Cloud Computing & Big Data

PARALLEL & SCALABLE MACHINE LEARNING & DEEP LEARNING

Dr. – Ing. Gabriele Cavallaro

Deputy Research Group Leader, Juelich Supercomputing Centre, Forschungszentrum Juelich, Germany

@Morris Riedel @MorrisRiedel @MorrisRiedel

LECTURE 6

Deep Learning driven by Big Data

October 20, 2020
Online Lecture

EUROPEAN OPEN SCIENCE CLOUD EOSC NORDIC EuroHPC Joint Undertaking ADMIRE EURO

UNIVERSITY OF ICELAND
SCHOOL OF ENGINEERING AND NATURAL SCIENCES
FACULTY OF INDUSTRIAL ENGINEERING, MECHANICAL ENGINEERING AND COMPUTER SCIENCE

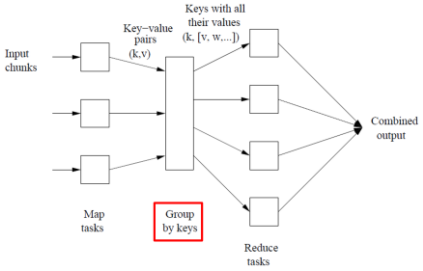
JÜLICH Forschungszentrum | JÜLICH SUPERCOMPUTING CENTRE

DEEP Projects HELMHOLTZ AI | ARTIFICIAL INTELLIGENCE COOPERATION UNIT

REVIEW OF PRACTICAL LECTURE 5.1

Understanding Map-Reduce in Cloud Applications

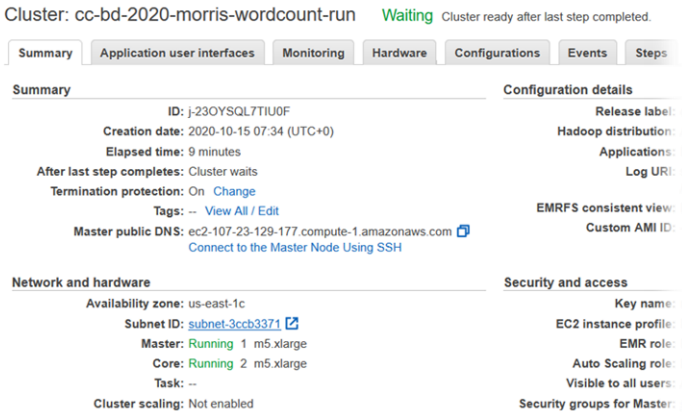
- Map-Reduce Approach

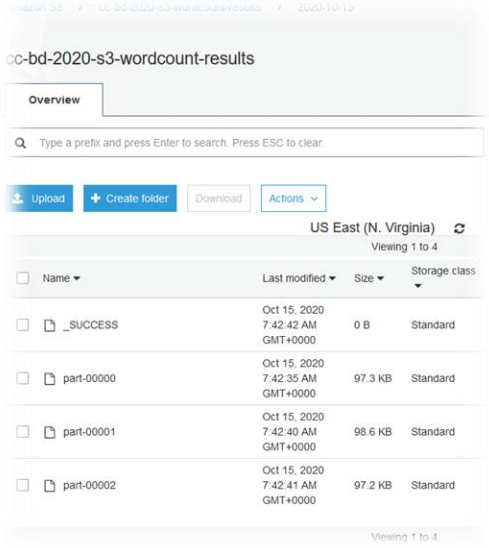


(origin in traditional computer & computational sciences)

HTC

(group/shuffle/sort done by the framework)





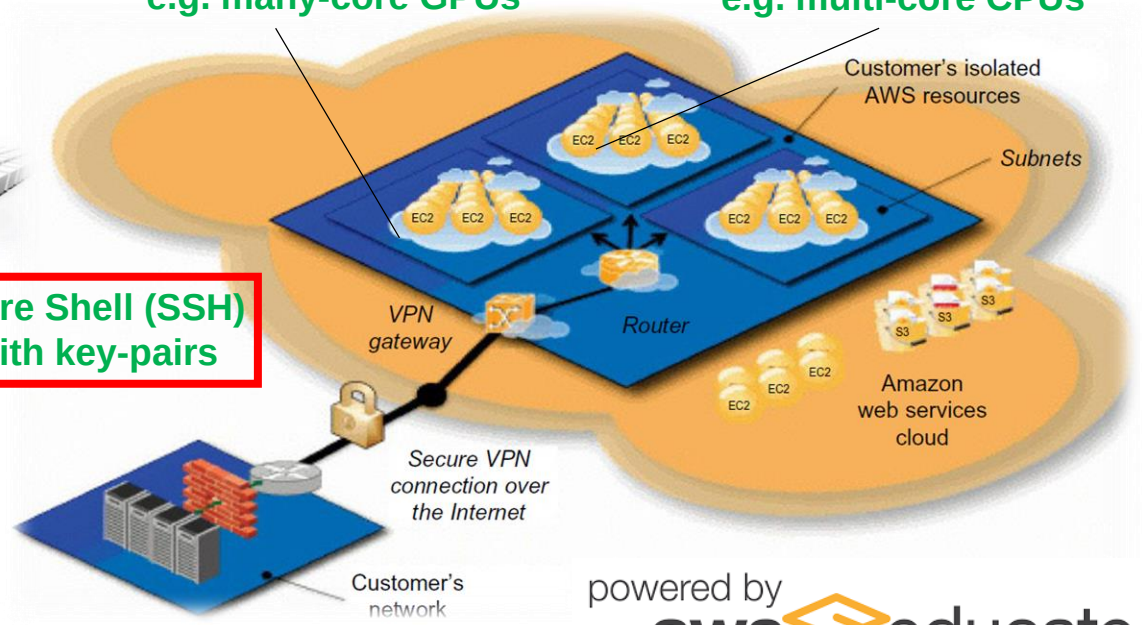
[2] Apache Hadoop

[9] Jupyter

e.g. many-core GPUs

e.g. multi-core CPUs

e.g. Secure Shell (SSH) access with key-pairs



powered by aws educate

[8] AWS Educate Web page

HDInsight

Google Cloud

Microsoft Azure

Google

[3] MapReduce: Simplified Data Processing on Large Clusters, 2004

Modified from [1] Mining of Massive Datasets

[7] Distributed & Cloud Computing Book

[4] Google Dataproc service

[5] Microsoft Azure HDInsight Service

[6] AWS EMR Web page

OUTLINE OF THE COURSE

1. Cloud Computing & Big Data Introduction
2. Machine Learning Models in Clouds
3. Apache Spark for Cloud Applications
4. Virtualization & Data Center Design
5. Map-Reduce Computing Paradigm
6. Deep Learning driven by Big Data
7. Deep Learning Applications in Clouds
8. Infrastructure-As-A-Service (IAAS)
9. Platform-As-A-Service (PAAS)
10. Software-As-A-Service (SAAS)
11. Big Data Analytics & Cloud Data Mining
12. Docker & Container Management
13. OpenStack Cloud Operating System
14. Online Social Networking & Graph Databases
15. Big Data Streaming Tools & Applications
16. Epilogue

+ additional practical lectures & Webinars for our hands-on assignments in context

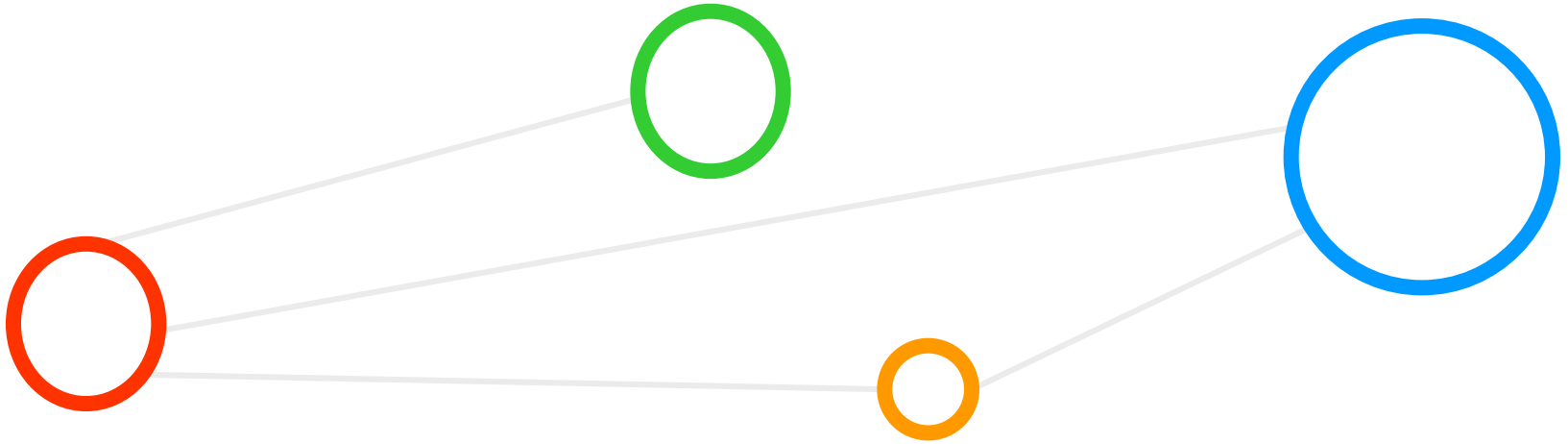
- Practical Topics
- Theoretical / Conceptual Topics

OUTLINE

- Machine Learning Fundamentals Revisited
  - Linear Perceptron Model
  - Limitations in non-linearly separable data sets
- Multilayer Perceptron
  - Neural Networks
- Deep learning
  - Deep Neural Networks
  - Convolutional Neural Networks (CNNs)
  - Backpropagation Algorithm
- Big Remote Sensing Data

- **Promises from previous lecture(s):**
- ***Practical Lecture 0.1:*** **Lecture 6 & 7 will provide more insights into deep learning algorithms and networks including the use of TensorFlow and Keras libraries**
- ***Practical Lecture 0.1:*** **Lecture 6 & 7 will provide more details on how artificial neural networks (ANNs) and deep learning networks can be used with this data**
- ***Lecture 2:*** **Lectures 6 & 7 offer more details on feature selection concepts including working with spatial aspects in image recognition tasks**
- ***Lecture 3:*** **Lecture 6 & 7 offer insights of how to use deep learning with cutting-edge GPUs via Google 'colab' notebooks within the Google Cloud**

MACHINE LEARNING FUNDAMENTALS REVISITED



MACHINE LEARNING

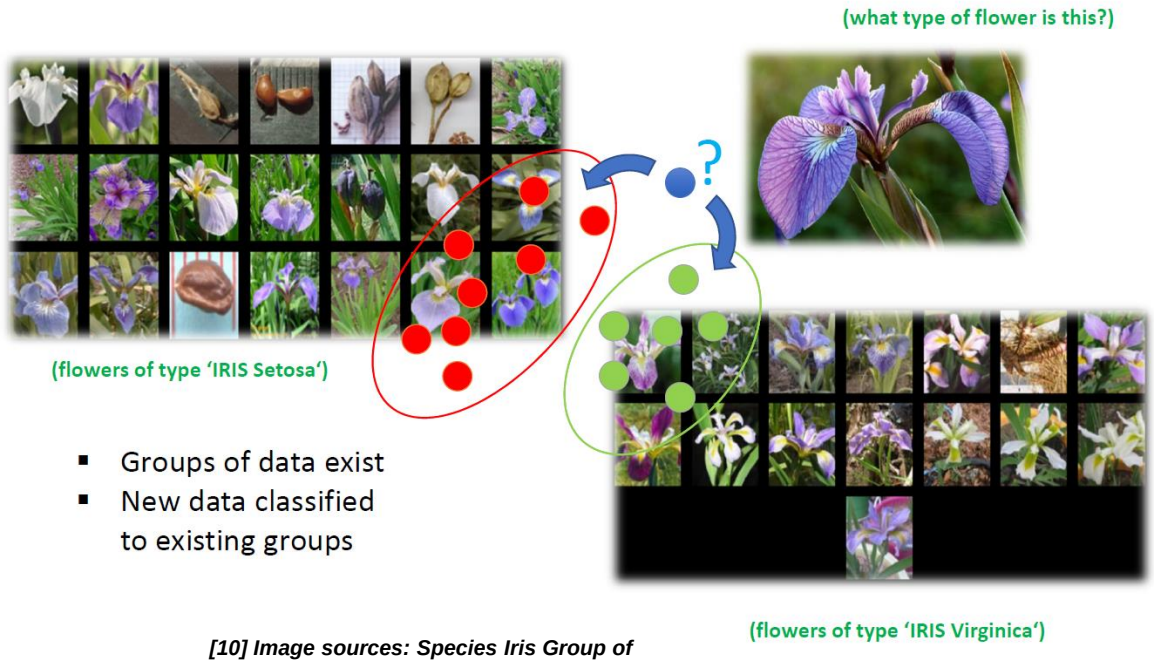
The domain

- Involves tasks for which there is **no known direct method** to compute a desired output from a set of inputs

- The strategy adopted is for the computer to **“learn” from** a set of representative **examples**

- The goal of machine learning:
  - Learn **patterns** from **examples**
  - Be able to **generalize** them to new examples

- Prerequisites:
  - **Some pattern exists**
  - **No exact mathematical formula**
  - **Data exists**



[10] Image sources: Species Iris Group of North America Database, www.signa.org

PERCEPTRON

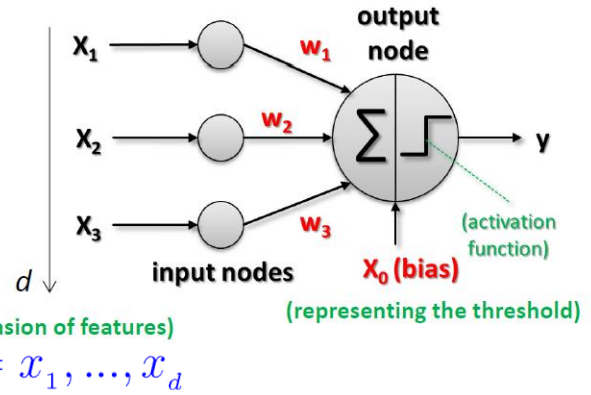
Revisited

- Human analogy in learning (brain: **neurons** and **strength of their connections** w_i)
- **Training** a perceptron model means **adapting the weights** w_i
  - Done until they fit input-output relationships of the given '**training data**'

$$(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)$$

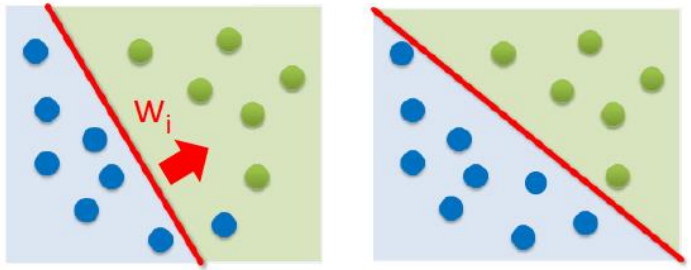
(training data)

[11] F. Rosenblatt, 1957



$$\mathbf{x} = x_1, ..., x_d$$

- When solving a **linear classification** problem (i.e., linearly separable data)
  - Goal: learn a simple value (+1/-1) above/below a certain threshold
  - Class label renamed: Irisi-setosa = -1 and Iris-virginica = +1
  - Decision boundary: perpendicular vector w_i fixes orientation of the line



$$h(\mathbf{x}) = sign(\mathbf{w}^T\mathbf{x})$$

[12] F. Rosenblatt, 1958

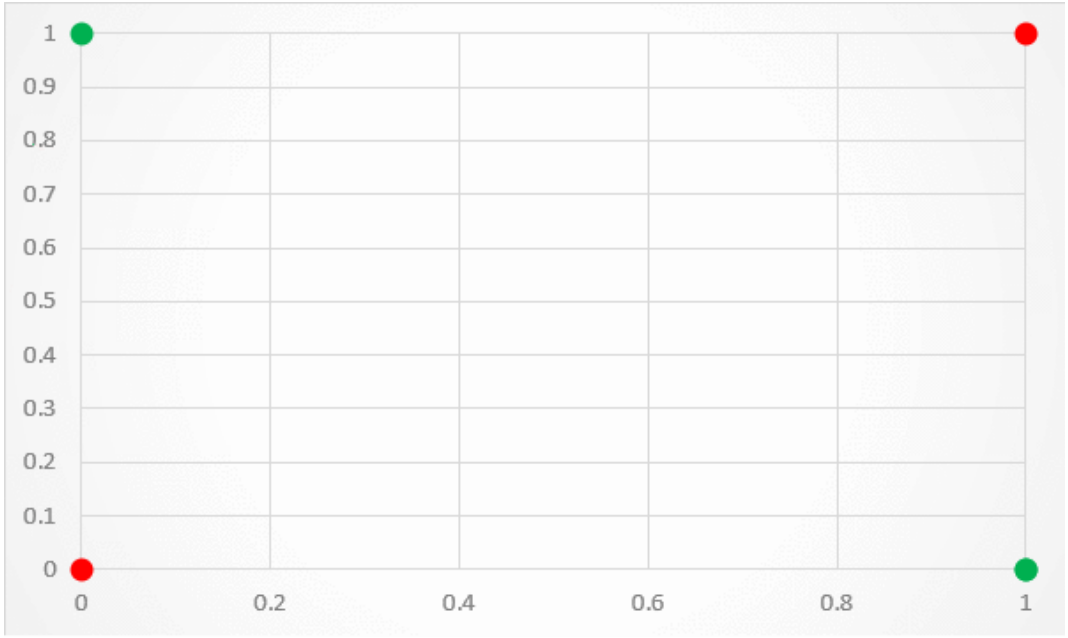
PERCEPTRON

XOR Problem

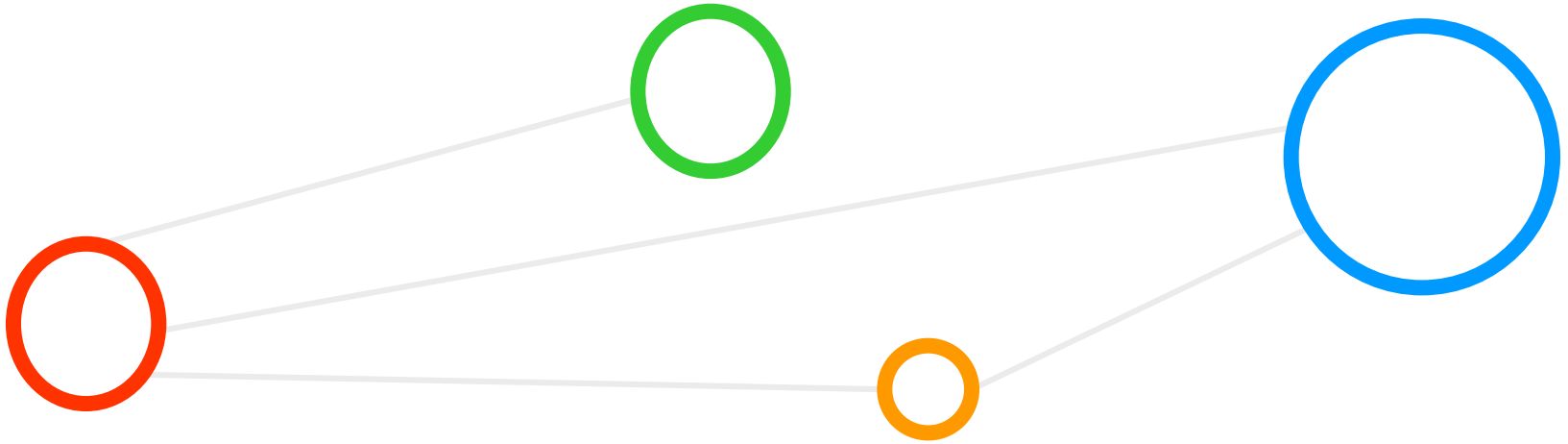
- Perceptron is only capable of separating data points with a single line
- XOR inputs are not linearly separable
  - There is no way to separate the 1 and 0 predictions with a single classification line

