Fundamentals of Training AI Models for Computer Vision and Video Analytics Applications, Part 1

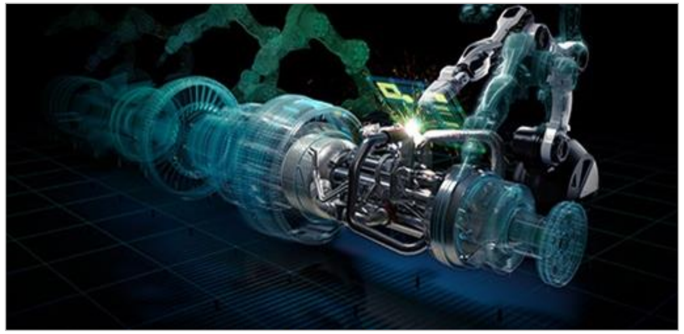
Ekaterina Sirazitdinova
Data Scientist
NVIDIA

Agenda

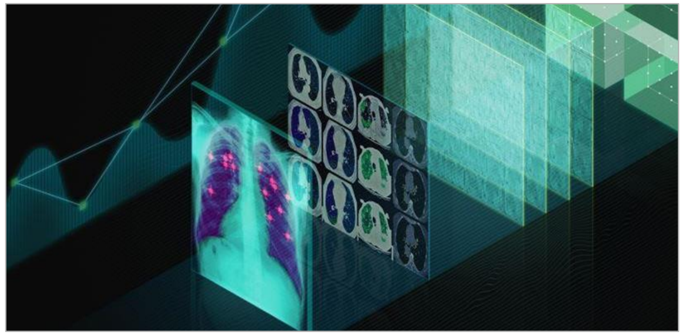
- AI and deep learning for vision processing
  - Embedded vison use cases
- Deep learning mechanism
  - How it works and how neural networks learn and how to prepare data for a supervised training
- Possible problems during training and their mitigation
  - Observing training behaviour and reacting accordingly
- Formulating a vison task as a deep learning problem
  - Kinds of tasks solved by deep learning in computer vision and video analytics
- Where to start
  - Deep learning frameworks and community resources

AI and Deep Learning for Vision Processing

AI & Deep Learning are Changing the World

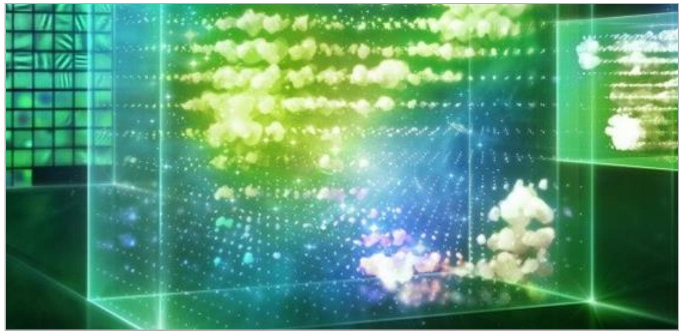


Robotics
Manufacturing, construction, navigation



Healthcare
Cancer detection, drug discovery, genomics

Autonomous Vehicles
Pedestrian & traffic sign detection, lane tracking



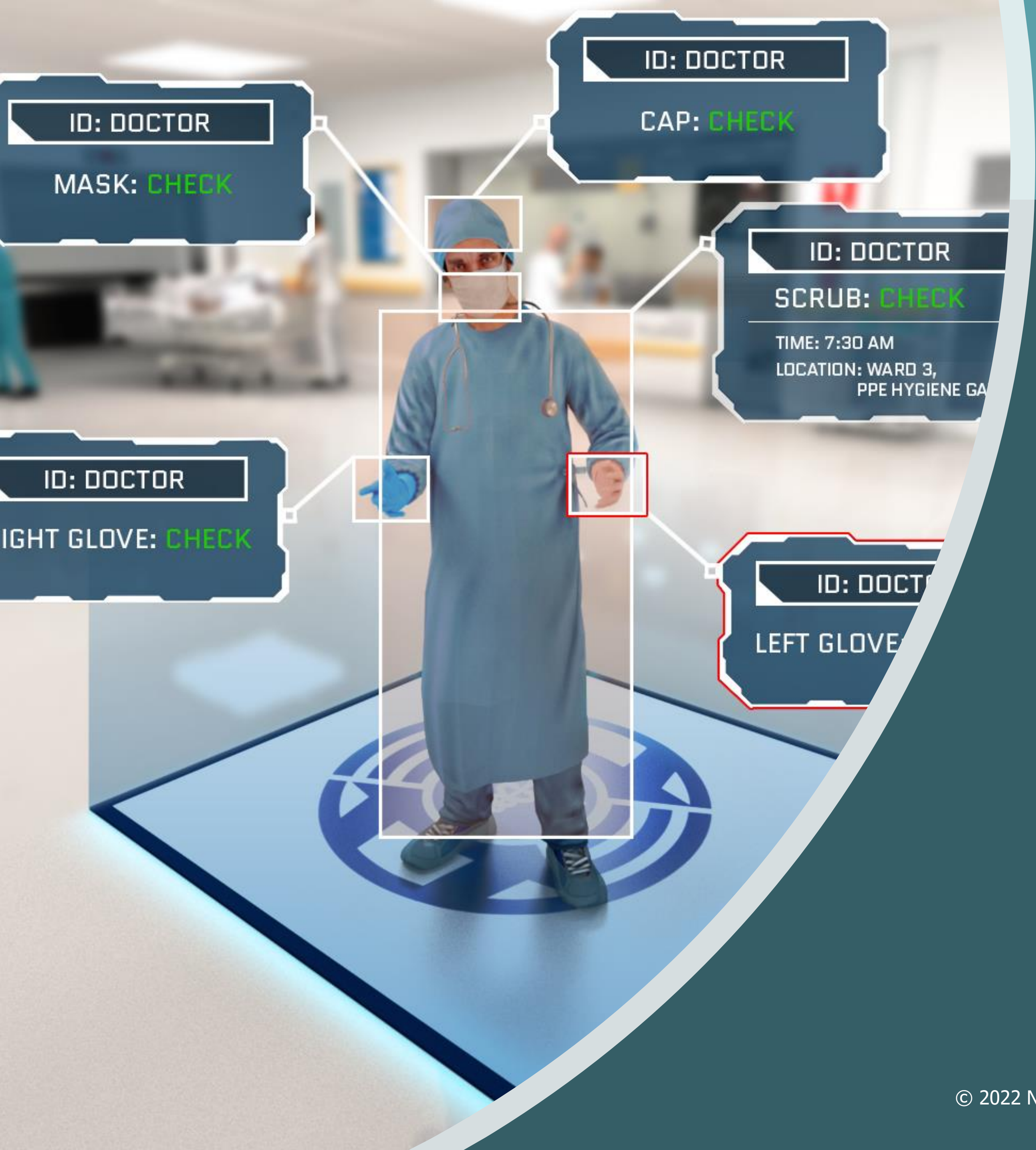
Internet Services
Image classification, speech recognition, NLP

Media & Entertainment
Digital content creation

Intelligent Video Analytics
AI cities, urban safety

Embedded AI Applications: Industrial Inspection

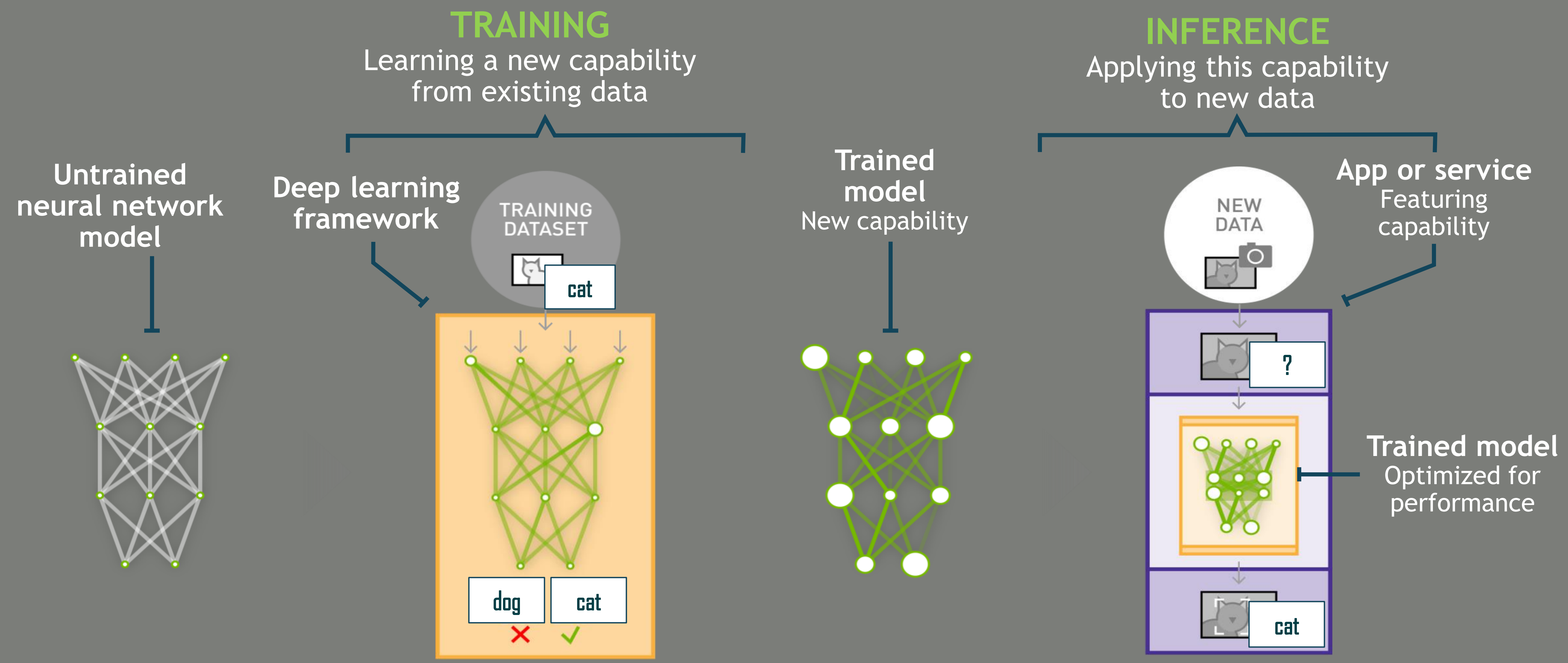
Embedded AI Applications: Agriculture



Embedded AI Applications: Healthcare

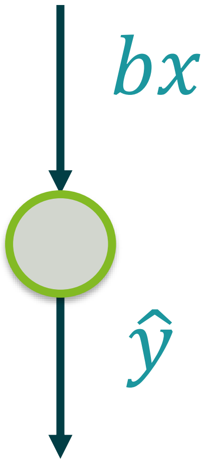
Deep Learning Mechanism

How Does Deep Learning Work?

