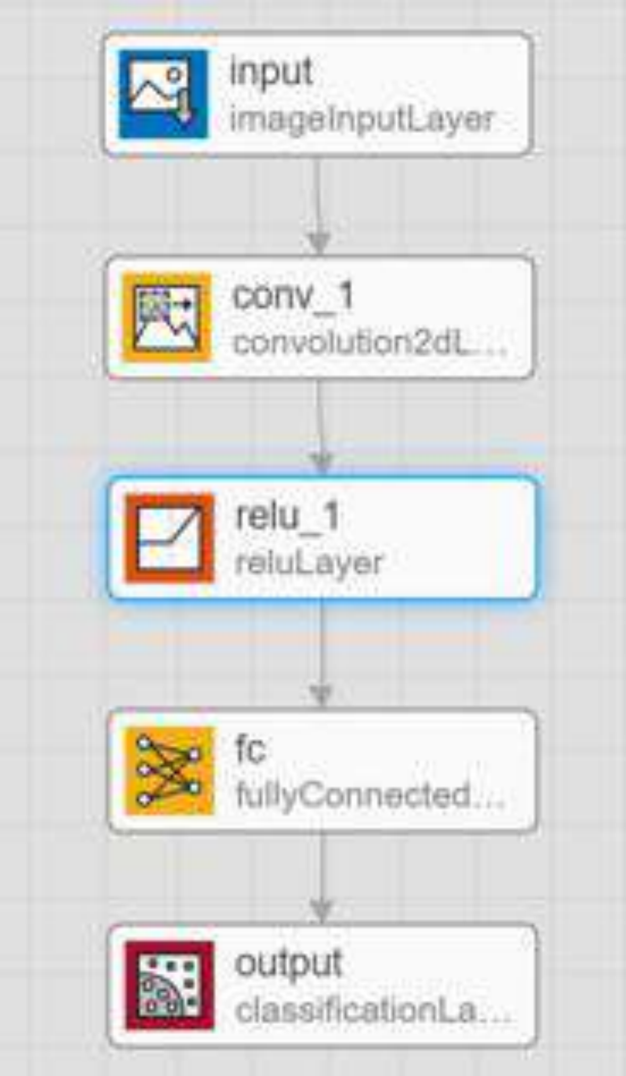


Data cleaning/feature engineering

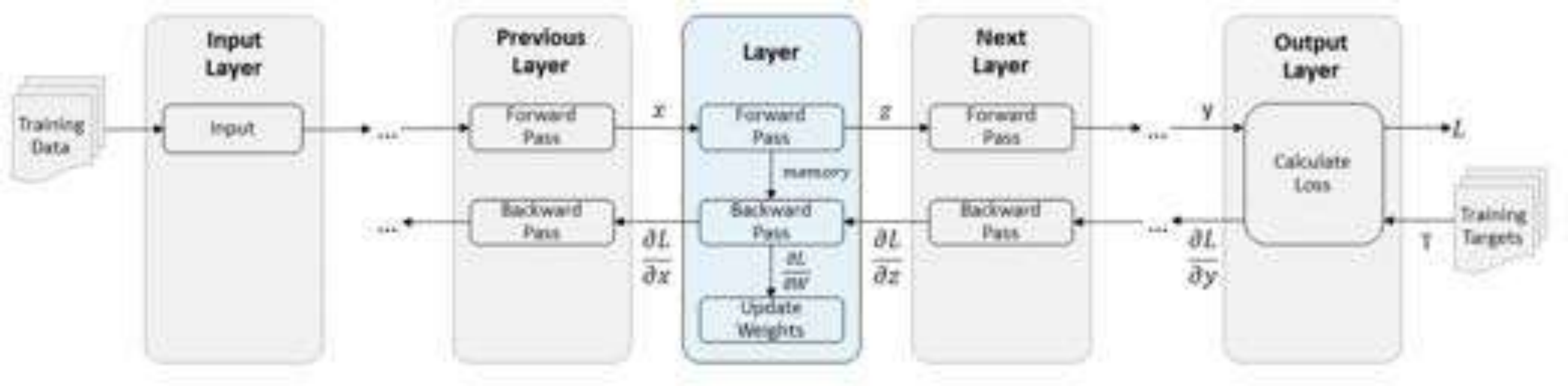
Theoretical foundations
Design the architecture

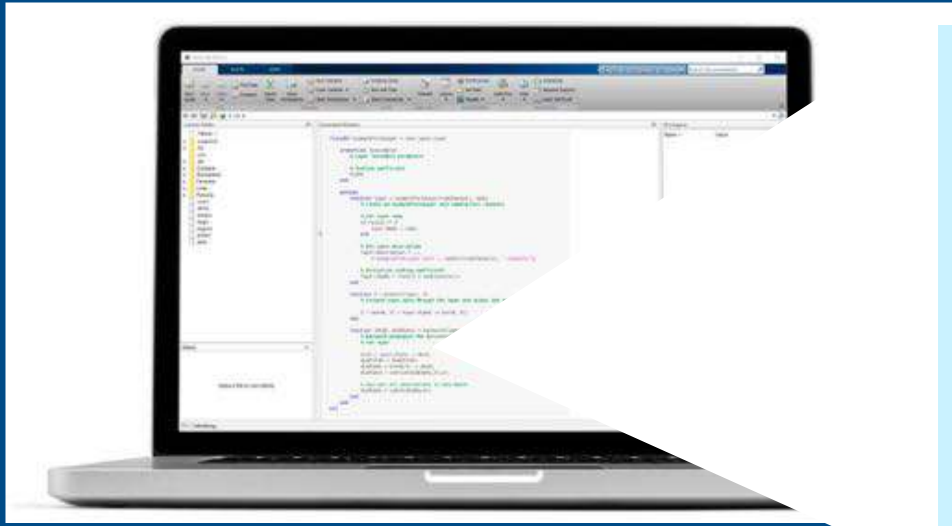
```
%% Define Network Architecture
% Define the convolutional neural network
architecture.
layers = [
    imageInputLayer([28 28 1])

    convolution2dLayer(3,16,'Padding',1)
    reluLayer

    fullyConnectedLayer(10)
    softmaxLayer
    classificationLayer];
```





Define custom layers





```
function [dLdX, dLdAlpha] = backward(layer, X, Z, dLdZ, memory)
% Backward propagate the derivative of the loss function through
% the layer

    dLdX = layer.Alpha .* dLdZ;
    dLdX(X>0) = dLdZ(X>0);
    dLdAlpha = min(0,X) .* dLdZ;
    dLdAlpha = sum(sum(dLdAlpha,1),2);

    % Sum over all observations in mini-batch
    dLdAlpha = sum(dLdAlpha,4);
end
```


Theoretical foundations
Design the architecture

- Define layer graph
  – Choose different types of layers
  – Stack layers in the right order

- Set training options
  – Choose solver, mini-batch size, learning rate, training environment…

Set training options

```
%% Specify Training Options
options = trainingOptions('sgdm',...
    'MaxEpochs',3, ...
    'ValidationData',valDigitData,...
    'ValidationFrequency',30,...
    'Verbose',false,...
    'Plots','training-progress');
```

```
options =

  TrainingOptionsSGDM with properties:

                     Momentum: 0.9000
             InitialLearnRate: 0.0100
    LearnRateScheduleSettings: [1×1 struct]
             L2Regularization: 1.0000e-04
      GradientThresholdMethod: 'l2norm'
            GradientThreshold: Inf
                    MaxEpochs: 4
                MiniBatchSize: 128
                      Verbose: 0
             VerboseFrequency: 50
               ValidationData: [1×1
matlab.io.datastore.ImageDatastore]
          ValidationFrequency: 30
           ValidationPatience: 5
                      Shuffle: 'once'
               CheckpointPath: ''
         ExecutionEnvironment: 'auto'
                   WorkerLoad: []
                    OutputFcn: []
                        Plots: 'training-progress'
               SequenceLength: 'longest'
         SequencePaddingValue: 0
```


Theoretical foundations
Design the architecture

- Define the layer graph
  - Choose different types of layers
  - Stack layers in the right order
- Set training options
  - Choose solver, mini-batch size, learning rate, training environment…
- Train a neural network

Train a neural network from scratch

```
%% Define Network Architecture
% Define the convolutional neural network
architecture.
layers = [
    imageInputLayer([28 28 1])

    convolution2dLayer(3,16,'Padding',1)
    reluLayer

    fullyConnectedLayer(10)
    softmaxLayer
    classificationLayer];
```

