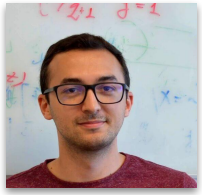
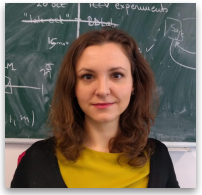
Deep Learning

1. Introduction to Deep Learning

Course Team



Florin Brad

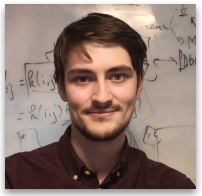


Elena Burceanu

Marius Dragoi

Florin Gogianu

Emanuela Haller



Andrei Manolache

Andrei Nicolicioiu

{fbrad, eburceanu, mdragoi, fgogianu, ehaller, amanolache, anicolicioiu}@bitdefender.com

Invited Speaker

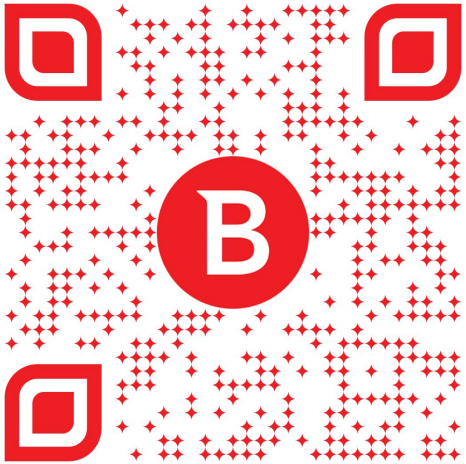
Iulia Duță

University of Cambridge

id366@cam.ac.uk

Our Team

Team page



bit-ml.github.io

AI / ML Community

Bucharest

- Bucharest Deep Learning
- Bucharest AI

Events:

- Romanian AI Days
- EEML Summer School

AI Romania

https://www.airomania.eu/

Eastern European Machine Learning Summer School

6-14 July 2022, Vilnius Lithuania
(hybrid event)

https://www.eeml.eu/

- **BONUS** points for acceptance

- **Deep Learning** lectures
- Tutorial sessions
- Paper reading sessions
- Poster sessions
- Top speakers from **DeepMind**, **Google**, **McGill**, **Mila**, **Cambridge**, **Harvard**, **Oxford**, **Microsoft**, **Samsung**

- Application
  - **Extended abstract**
  - **Deadline: 7 April 2022**

AI / ML Community

Research Centers:

FMI: Marius Popescu, Bogdan Alexe, Radu Ionescu et al.

UPB: Traian Rebedea, Marius Leordeanu et al.

IMAR: Marius Leordeanu et al.

UT (Cluj): Lucian Buşoniu et al.

Bitdefender: **bit-ml.github.com** team

Siemens (Braşov): Lucian Itu et al.

ROVIS (Braşov): Sorin Grigorescu et al.

Industry:

Bitdefender: ML

Arnia: Outsourcing CV, NLP

Sparktech: Outsourcing CV, NLP

Xperi (FotoNation): CV

Siemens: Medical CV, ML

BOSCH (Cluj): Autonomous driving

Continental (Timişoara): Autonomous driving

Administrative

Program

- **Lectures**
  - Each Wednesday: 18:00 - 20:00
  - 3 speed tests
    - 5 minutes
    - 1-2 short questions
- **Labs**
  - Each Wednesday: 16:00 - 18:00

- **Course Materials**

https://drive.google.com/drive/folders/1l_ujnkkVxOSMxOa8IAuRpoNmaXzVraXF?usp=sharing

Assignments and Projects

- **2 Assignments (PyTorch)**
  - Related to the lab content
  - Submission deadline: 2 or 3 weeks after publishing
    - penalties for delays
  - Both of them will be presented in Week 9