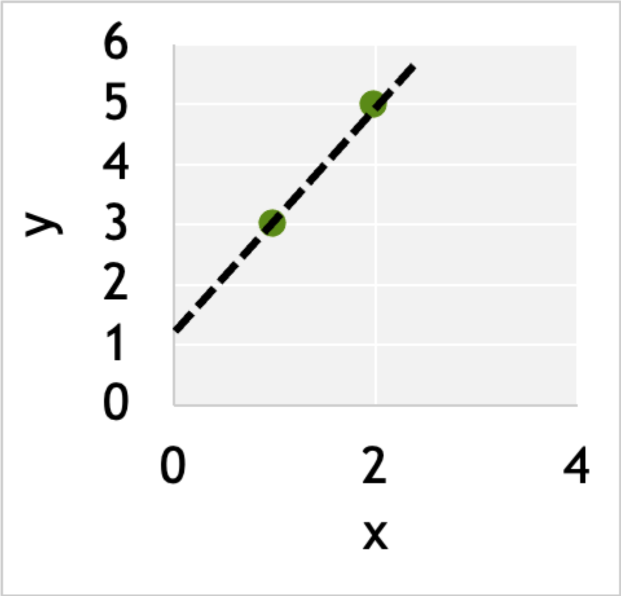


A Simpler Case: Linear Model

$$\hat{y} = bx + a$$

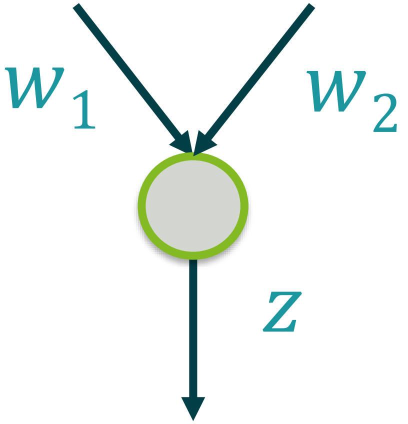


$$b = ?$$
$$a = ?$$



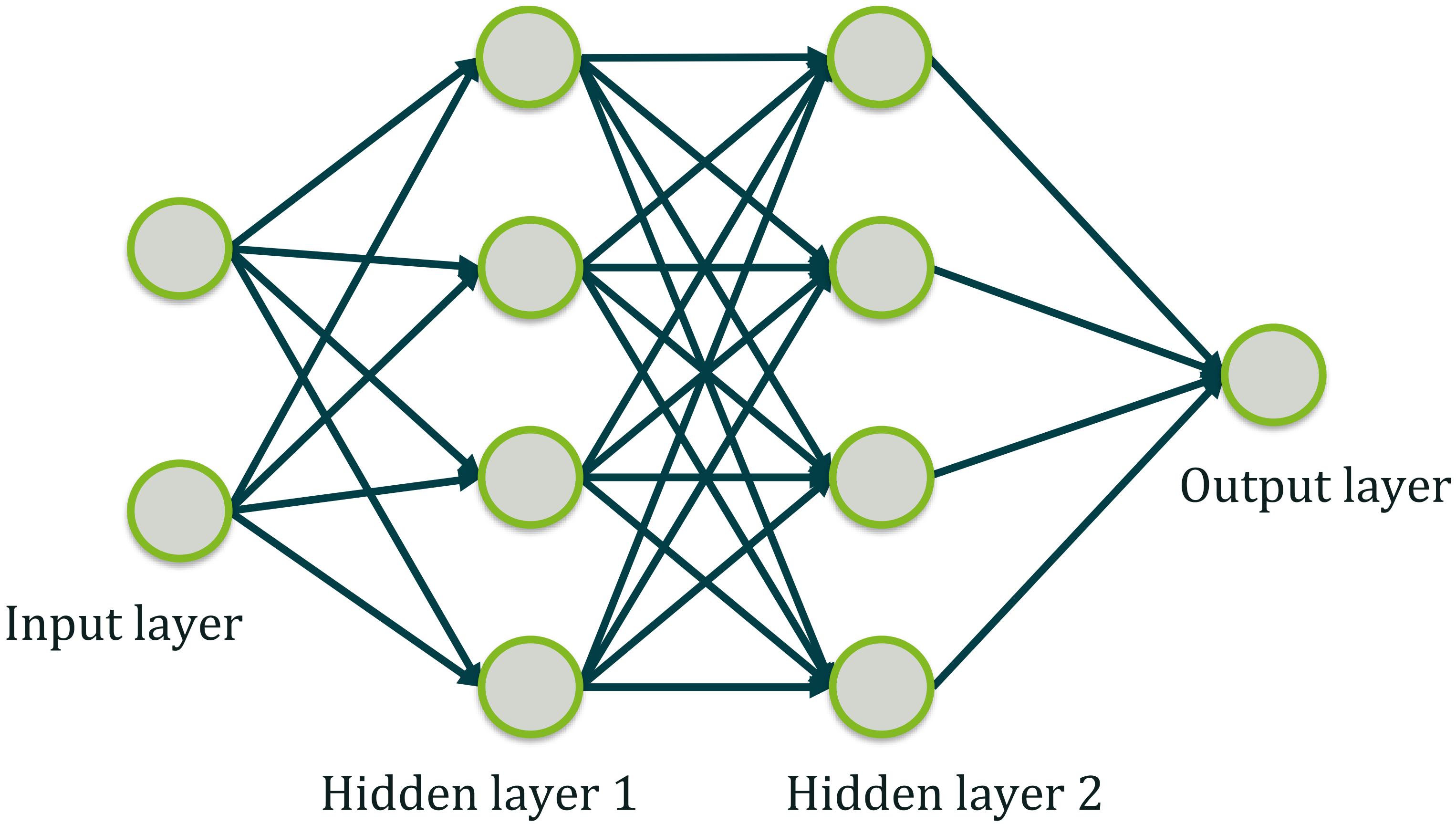
Weights and ***biases*** are the learnable parameters of a machine learning model

From Neuron to Network



Score:

$$z = x_1 w_1 + \cdots + x_n w_n$$



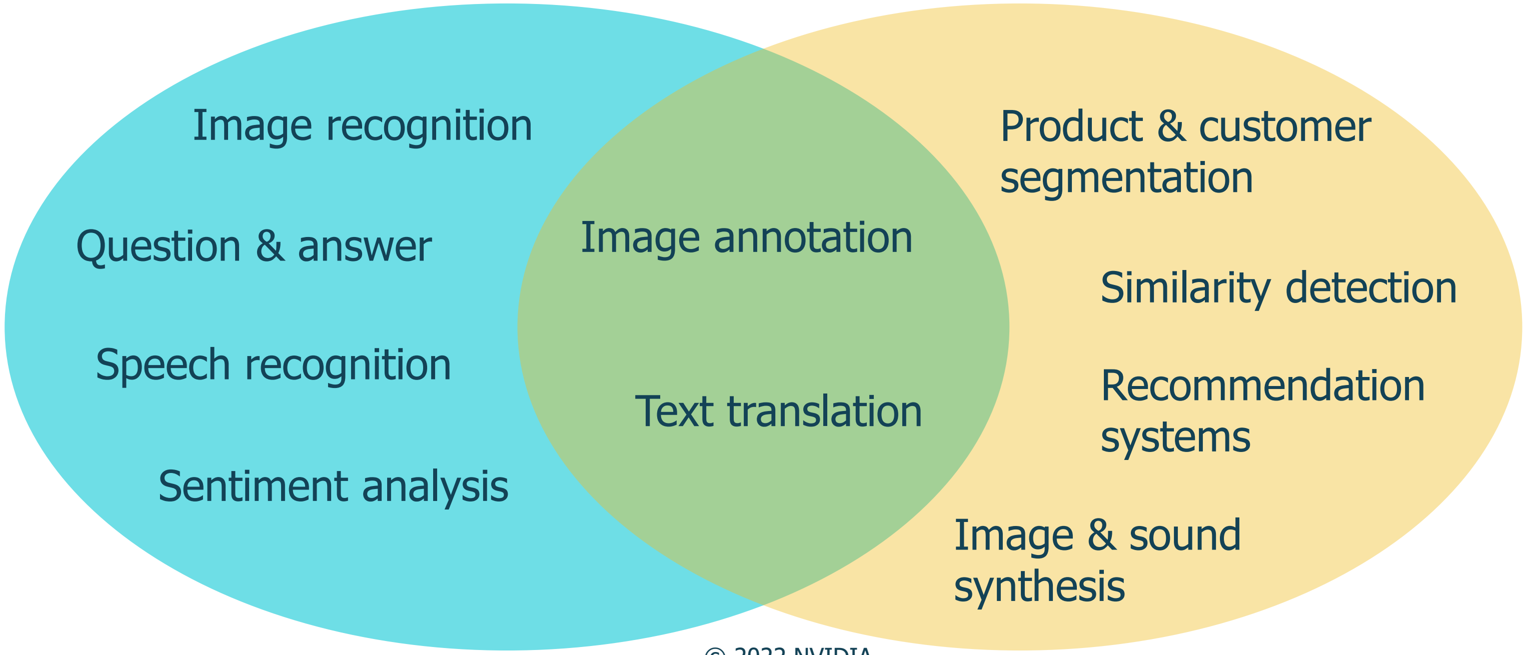
Supervised and Unsupervised Learning

Supervised methods

- Defined by its use of labelled datasets
- Two problem types: ***classification*** and ***regression***