Input 1	Input 2	Output
0	0	0
0	1	1
1	1	0
1	0	1

[13] The XOR Problem



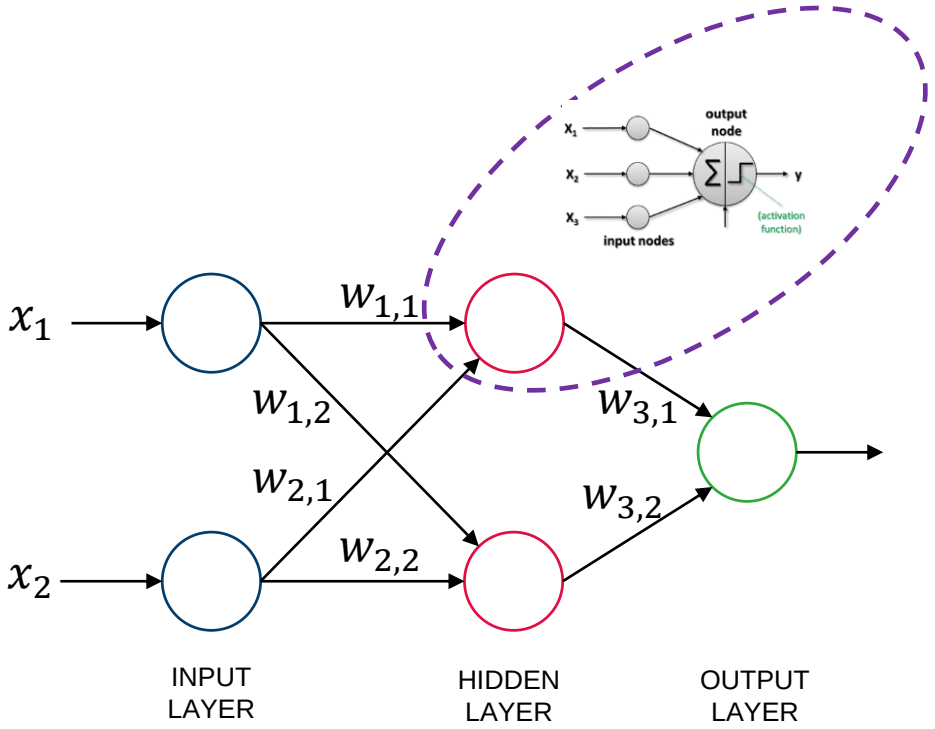
MULTILAYER PERCEPTRON



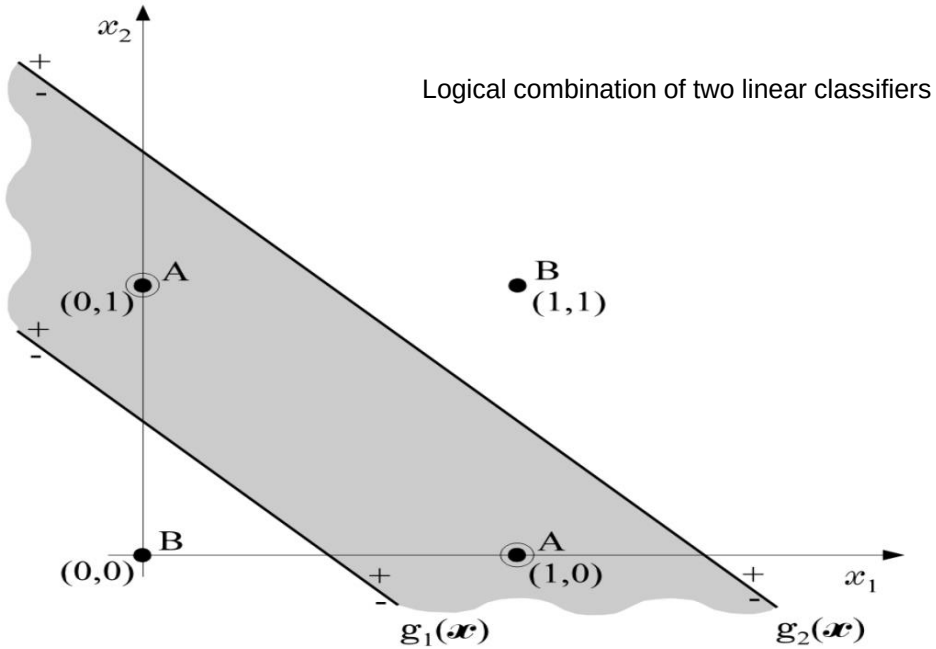
MULTILAYER PERCEPTRON (MLP)

XOR Problem

- The XOR problem can be solved by a **three layers** network
  - Hidden layer: additional layer of units without any direct access to the outside world



Logical combination of two linear classifiers

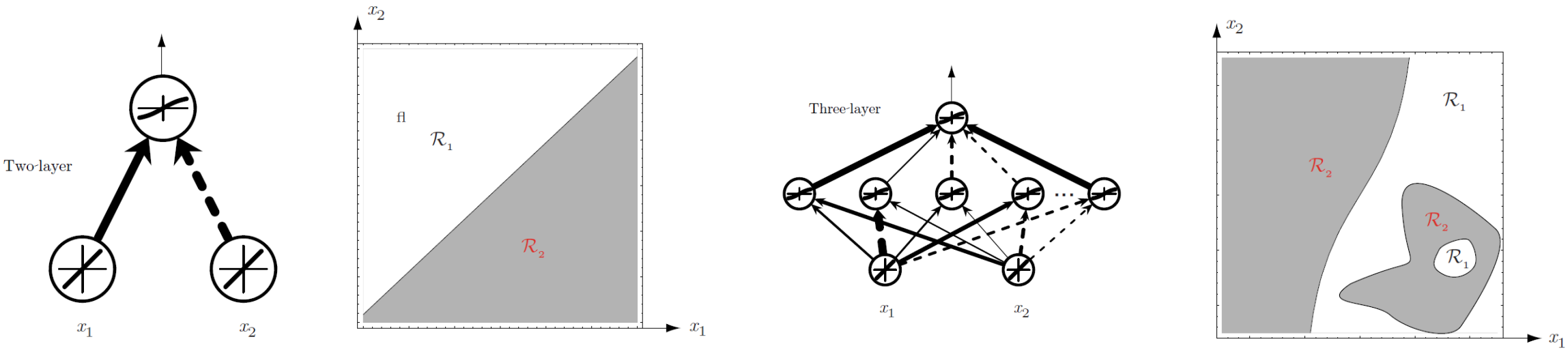


[14] Multilayer Perceptron

MULTILAYER PERCEPTRON (MLP)

Decision Boundaries

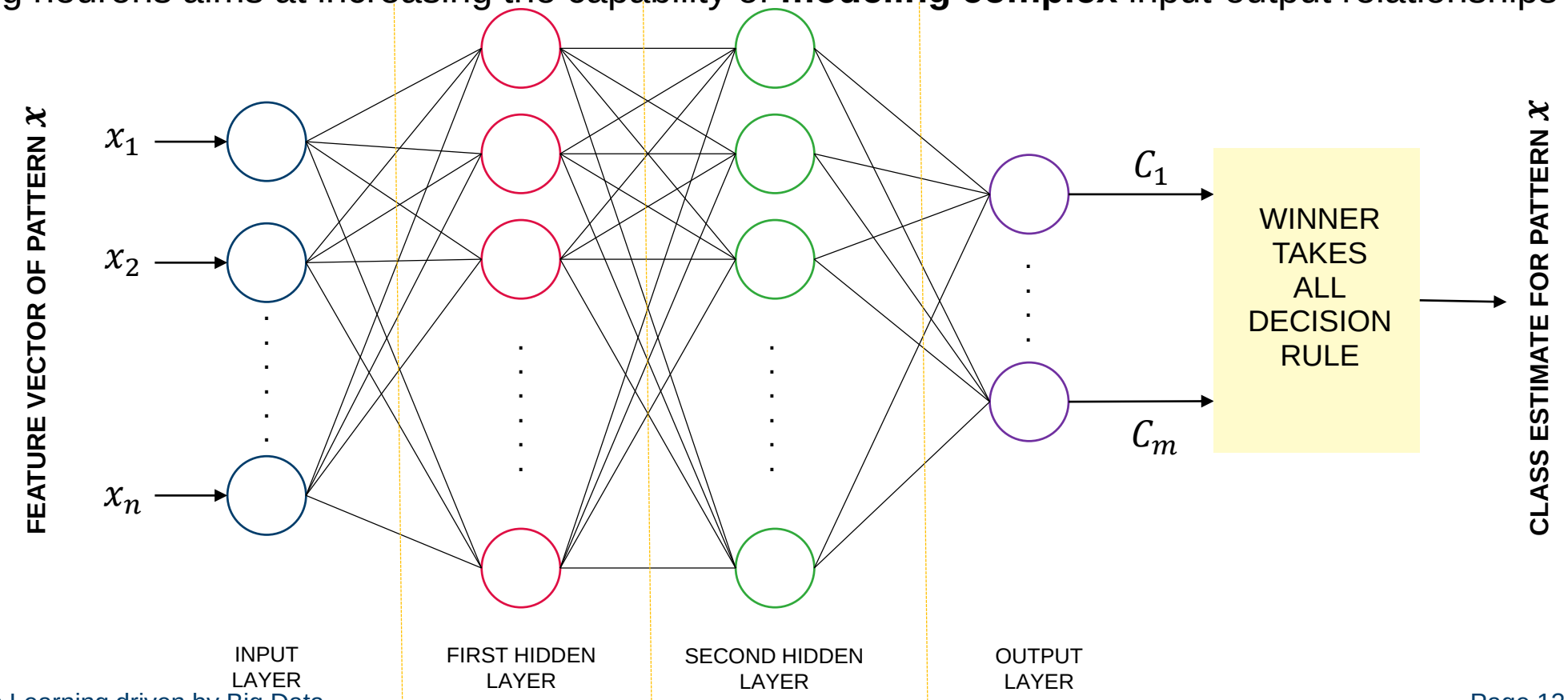
- A **two-layer** network classifier can only implement a **linear decision boundary**
- With a sufficient **number of hidden units**, networks can implement **arbitrary decision boundaries**
  - The decision regions need not be convex, nor simply connected



[15] R. O. Duda et al.

MULTILAYER PERCEPTRON (MLP) NEURAL NETWORK

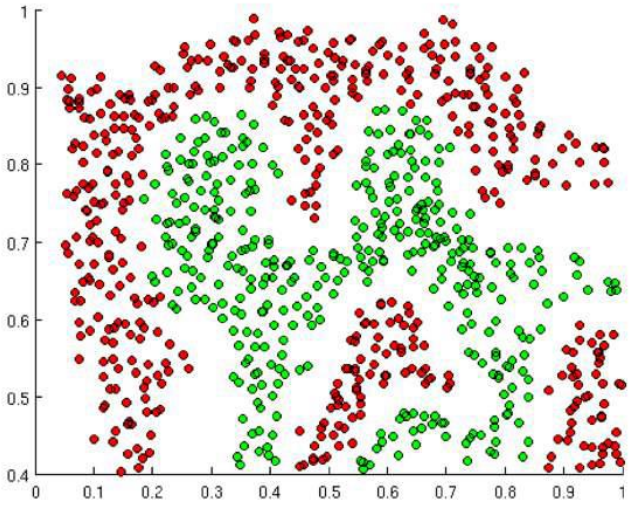
- **Forward interconnection** of several layers of perceptrons
- MLPs can be used as **universal approximators**
- In classification problems, they allow modeling **nonlinear discriminant functions**
- Interconnecting neurons aims at increasing the capability of **modeling complex** input-output relationships

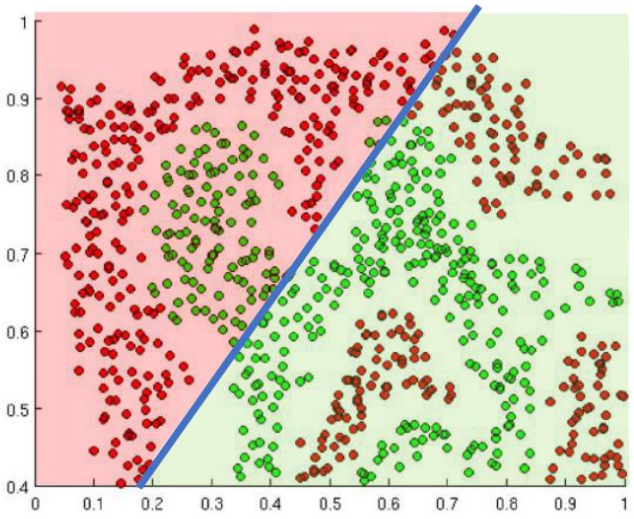


ACTIVATION FUNCTIONS

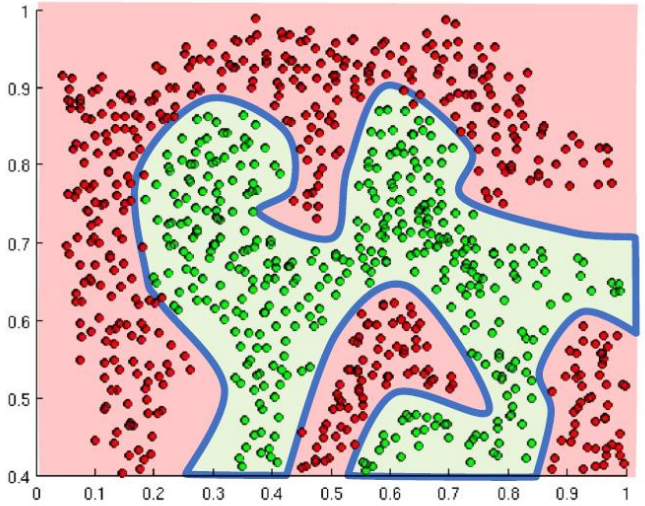
Introduce non-linearities into the network

- E.g., How to build a Neural Network to distinguish green vs red points?





Linear Activation functions produce linear decisions no matter the network size

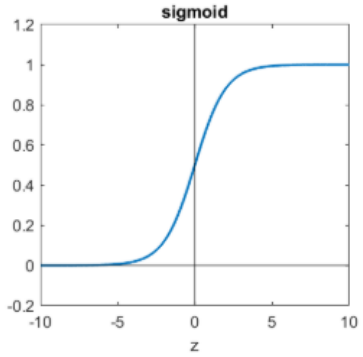


Non-linearities allow us to approximate arbitrarily complex functions

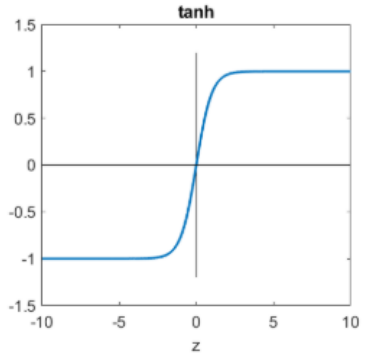
ACTIVATION FUNCTIONS

Non-linear transformation

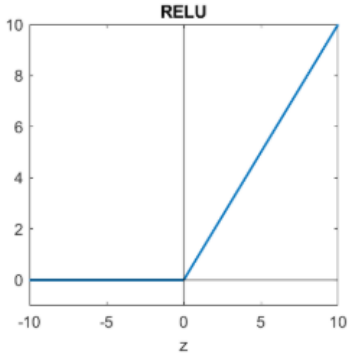
- The choice of the architecture and the **activation function** plays a key role in the definition of the network
- Each activation function takes a single number and performs a certain fixed mathematical operation on it



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

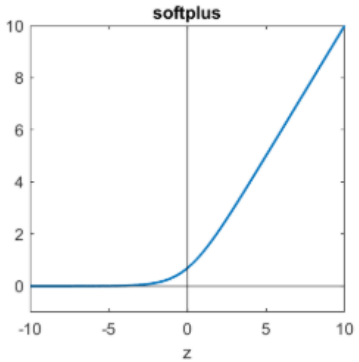


$$h(z) = \tanh z$$

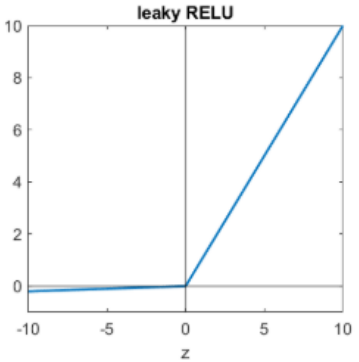


$$h(z) = \max(z, 0)$$

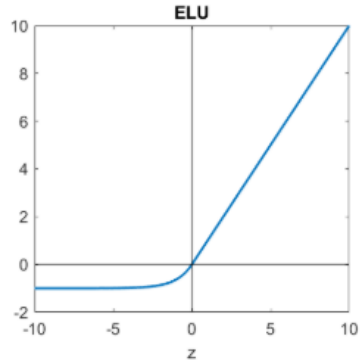
[16] Understanding the Neural Network



$$h(z) = \log(1 + e^{z})$$



$$h(z) = \max(z, z\alpha)$$
$$0 < \alpha < 1$$

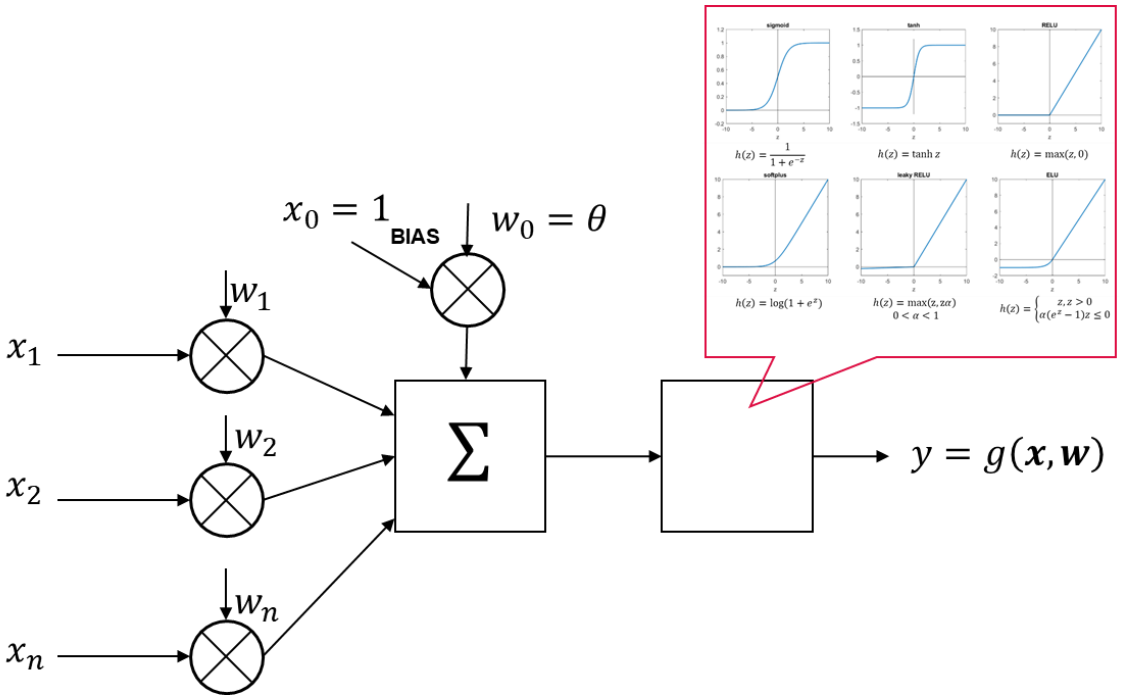


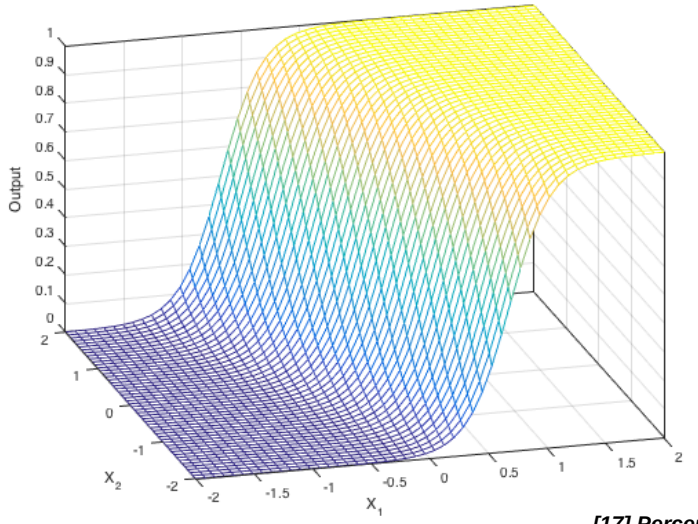
$$h(z) = \begin{cases} z, z > 0 \\ \alpha(e^{z} - 1) z \leq 0 \end{cases}$$

PERCEPTRON WITH NON LINEAR ACTIVATION FUNCTIONS

Can perform nonlinear classification?

- Perceptron **can't perform nonlinear** classification **regardless** of the choice of **activation function**
  - The input is projected onto the weight vector and scaled/shifted along this direction
  - This is a linear operation that reduces the input to a single value
  - Which is then passed through the (possibly nonlinear) activation function





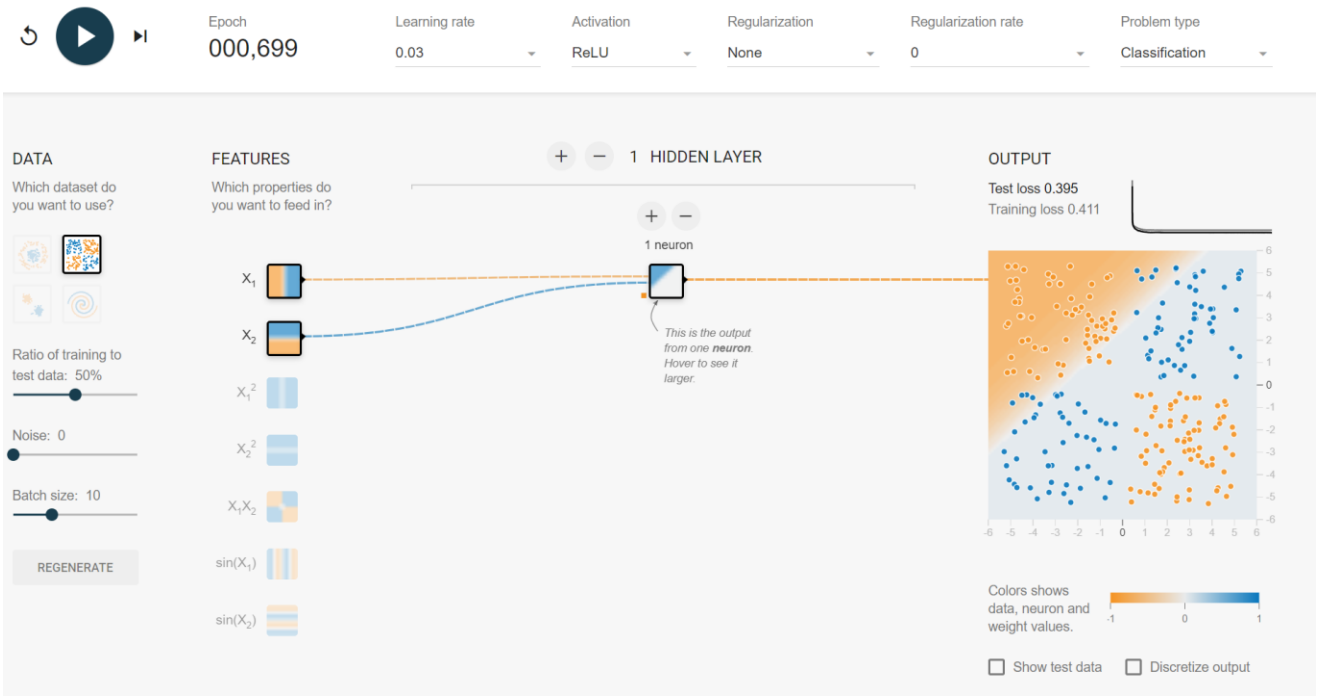
[17] Perceptron with sigmoid

- E.g., Perceptron with a logistic sigmoid
- Function: surface bent into a sigmoidal shape along the direction of the weight vector
- Changing the weights can rotate the direction of the sigmoidal surface, and stretch or shift it
- But, the fundamental sigmoidal shape will always remain

LIVE DEMONSTRATION

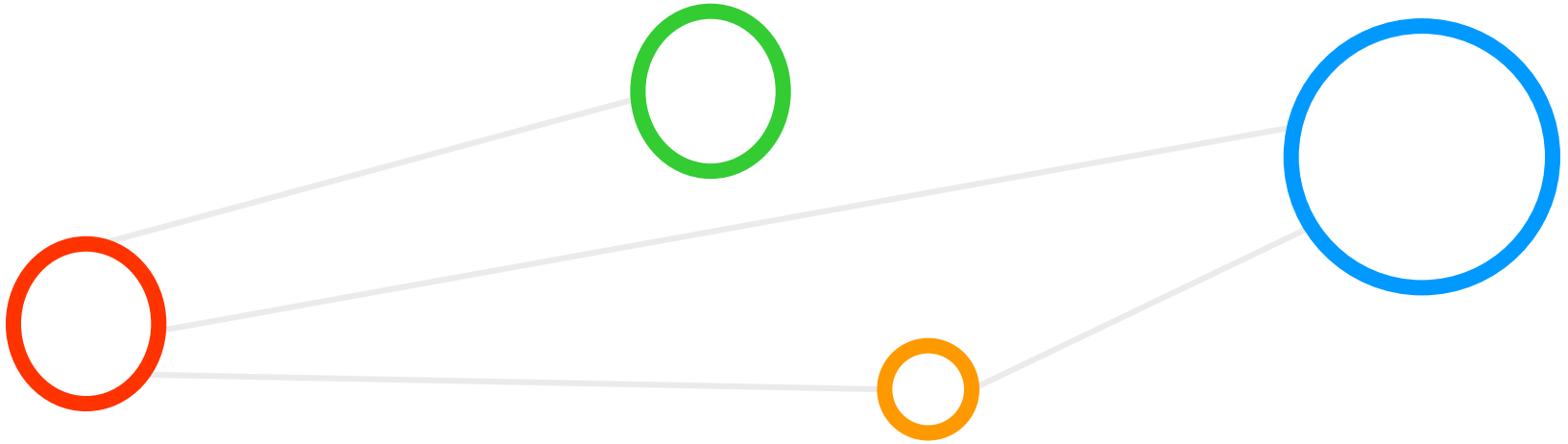
Website

- Illustration of connection between neural network architecture, hyperparameters, and dataset characteristics
- Explore this connection yourself at: https://playground.tensorflow.org/



[18] A Neural Network Playground - TensorFlow

DEEP LEARNING



CLASSIFICATION OF FLOWERS

Revisited

- What **attributes (features)** about the data help?