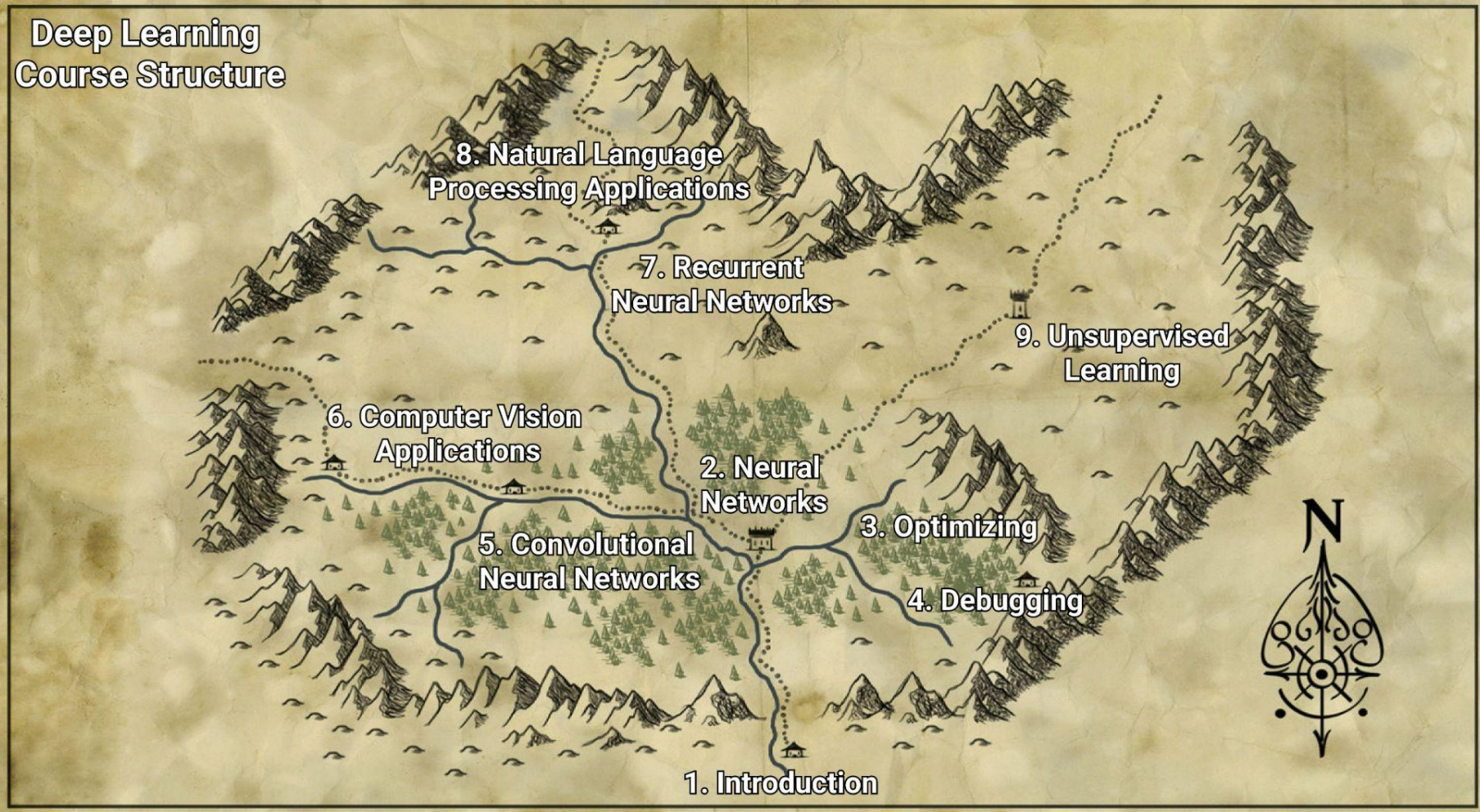
- **1 Project (PyTorch)**
  - Teams of 2 members
  - Subjects will be proposed in Week 4
  - Each student should choose a subject until Week 6
  - Will be presented in Week 10 - Poster Session

Course Structure

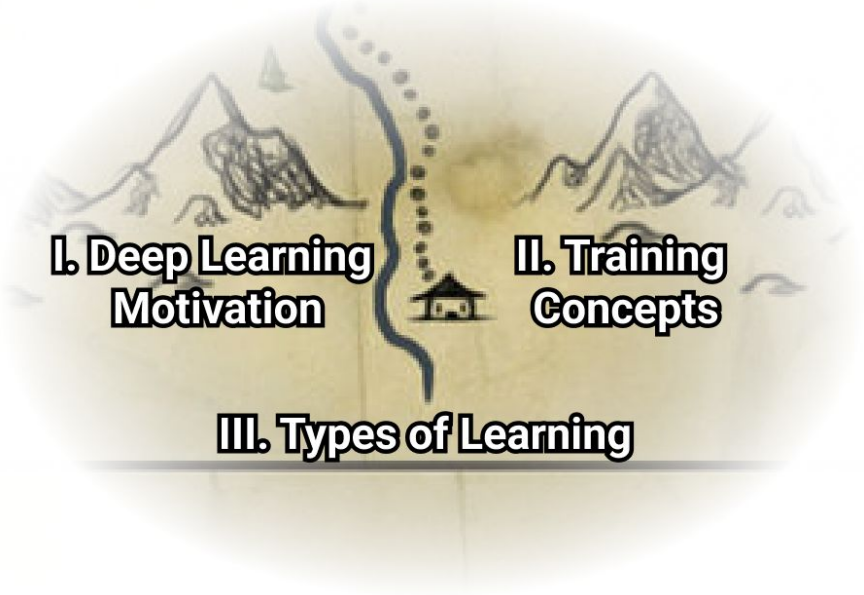
Week	Lecture	Speed Test	Laboratory	Event	Deadline
1	Introduction to Deep Learning		PyTorch and Tensors		
2	Neural Networks		Neural Networks		
3	Optimization		Training and Optimization	Assignment 1 out	
4	Debugging Neural Networks	Test 1	Debugging	Project proposals	
5	Convolutional Neural Networks		ConvNets (1)	Assignment 2 out	Assignment 1
6	Computer Vision Applications		ConvNets (2)		Project selection
7	Recurrent Neural Networks	Test 2	RNN		
8	Natural Language Processing Applications		NLP Applications		Assignment 2 EEML - April 7
9	Unsupervised Learning	Test 3	Assignments Evaluation		
10	Project Evaluation (Poster Session)		Project Evaluation		Project