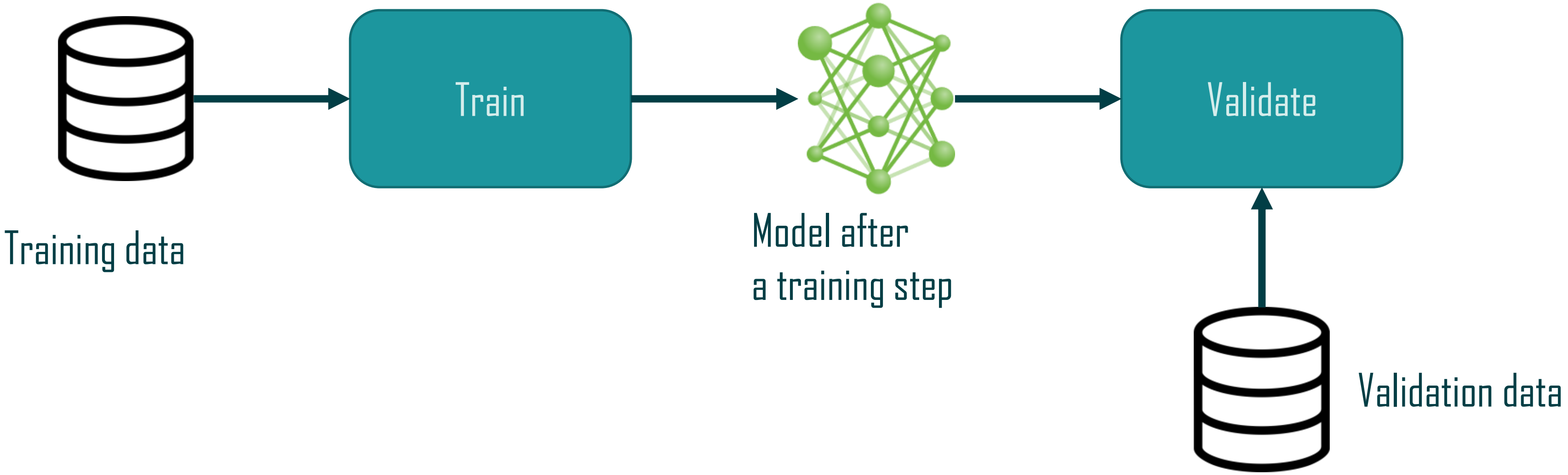
Unsupervised methods

- Analyze and cluster unlabelled data sets
- Discover hidden patterns in data
- No need for human intervention



Preparing Datasets for Supervised Training

- Acquiring *enough* data
- Ground truth labelling
- Balancing classes
- Splitting data into ***training***, ***validation*** and ***testing*** sets



Forward Pass and Activations

Some popular activation functions

- ReLU:

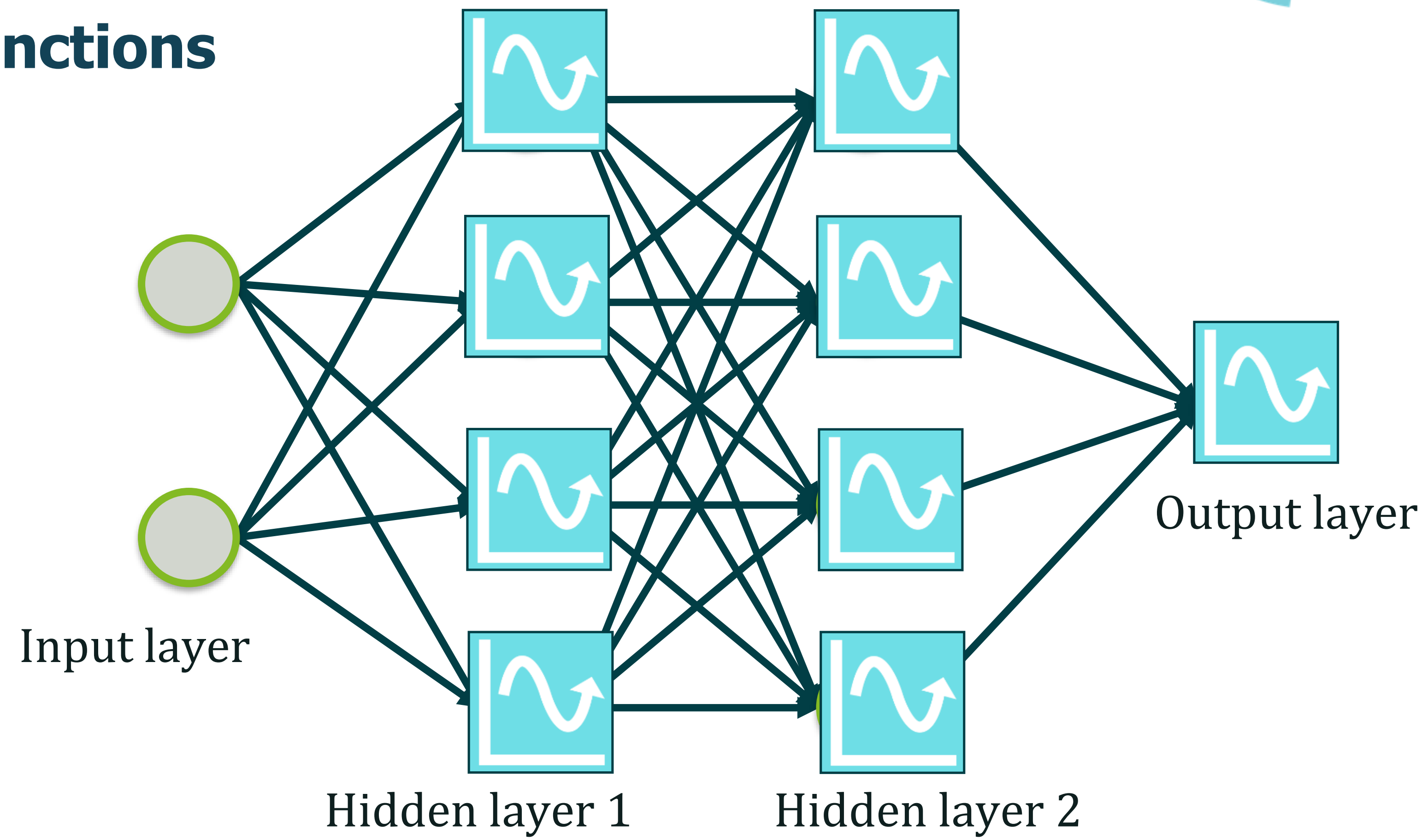
$$R(z) = \max(0, z)$$

- Tanh:

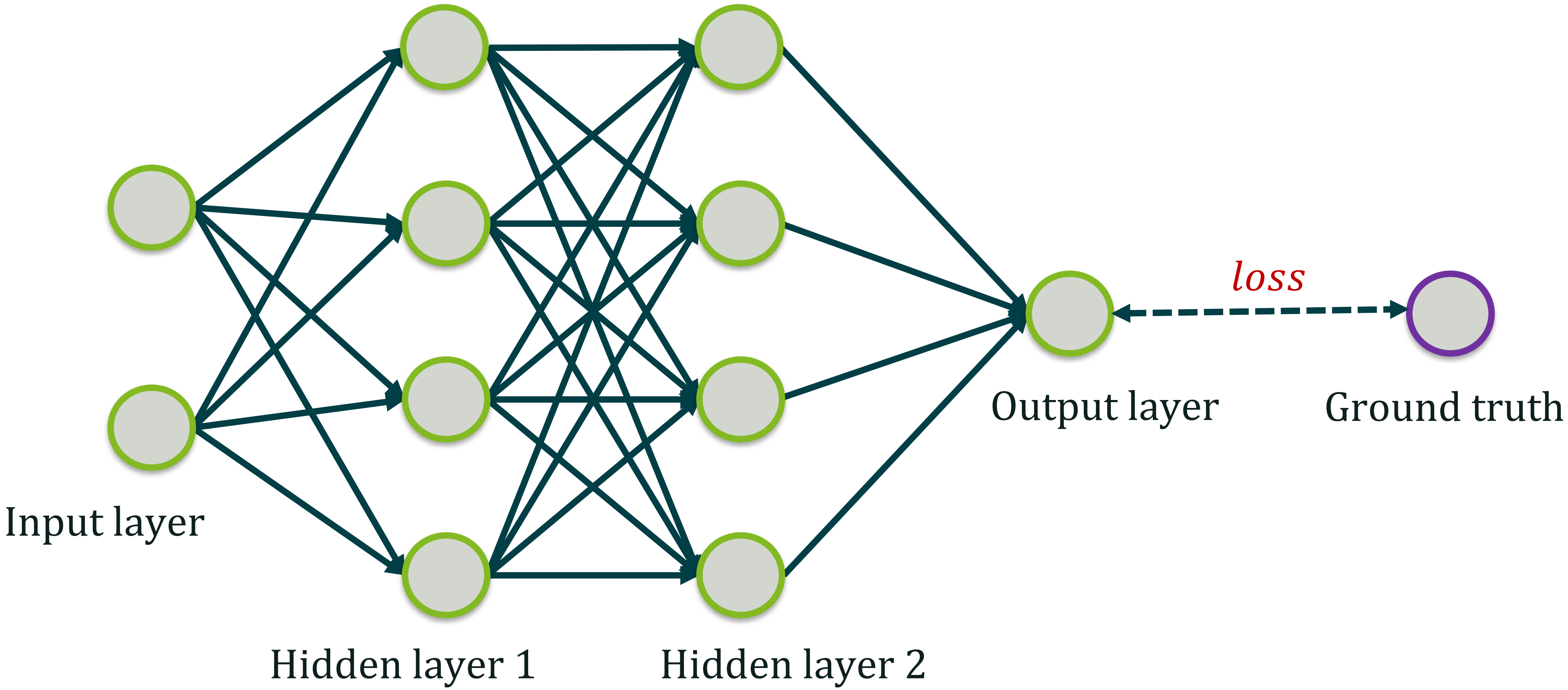
$$\tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