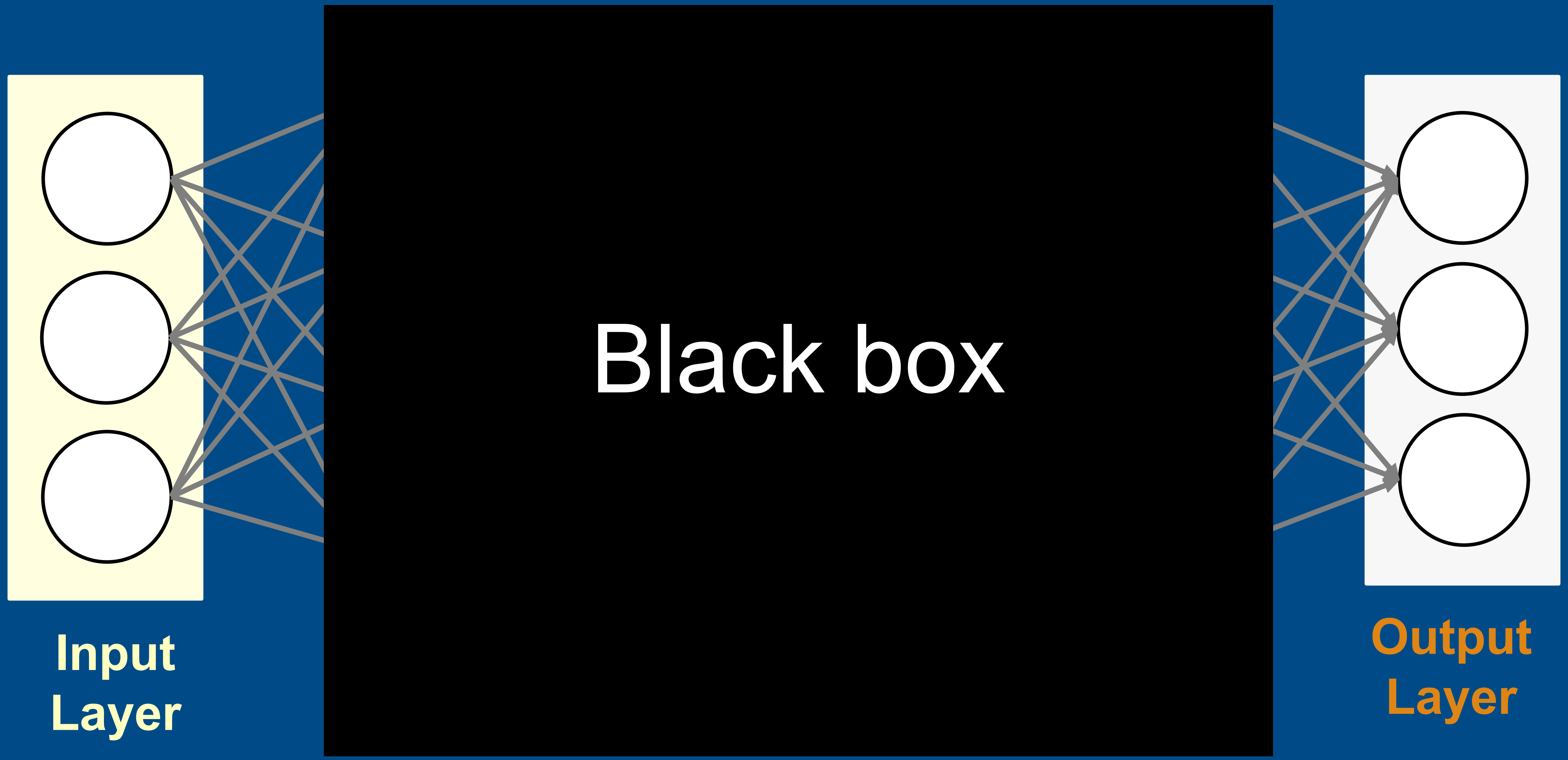
```
%% Specify Training Options
% After defining the network structure, specify
the training options.
options = trainingOptions('sgdm',...
    'MaxEpochs',3, ...
    'ValidationData',valDigitData,...
    'ValidationFrequency',30,...
    'Verbose',false,...
    'Plots','training-progress');
```

```
%% Train Network Using Training Data
% Train the network using the architecture defined
by |layers|, the training data, and the training
options.
net = trainNetwork(trainDigitData,layers,options);
```

A typical deep learning course looks like…

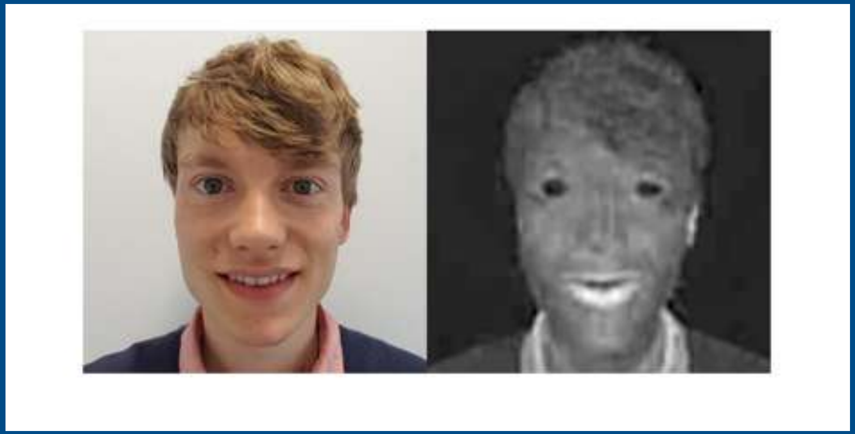
- Introduction to deep learning
  - Historical context, reason of success, etc.
- Theoretical foundations
  - Mathematics basics
  - Neural networks
- **Visualization and debugging of neural networks**
- Pretrained models, advanced architectures
- Applications