Grading

		Requirements
Assignment 1	2 points	50%
Assignment 2	2 points	
3 x Speed Tests	1 point	
[**BONUS**] Good answers / questions during lectures and labs	max 2 points	
Project	5 points	50%
[**BONUS**] Good interventions / questions during the poster session	max 1 point	


Deep Learning
Course Structure
8. Natural Language
Processing Applications
7. Recurrent
Neural Networks
9. Unsupervised
Learning
6. Computer Vision
Applications
2. Neural
Networks
3. Optimizing
5. Convolutional
Neural Networks
4. Debugging
N
1. Introduction

Lecture Overview



I. Deep Learning Motivation

Artificial Intelligence

Artificial Intelligence

Any technique that enables computers to mimic human behavior

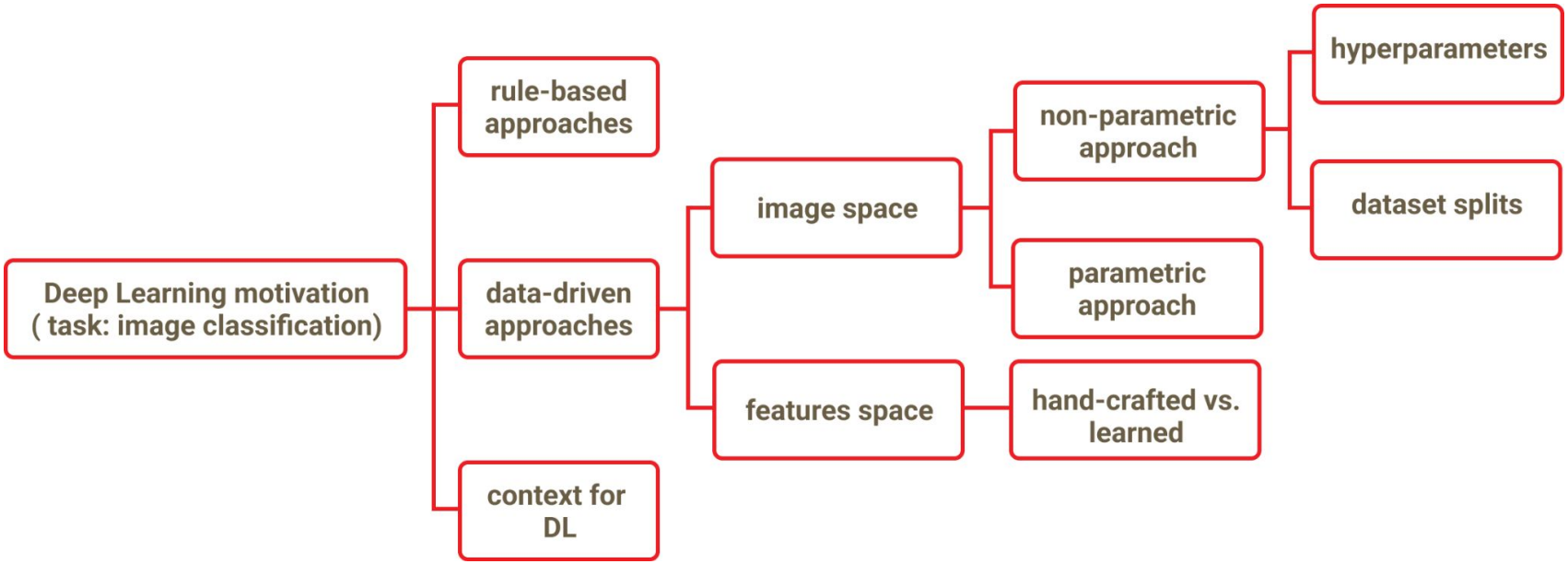
Machine Learning

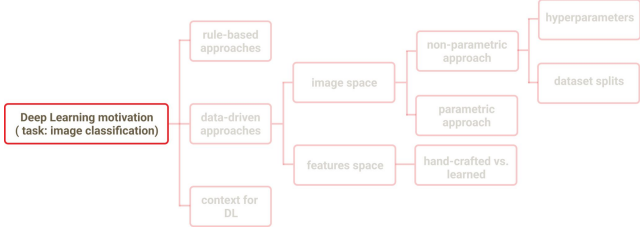
Ability to learn without explicitly being programmed