- Sigmoid:

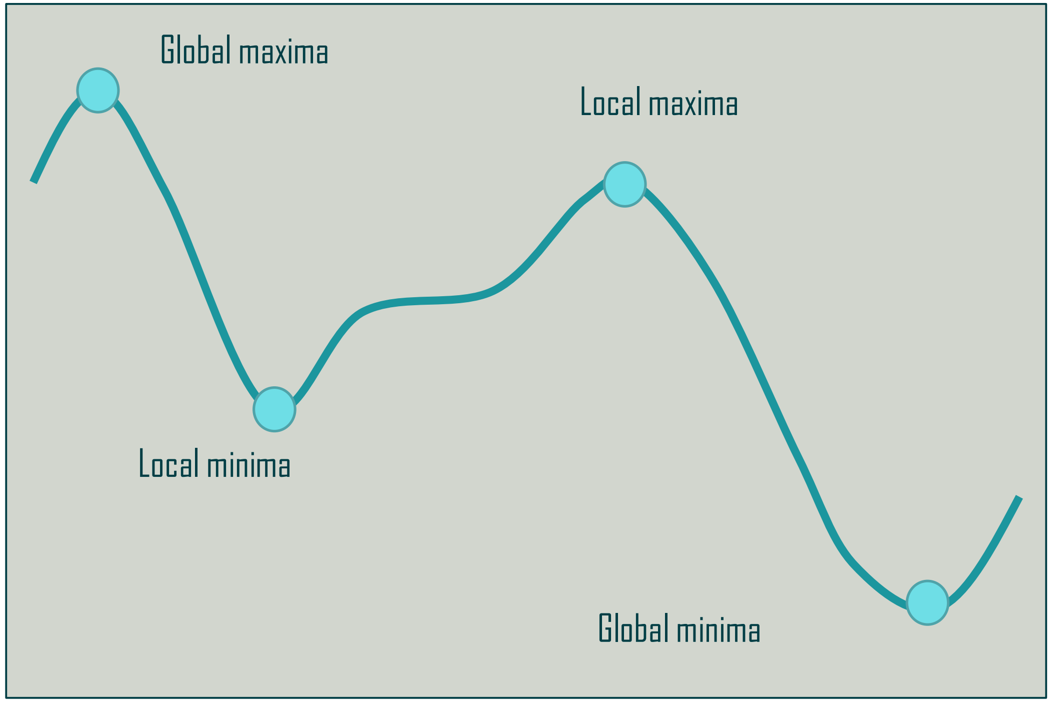
$$S(z) = \frac{1}{1 + e^{-z}}$$

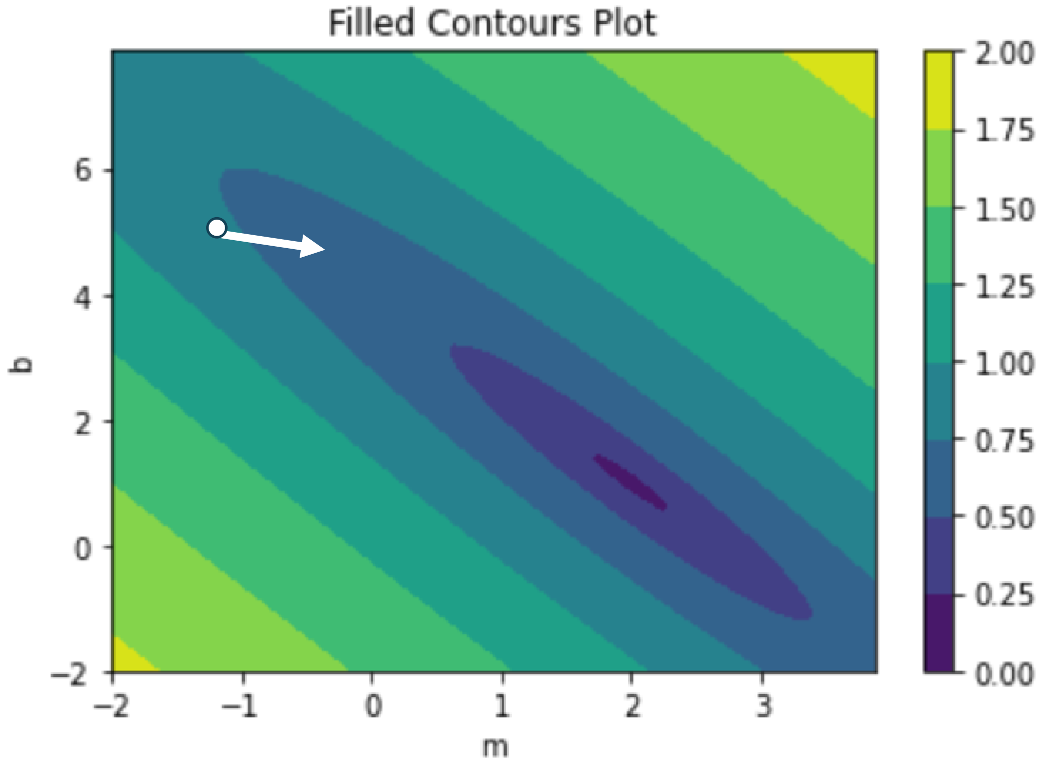


Loss Function: Calculating the Total Error



Function Optimization

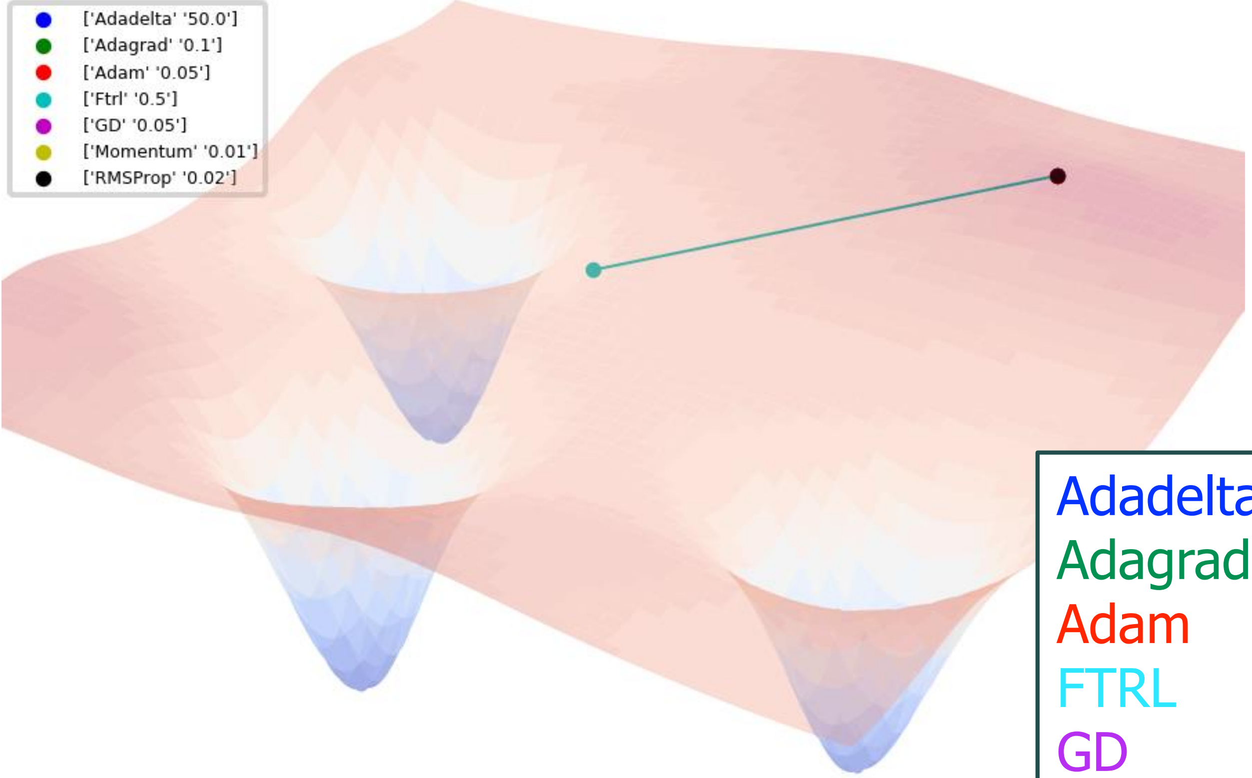


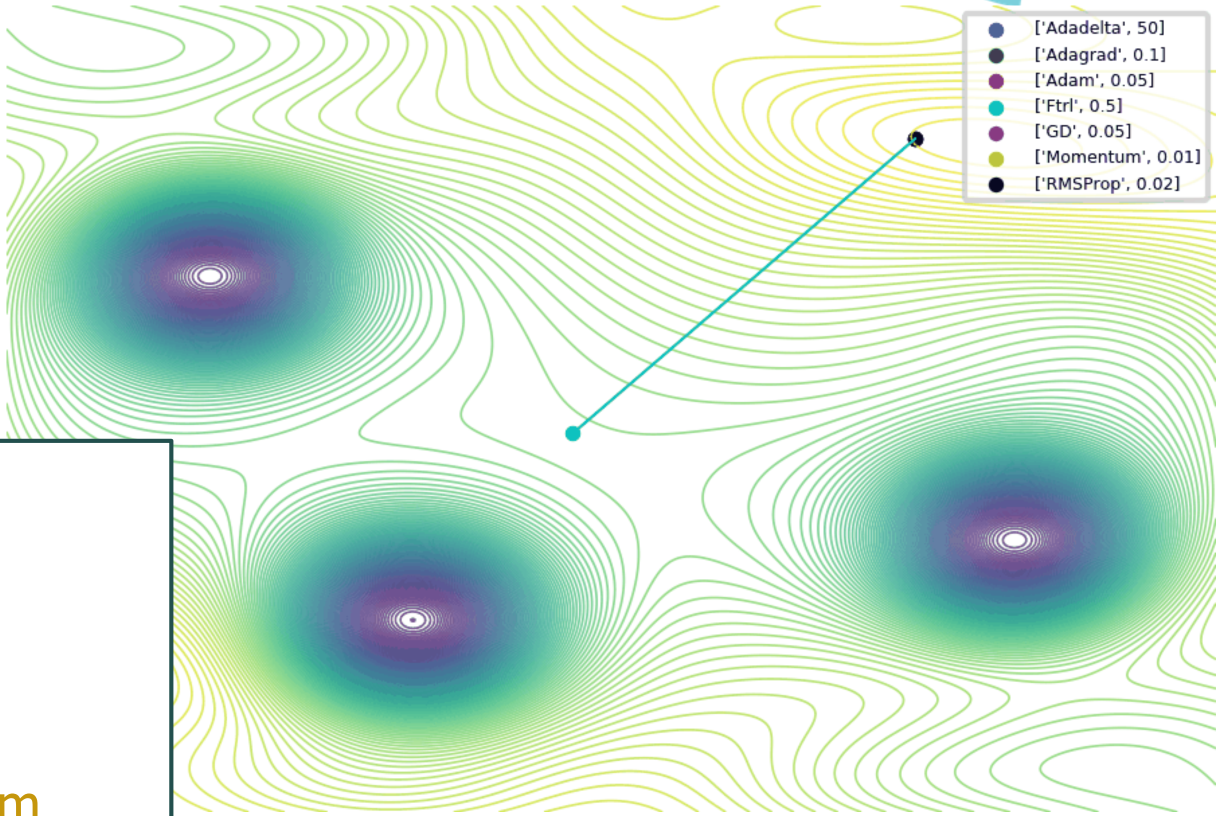


Gradient Descent and Hyperparameters

- Gradient: which direction loss decreases the most