(flowers of type 'IRIS Setosa')

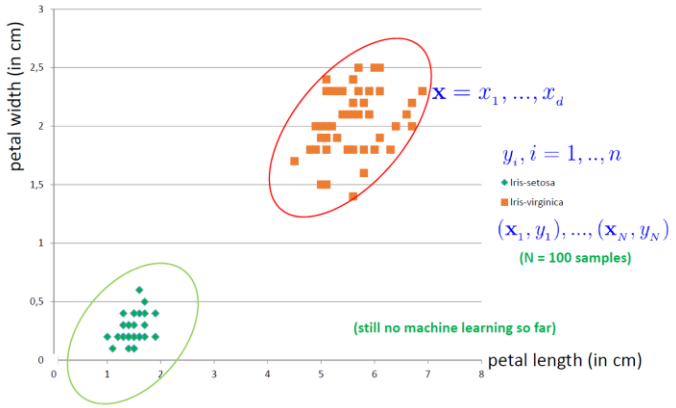
(flowers of type 'IRIS Virginica')

[10] Image sources: Species Iris Group of North America Database, www.signa.org

[19] Wikipedia 'Sepal'

(four data attributes for each sample in the dataset)
- sepal length in cm
- sepal width in cm
- petal length in cm
- petal width in cm

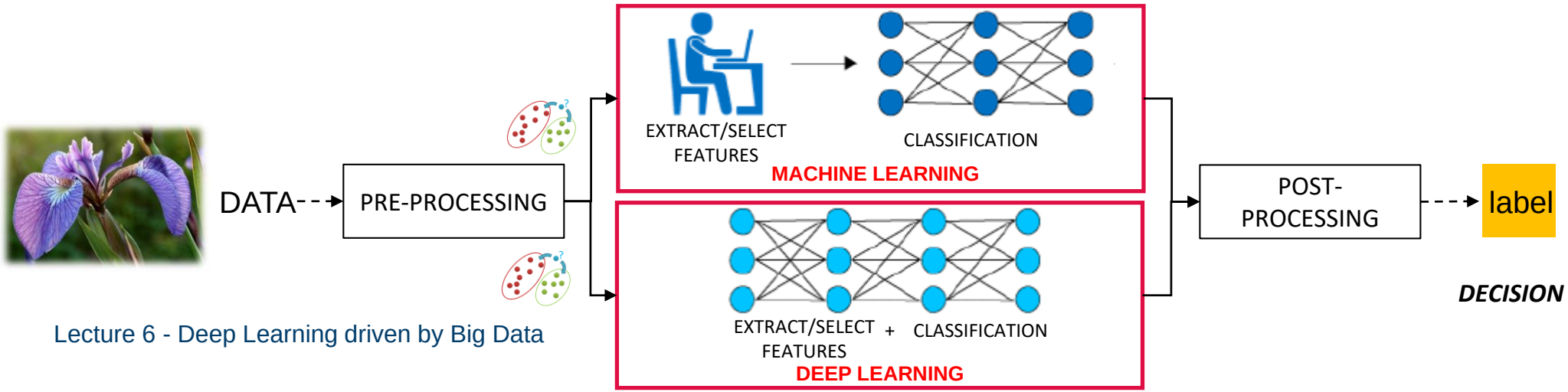
(one class label for each sample in the dataset)
- class: Iris Setosa, or Iris Versicolour, or Iris Virginica



CLASSIFICATION

General pipeline

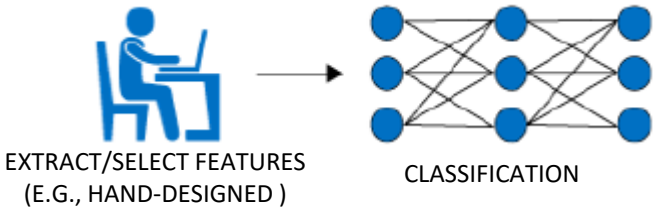
- **Pre-processing**
  - Standardize data, rescaling, etc.
- **Feature extraction/selection**
  - Extraction of information parameters
  - Selection of information parameters
- **Classification**
  - Based on the information parameters previously extracted and selected



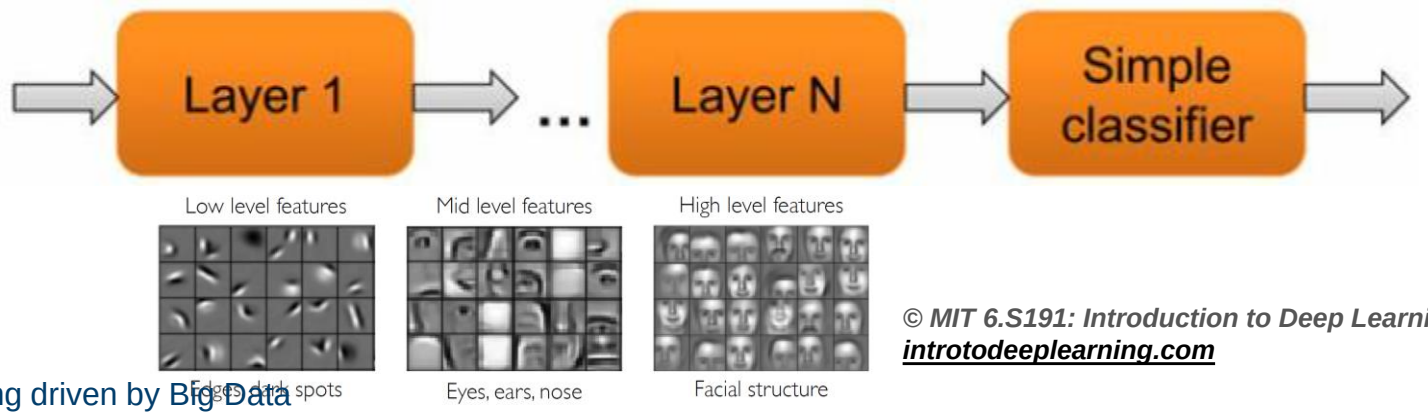
DEEP LEARNING AND SHALLOW LEARNING

Can we learn a hierarchy of features directly from the data instead of hand engineering?

- **Shallow learning**: learning networks that usually have at most one to two layers
  - They compute linear or nonlinear functions of the data (**often hand-designed features**)



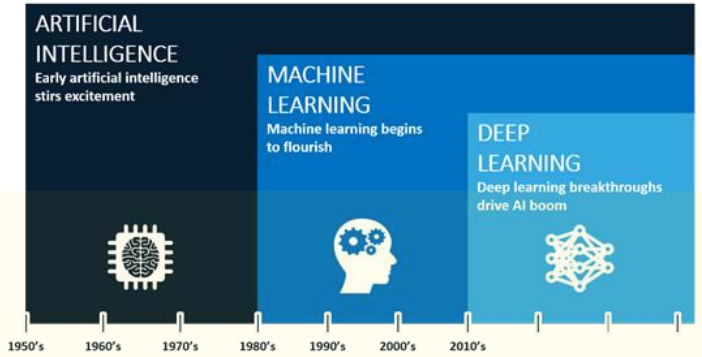
- **Deep learning:** means a deeper network with many layers of non-linear transformations
  - No universally accepted definition of how many layers constitute a “deep” learner



© MIT 6.S191: Introduction to Deep Learning
introtodeeplearning.com

DEEP LEARNING

Emerging as the leading AI technique



AI: Intelligence demonstrated by machines rather than humans or animals.

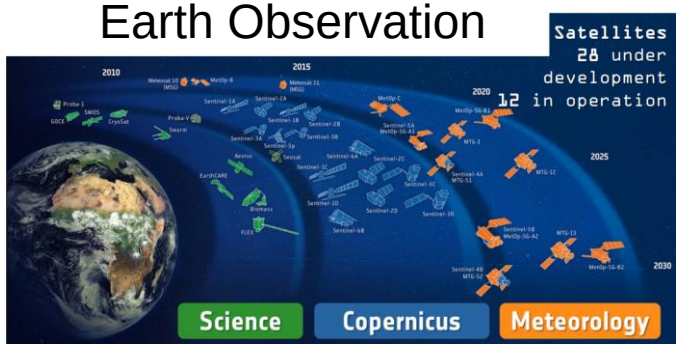
ML: Giving computers the skills to learn without explicit programming

DL: Is an ML subset, examining algorithms that learn and improve on their own.

[20] AI, ML and DL

- Current convergence of **scalable computing capability**
- Easy access to **large volumes of data**
- Emergence of new algorithms enabling robust training of large-scale **deep neural networks**

Earth Observation

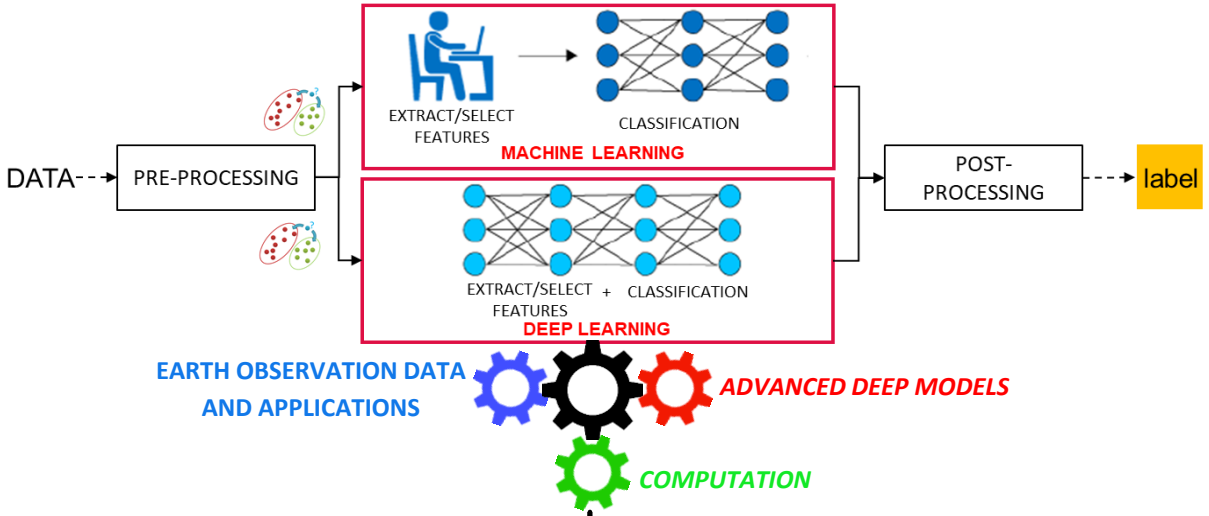


[21] Gordon Campbell

Continuous developments of RS platforms and sensor technologies

Unprecedented large **volume** and **variety** of raw data

Free data policy of Earth Observation programs



Desktop Computers

[22] Desktop PC

Cloud computing
High Throughput Computing (HTC)

[23] The Benefits of Cloud Computing

High Performance Computing (HPC)
Supercomputing
Exascale computing

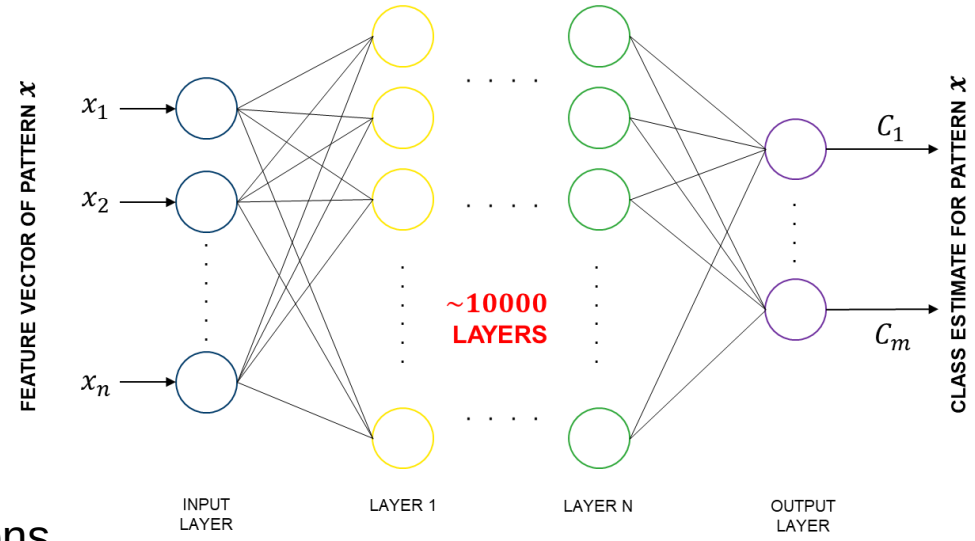
[24] JUWELS

Quantum Computing

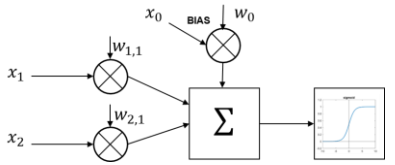
[25] D-Wave systems

DEEP LEARNING

- **Deep Learning**: using a generic, flexible model family
  - **“Neural” Networks** with multiple layers
  - Based on stacked, repetitive operations
  - Executed by simple, generic units
  - With parameters (“weights”) adapted from incoming data



- **Deep Neural Networks**
  - Most models can be cooked down to stacked repetitive operations
  - They are executed by simple units
  - **Linear summation + non-linear transfer function**



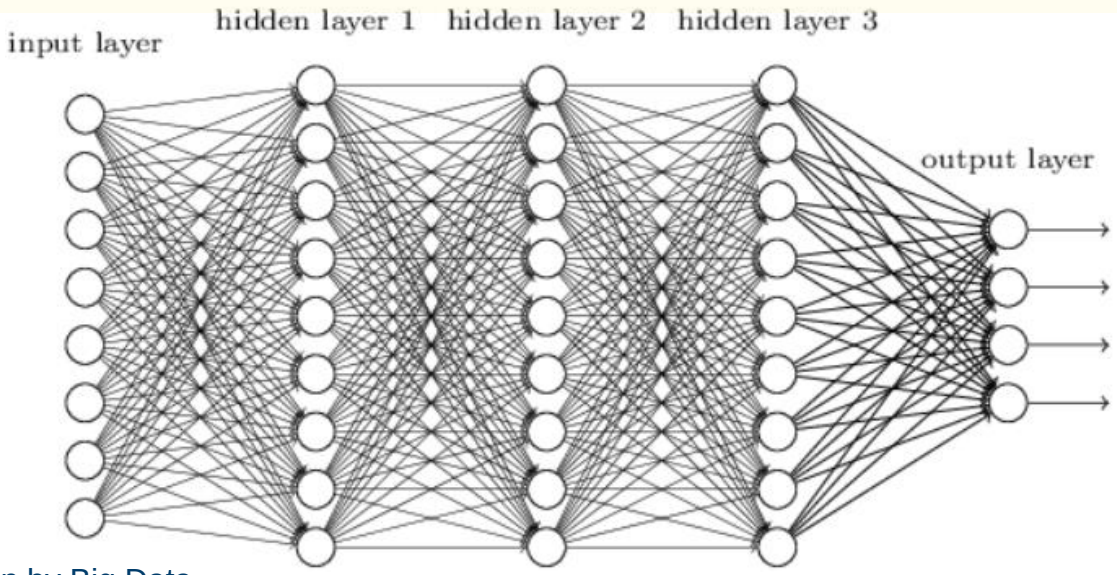
- The units are very simple
- The complexity arise when combining more of them
- The are many weights (connections) that are adjustable from the data (they are not fixed)

FULLY CONNECTED NEURAL NETWORK (FCNN)

Architecture Overview

- Receives an input a **single vector** and transforms it through a series of **hidden layers**
- Each hidden layer is made up of a set of **neurons** that have **learnable weights** and **biases**
  - Each neuron receives some inputs, performs a dot product and **follows it with a non-linearity**
  - Each neuron is **fully connected** to all neurons in the previous layer
  - Neurons in a single layer function completely independently and **do not share any connections**
- The last fully-connected layer is the “output layer” and in classification settings it represents the class scores

Bias: additional set of weights that require no input, and this it corresponds to the output of the network when it has zero input (not tied to any previous layer)



Universal approximator
Provide posteriors in output
Numerous training algorithm available
Training computationally demanding

HOW TO USE SPATIAL STRUCTURE IN THE INPUT

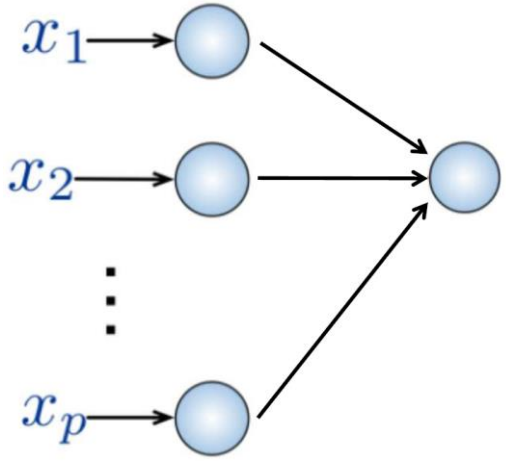
To inform the architecture of the network?