Deep Learning

Extract patterns from data using neural networks

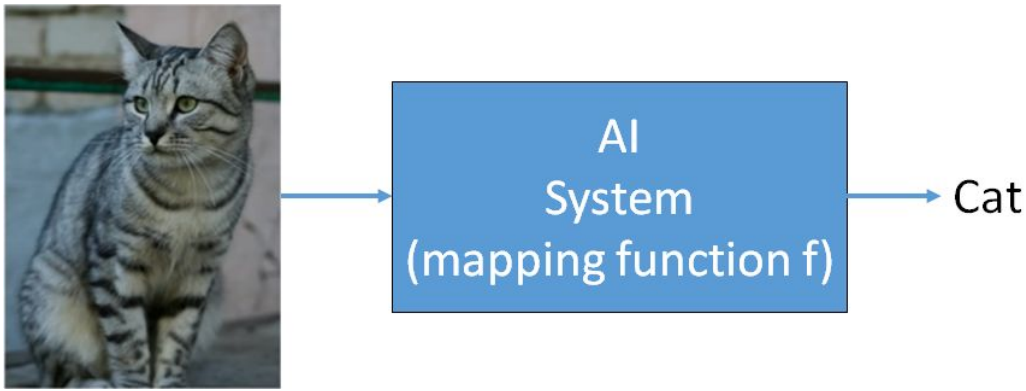
Deep Learning Motivation

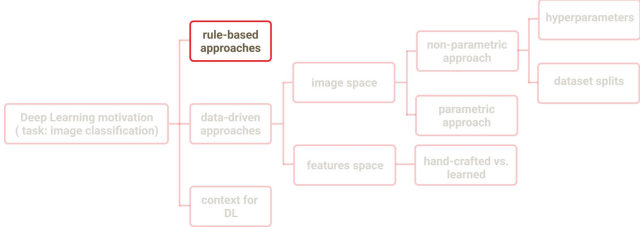




Task: Image Classification

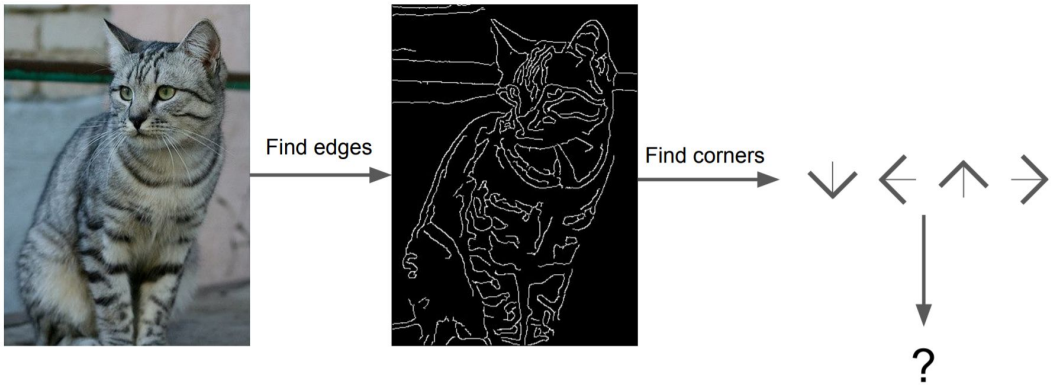
- Classify an image according to its visual content
- Consider a discrete set of labels
  - e.g. {dog, airplane, car, cat}