- Learning rate: how far to travel
- Step: an update to the weight parameters
- Epoch: a model update with the full dataset
- Batch: a sample of the full dataset
- Momentum: accelerates the optimization process

Popular Optimizers

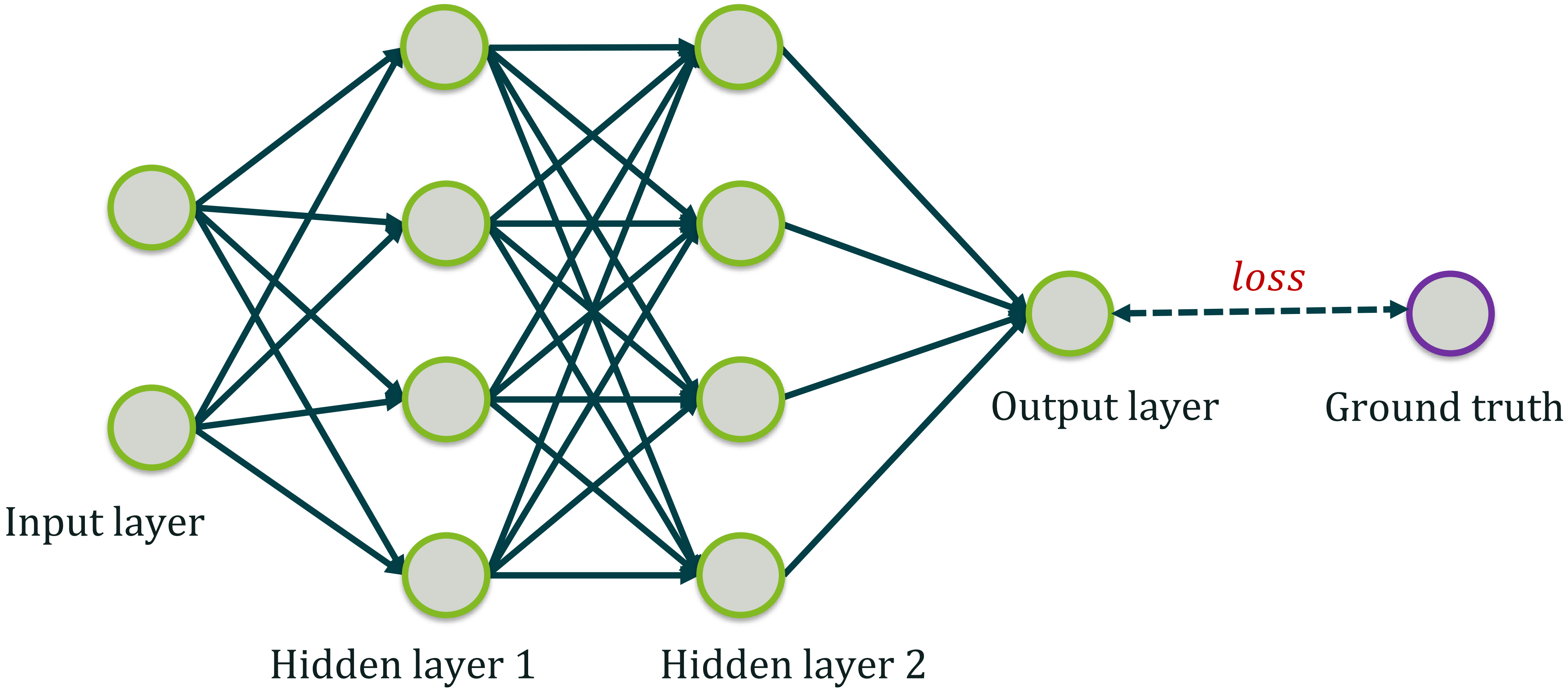




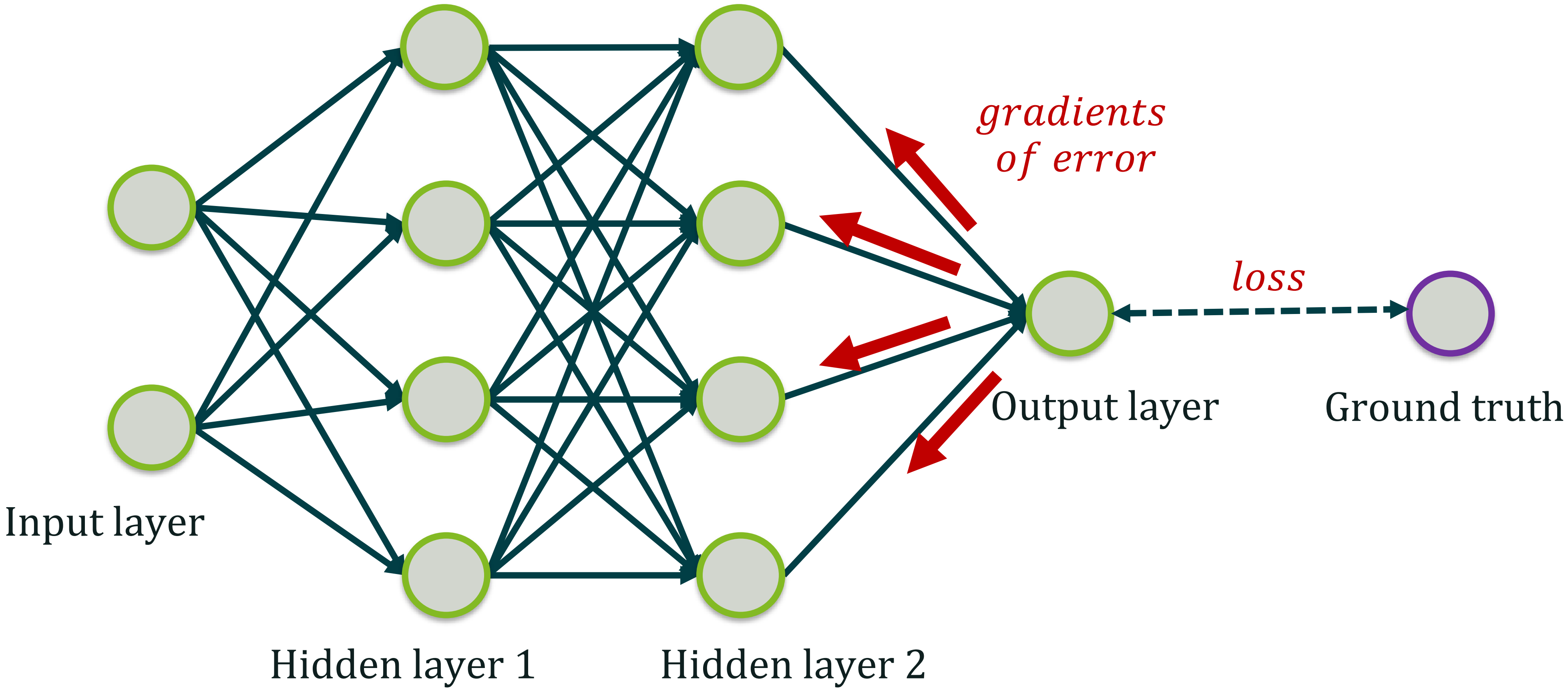
Adadelta
Adagrad
Adam
FTRL
GD
Momentum
RMSProp