Reference:
- Stanford CS230, 231n
- UMD CMSC 828L
- Dartmouth CS078/178

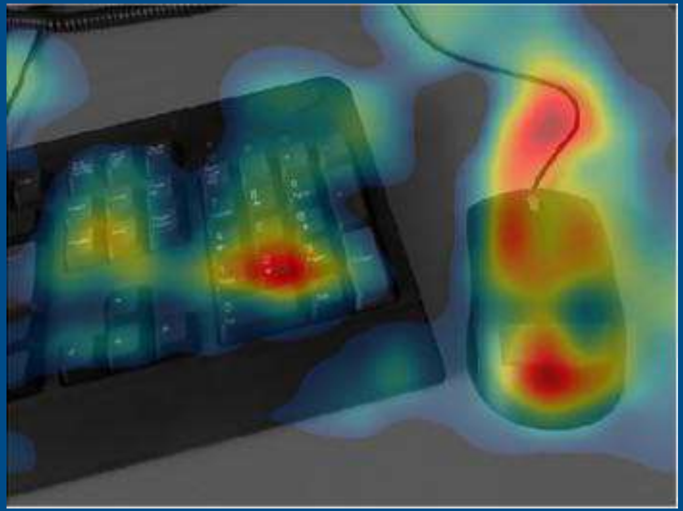

Black box
Input
Layer
Output
Layer

Theoretical foundations

Demystify the black box



Layer Activations

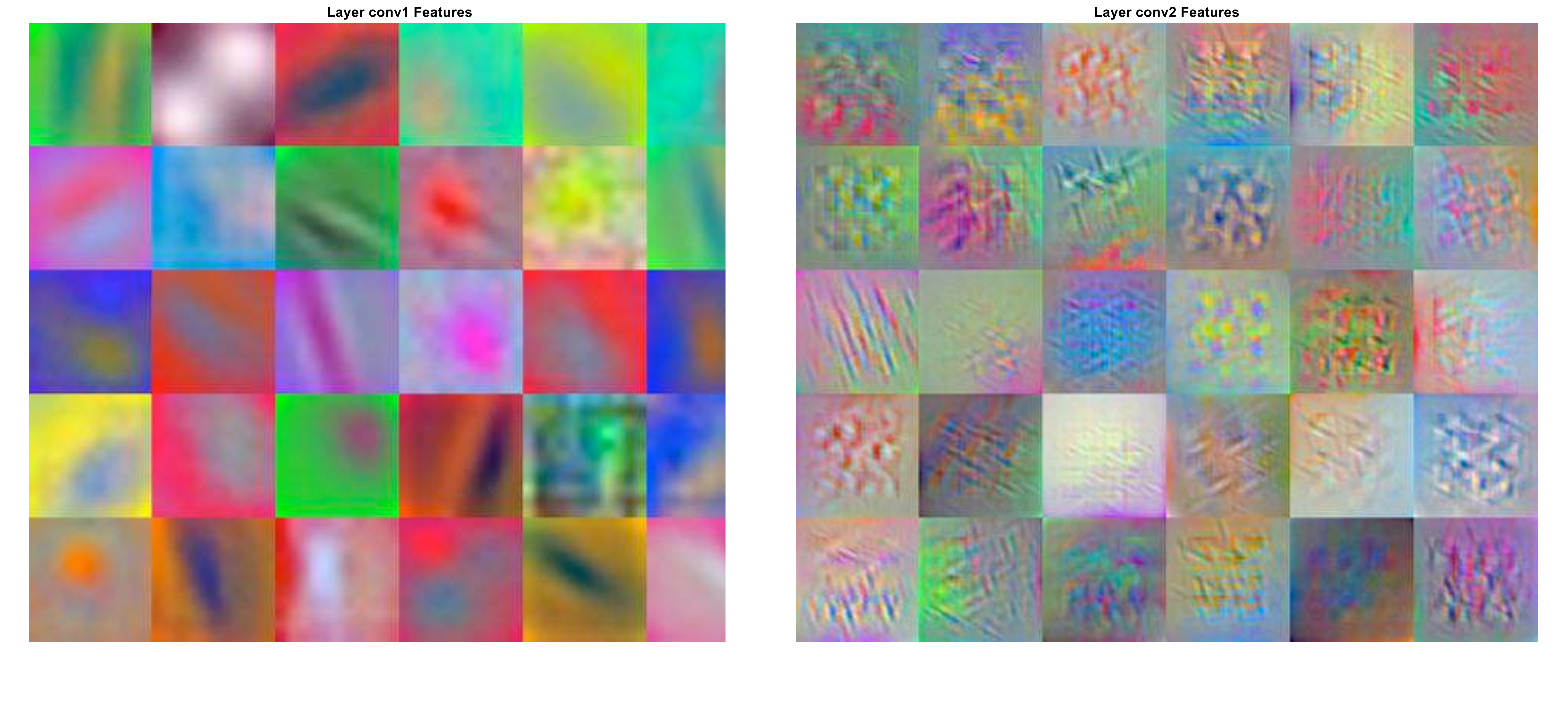


Class Activations

DeepDream Images

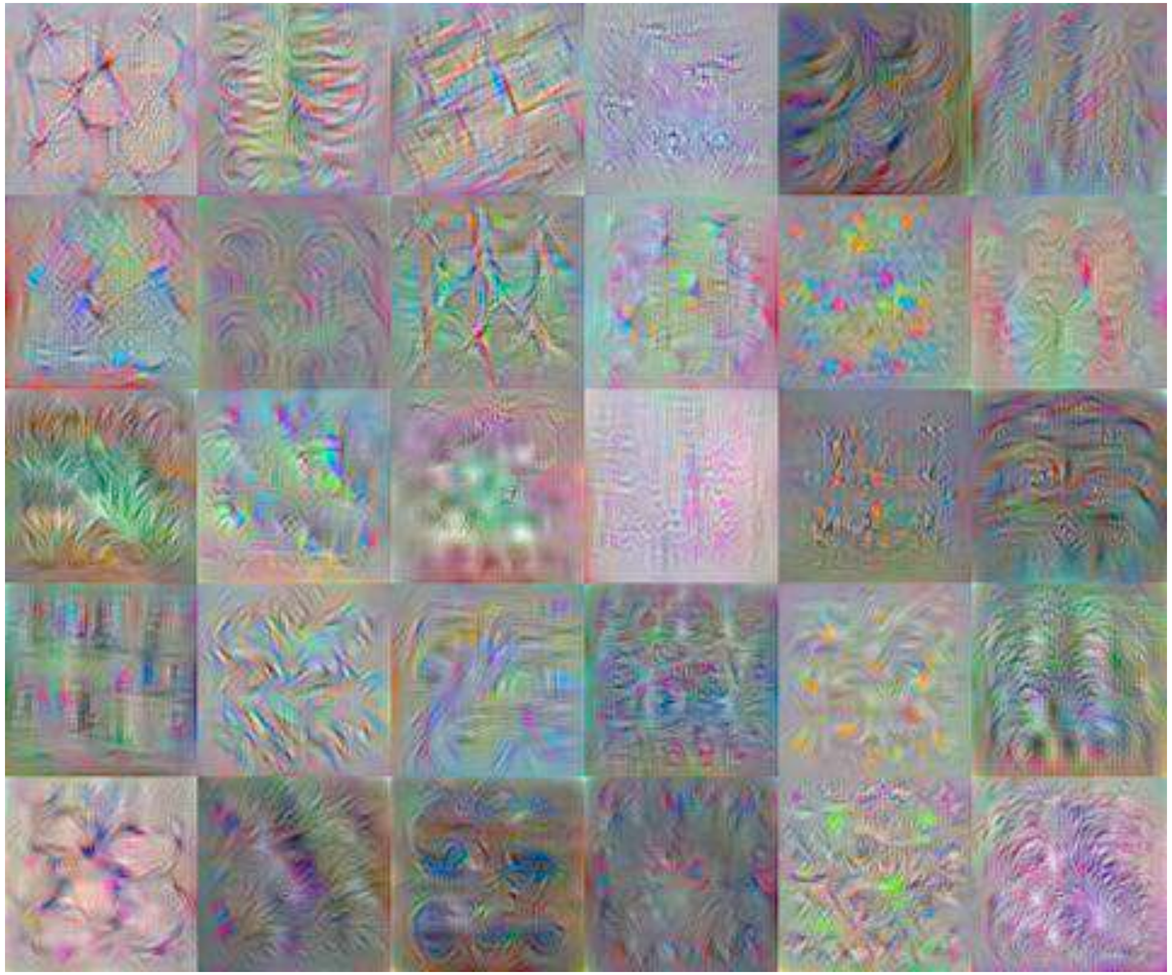
Visualization and debugging of neural networks