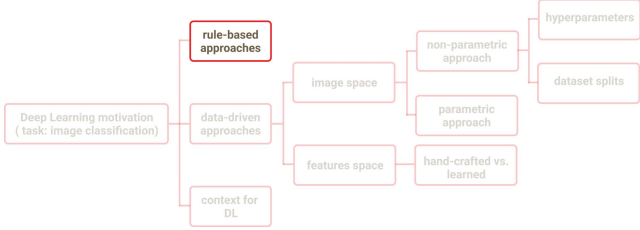


Rule-Based Approach

- Hand crafted features and rules



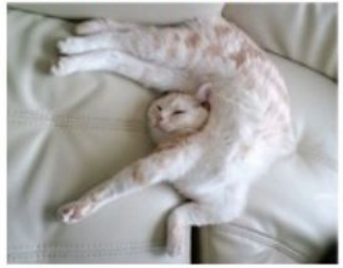
Challenges



Viewpoint variation

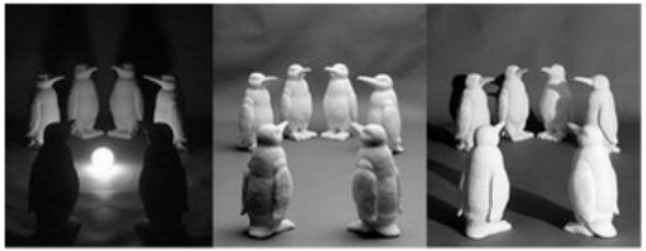
Scale variation

Deformation



Occlusion

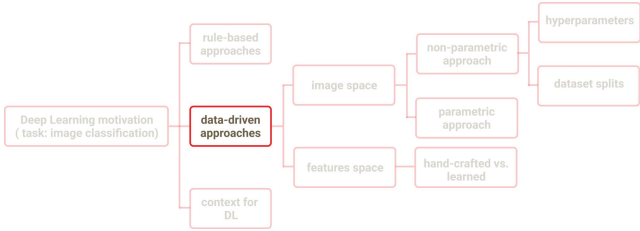
Illumination conditions



Background clutter

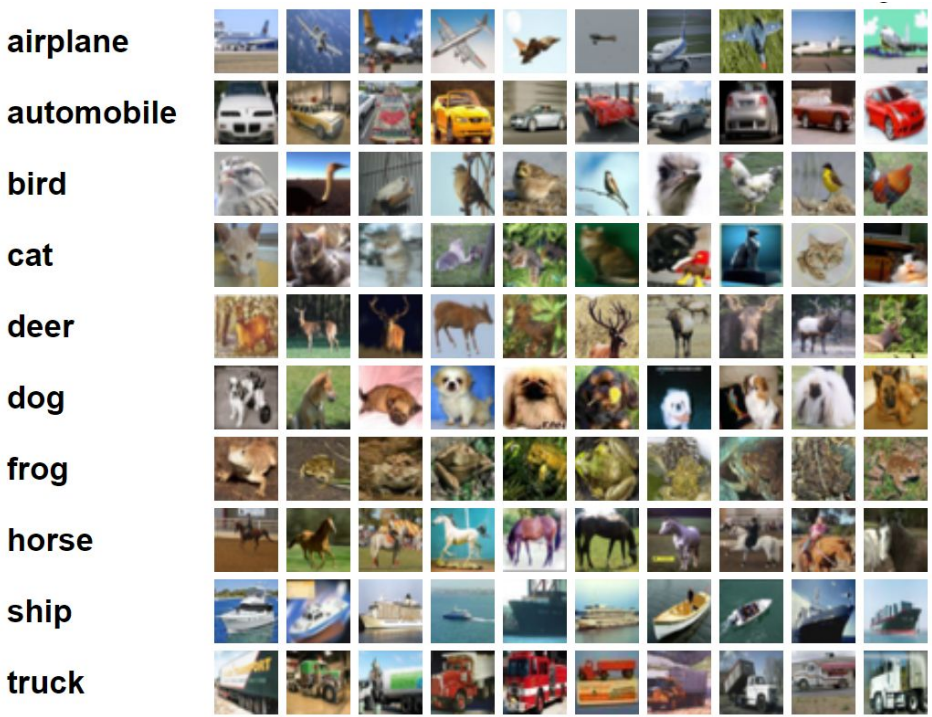
Intra-class variation

Data-Driven Approach



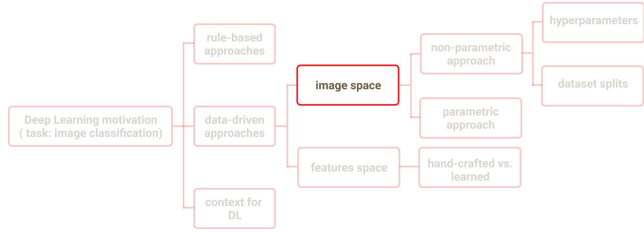
- Dataset
  - with images and associated labels