- FCNN don’t scale well to full images
  - Moreover, when adding several neurons, the **parameters grows quickly**
  - The fully connectivity is wasteful, and the huge number of parameters can lead to **overfitting**

Input:

- 2D image

- Transformed into a vector of pixel values



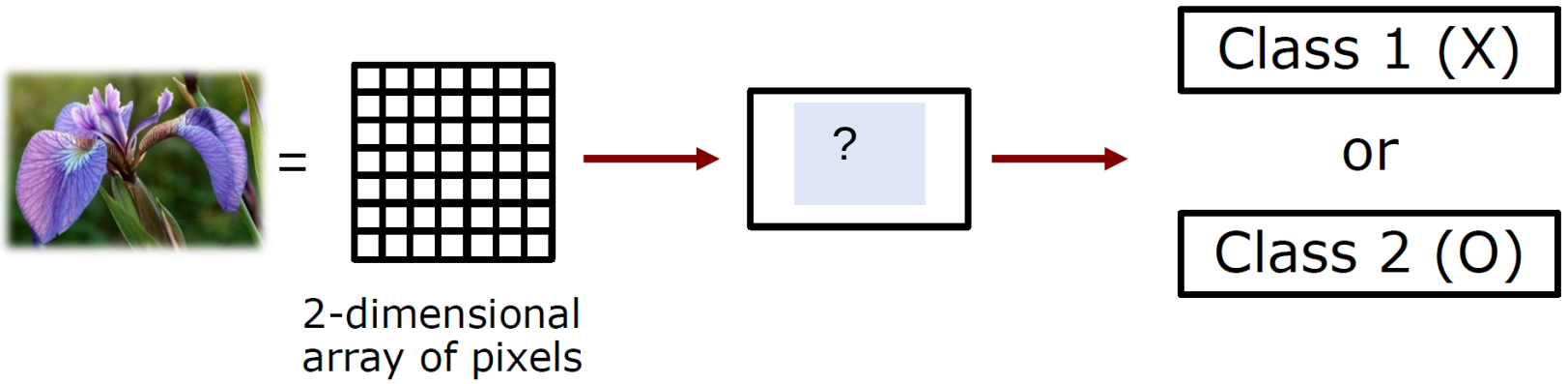
Fully connected:

- Connect neuron in hidden layer to all neurons in input layer
- **No spatial information**
- **Large number of parameters**

INPUT

Spatial Structures

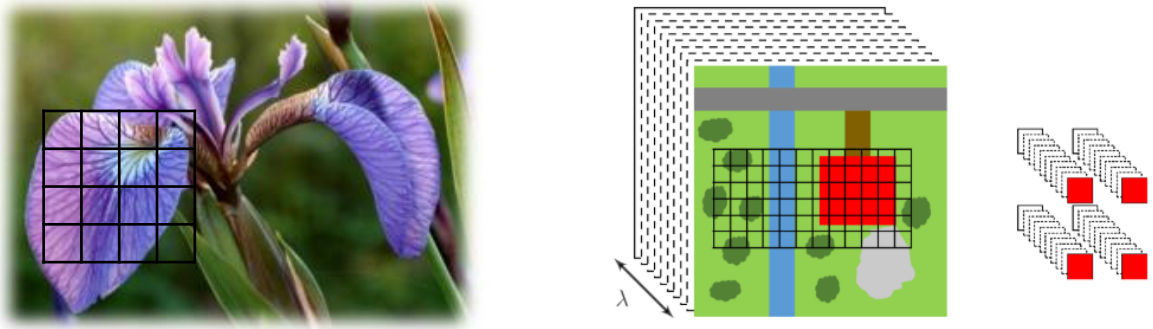
- Need of an **architecture** that makes the explicit assumption that the inputs are **images**
- The whole network still expresses a **single differentiable loss function**
  - From the raw image pixels on one end to class scores at the other



LOCALITY

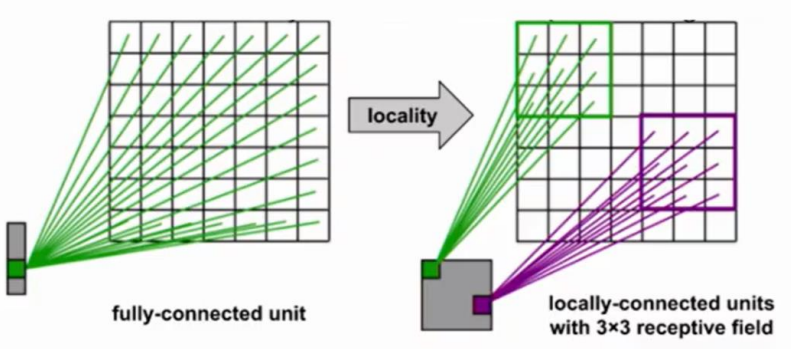
Incorporating the assumption

- **Locality**: objects tend to have a local spatial support
  - Can be defined since an image lies on a grid/lattice



Exploit the neighboring relationship

- How to define an architecture which exploit this property?
  - Make **fully**-connected layer **locally**-connected
  - Each neuron is connected to a local area (i.e., **receptive field**)
  - Different neurons connected to different locations

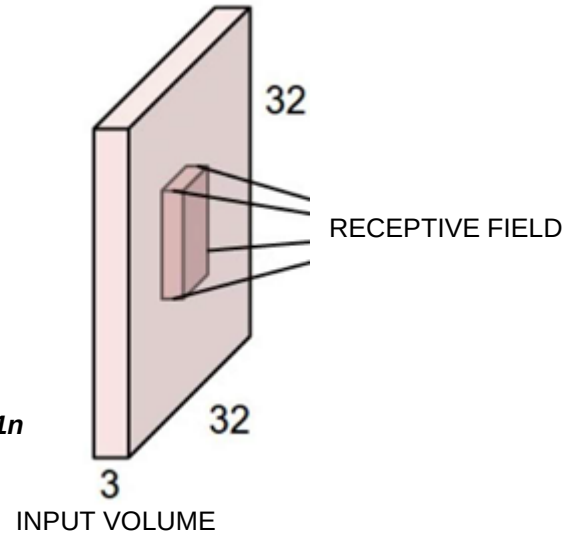


[26] Deep Learning: Practices and Trends

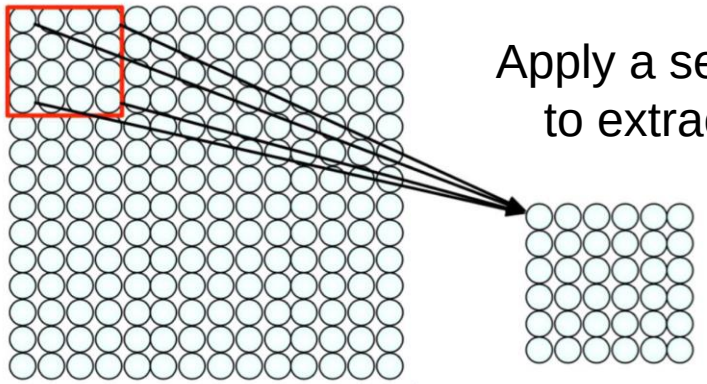
LOCALITY

Incorporating the assumption

- With **images**: not efficient to connect neurons to all neurons in the previous layer
- It is better to connect each neuron to only a **local region** of the input volume
  - Connect **patches** of input to neurons in hidden layer
- The spatial extent of this connectivity is a **hyperparameter** called the **receptive field** (filter size)



[27] CS231n



Apply a set of weights (**filter**) to extract **local features**

4x4 filter: matrix of weights w_{ij}

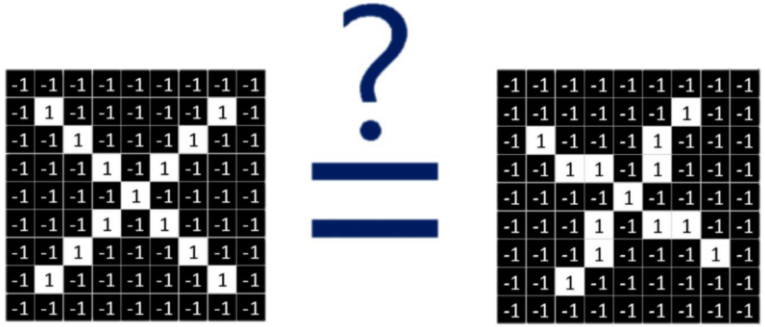
$$\sum_{i=1}^{4}\sum_{j=1}^{4} w_{ij}\, x_{i+p,j+q} + b$$

for neuron (p,q) in hidden layer

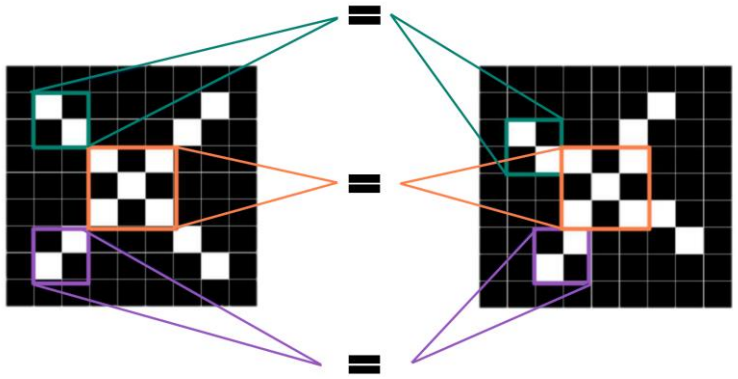
REVIEW THE CONVOLUTION

Example of classification task: classify X from a set of binary images

- For classification: not be possible to simply compare the two matrices and check if they are equal
  - The classier needs to classify an X as an X even if its **shifted**, **shrunk**, **rotated**, **deformed**, etc..

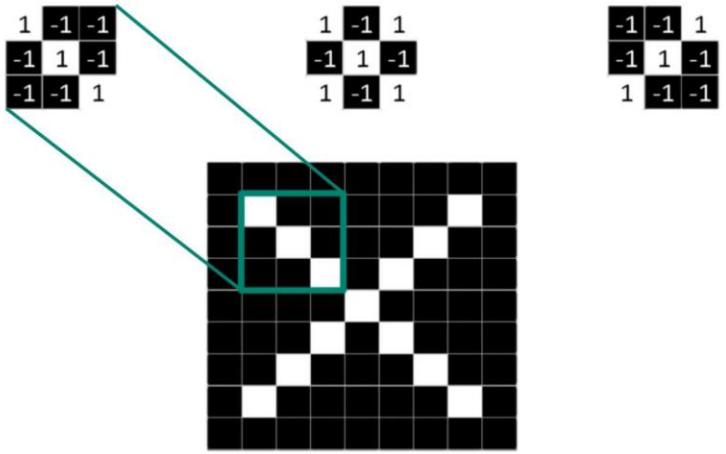


- **Effective approach**: compare the images **piece by piece**
  - Search the important parts that define an X as an X (the meaningful **features**)



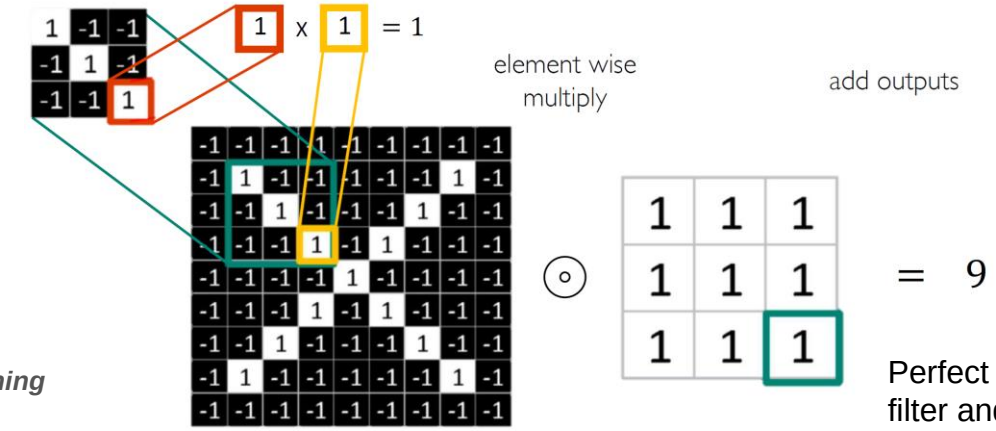
REVIEW THE CONVOLUTION

Filters to detect X features



- Think about each feature as small patches
- Use filters of weights for the convolutional operations
  - To detect the corresponding features

- Convolution preserves the spatial relationship between pixels
  - By learning image features in small squares of the input



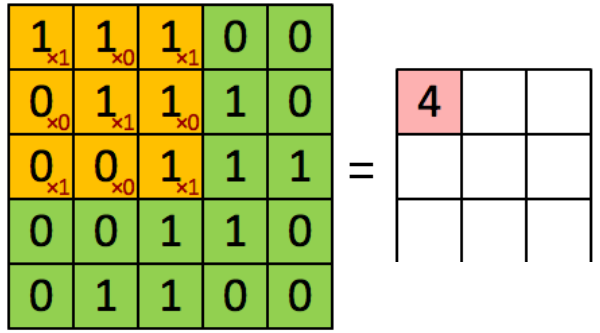
Perfect correspondence between the filter and the region of the input that is convolved

REVIEW THE CONVOLUTION

Producing feature maps

- **Feature map:** reflects where in the input was activated by the applied filter

- E.g., Slide the 3x3 filter over the input image,
  element-wise multiply, and add the outputs

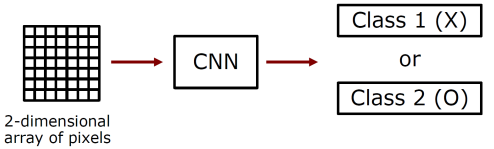


- Different filters can be used to produce different filter maps
  - E.g., With 3 different convolutional filters (i.e., different weights)

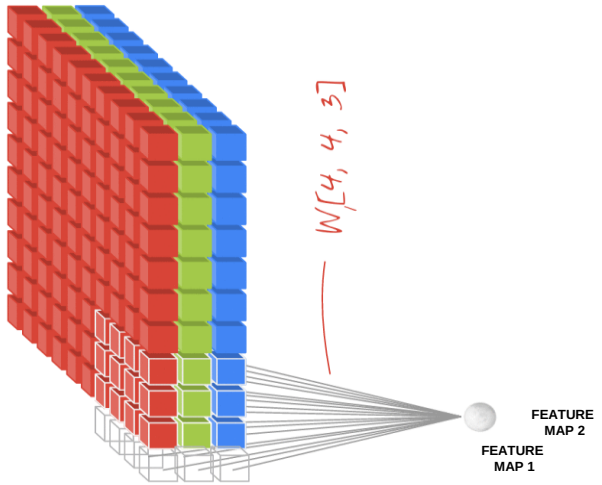
[27] CS231n

CONVOLUTIONAL LAYER

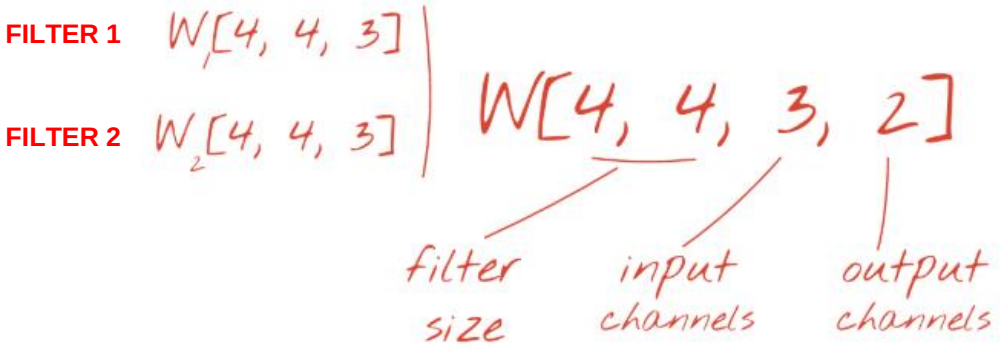
Within a Convolutional Neural Network (CNN)



- Within a single **convolutional layer** there are **N filters** (with different set of weights)
  - The **output** of a convolutional layer has a volume (i.e., **N feature maps**)
- Slide the patch of weights across the image in both directions (x,y)
  - With **padding**, output as many values as there were pixels in the image



[28] M. Görner et al.



Output: **3D = 2D x N** matrix

CONVOLUTIONAL LAYER

Locally-connected layer with weight sharing (translation invariance)

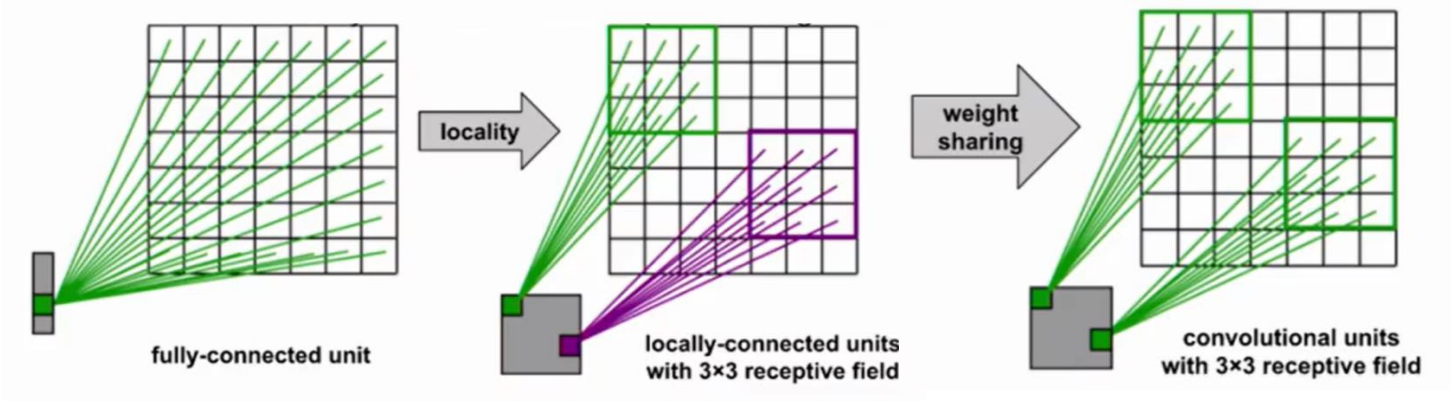
- **Locally-connected** layer: rarely used, beneficial only in specific cases
  - E.g., Face recognition: different features learned on one side of the image than another (eye/hair-specific features)
  - Not optimal when computing with GPUs

- **Translation invariance**: object appearance is independent of location

Objects look the same in different parts on an image

[29] Matt Krause, Invariance

- **Weight-sharing** layer
  - Neurons connected to different locations have the same weights (i.e., each neuron is applied to all locations)



[26] Deep Learning: Practices and Trends

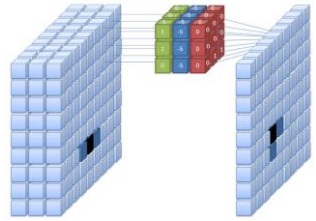
LOCALITY ASSUMPTION

TRANSLATION INVARIANCE ASSUMPTION

CONVOLUTIONAL NEURAL NETWORK (CNN)

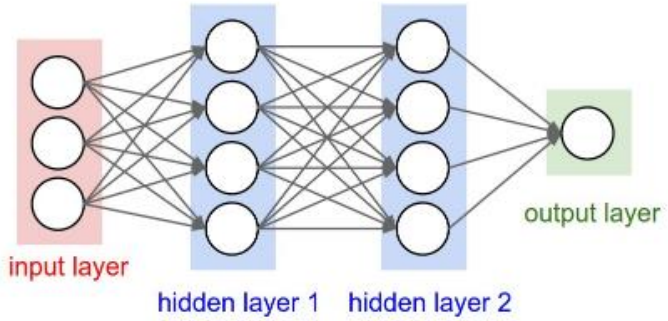
Three main operations

- Sequence of **layers** that transforms one volume of activations to another through a differentiable function
  - **Convolutional Layer**
    - Apply filters with learned weights to generate feature maps
  - **Pooling Layer**
    - Downsampling operation on each feature map
  - **Fully-Connected Layer**
    - Perform classification

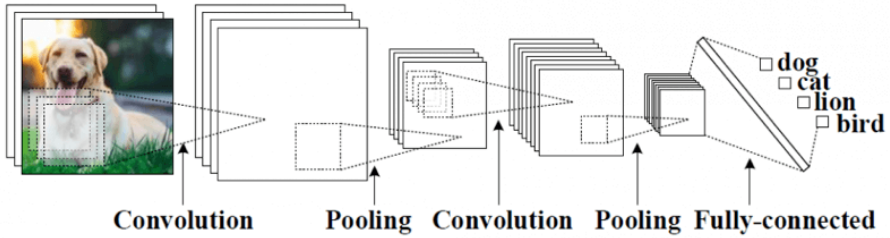


[30] Convolutional Neural Network

[21] CS231n



Regular 3-layer Neural Network



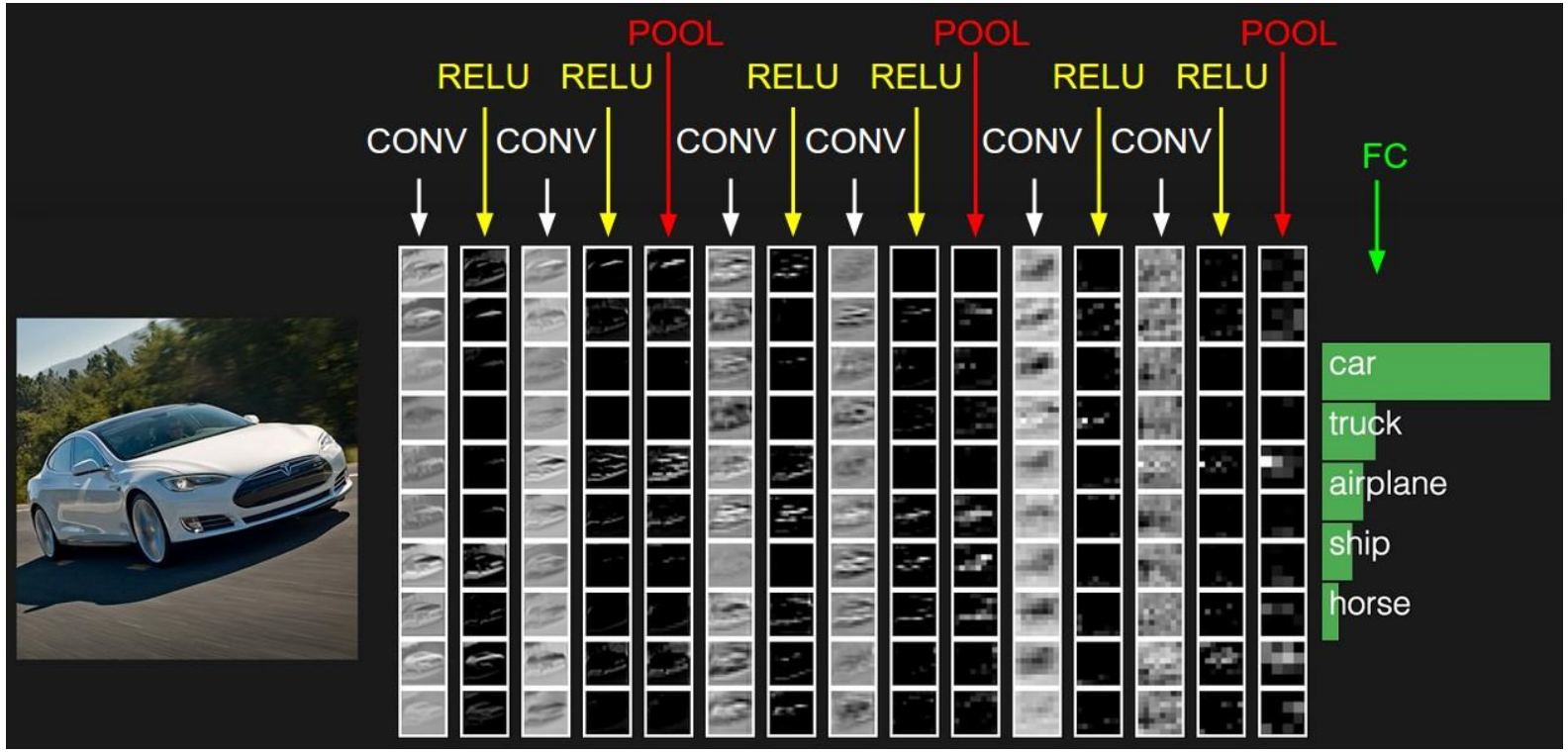
[31] Gunnar Carlsoon

Convolutional Neural Network

CONVOLUTIONAL NEURAL NETWORK (CNN)