Visualization and debugging of neural networks

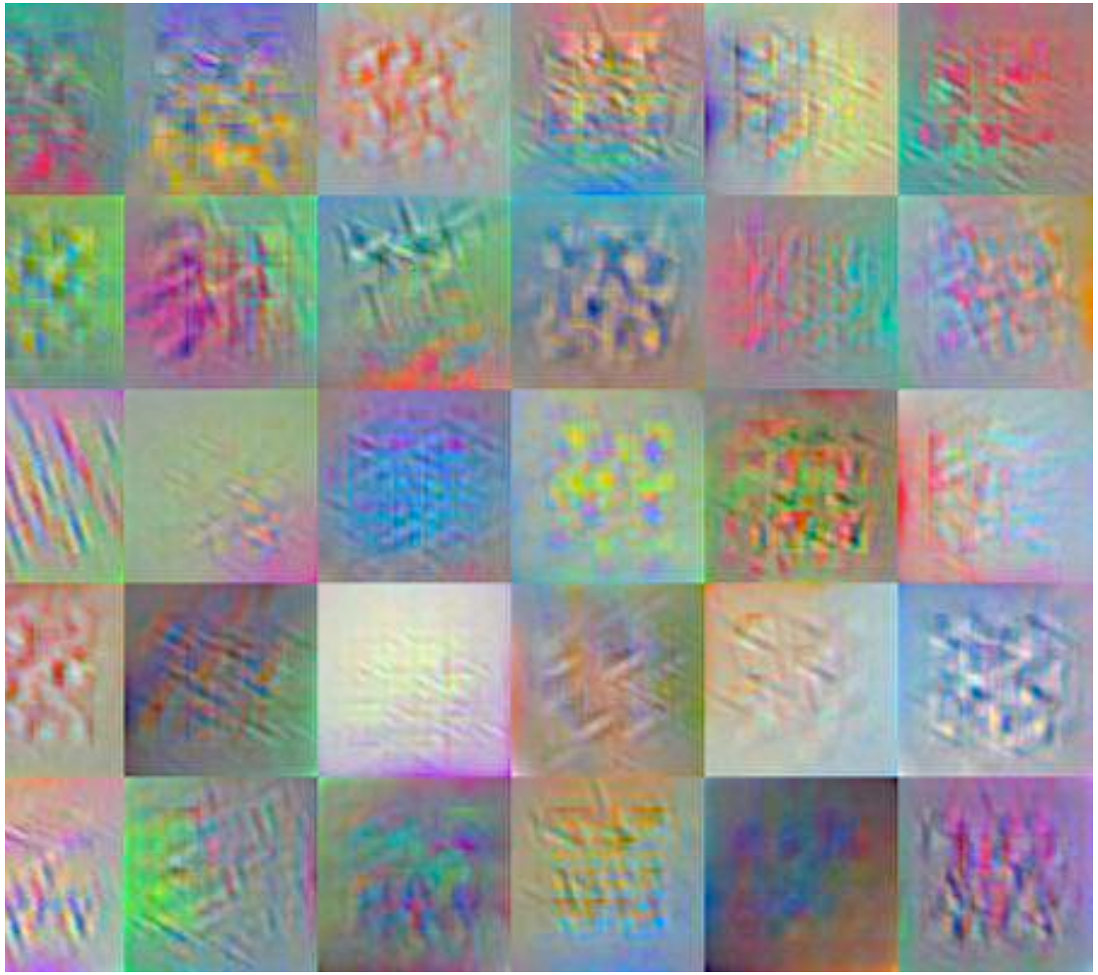


Visualization and debugging of neural networks

Layer conv3 Features

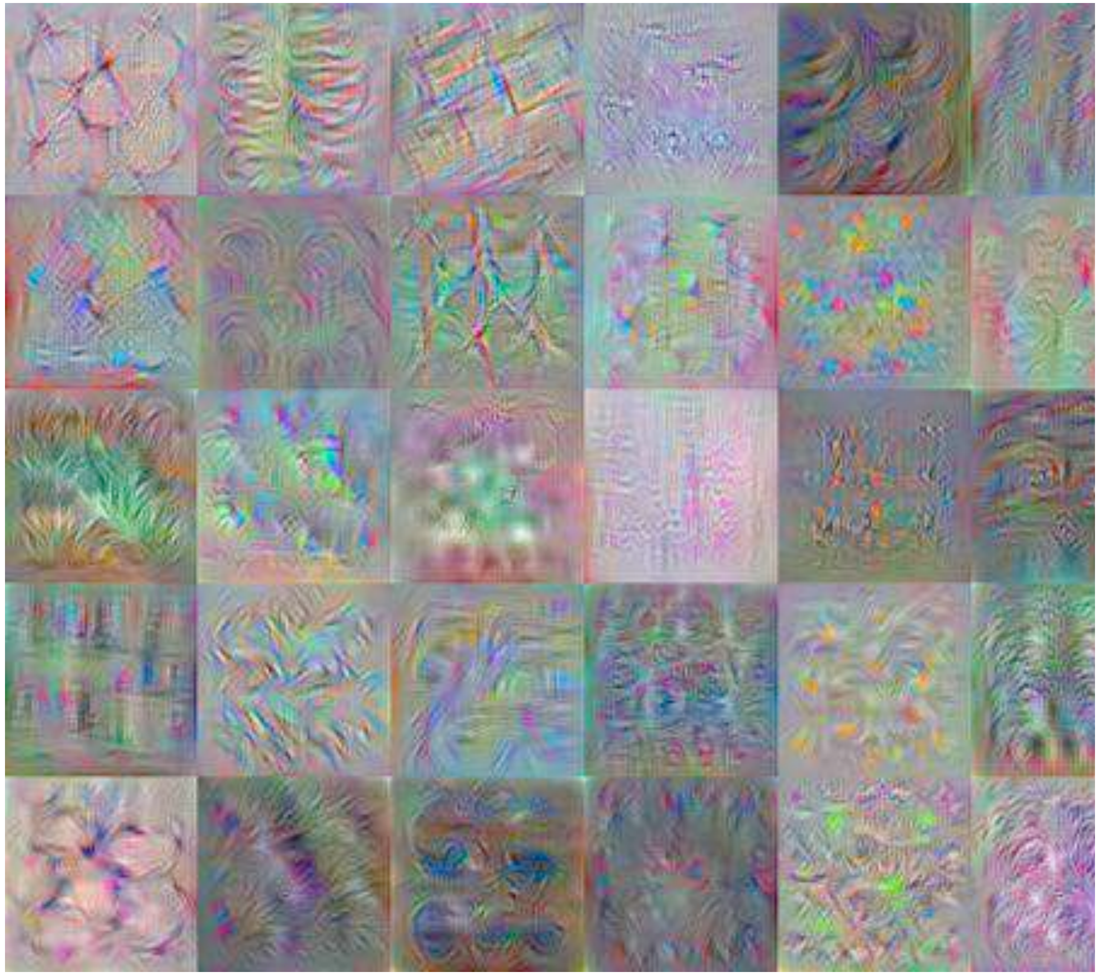


Layer conv2 Features

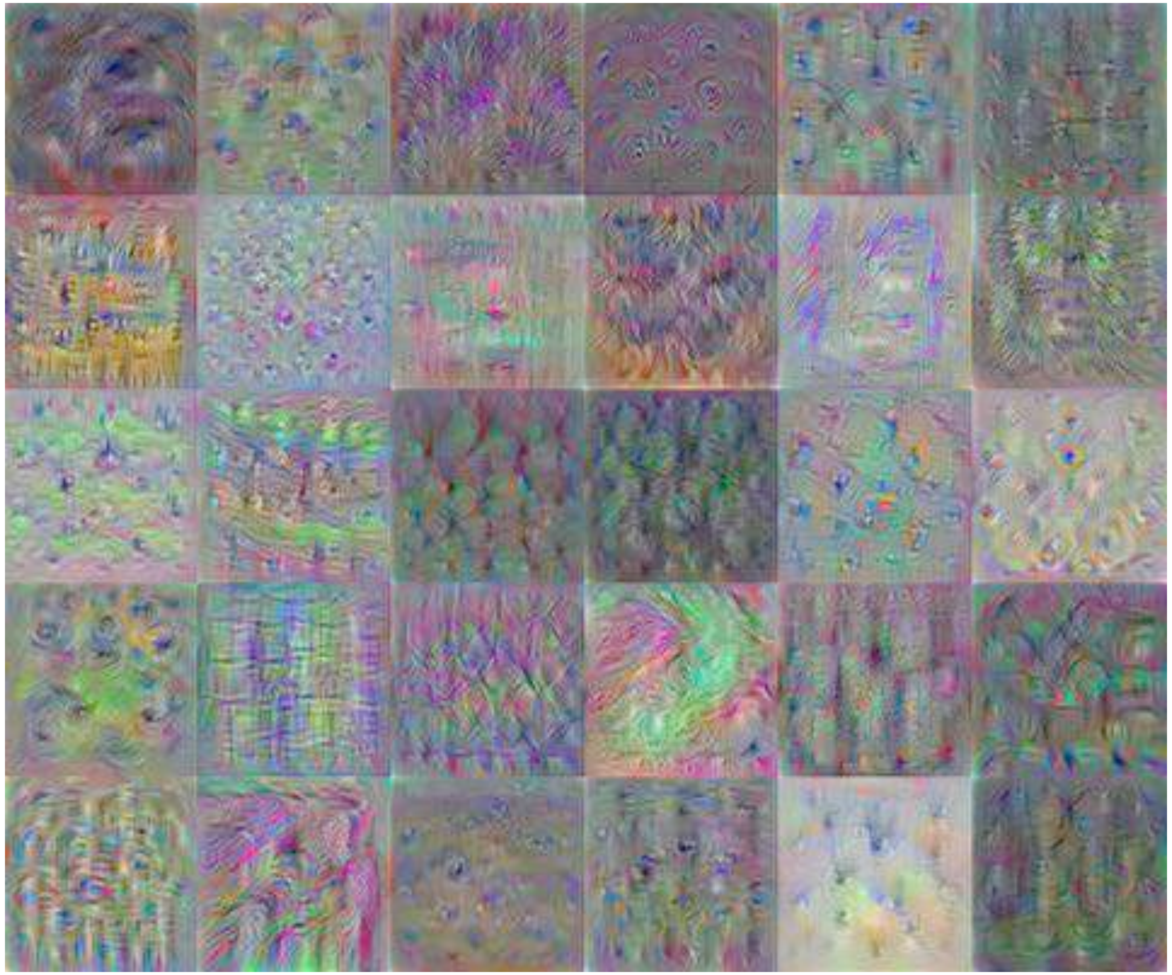


Visualization and debugging of neural networks

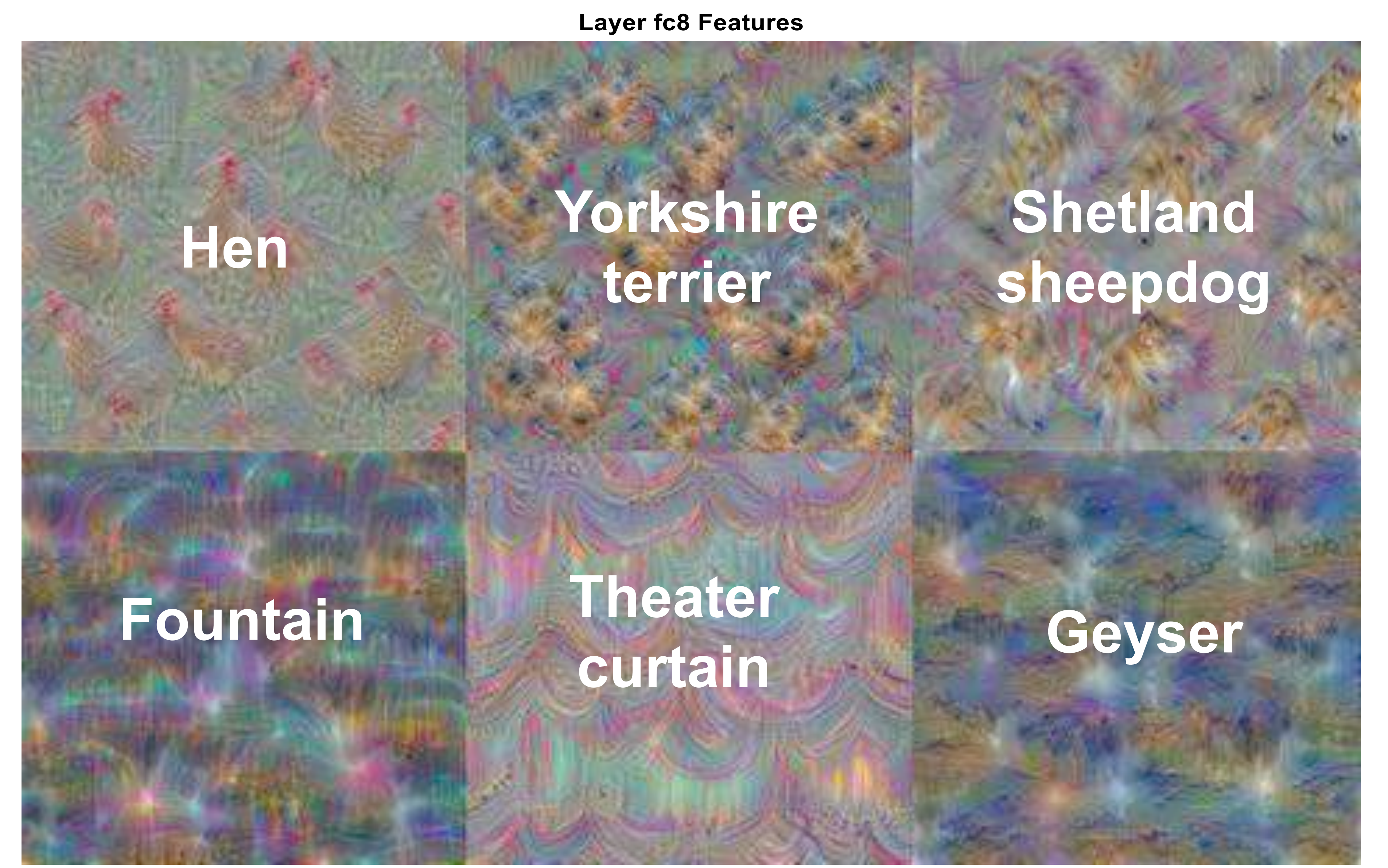
Layer conv3 Features



Layer conv4 Features



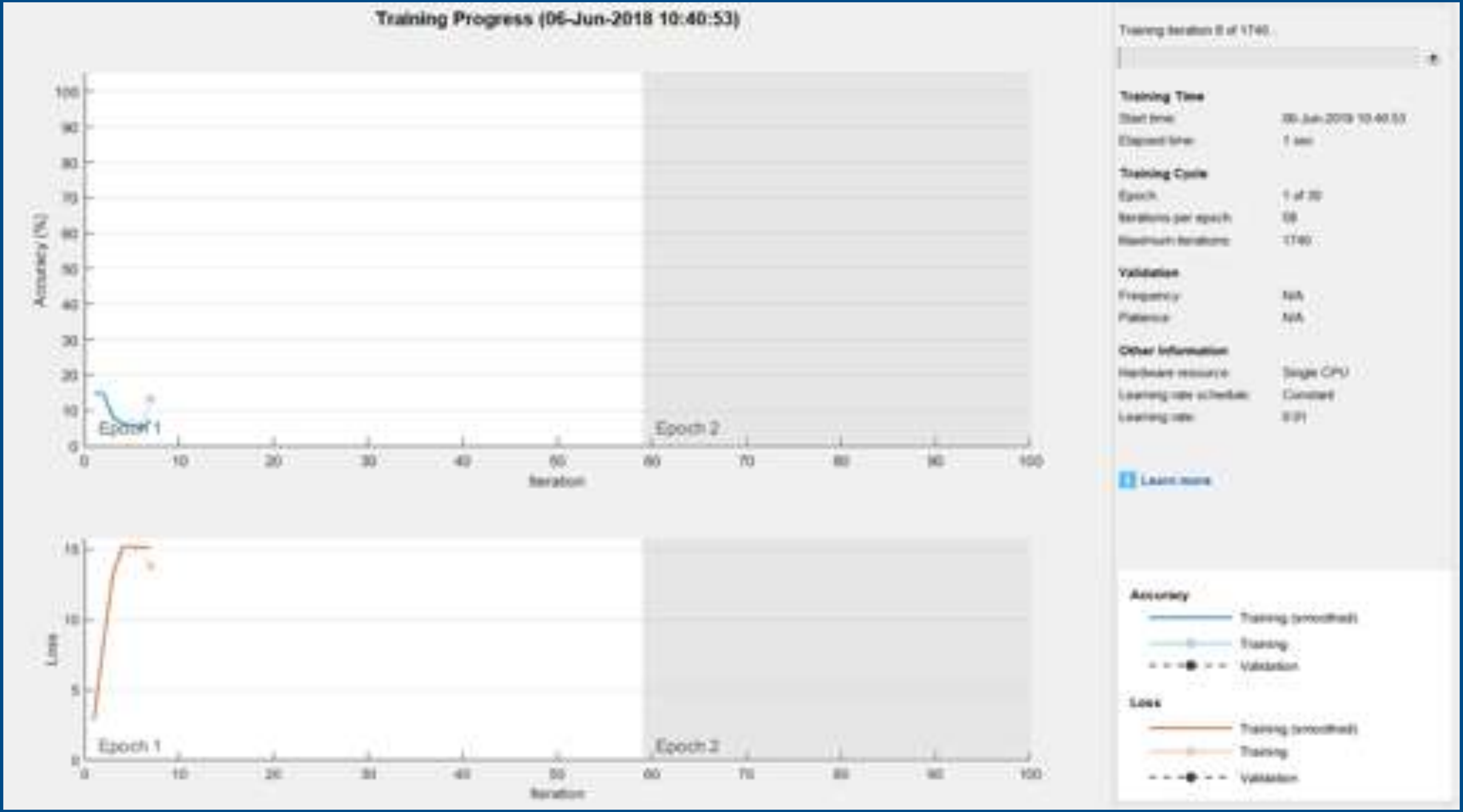
Visualization and debugging of neural networks



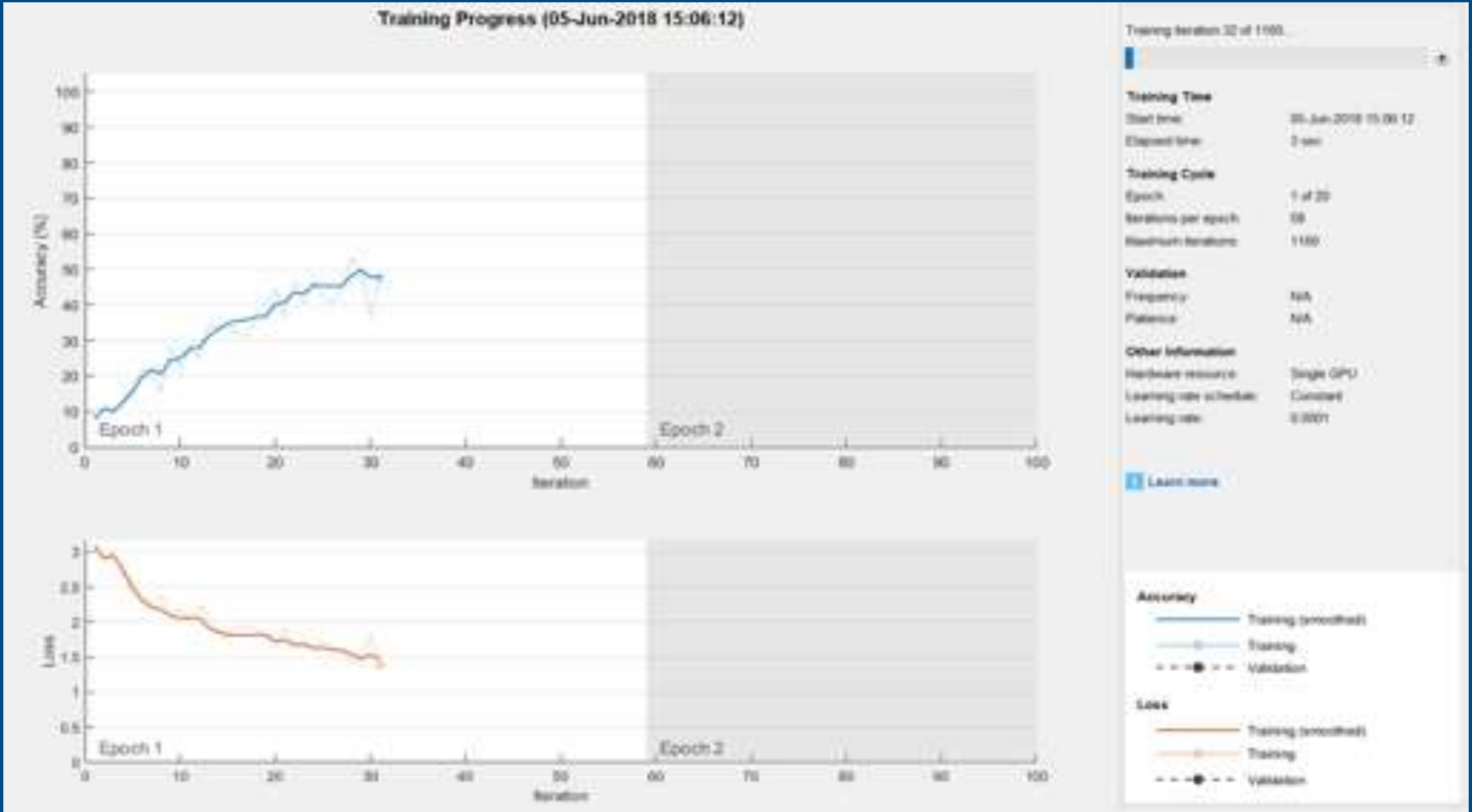
Visualization and debugging of neural networks

- Show features learned in the network
  - Visualize activations, `deepDreamImage`

- Monitor training progress
  - `options = trainingOptions('sgdm', 'Plots','training-progress');`



Training Progress (06-Jun-2018 10:40:53)
Accuracy (%)
Epoch 1
Epoch 2
Iteration
Loss
Training Time
Training Cycle
Epoch
Validation
Frequency
N/A
N/A
Other Information
Hardware resource
Single CPU
Learning rate schedule
Constant
Learning rate
Learn more
Accuracy
Training (smoothed)
Training
Validation
Loss
Training (smoothed)
Training
Validation