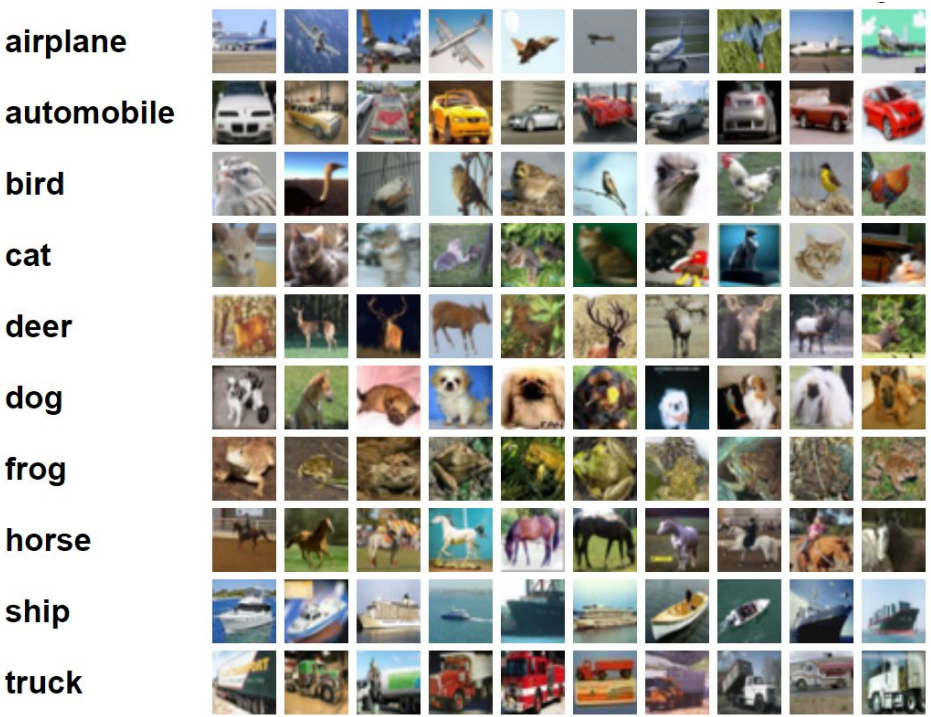
- CIFAR-10
  - 10 classes
  - 60 000 examples
  - 32 x 32 x 3



CIFAR-10

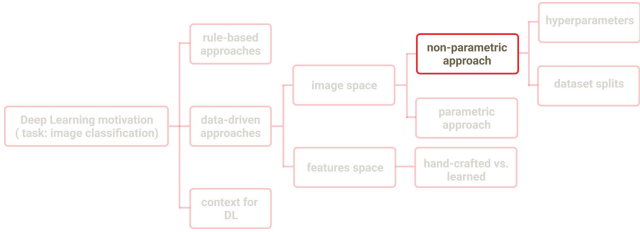
Image Space



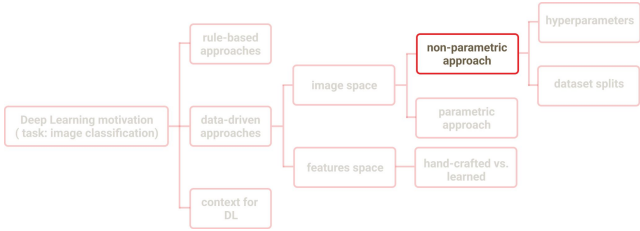


CIFAR-10

Image Space
Non-Parametric Approach
K-Nearest Neighbor Classifier



- Memorize all examples and corresponding labels
- Predict label of top K most similar examples
- Define distance between two images

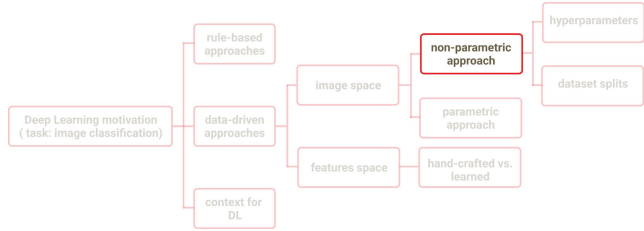


Distance metric

L1 distance: $d(I_1, I_2) = \sum_x |I_{1;x} - I_{2;x}|$, where $I_{1;x}$ is the x'th element of image I_1

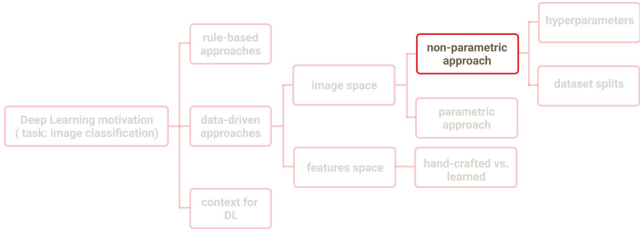
test image					training image					pixel-wise absolute value differences				
56	32	10	18	-	10	20	24	17	=	46	12	14	1	→ 456
90	23	128	133		8	10	89	100		82	13	39	33	
24	26	178	200		12	16	178	170		12	10	0	30	
2	0	255	220		4	32	233	112		2	32	22	108	