Example Feature Maps

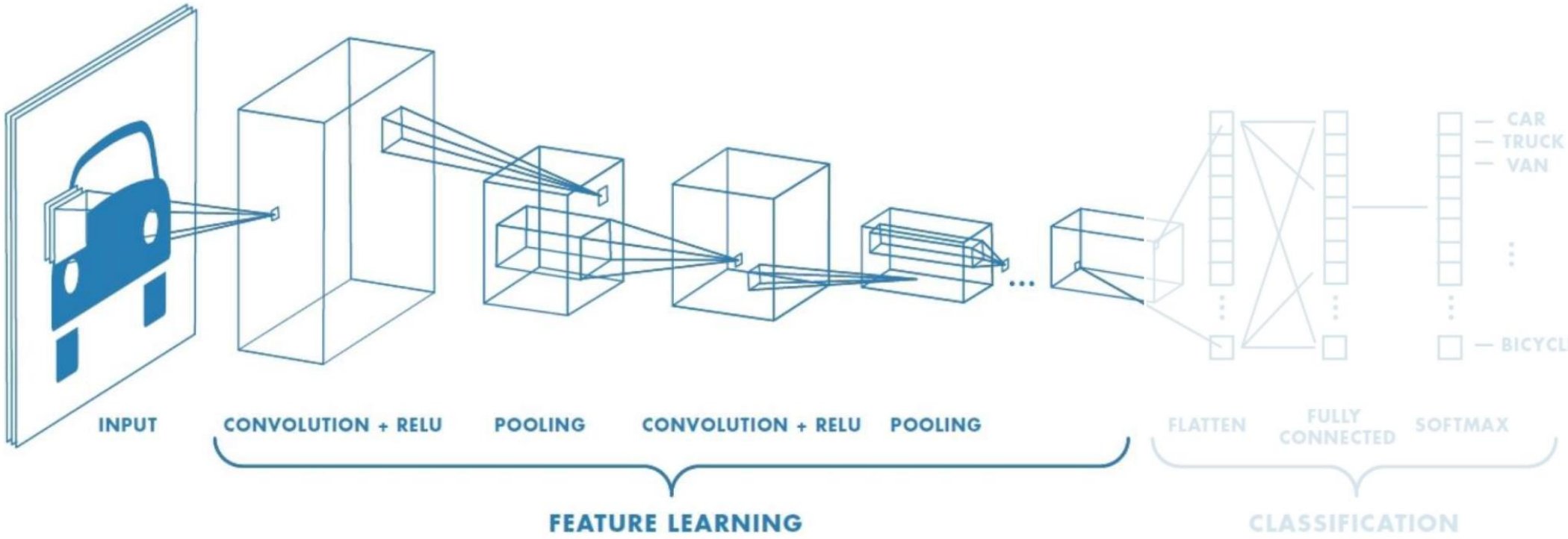
- The initial volume stores the raw image pixels
- Volume of **activations** shown as a column (i.e., lay out each volume's slices in rows)
- The last volume holds the scores for each class



[27] CS231n

CNN FOR CLASSIFICATION

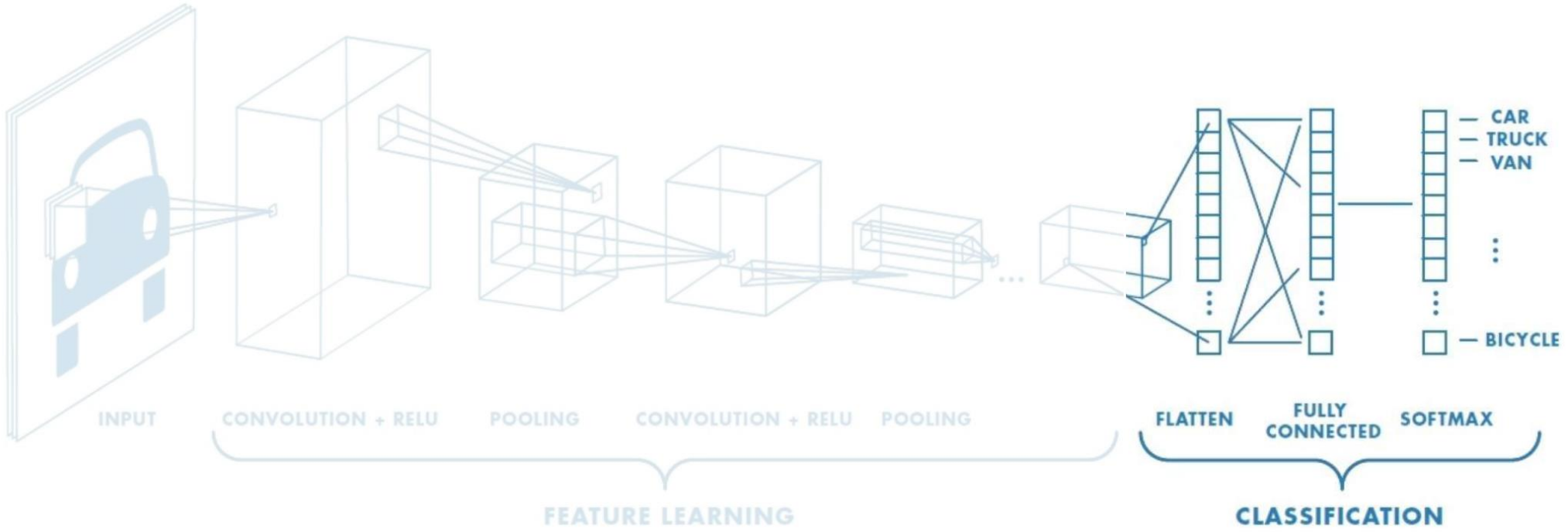
Feature learning pipeline



1. Learn features in input image through **convolution**
2. Introduce non-linearity through **activation function** (real-world data is non-linear!)
3. Reduce dimensionality and preserve spatial invariance with **pooling**

CNN FOR CLASSIFICATION

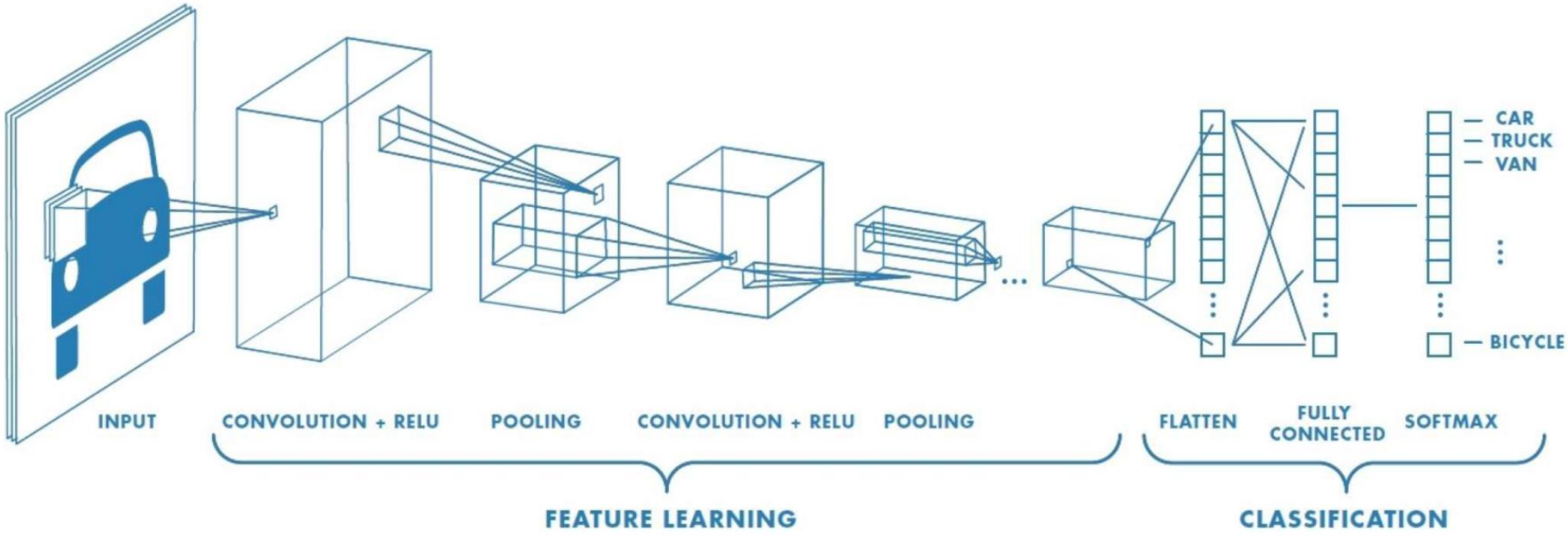
Class Probabilities



- Convolutional and Pooling layers output high-level features of input
- **Fully connected layer** uses these features for classifying input image
- Express output as **probability** of image belonging to a particular class

CNN FOR CLASSIFICATION

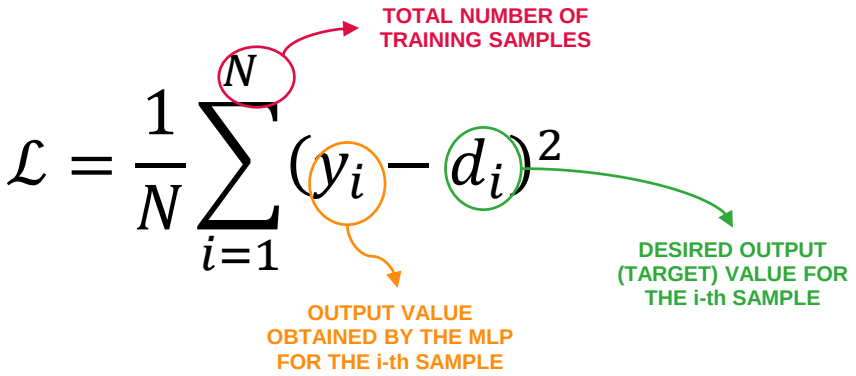
Training with Backpropagation



- Learn weights for convolutional filters and fully connected layers

BACKPROPAGATION ALGORITHM

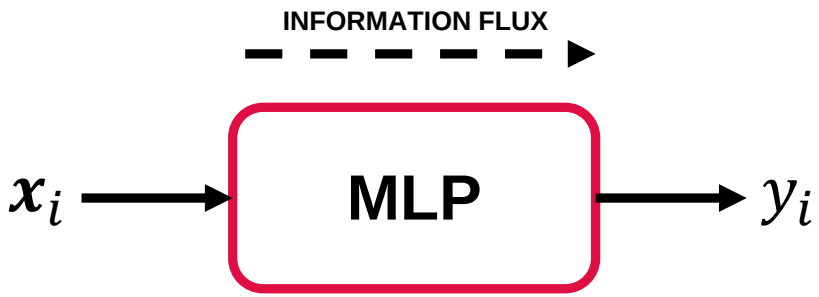
- What weights should be modified (and how much) to obtain correct classification?
  - I.e., Understand what connections are increasing or reducing to the error in the output
- Looking for an algorithm which modifies the different weights to **minimize the error rate**
- **Backpropagation:** iterative algorithm which has hugely contributed to neural network fame
- It is a **gradient-based search** method which allows finding a minimum of the **sum of squared error criterion**



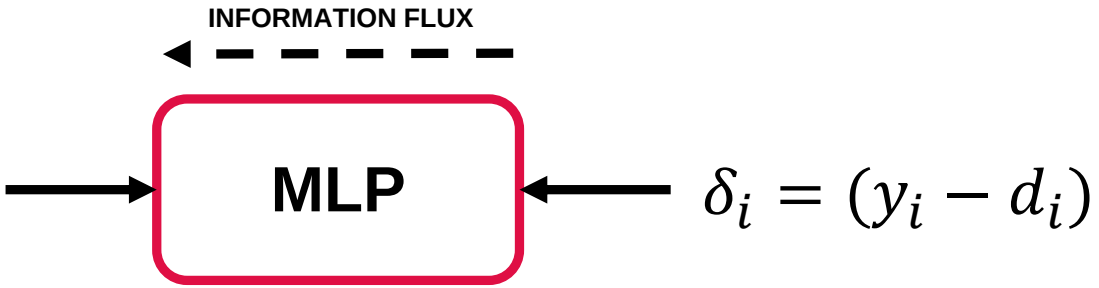
BACKPROPAGATION ALGORITHM

Three Phases

- Forward propagation phase



- Backward propagation phase



- Weight updating phase

LOSS OPTIMIZATION

- How to use the **loss** to iteratively **update the weights** over time given the **training data**?
- **Objective**: find the weights that minimize the **empirical loss**

$$\boldsymbol{W}^* = \arg\min \sum_{i=1}^{N} \mathcal{L}(f(x_i; \boldsymbol{W}), d_i)$$

$$\boldsymbol{W}^* = \arg\min_{\boldsymbol{W}} \mathcal{L}(\mathbf{W})$$

$\mathbf{W} = \{\mathbf{W}^{(0)}, \mathbf{W}^{(1)}, ...\}$ Set of all the weights in the network (first layer, second layer, etc..)

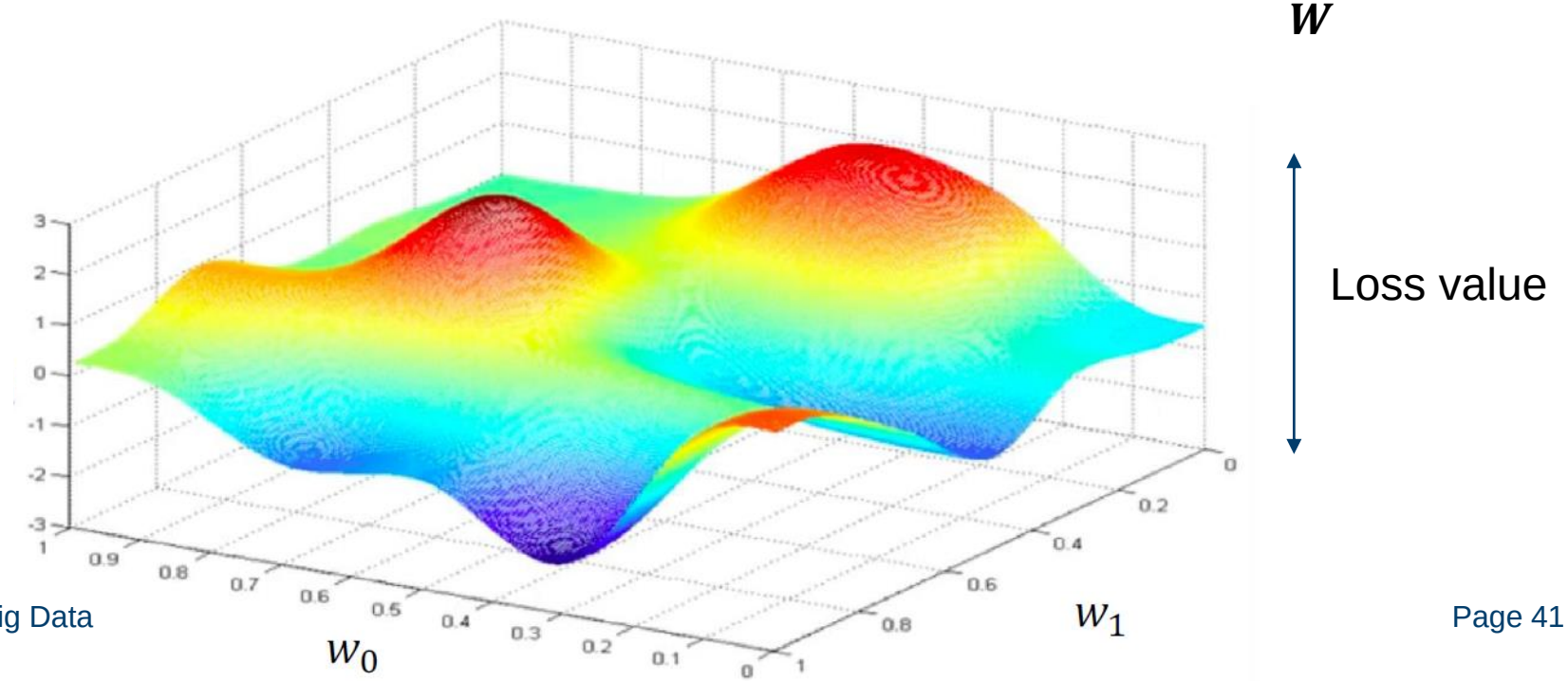
Compute this optimization problem over all these weights

LOSS OPTIMIZATION

illustration

- Loss function: takes as input the weights and gives the loss
  - It is a function of the network weights
- **Find the lowest point** in this landscape that correspond to the minimum loss
  - I.e., Find the correspondent weights

$$\boldsymbol{W}^* = \arg\min_{\boldsymbol{W}} \mathcal{L}(\mathbf{W})$$

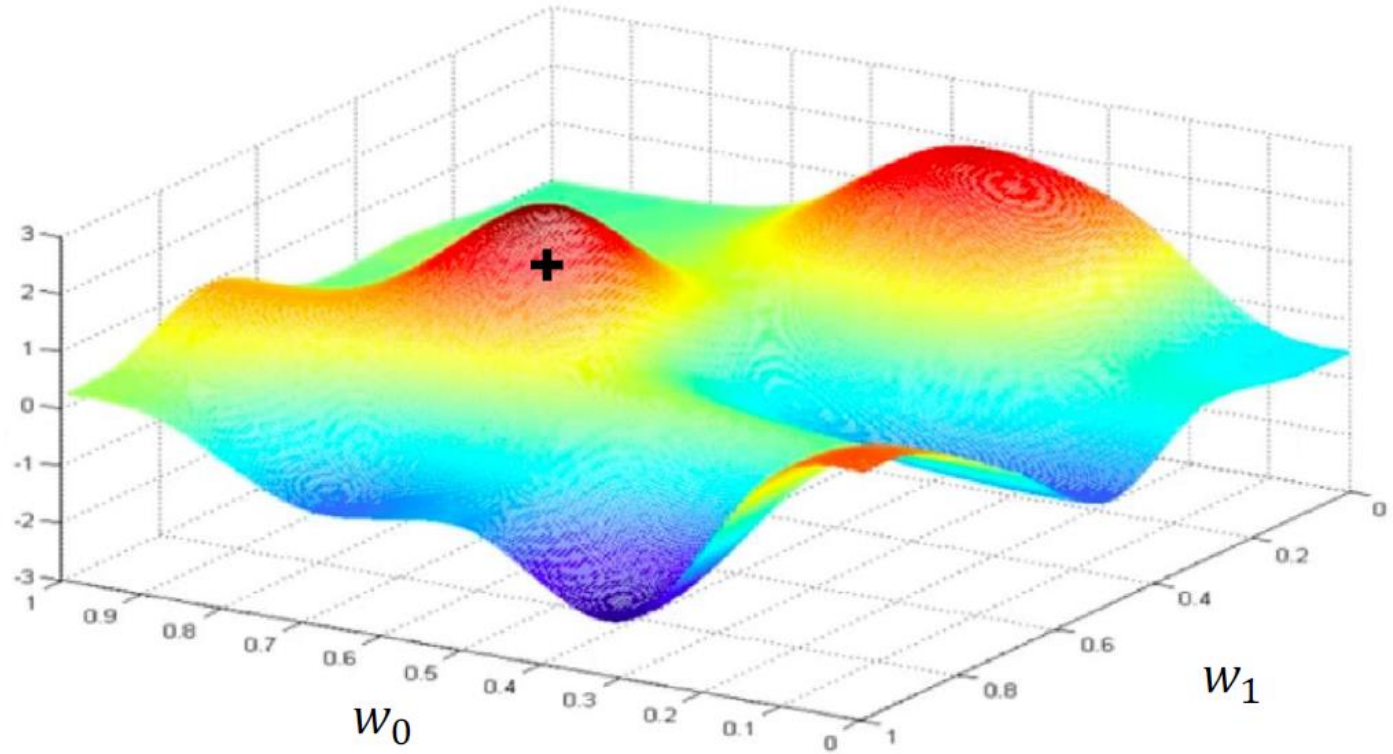


LOSS OPTIMIZATION

illustration

- Randomly pick an initial (w_0, w_1)

$\mathcal{L}(w_0, w_1)$

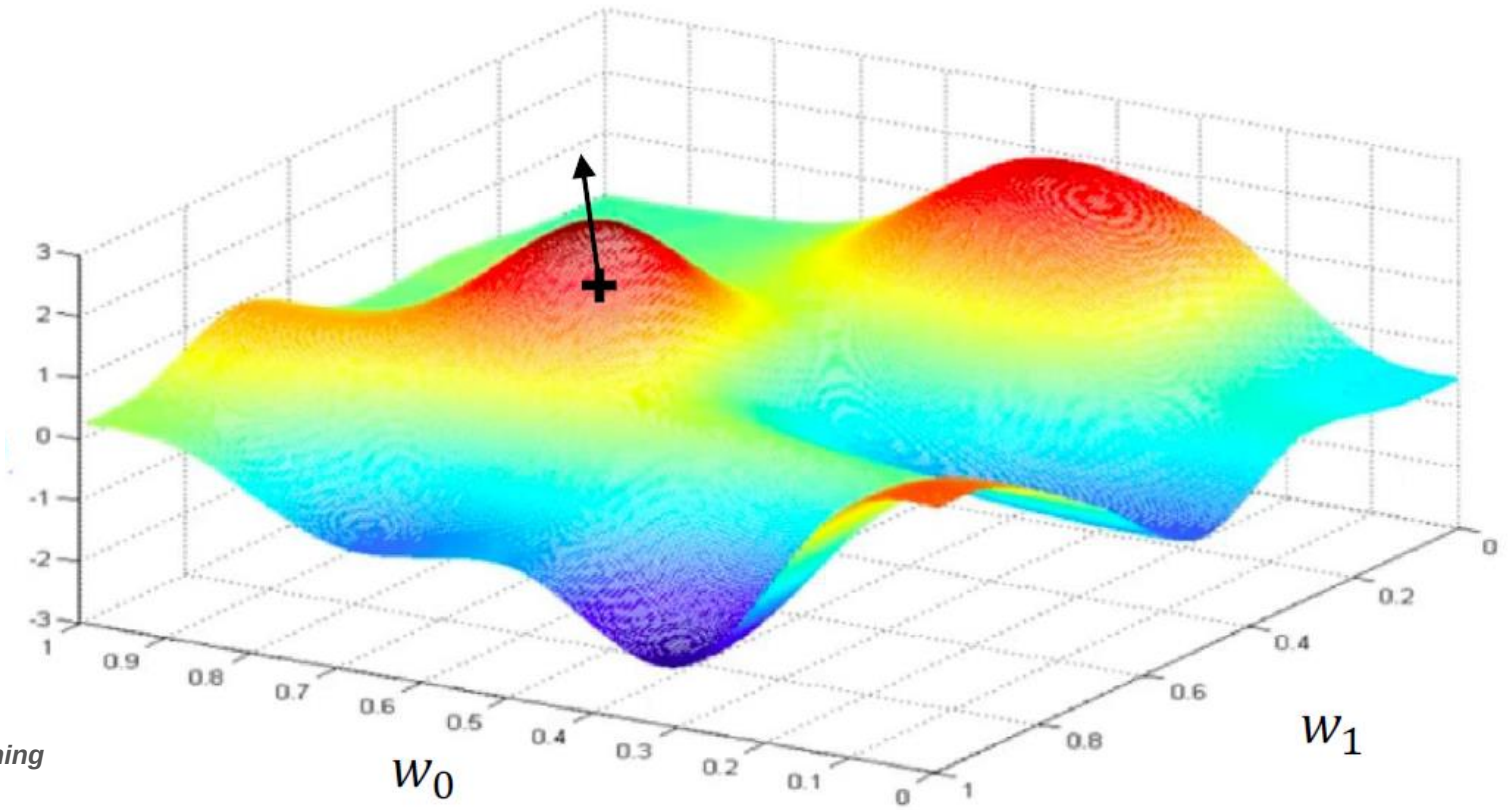


LOSS OPTIMIZATION

illustration

- Compute **gradient** at this local point $\frac{\partial \mathcal{L}(\boldsymbol{W})}{\partial \boldsymbol{W}}$
  - In this landscape the gradient tells us **the direction** of the maximum (steepest) **ascent**