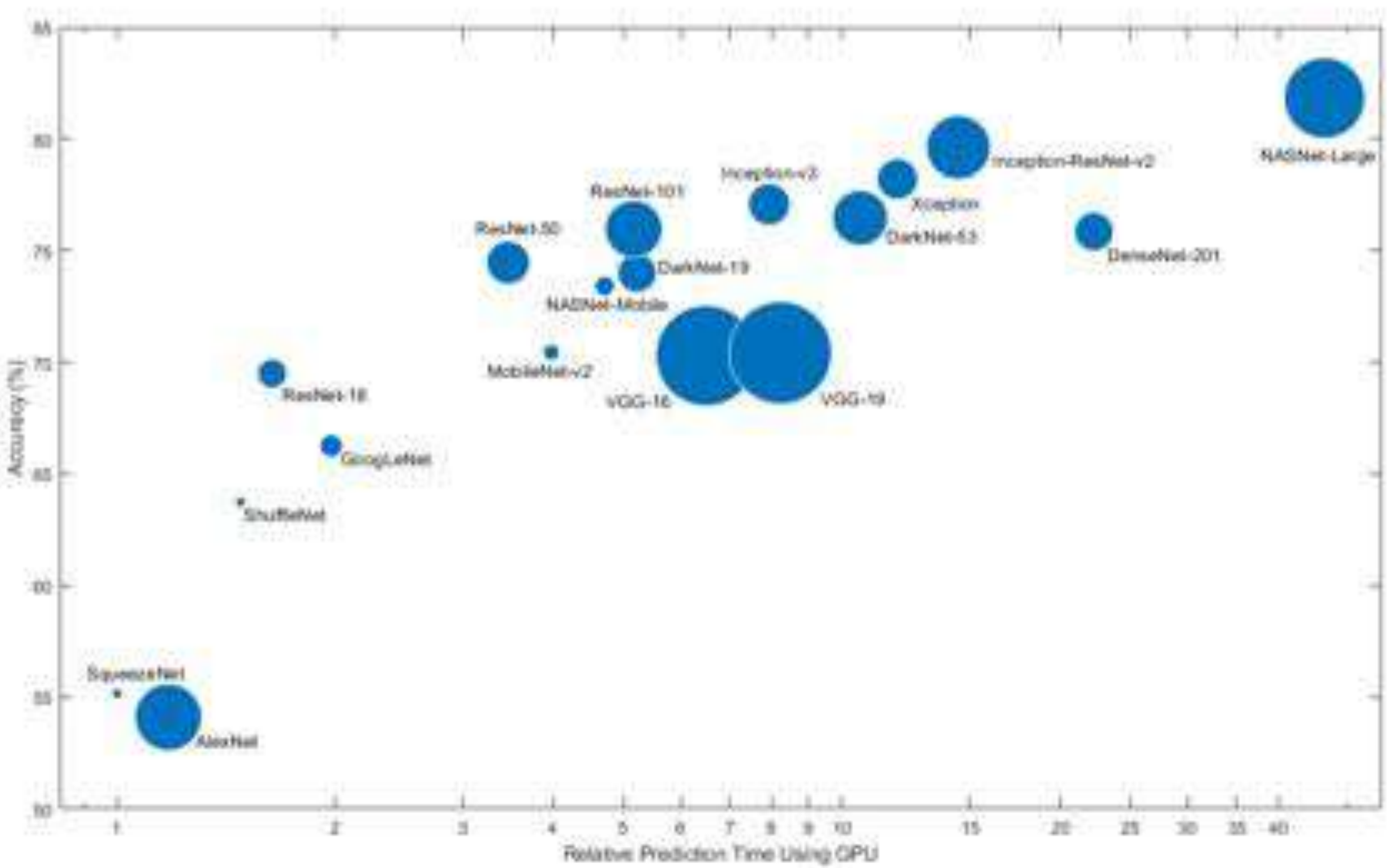
Training Progress (05-Jun-2018 15:06:12)
Epoch 1
Epoch 2
Epoch 1
Epoch 2

A typical deep learning course looks like…

- Introduction to deep learning
  - Historical context, reason of success, etc.
- Theoretical foundations
  - Mathematics basics
  - Neural networks
- Visualization and debugging of neural networks
- **Pretrained models, advanced architectures**
- Applications

Reference:
- Stanford CS230, 231n
- UMD CMSC 828L
- Dartmouth CS078/178

Pretrained models

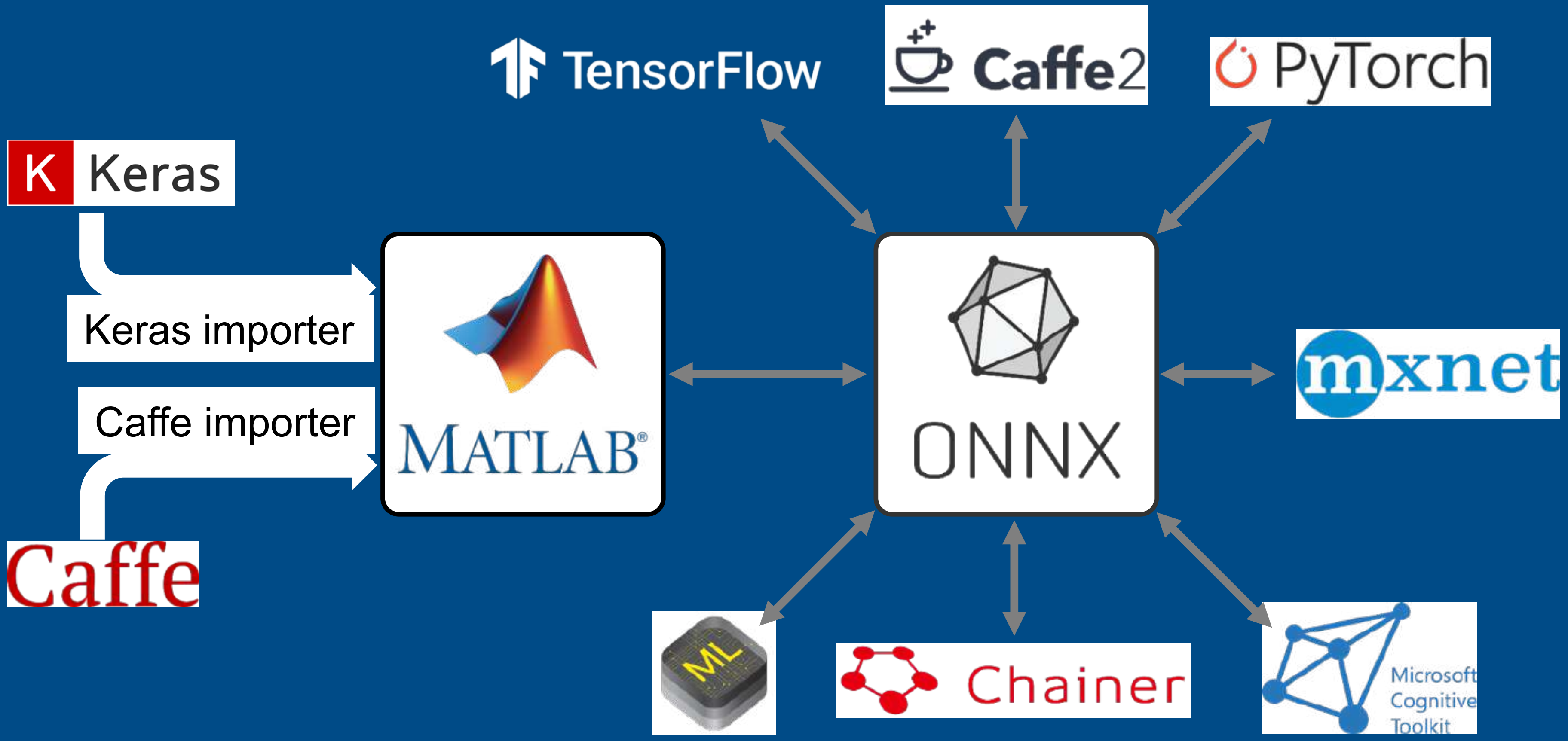


https://www.mathworks.com/help/deeplearning/ug/pretrained-convolutional-neural-networks.html

MathWorks

MATLAB interoperates with other frameworks

Supports ONNX and can exchange models with PyTorch, TensorFlow, and other frameworks.



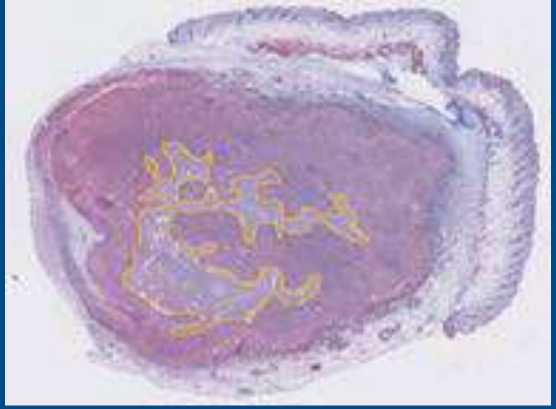
A typical deep learning course looks like…

- Introduction to deep learning
  - Historical context, reason of success, etc.
- Theoretical foundations
  - Mathematics basics
  - Neural networks
- Visualization and debugging of neural networks
- Pretrained models, advanced architectures
- **Applications**

Reference:
- Stanford CS230, 231n
- UMD CMSC 828L
- Dartmouth CS078/178

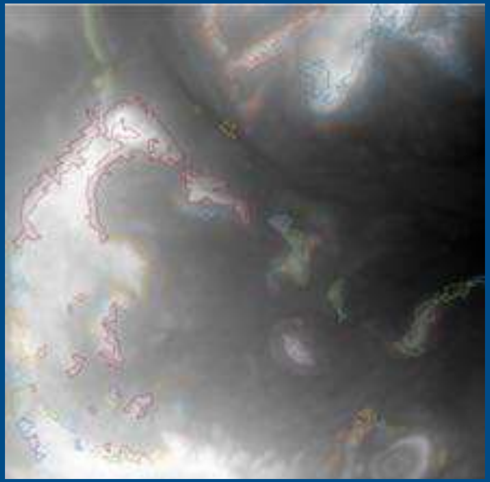
Deep Learning Used in Many Industries for Many Applications