K-Nearest Neighbor Classifier

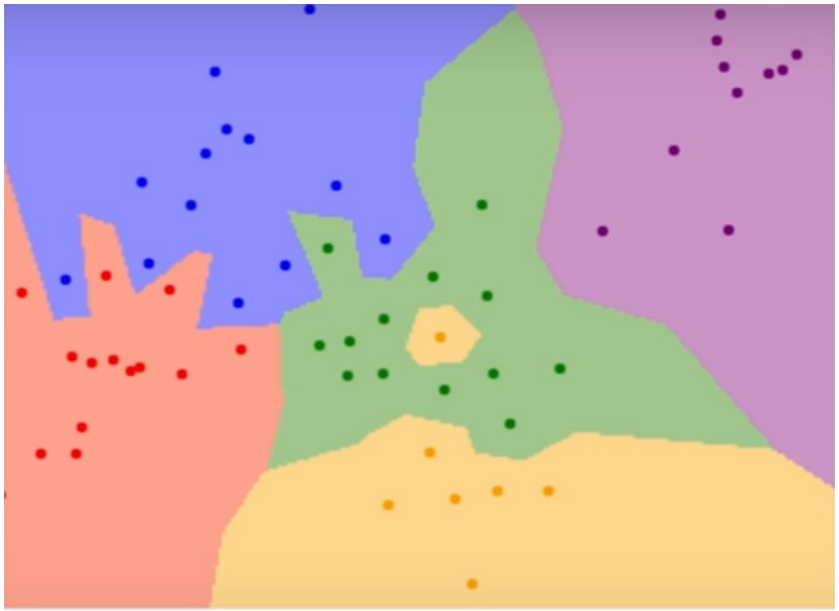


- K=1

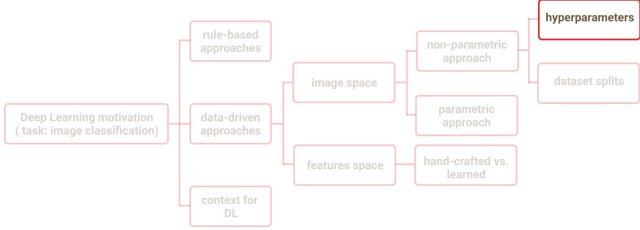
K-Nearest Neighbor Classifier



- Decision regions
  - K = 1

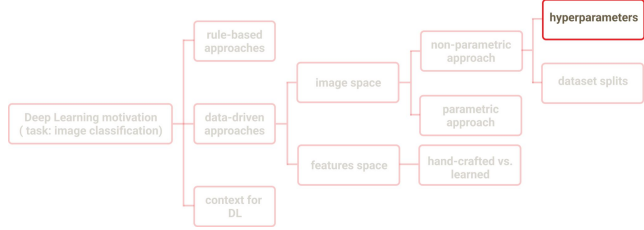


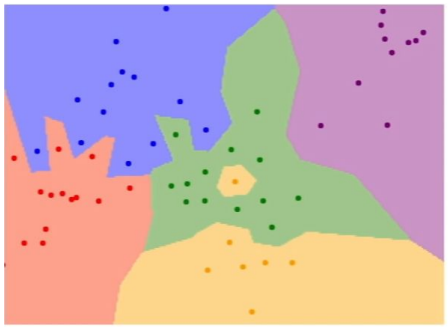
K-Nearest Neighbor Classifier



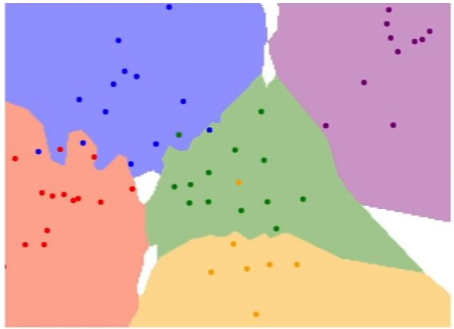
- Hyperparameters
  - K
  - Distance metric