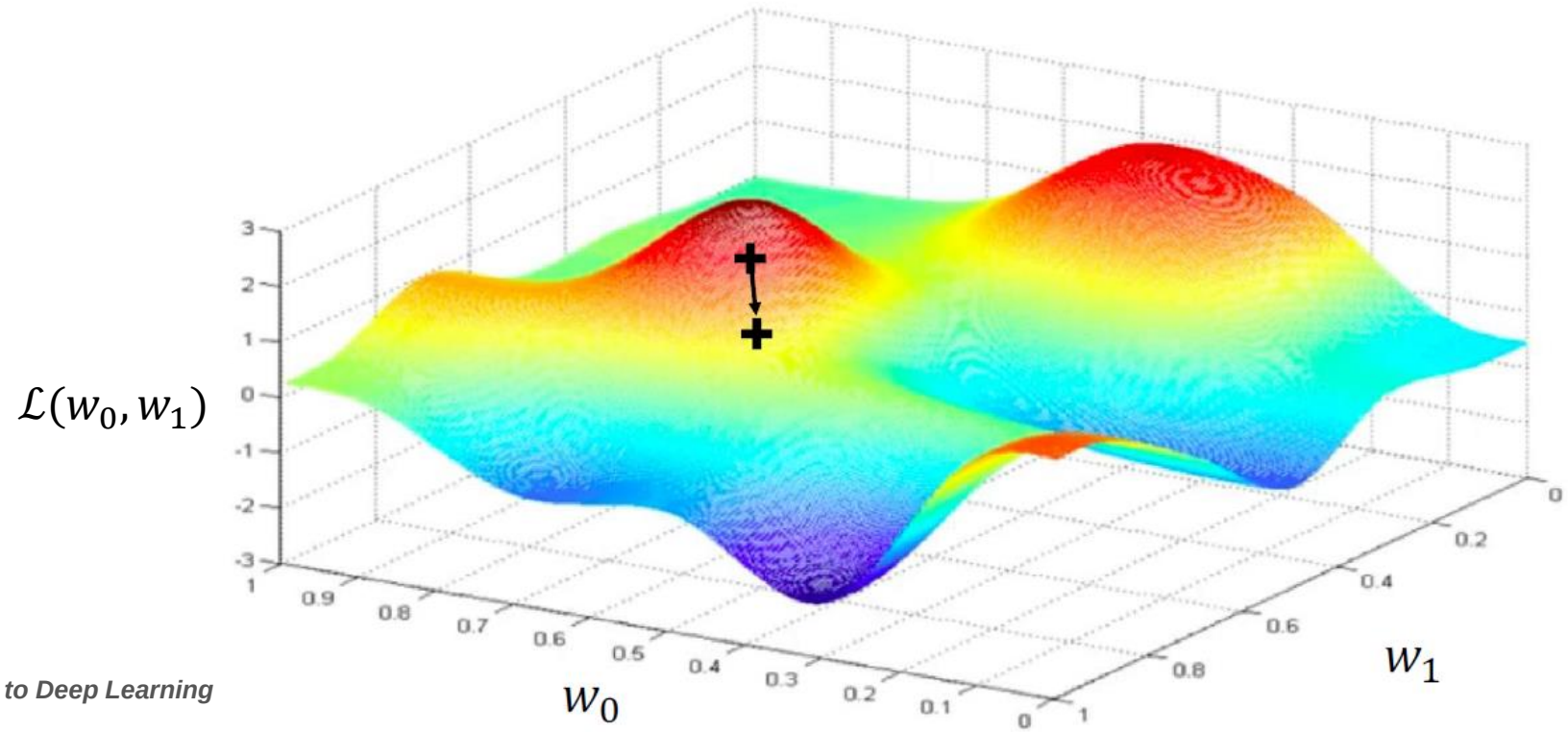
$\mathcal{L}(w_0, w_1)$



LOSS OPTIMIZATION

illustration

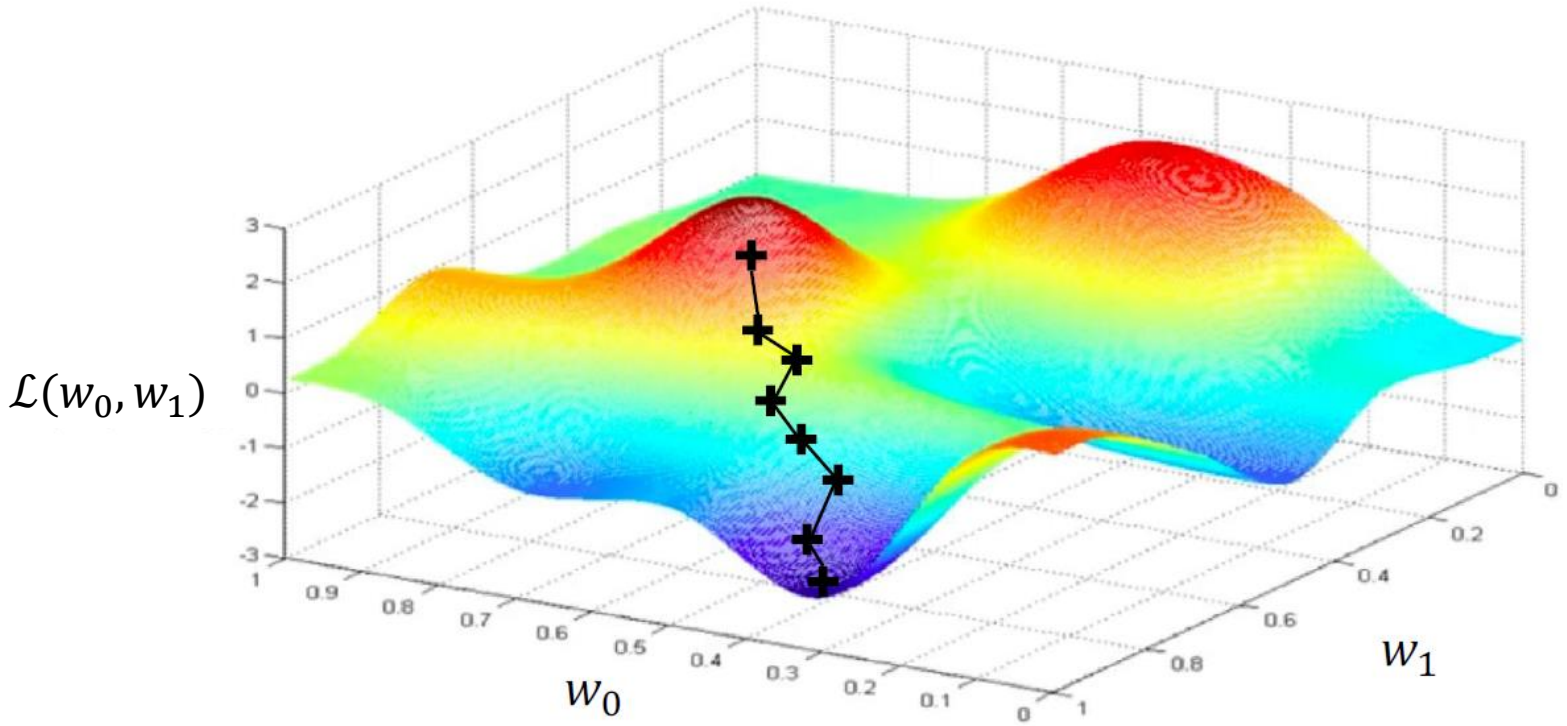
- Reverse the gradient and take a small step in opposite direction



GRADIENT DESCENT

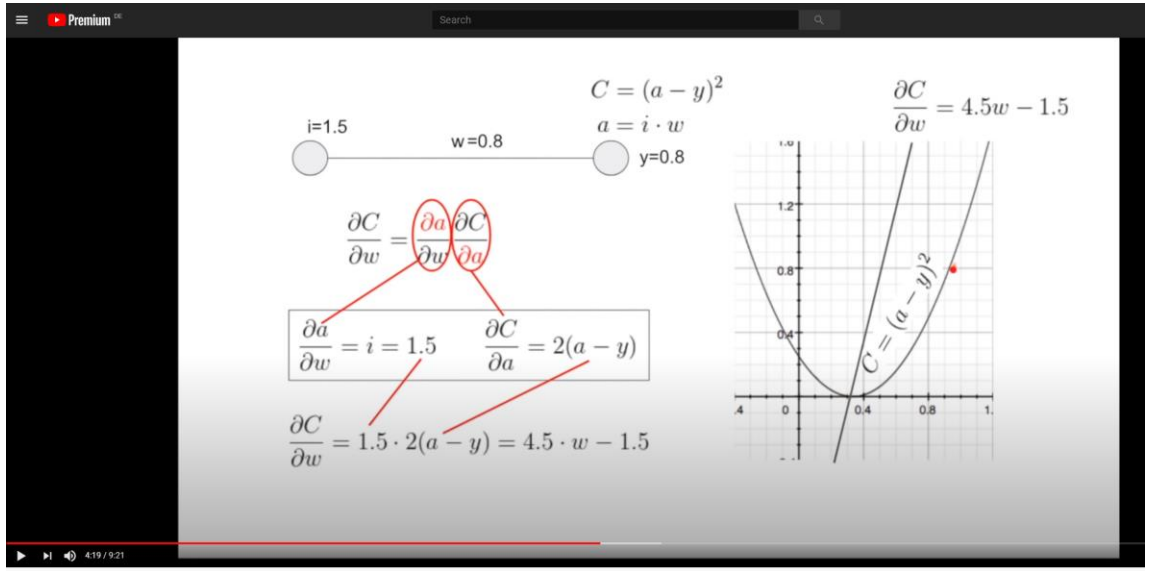
illustration

- Repeat until convergence
  - The gradient is computed over and over



BACKPROPAGATION ALGORITHM

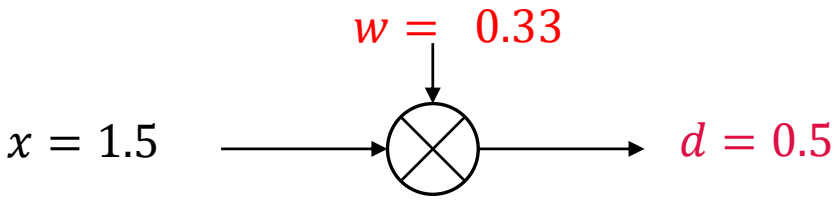
Video

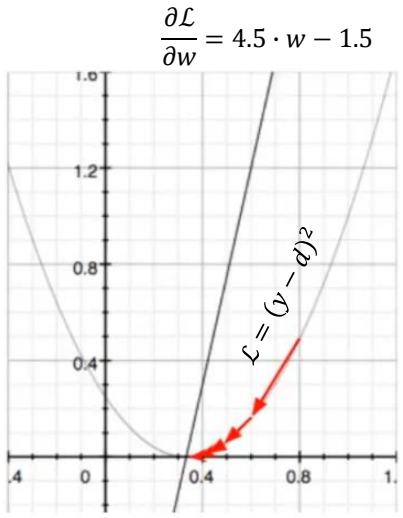


[32] The Absolutely Simplest Neural Network Backpropagation Example

$$w_1 = w_0 - \eta \frac{\partial \mathcal{L}}{\partial w} = w_0 - 0.1 \cdot (4.5 \cdot w_0 - 1.5)$$

w_0	w_1
0.8	0.59
0.59	0.4745
0.4745	0.410975
0.410975	0.37603625
0.37603625	0.3568199375
……	
→ 0.333333	





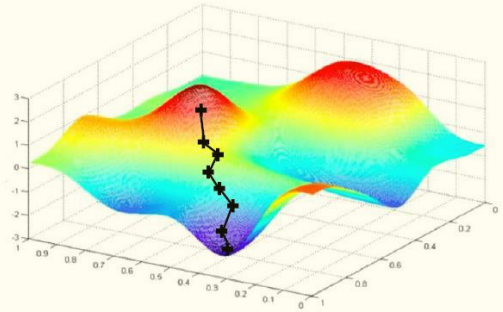
- The slope of the $\mathcal{L}$ determines the adjustment direction
- Adjust weights proportional to the negative of the gradient
- The adjustment is ''backpropagated'' through all layers

BACKPROPAGATION ALGORITHM

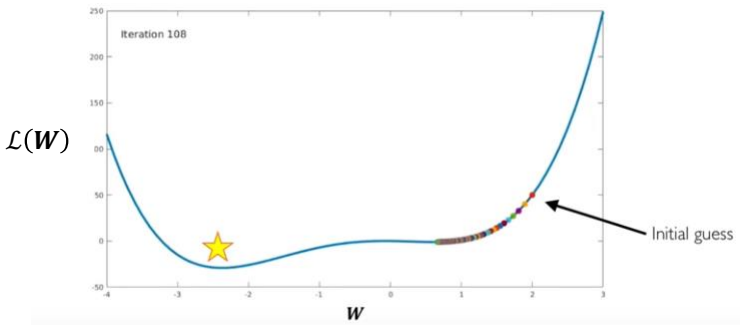
1. Initialize weights randomly $\sim \mathcal{N}(0, \sigma^2)$
2. Loop until convergence
3. Compute gradient $\frac{\partial \mathcal{L}(\boldsymbol{W})}{\partial \boldsymbol{W}}$ (it explains how the loss changes with respect to each of the weights)
4. Update weights $\boldsymbol{W} \coloneqq \boldsymbol{W} - \eta \frac{\partial \mathcal{L}(\boldsymbol{W})}{\partial W}$ (in the opposite direction of the gradient)
5. Return weights

Most computational part when there is a high number of weights

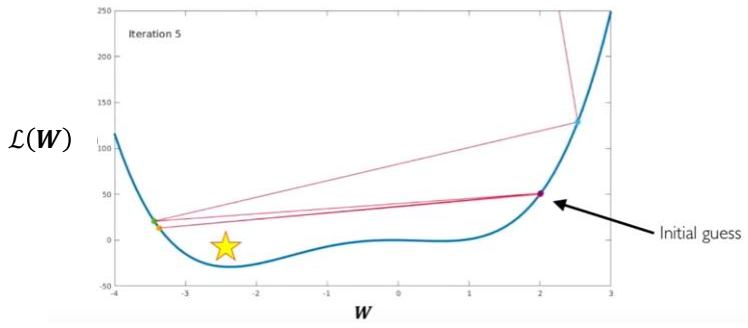
[33] NVIDIA Launches Tesla V100s



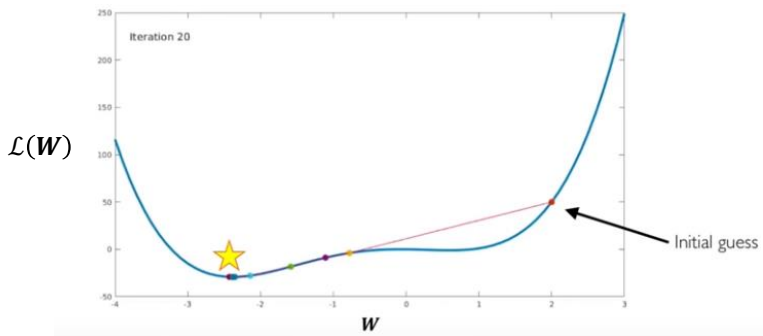
Learning rate determines the adjustment magnitude
How much do you trust the computed gradient



Small learning rate converges slowly and gets stuck in false local minima



Large learning rates overshoot, become unstable and diverge

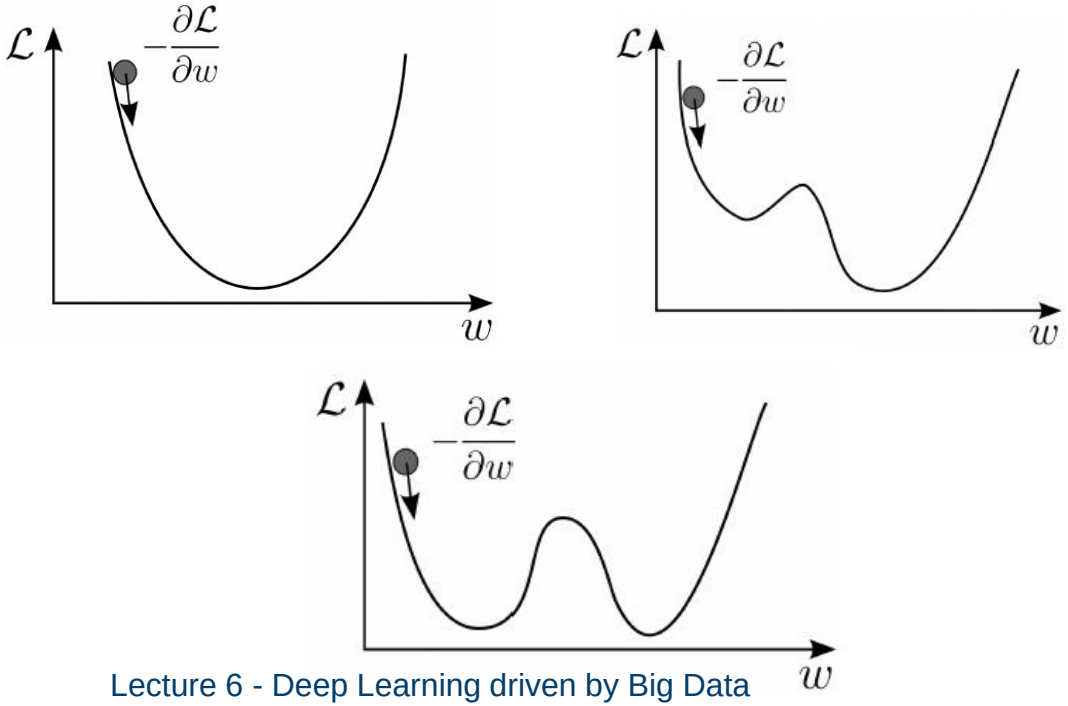


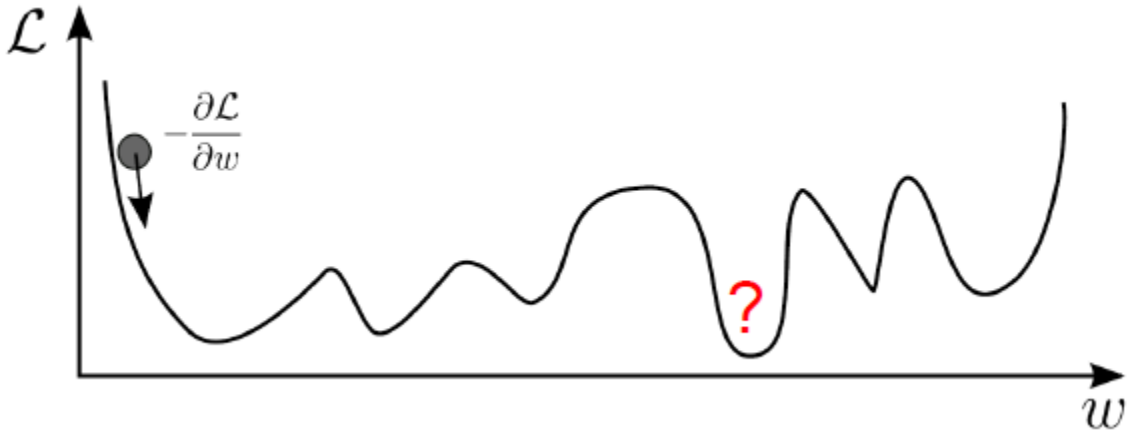
Stable learning rates converge smoothly and avoid local minima

TRAINING DEEP NEURAL NETWORKS

Loss minimization is highly non-trivial

- How to ensure to get to a global minima (instead of a local minima)?
  - There is no guarantee
- Also, there are many local minima
- Finding the optimal true minimum is difficult



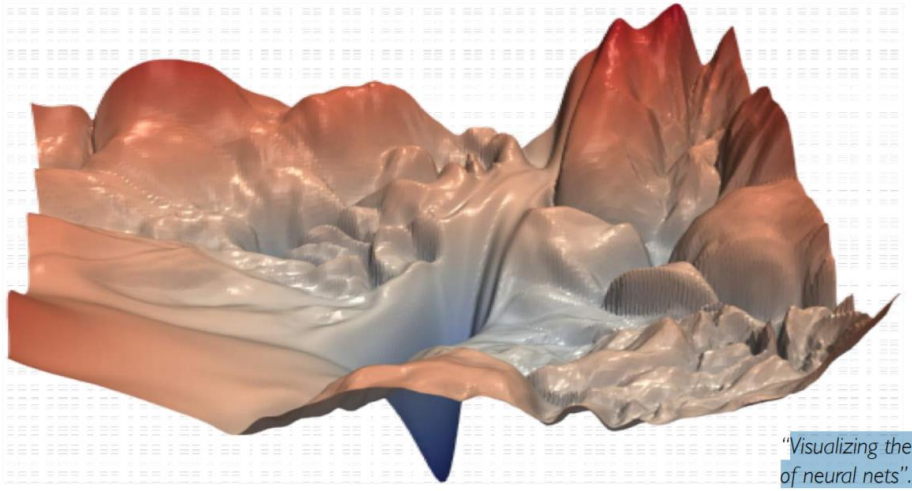


TRAINING DEEP NEURAL NETWORKS

Loss minimization is highly non-trivial

- There is one fundamental problem of machine learning
  - The **real/true loss function** is **hidden**
- You always work with the limited training data
  - No matter how large it is
  - It is tiny **subset** of the **real problem**
- We compute the **empirical loss** (the loss on the training data)

- How to estimate the **true loss**? ,
- i.e., How good or bad the network performs on the data that the network did not see during the training?