Shell: Machinery Identification

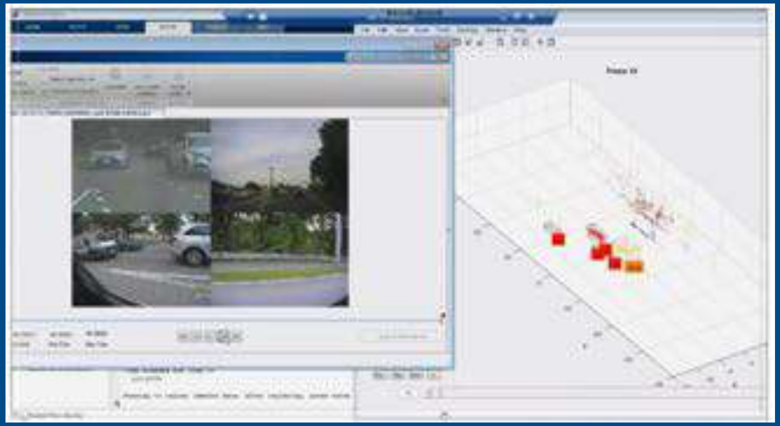


Genentech: Pathology Analysis

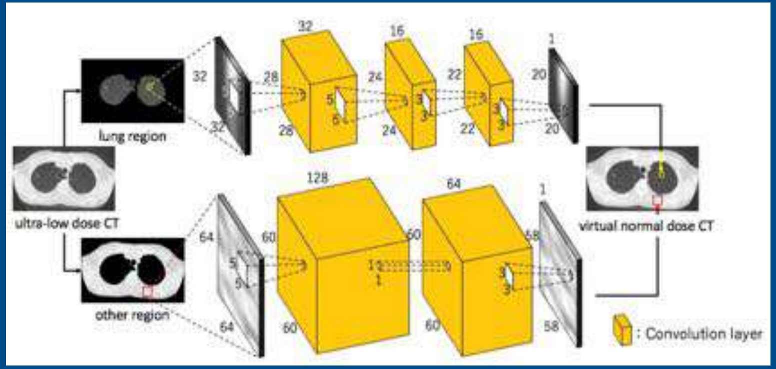
Musashi Seimitsu Industry Co.
Detect Abnormalities in Auto Parts



Shell: Satellite Terrain Recognition



AutoLiv: Lidar Object Detection



Ritsumeikan University
Reduce Exposure in CT Imaging

Types of Data

Numeric Data

ID	WC_TA	RE_TA	EBIT_TA	MVE_BVTD	S_TA	Industry	Rating
62394	0.013	0.104	0.036	0.447	0.142	3	BB
48608	0.232	0.335	0.062	1.969	0.281	8	A
42444	0.311	0.367	0.074	1.935	0.366	1	A
48631	0.194	0.263	0.062	1.017	0.228	4	BBB
43768	0.121	0.413	0.057	3.647	0.466	12	AAA
39255	-0.117	-0.799	0.01	0.179	0.082	4	CCC
62236	0.087	0.158	0.049	0.816	0.324	2	BBB
39354	0.005	0.181	0.034	2.597	0.388	7	AA
40326	0.47	0.752	0.07	11.596	1.12	8	AAA
51681	0.11	0.337	0.045	3.835	0.812	4	AAA

Machine Learning or RNN (LSTM)