Animation credit: https://github.com/Jaewan-Yun/optimizer-visualization

Backpropagation: Update Each of the Weights in the Network



Backpropagation: Update Each of the Weights in the Network



Neural Network: Math Behind (Matrix Form)

- **Forward pass**

$$x_i = f_i(W_i\, x_{i-1})$$

$$E = \|x_L - y\|_2^2$$

- **Backward pass**

$$\delta_L = (x_L - y) \circ f'_L(W_L\, x_{L-1})$$

$$\delta_i = W_{i+1}^T\ \delta_{i+1} \circ f'_i(W_i\, x_{i-1})$$

Here $\circ$ is the Hadamard (element-wise) product

- **Weight update**

$$\frac{\partial E}{\partial W_i} = \delta_i\, x_{i-1}^T$$

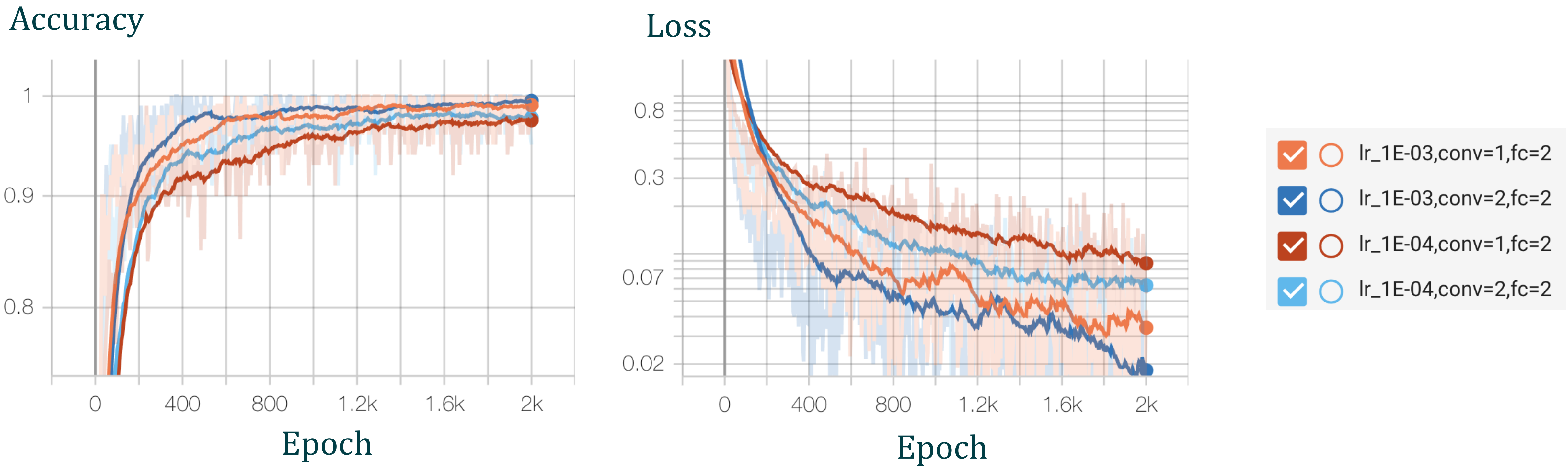
$$W_i = W_i - \alpha_{W_i} \circ \frac{\partial E}{\partial W_i}$$

L – number of network layers
x_0 – input vector; x_L – output vector
E – loss
$W_{\overline{1,L}}$ – weight matrices
$f_{\overline{1,L}}$ – activation functions
α_{W_i} – weight scalar

Fundamentals of Training AI Models for Computer Vision and Video Analytics Applications, Part 2

Ekaterina Sirazitdinova
Data Scientist
NVIDIA

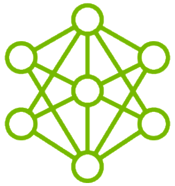
Training Visualization: Loss Curves



https://tensorboard.dev/

Transfer Learning

Train a model with a dataset



Train the same model with another dataset:

- With different distribution of classes
- With different classes

Objective:

- Model repurposing (brand new classes, different modalities, scene adaptation)
- Extending the model to new classes

Fine Tuning

Train a model with ~90% of a dataset → Train the model with the remaining ~10%

During re-training:

- Smaller learning rate
- Freezing some layers

Objective: increased accuracy

Possible Problems During Training, and Their Mitigation

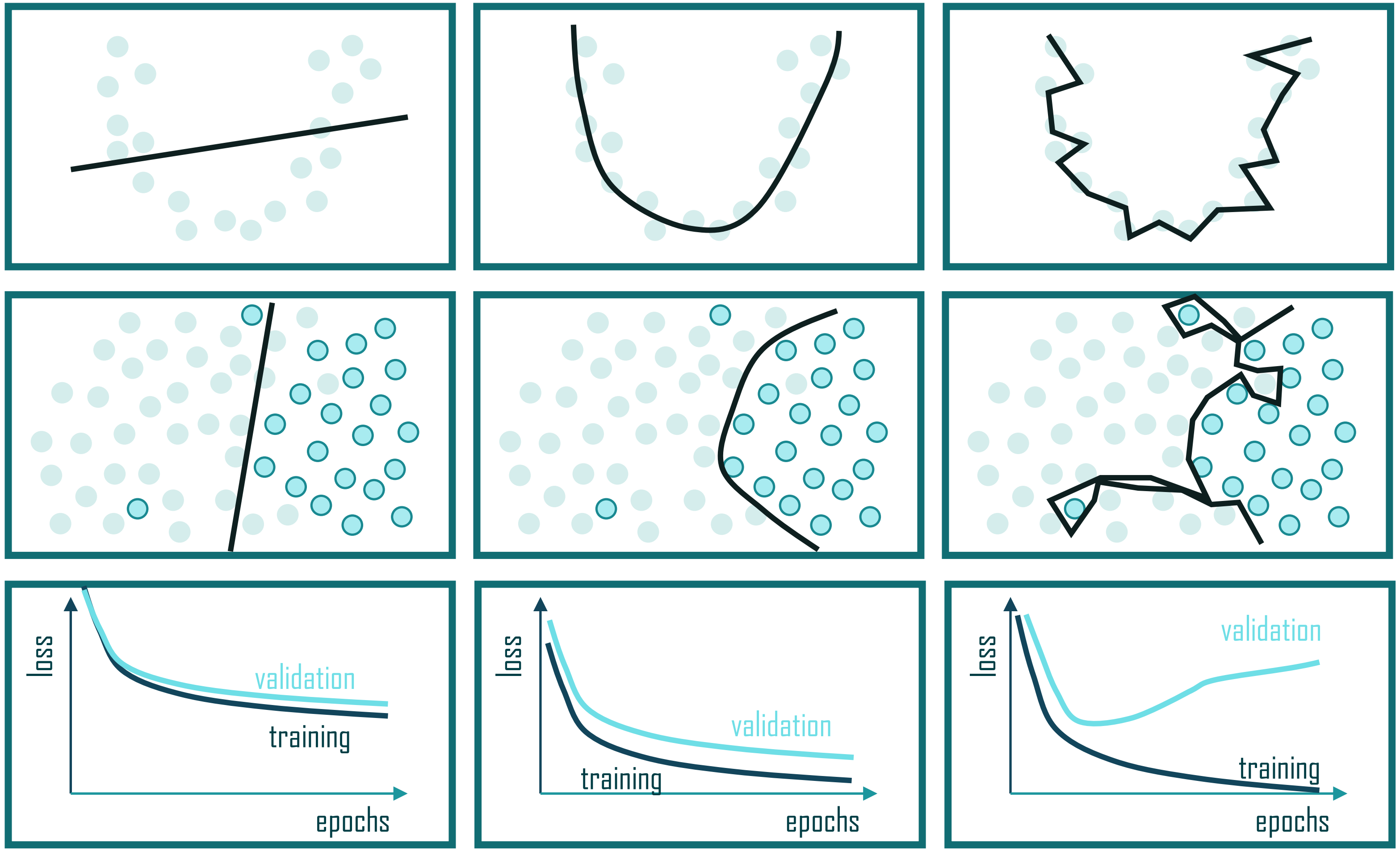
NVIDIA

Underfitting – Just Right – Overfitting

Regression

Segmentation

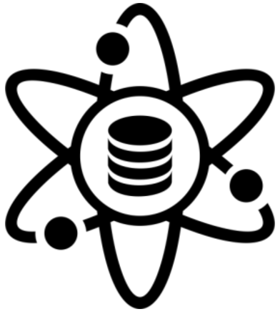
Deep learning loss curves



Preventing Over- and Underfitting

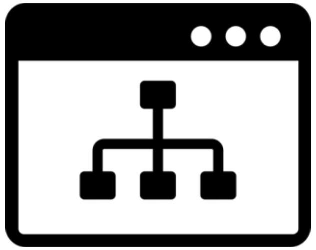
With the help of data

- Add more examples
- Check that your data set is balanced
- Use a separate test set
- Apply data augmentation