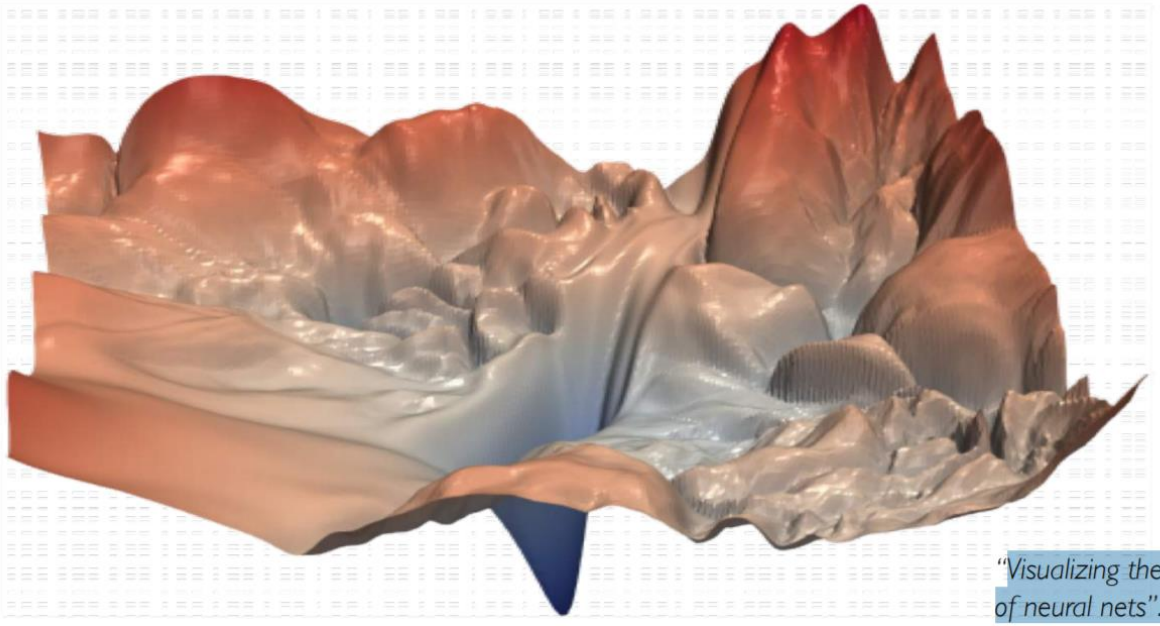
"Visualizing the loss landscape of neural nets". Dec 2017.

[34] Tom Goldstein: "What do neural loss surfaces look like?"

TRAINING DEEP NEURAL NETWORKS

Is it possible?

- One would expect that ML would never work (i.e., this kind of learning will just fail)
  - E.g., **Overfitting**, adapting too much to the training data (lookup table procedure - memorize)
- Previous common wisdom: learning in deep architectures not tractable



"Visualizing the loss landscape of neural nets". Dec 2017.

[34] Tom Goldstein: "What do neural loss surfaces look like?"

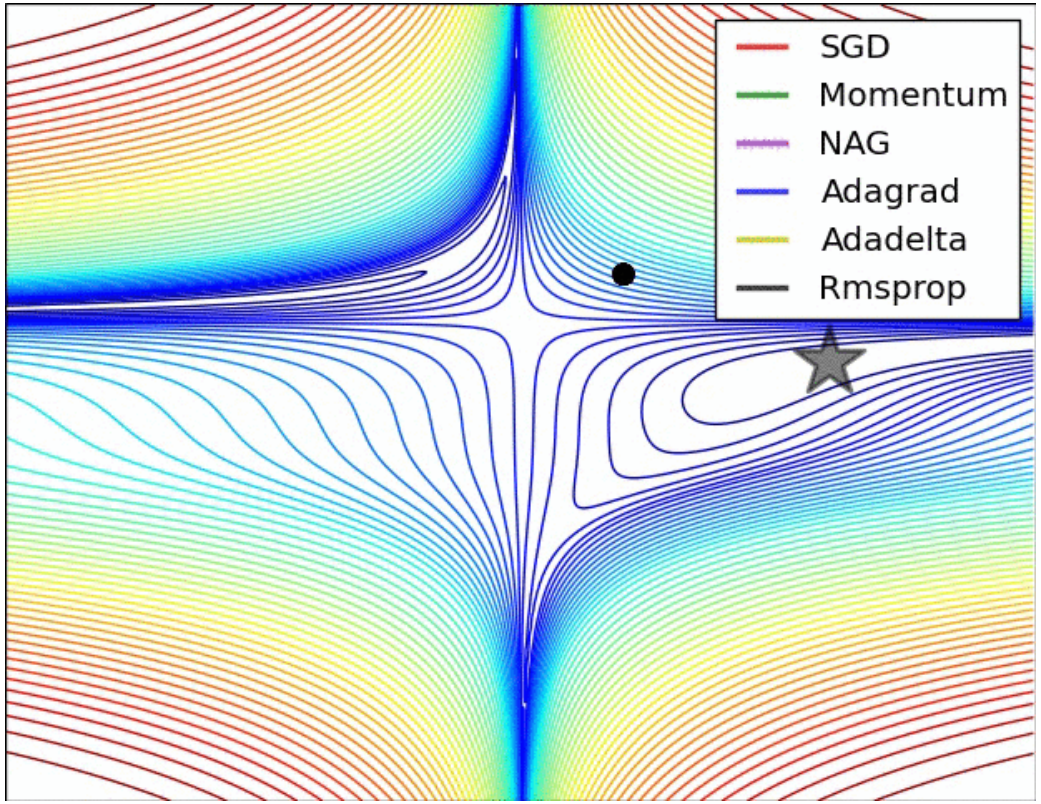
TRAINING DEEP NEURAL NETWORKS – KEY IMPROVEMENTS

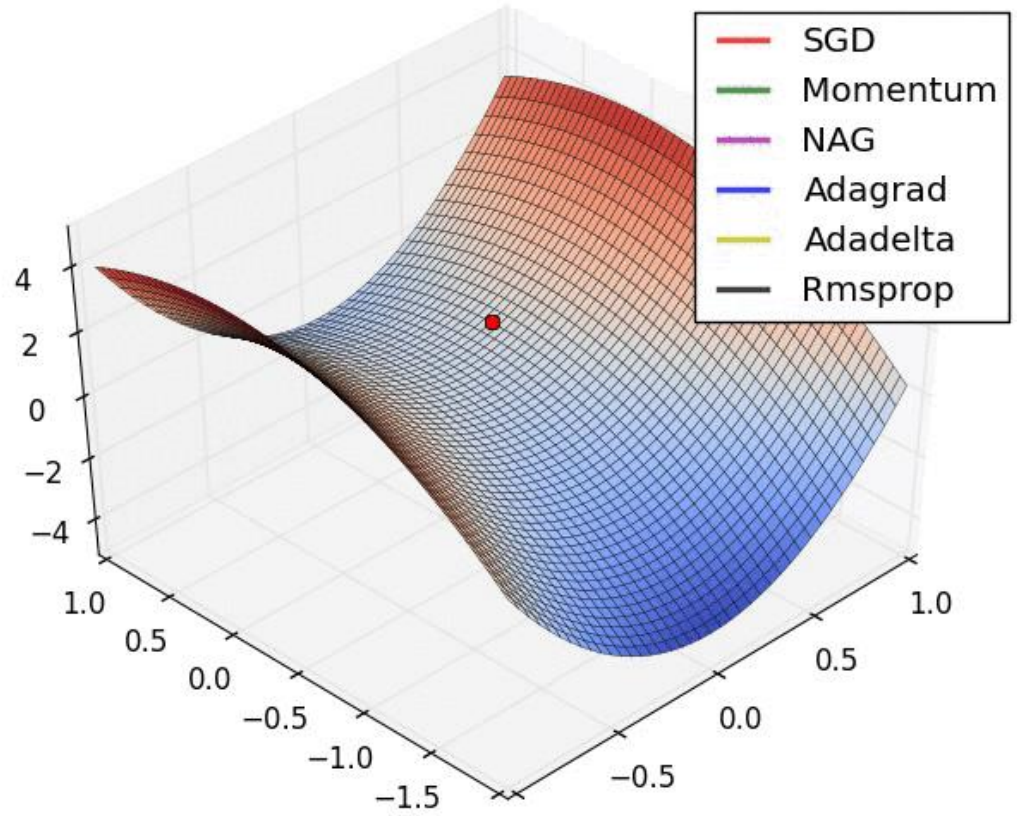
What are the reason for the boost in performance

- Improved **stochastic gradient descent (SGD)**
  - Preventing vanishing or exploding gradients over many layers
    - Proper random **weight initialization**
    - Suitable **transfer functions**
    - Network architecture modification (e.g., **skip connections**)
  - Update rules with adaptive momentum (Nesterov), **adaptive learning rate** (stochastic annealing; Adam, RMSProp)
- **Regularization** against overfitting (DropOut, Batch Normalization, decaying weights, sparse activation, etc)
- **Huge amounts of labeled data** (ImageNet, CoCo, etc), freely available
- **Data augmentation** techniques extending data sets
- **Parallelization**, specialized hardware (e.g, GPU, TPU) based acceleration

DEEP NEURAL NETWORKS: KEY IMPROVEMENTS

Momentum update, adaptive learning rate





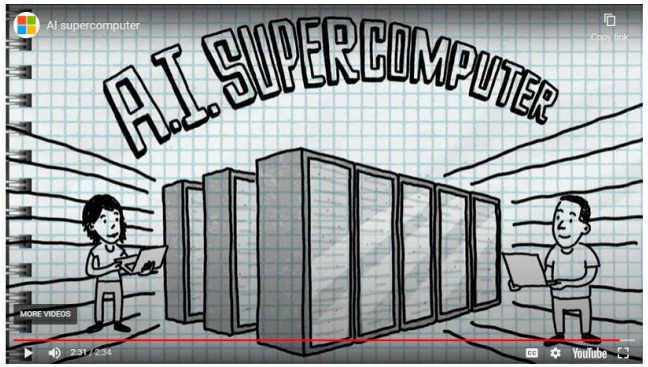
[35] An overview of gradient descent optimization algorithm

DL IS TRANSFORMING HOW COMPUTERS ARE DESIGNED

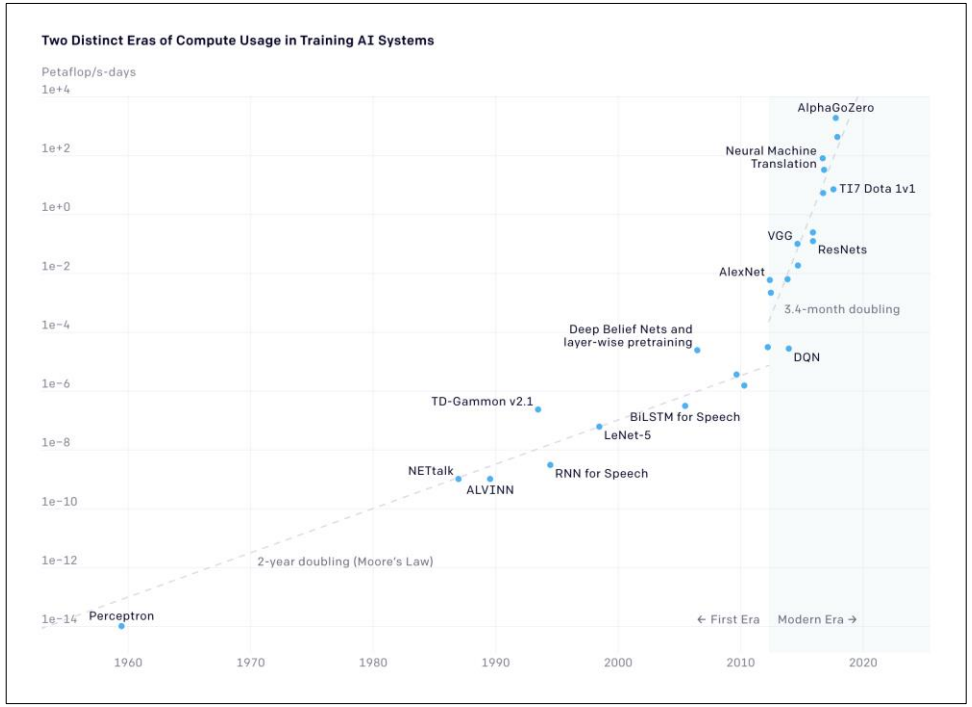
More computational power needed

- **Training** powerful but computationally-expensive deep models on:
  - Terabyte or petabyte-sized training datasets
  - Plus techniques like AutoML (“Learning to learn”, Neural Architecture Search, etc.) can multiply desired training computation by 5-1000X
- **Inference** using expensive deep models in systems with:
  - Hundreds of thousands of requests per second and billions of users
  - Latency requirements of tens of milliseconds

“Microsoft announces new supercomputer, lays out vision for future AI work”



[36] The AI Blog



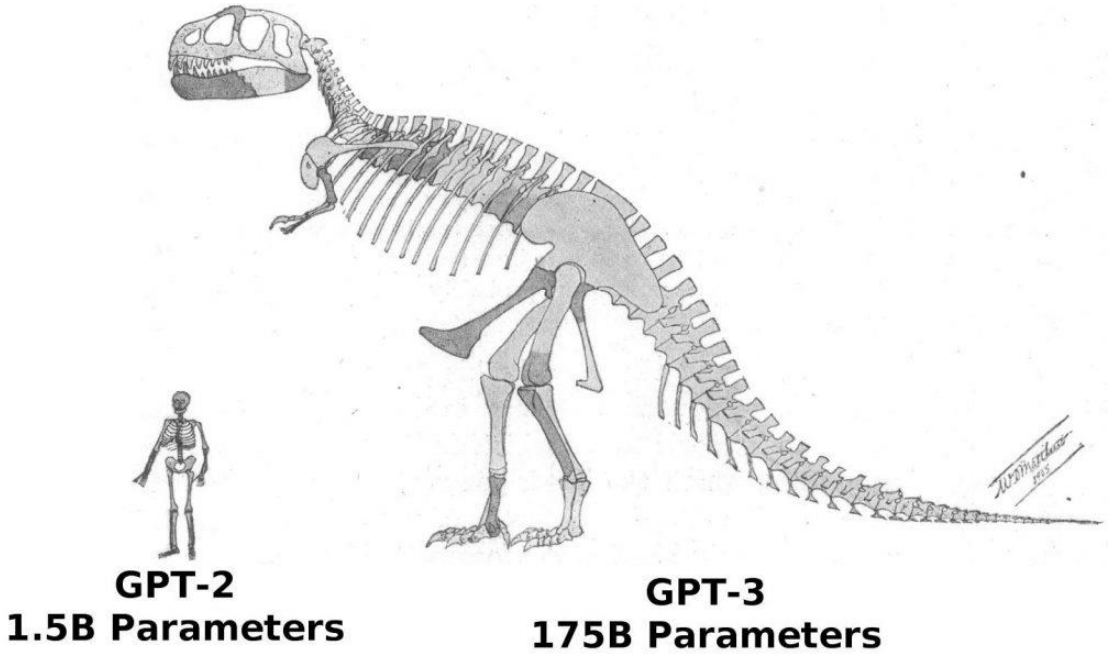
- Deep learning is creating insatiable computation demands
- Achievements in AI need exponentially growing computing power

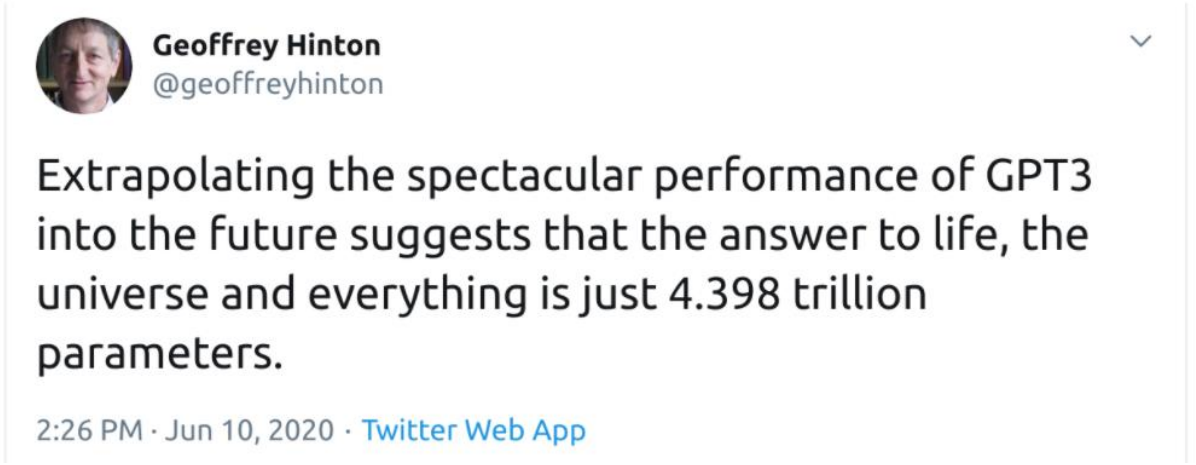
[37] OpenAI: AI and Compute

EXAMPLE

Generative Pretrained Transformer-3 (GPT-3)

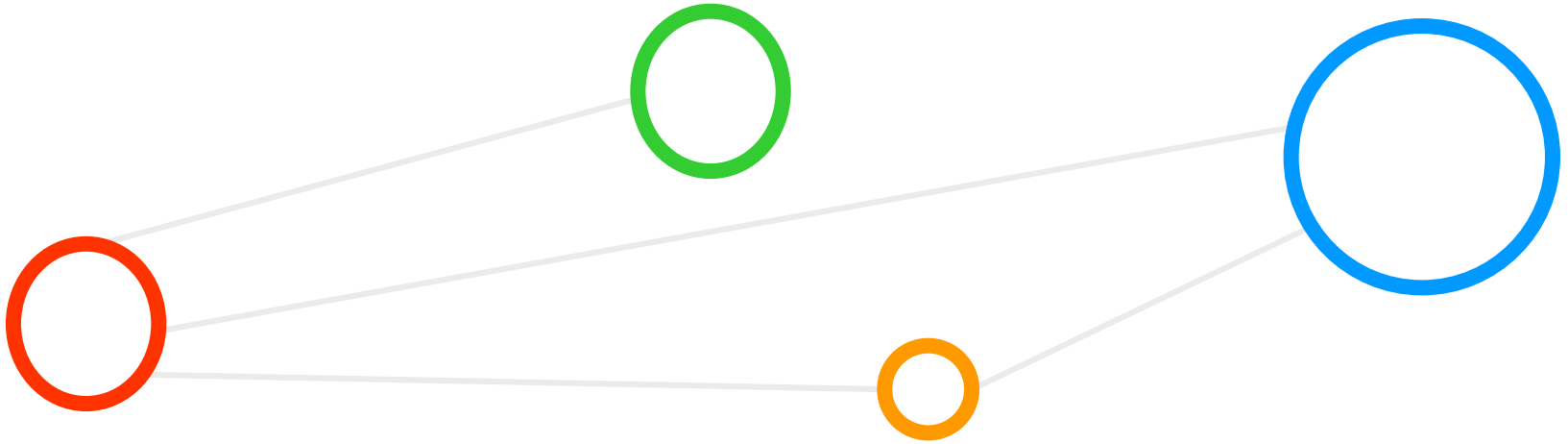
- GPT-3 is the largest natural language processing transformer released to date
  - Pre-training on 570GB of text compared to 40GB for GPT-2





[38] GPT-3: A New Breakthrough in Language Generation

BIG REMOTE SENSING DATA



WHAT IS BIG DATA?