Time Series/ Text Data

RNN (LSTM) or CNN

Image Data

CNN

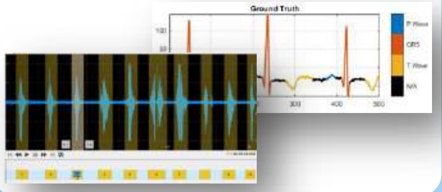
Deep Learning Workflow

Create and Access Datasets

Data sources

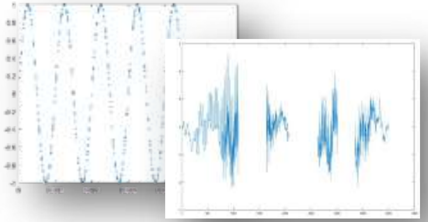
Simulation and augmentation

Data Labeling

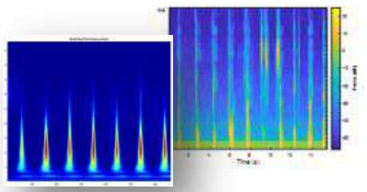


Preprocess and Transform Data

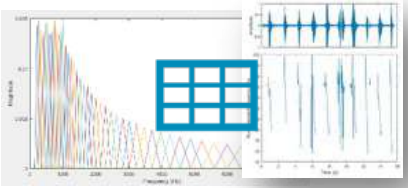
Pre-Processing



Transformation



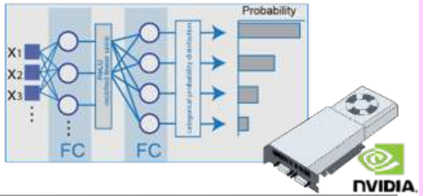
Feature extraction



Develop Predictive Models

Import Reference Models/ Design from scratch

Hardware-Accelerated Training



Analyze and tune hyperparameters

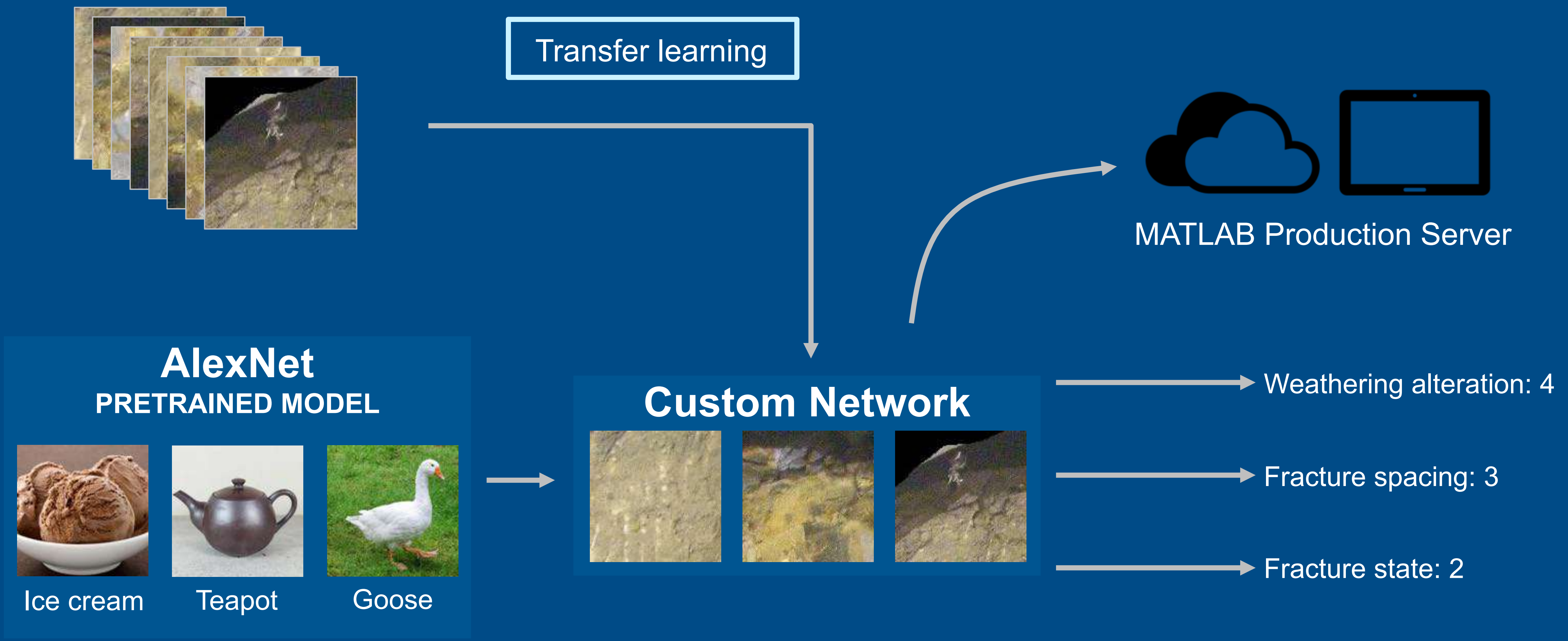
Accelerate and Deploy

Desktop Apps

Enterprise Scale Systems

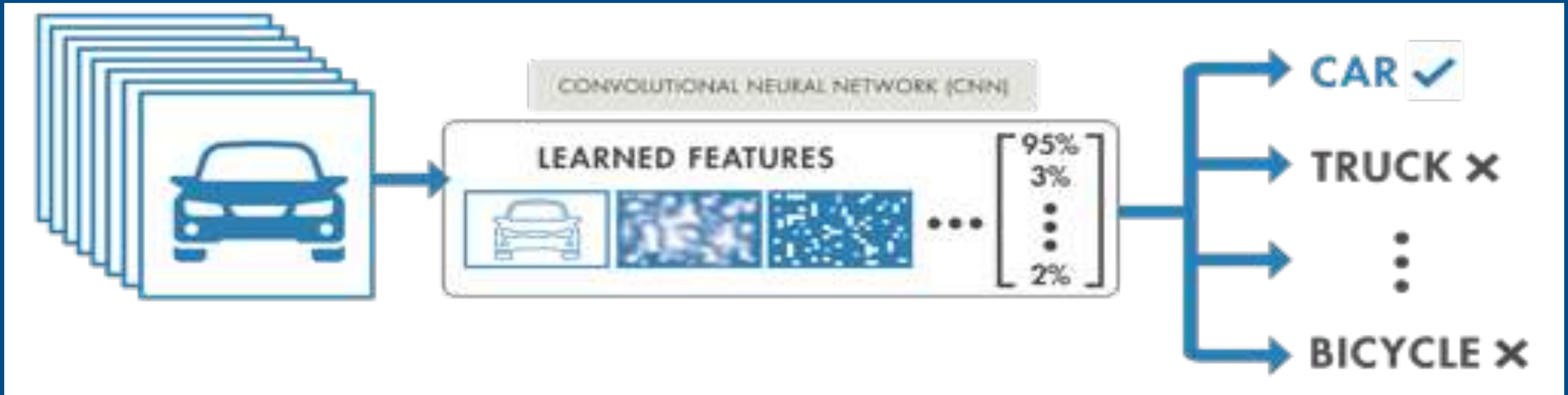
Embedded Devices and Hardware

Efficient tunnel drilling with deep learning
Obayashi Corporation

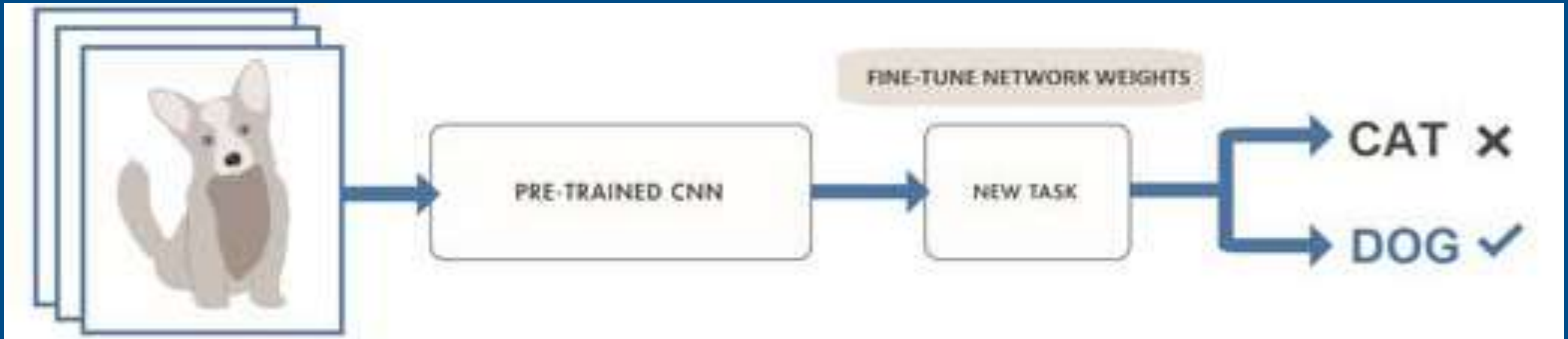


Two Approaches for Deep Learning

1. Train a Deep Neural Network from Scratch

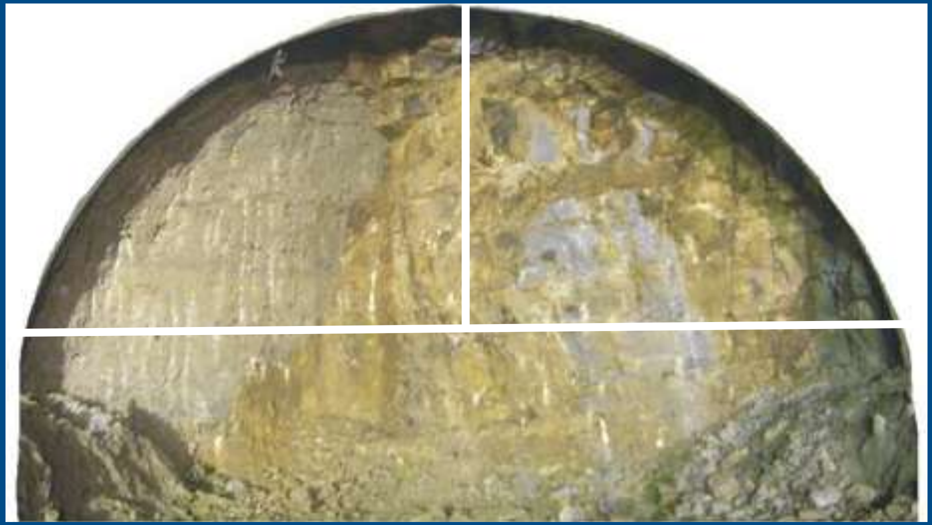


2. Fine-tune a pre-trained model (transfer learning)



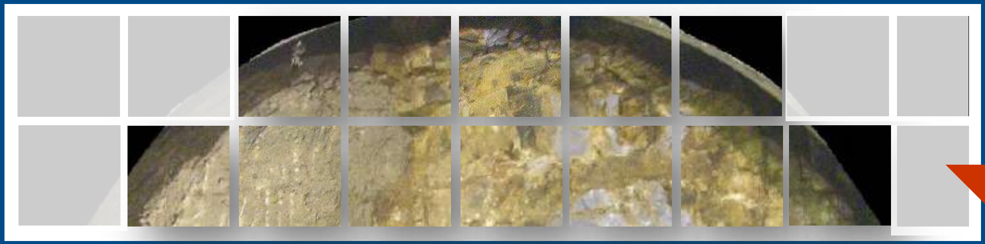
Efficient tunnel drilling with deep learning

Obayashi Corporation



Split into sub-images

Create and Access Datasets



Label each sub-image

Preprocess and Transform Data

Done by geologists

Image	Weathering Alteration (1-4)	Fracture Spacing (1-5)	Fracture State (1-5)
	3	3	2
	4	1	1
	2	3	2
	3	3	2
⋮	⋮	⋮	⋮

Efficient tunnel drilling with deep learning
Obayashi Corporation

CREATE AND ACCESS DATASETS

PREPROCESS AND TRANSFORM DATA

Transfer learning

DEVELOP PREDICTIVE MODELS

ACCELERATE AND DEPLOY

MATLAB Production Server

AlexNet
PRETRAINED MODEL

Ice cream Teapot Goose

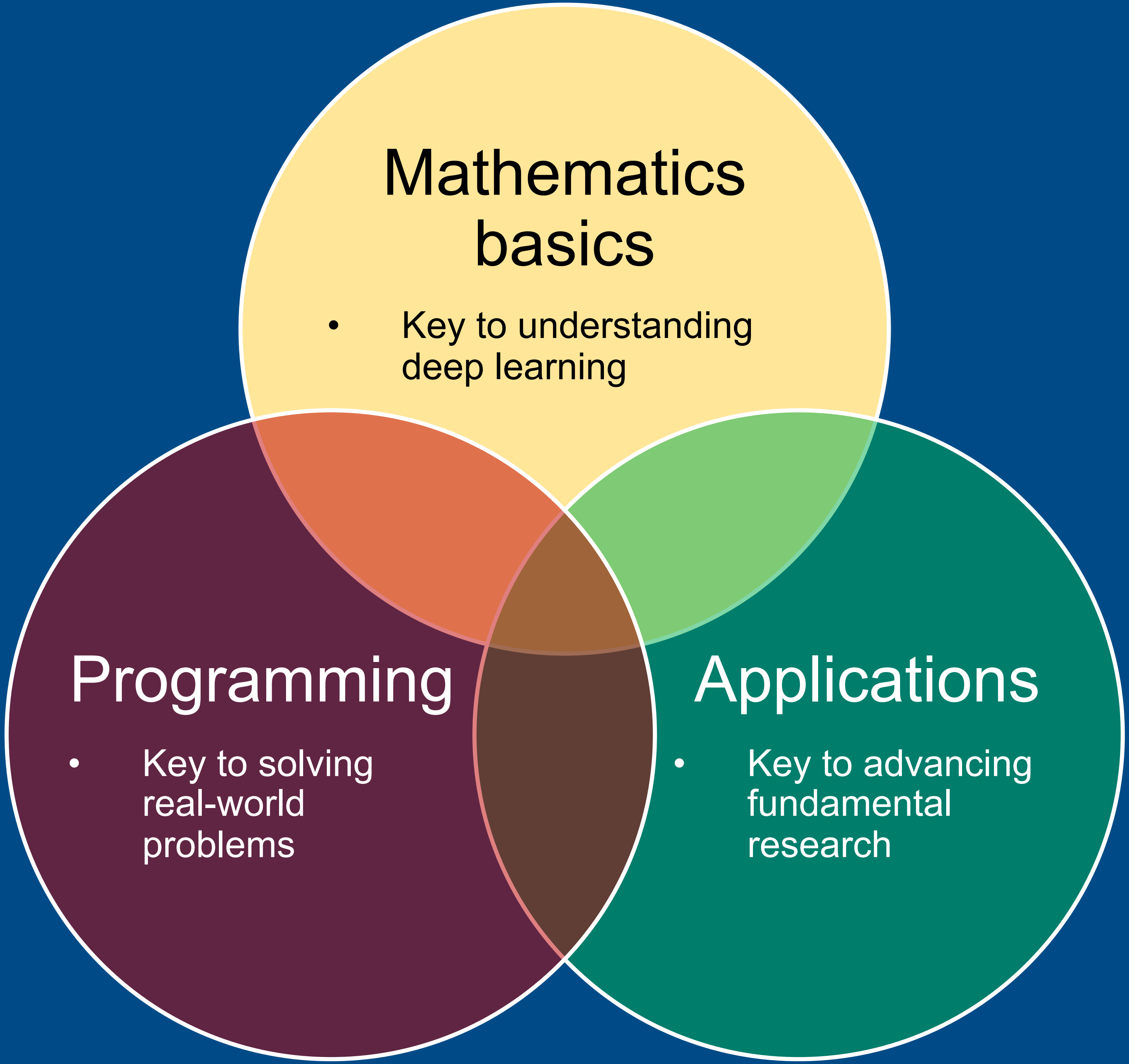
Custom Network

Weathering alteration: 4

Fracture spacing: 3

Fracture state: 2

What does a successful deep learning course look like?


Mathematics basics
• Key to understanding deep learning
Programming
• Key to solving real-world problems
Applications
• Key to advancing fundamental research