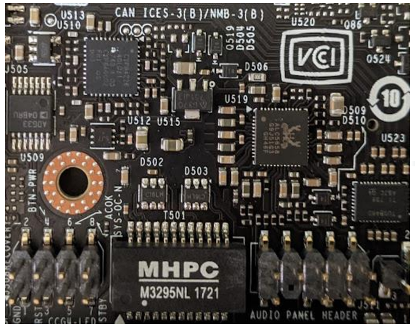
Applying techniques

- Early stopping / increasing number of epochs
- Changing network complexity
- Regularization
  - Pruning
  - Dropout
  - Loss penalty (L1 and L2)
- Ensembling
- Transfer learning

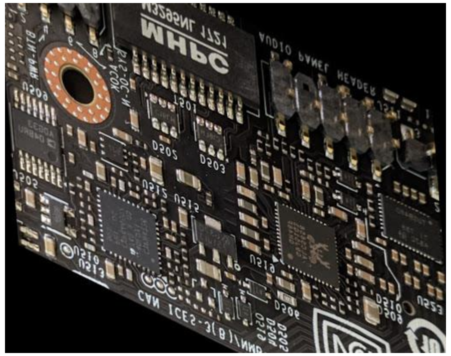


Data Augmentation: Improve Accuracy with Limited Dataset

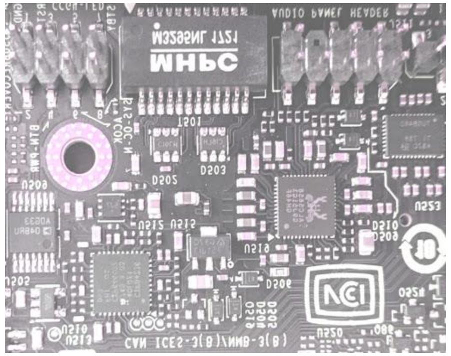
Original



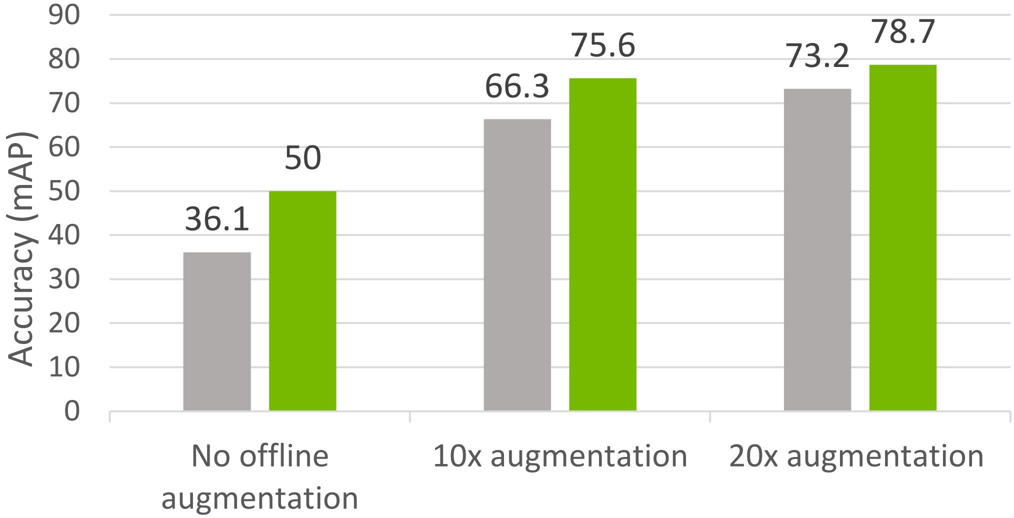
Rotated



Brightness adjusted

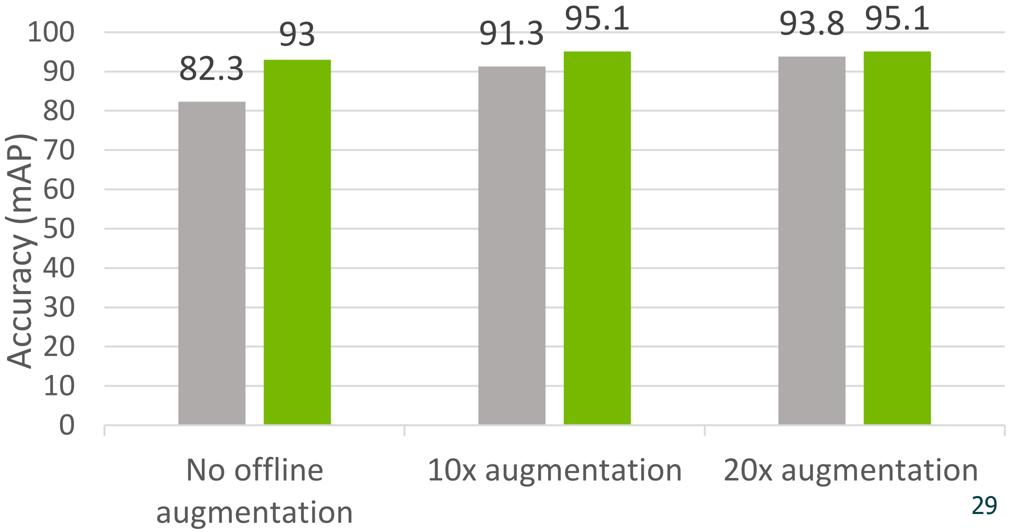


100 Images Experiment



No online augmentation
Online augmentation

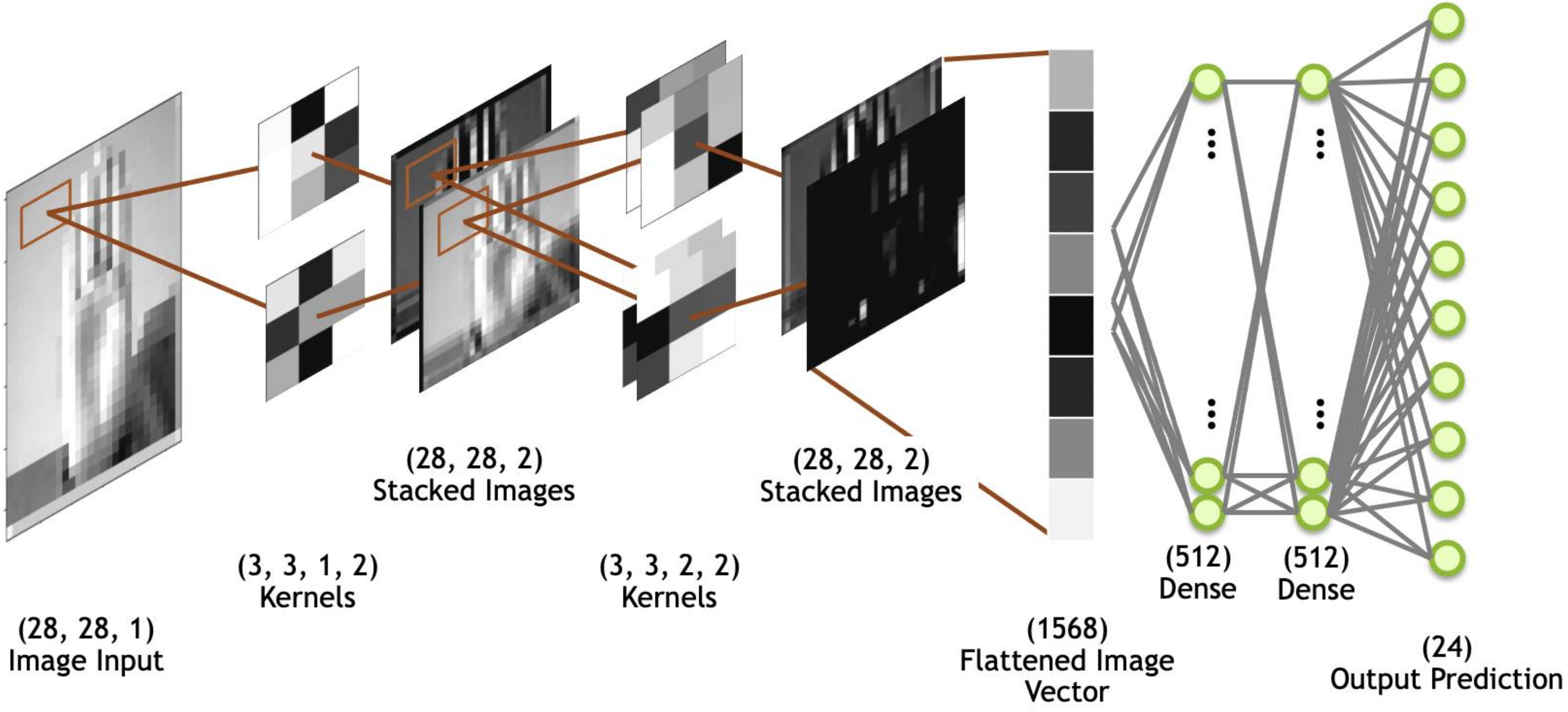
500 Images Experiment



Synthetic Data Generation

Formulating a Vision Task as a Deep Learning Problem

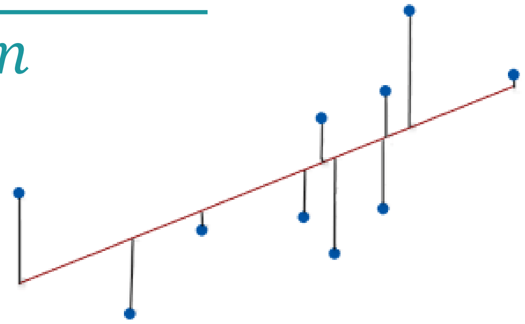
Convolutional Networks



Regression Problem

- Predict a real-value quantity
- **Output layer configuration**: one node with a linear activation unit
- **Loss function**: mean squared error (MSE)

$$Loss_{MSE} = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$



Example: regression of key-points for human pose estimation in 2D

Binary Classification Problem

- Classify an example as belonging to one of two classes
- **Output layer configuration**: one node with a *sigmoid* activation unit
- **Loss function**: binary cross-entropy (logarithmic) loss