- **Buzzword** in science and engineering – what does this mean?
  - When does ‘big data’ start – with hundreds of MB / GB / TB / PB … EB?
  - Exact definition (in terms of volume) of big data is hard to find…
  - We have to look on concrete examples to find answers in Cloud context
  - Initially referred to **VVV (Volume, Velocity, Variety)**
  - Being constantly extended to n ‘Big Data Vs‘ (Veracity, Validitv. …)
- Selected attempts of definitions for **‘Big Data’** :

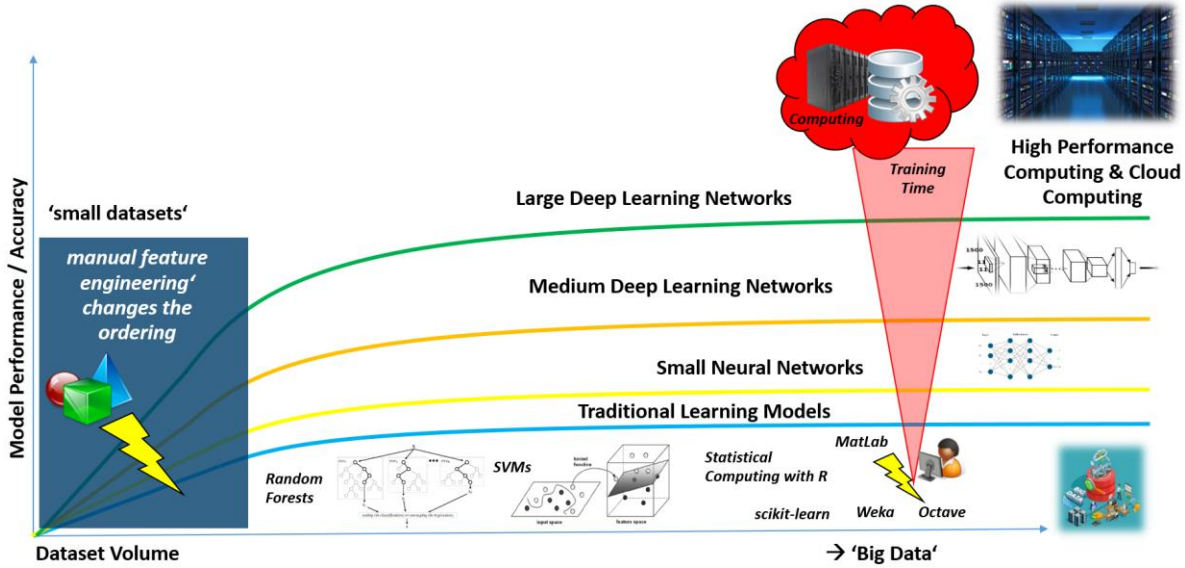
> **‘Big Data’ is the term for a collection of data sets so large and complex that it becomes difficult to process using on-hand database management tools or traditional data processing applications**

[40] Wikipedia ‘Big Data’ Online

> **‘Big Data’is data that becomes large enough that it cannot be processed using conventional methods.’**

[41] O’Reilly Radar Team, ‘Big Data Now: Current Perspectives from O'Reilly Radar’

[42] www.big-data.tips



COPERNICUS

World's Largest Single Earth Observation Programme (EO)

[42] Copernicus programme

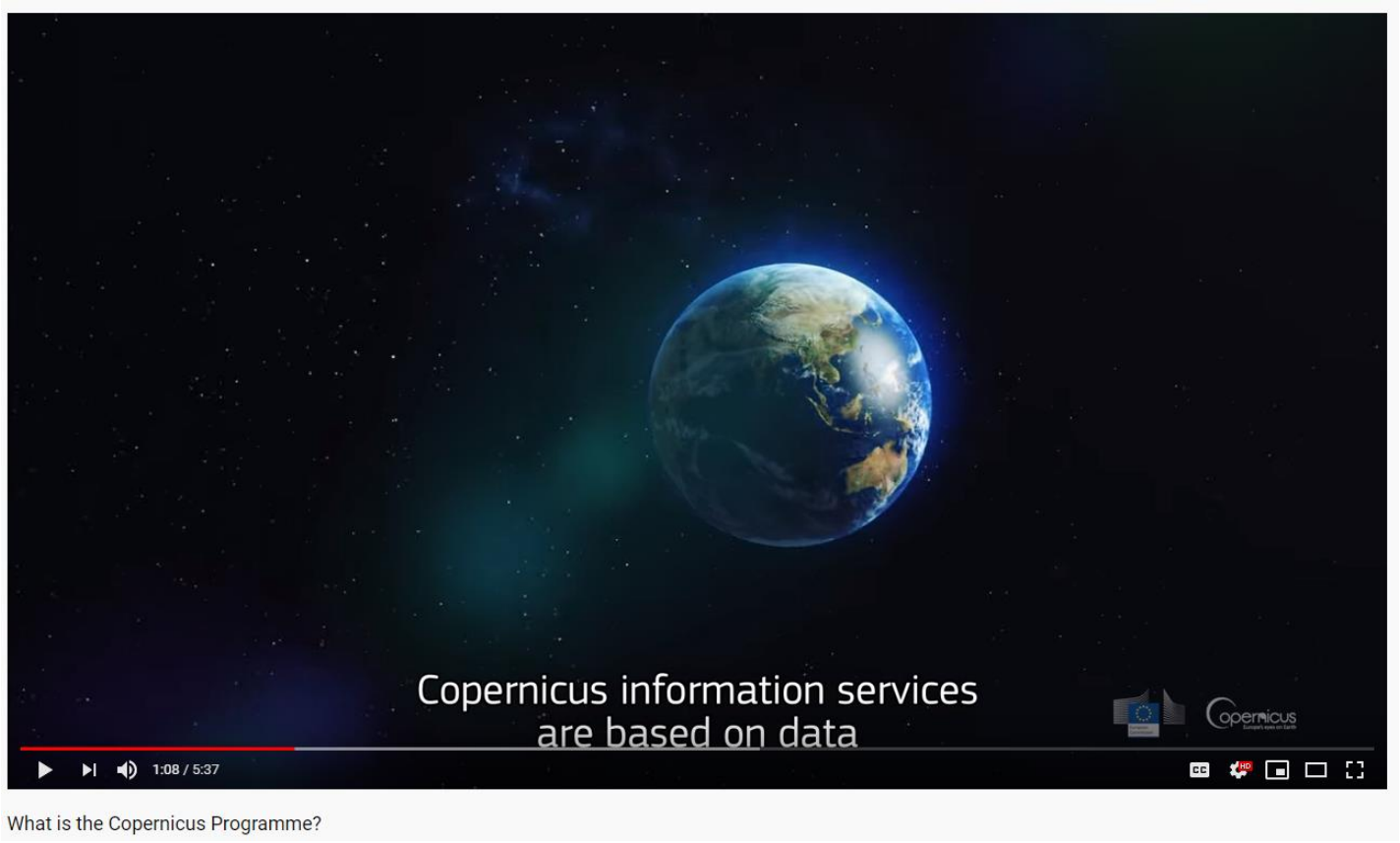
- Directed by the European Commission in partnership with the European Space Agency (ESA)
  - Monitors the Earth, its environment and ecosystems
  - **Free and open data policy**
  - Features: Continuity, Global coverage, Frequent updates and Huge data volumes

[43] Sentinel Space

Served by a set of dedicated satellites (the Sentinel families)

COPERNICUS PROGRAMME

Video

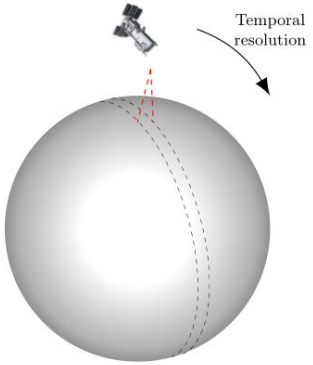


[44] What is the Copernicus Programme?

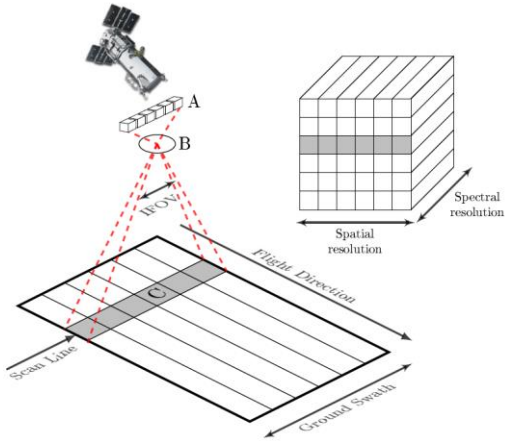
REMOTE SENSING DATA

Fit the V's of Big Data

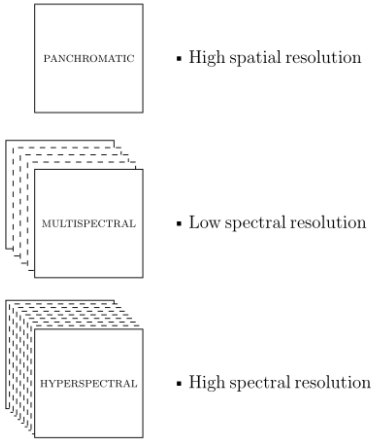
- **Velocity**: acquisition



Volume: generated



Variety: data

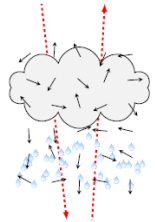


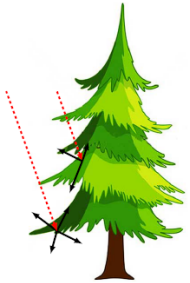
- **Velocity**: data processing and analysis must confront with the rapidly **growing rate of data generation** (i.e., frequent revisits)
  - Data should preferably be **processed in real time** in order to achieve a given task
  - Despite significant progress, challenges remain in developing predictive algorithms that can deal with the velocity of data arriving in real time
- **Volume**: increasing scale of **archived data** (i.e., beyond the petabytes) raises not only data storage but also massive **analysis issues**
- **Variety:** data is delivered by sensors acting over **different spatial, spectral, and temporal resolutions**
  - E.g., World-View-3 satellite (spatial resolution of 0.31 m), AISA Dual airborne (500 bands with spectral resolution of 2.9 nm), NOAA satellite with a re-visit cycle of a few hours

REMOTE SENSING DATA

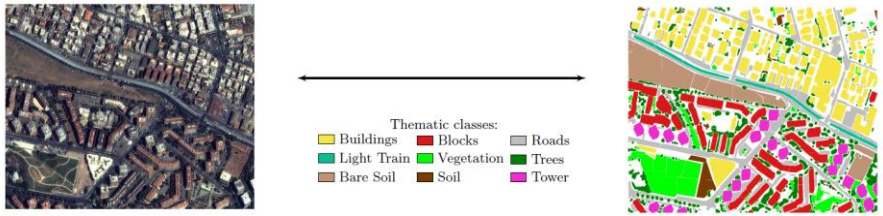
Fit the V's of Big Data

- **Veracity:** uncertainty
  - Massive amount of sensed data coming in at high speed
  - Cannot be directly used by the applications
  - Data corruption

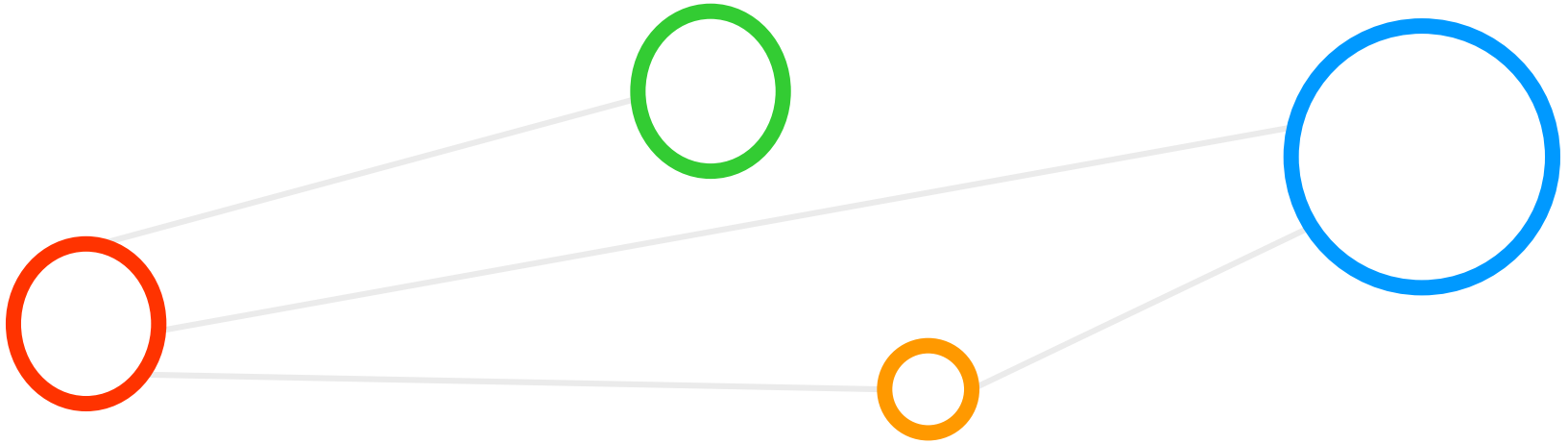




- **Value:** hidden information
  - The **interpretation** of remote sensing data is **not straightforward**
  - It **requires** a powerful yet highly **accurate processing scheme** to extract reliable and valuable **information**
  - The value of the data depends on its potential for extracting the information from it
  - Analytical methods such as **deep learning** can be exploited in order to derive this value



LECTURE BIBLIOGRAPHY



REFERENCES

- [1] Mining of Massive Datasets
  Online: http://infolab.stanford.edu/~ullman/mmds/book.pdf
- [2] Apache Hadoop Web page
  http://hadoop.apache.org/
- [3] J. Dean, S. Ghemawat, ‘MapReduce: Simplified Data Processing on Large Clusters’, OSDI'04: Sixth Symposium on Operating System Design and Implementation, December, 2004
- [4] Google DataProc Service
  Online: https://cloud.google.com/dataproc/
- [5] Microsoft Azure HDInsight Service
  Online: https://azure.microsoft.com/en-us/services/hdinsight/
- [6] Amazon Web Services (AWS) Elastic Map-Reduce (EMR)
  Online: https://aws.amazon.com/emr
- [7] K. Hwang, G. C. Fox, J. J. Dongarra, ‘Distributed and Cloud Computing’, Book
  Online: http://store.elsevier.com/product.jsp?locale=en_EU&isbn=9780128002049
- [8] Amazon Web Services Educate Web Page
  Online: https://aws.amazon.com/education/awseducate/
- [9] Jupyter Web page
  Online: http://jupyter.org/
- [10] Species Iris Group of North America Database
  Online: http://www.signa.org

REFERENCES

- [11] F. Rosenblatt, ‘The Perceptron - a perceiving and recognizing automaton’, Report 85-460-1, Cornell Aeronautical Laboratory, 1957
  Online: https://blogs.umass.edu/brain-wars/files/2016/03/rosenblatt-1957.pdf
- [12] Rosenblatt, ’The Perceptron: A probabilistic model for information storage and orgainzation in the brain’, Psychological Review 65(6), pp. 386-408, 1958
  Online: https://psycnet.apa.org/doi/10.1037/h0042519
- [13] The XOR Problem in Neural Networks
  Online: https://medium.com/@jayeshbahire/the-xor-problem-in-neural-networks-50006411840b
- [14] Multilayer Perceptron
  Online: https://www.eecs.yorku.ca/course_archive/2012-13/F/4404-5327/lectures/10%20Multilayer%20Perceptrons.pdf
- [15] R. O. Duda, P. E. Hart and D. G. Stork. Pattern Classification. Second Edition, New York: John Wiley & Sons Inc, 2001.
- [16] Understanding the Neural Network
  Online: http://www.cs.cmu.edu/~bhiksha/courses/deeplearning/Fall.2019/www/hwnotes/HW1p1.html
- [17] Can a perceptron with sigmoid activation function perform nonlinear classification?
  Online: https://stats.stackexchange.com/questions/263768/can-a-perceptron-with-sigmoid-activation-function-perform-nonlinear-classificati
- [18] A Neural Network Playground - TensorFlow
  Online: https://playground.tensorflow.org/
- [19] Wikipedia ‘Sepal‘
  Online: https://en.wikipedia.org/wiki/Sepa
- [20] The significant difference between AI, ML and Deep Learning
  Online: https://www.usoft.com/blog/difference-between-ai-ml-and-deep-learning

REFERENCES

- [21] Gordon Campbell, ‘’Earth Observation Science and Applications’’, Workshop on Quantum for Earth Observation
  Online: https://esamultimedia.esa.int/docs/EarthObservation/GCampbell_ESA.pdf
- [22] Desktop PC
  Online: https://praxistipps.chip.de/was-ist-ein-desktop-pc-einfach-erklaert_108897
- [23] The Benefits of Cloud Computing
  Online: https://sevaa.com/blog/2018/07/cloud-computing/
- [24] JUWELS Jülich Wizard for European Leadership Science
  Online: https://www.fz-juelich.de/ias/jsc/EN/Expertise/Supercomputers/JUWELS/JUWELS_news.html
- [25] D-Wave systems
  Online: https://www.dwavesys.com/
- [26] Deep Learning: Practices and Trends - NIPS 2017
  Online: https://www.facebook.com/nipsfoundation/videos/1552060484885185/
- [27] CS231n Convolutional Neural Networks for Visual Recognition
  Online: http://cs231n.github.io/
- [28] Martin Görner, Theory: convolutional networks
  Online:https://sites.google.com/site/nttrungmtwiki/home/it/data-science---python/tensorflow/tensorflow-and-deep-learning-part-3?tmpl=%2Fsystem%2Fapp%2Ftemplates%2Fprint%2F&showPrintDialog=1
- [29] Matt Krause, Invariance
  Online: https://stats.stackexchange.com/questions/208936/what-is-translation-invariance-in-computer-vision-and-convolutional-neural-netwo
- [30] Convolutional Neural Network
  Online: https://commons.wikimedia.org/wiki/File:Convolutional_Neural_Network_NeuralNetworkFeatureLayers.gif

REFERENCES

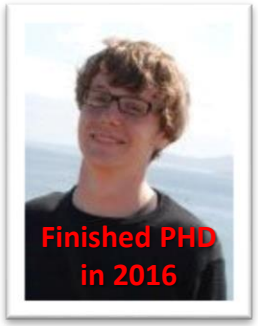
- [31] Gunnar Carlsoon, Topological Data Analysis
  Online: https://www.ayasdi.com/using-topological-data-analysis-understand-behavior-convolutional-neural-networks/
- [32] The Absolutely Simplest Neural Network Backpropagation Example
  Online: https://www.youtube.com/watch?v=khUVIZ3MON8&ab_channel=MikaelLaine
- [33] NVIDIA Launches Tesla V100s GPU AI Accelerator With Faster Clocks And 32GB HBM2
  Online: https://hothardware.com/news/nvidia-tesla-v100s-gpu-ai-accelerator-faster-clocks-32gb-hbm2
- [34] Tom Goldstein: "What do neural loss surfaces look like?“
  Online: https://www.youtube.com/watch?v=78vq6kgsTa8&t=1056s
- [35] An overview of gradient descent optimization algorithm
  Online: http://ruder.io/optimizing-gradient-descent/index.html#visualizationofalgorithms
- [36] The AI Blog
  Online: https://blogs.microsoft.com/ai/openai-azure-supercomputer/
- [37] OpenAI: AI and Compute
  Online: https://openai.com/blog/ai-and-compute/
- [38] GPT-3: A New Breakthrough in Language Generation
  Online: https://blog.exxactcorp.com/what-can-you-do-with-the-openai-gpt-3-language-model/
- [39]Wikipedia ‘Big Data’
  Online: https://en.wikipedia.org/wiki/Big_data
- [40] Big Data Now: Current Perspectives from O'Reilly Radar, O'Reilly Radar Team, O'Reilly Media, Inc., 2011, ISBN: 9781449315184

REFERENCES

- [41] Big Data Tips – Big Data Mining & Machine Learning, Online:
  Online: http://www.big-data.tips/
- [42] Copernicus programme: Europe's eyes on Earth
  Online: http://www.copernicus.eu/
- [43] Sentinel Space
  Online: http://earsc.org/news/airbus-selected-by-esa-for-copernicus-data-and-information-access-service-dias
- [44] What is the Copernicus Programme?
  Online: https://www.youtube.com/watch?v=MGJss4lDaBo

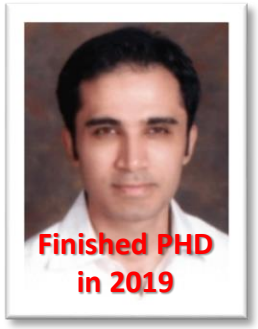
ACKNOWLEDGEMENTS

High Productivity Data Processing Research Group



PD Dr.
G. Cavallaro

Senior PhD
Student A.S. Memon



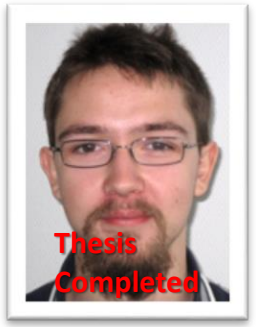
Senior PhD
Student M.S. Memon

PhD Student
E. Erlingsson

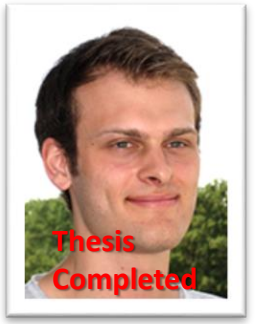
PhD Student
S. Bakarat

PhD Student
R. Sedona

Dr. M. Goetz
(now KIT)



MSc M.
Richerzhagen
(now other division)



MSc
P. Glock
(now INM-1)

MSc
C. Bodenstein
(now
Soccerwatch.tv)

MSc Student
G.S. Guðmundsson
(Landsverkjun)

This research group has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 763558 (DEEP-EST EU Project)


services
measurement
within
fusion
new
funding
Policy-based
concepts
device
Analysis
forms
cross-disciplinary
resources
Centre
climate
Computer
expertise
year
like
Computational
Test
E.g
Computing
disciplines
results
Approaches
construction
Fusion
Enable
Cross-Disciplinary
methods
High
big
nodes
PB
community
Environment
Modelling
different
EU
Science
centers
Key
scientific
important
files
etc
DKRZ
Use
storage
basis
Infrastructure
model
international
simulations
ENES
technologies
formats
physical
SMAQ
engineering
References
Big
brain
Project
Data
increasing
via
EUDAT
processing
IPCC
Common
TB
MapReduce
create
computing
Methods
Using
European
databases
GB
Supercomputing
Work
images
codes
analysis
stored
Resources
analyze
Large
Summary
data-intensive
Service
Scientific
levels
often
use
DLCL
Level
external
JSC
computational
Research
Matters
Concrete
performance
research
e.g
science
HPC
Juelich
system
datasets
acquisition
compute
Landscape
management
System
Climate
modelling
AR5
Hardware
distributed
large-scale
cores
access
hundreds
Nutshell
Services
Technologies
Earth
Structure
current
network
DLCLs
Understanding
structures
directory
project
General
Germany
Simulation
Provide
NASA
Energy
systems
day
Processing
Health