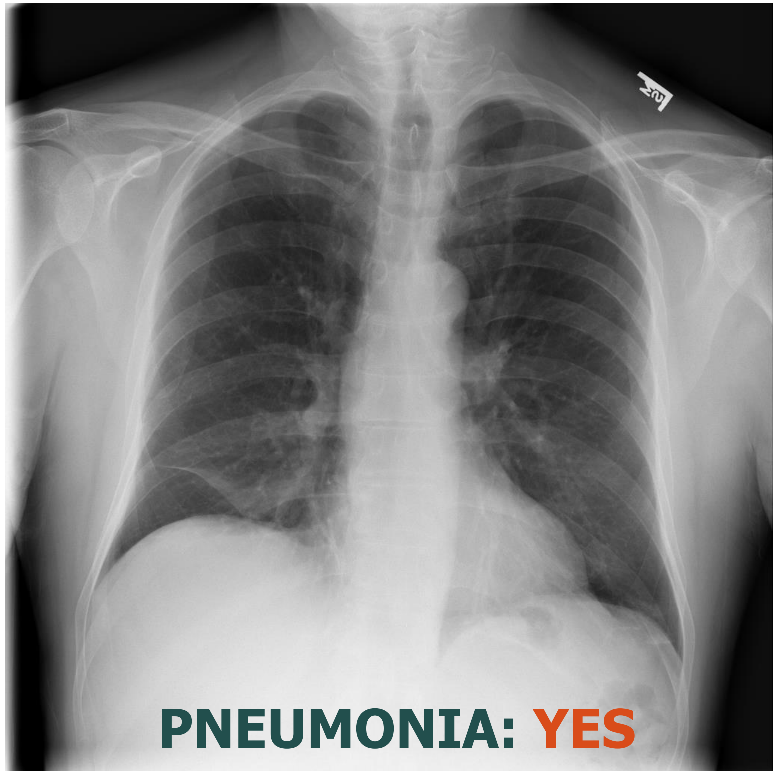
$$\hat{y}_i = f(z_i) = \frac{1}{1 + e^{-z_i}}$$

C – number of scalar values in the model output

→ Sigmoid → Cross-Entropy Loss

$$Loss_{CE} = -\frac{1}{C}\sum_{i=1}^{C}(y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i))$$

Example: identifying whether a patient has pneumonia or not



Multi-Class Classification Problem

- Classify an example as belonging to one of more than two classes
- **Output layer configuration**: one node for each class using the *softmax* activation function
- **Loss function**: categorical cross-entropy (logarithmic) loss

$$\hat{y}_i = f(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}}$$

$$Loss_{CE} = -\sum_{i=1}^{C} y_i \log \hat{y}_i$$

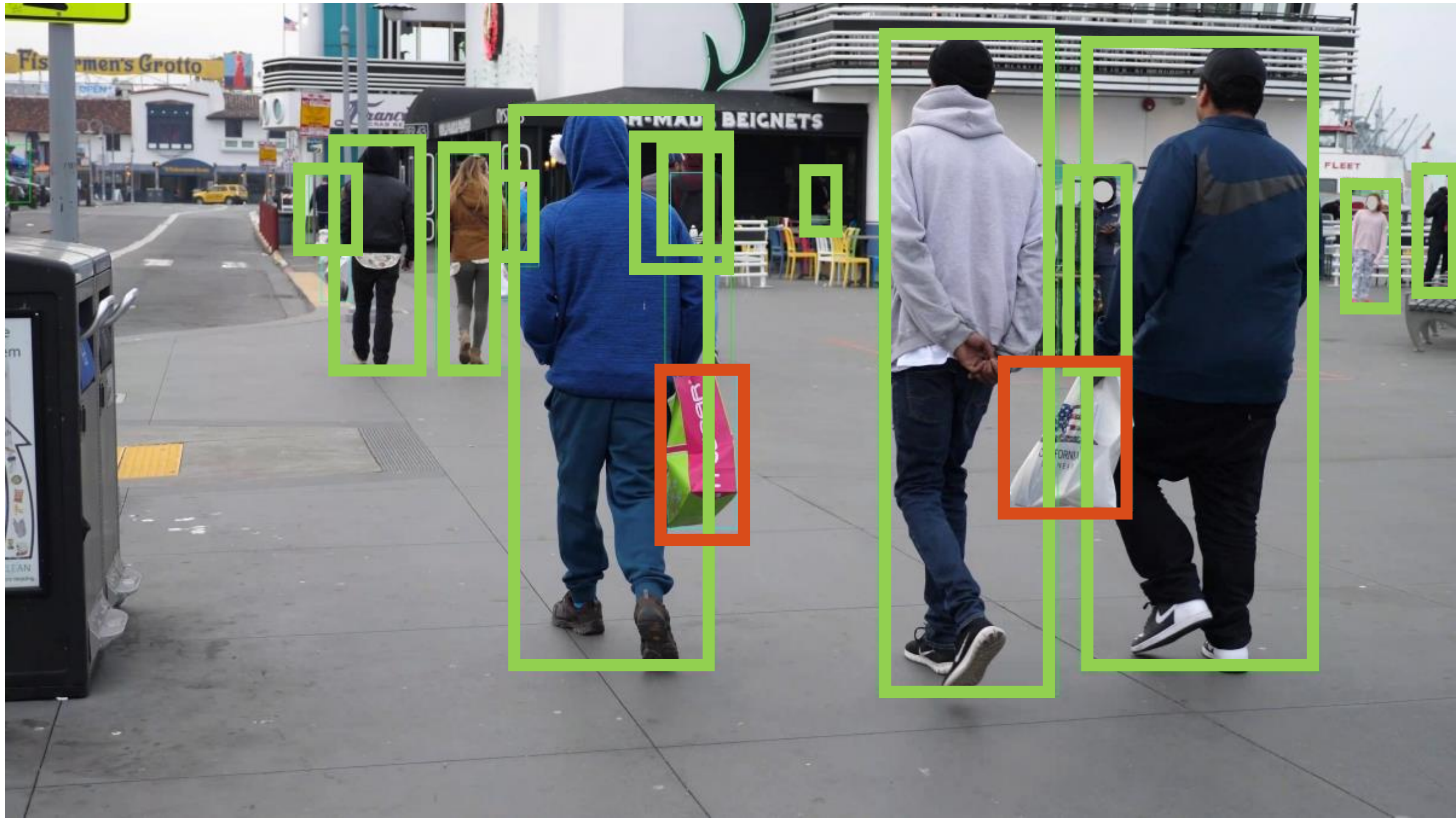
Example: identifying which body part an x-ray image represents

CHEST
SKULL
HAND
DENTAL
NECK

Detection Problem

- Detect boundaries of objects belonging to the classes of interest
- **Popular architectures:**
  - YOLOv3, YOLOv4
  - SSD
  - Faster RCNN
  - RetinaNet

Example: finding locations of all persons and bags in the scene



Segmentation Problem

- Detect contours of objects belonging to the classes of interest
- **Types:** *semantic* and *instance* segmentations
- **Popular architectures:**
  - U-net
  - Mask-RCNN

Example: finding precise contours of all persons in the scene

Where to Start?

Deep Learning Frameworks

embedded VISION summit

torch

Microsoft Cognitive Toolkit

Caffe2

fast.ai

mxnet

PyTorch

Caffe

TensorFlow

theano

Chainer

K Keras

DL4J

Summary: In this Talk You Have Learned

- What is a neural network and what does it mean to train it?
- What are important steps of deep learning training process and parameters affecting it?
- What can go wrong, and how to make it right
- How to prepare your data for training
- How to formulate a deep learning problem
- What public resources you can use to master deep learning

Resources: Learning

Community resources

Kaggle

https://www.kaggle.com/

Papers with code

https://paperswithcode.com/

MLPerf

https://mlperf.org/

From the industry and academy

Google AI Education

https://ai.google/education/

Machine learning course by Stanford (Coursera)

https://www.coursera.org/learn/machine-learning

NYU Deep Learning

https://cds.nyu.edu/deep-learning/

AI Conferences

NeurIPS

https://nips.cc/

CVF conferences

https://openaccess.thecvf.com

ICML

https://icml.cc/

Resources: NVIDIA

Resources

NVIDIA Technical Blog

https://developer.nvidia.com/blog/

Deep Learning Institute

https://www.nvidia.com/en-us/training/

NGC Software Hub

https://catalog.ngc.nvidia.com/

"NVIDIA Tools" at **2022 Embedded Vision Summit**

May 18

- 1:30pm - 2:00pm **"Accelerating the Creation of Custom, Production-Ready AI Models for Edge AI"** by Akhil Docca
- 2:05pm - 2:35pm **"Vision AI At the Edge: From Zero to Deployment Using Low-Code Development"** by Alvin Clark

Thank you!