Hyperparameters: K

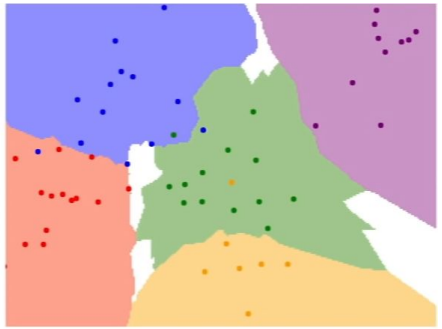




K = 1

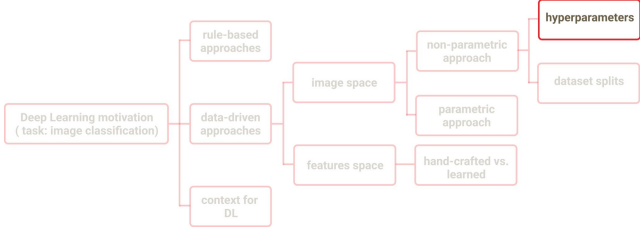


K = 3



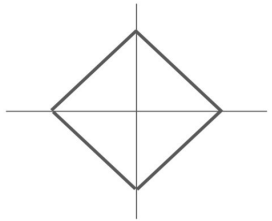
K = 5

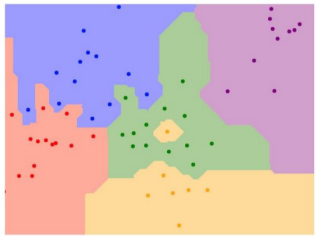
Hyperparameter: distance metric



L1 (Manhattan) distance

$$d_1(I_1, I_2) = \sum_x |I_{1;x} - I_{2;x}|$$

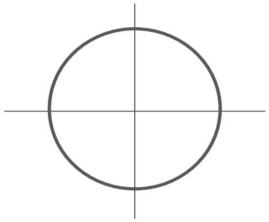


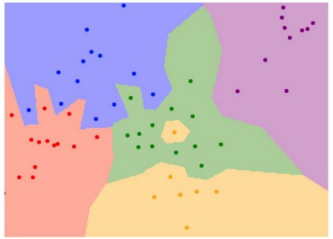


K = 1

L2 (Euclidean) distance

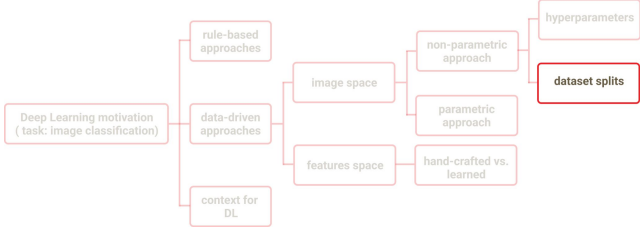
$$d_1(I_1, I_2) = \sqrt{\sum_x (I_{1;x} - I_{2;x})^2}$$



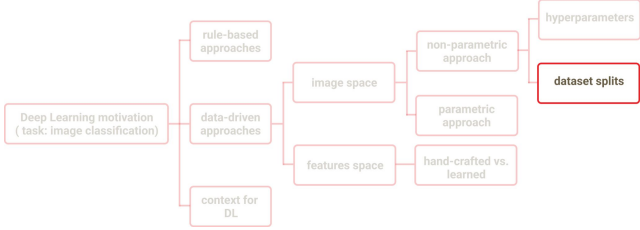


K = 1

Dataset splits

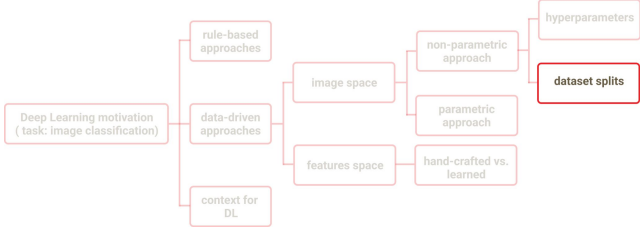


- Only train data



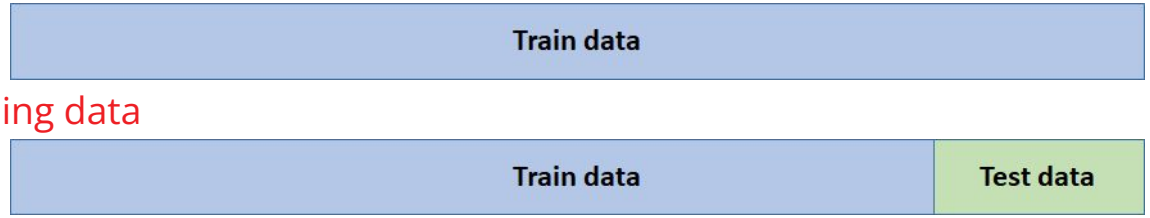
Dataset splits

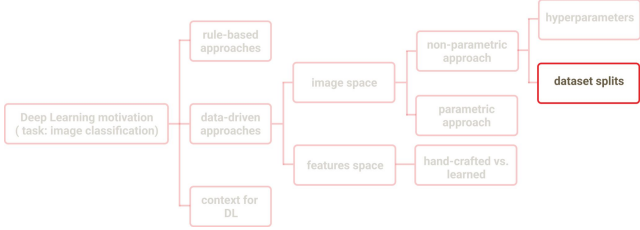
- Only train data
  - K=1 - works best for training data



Dataset splits

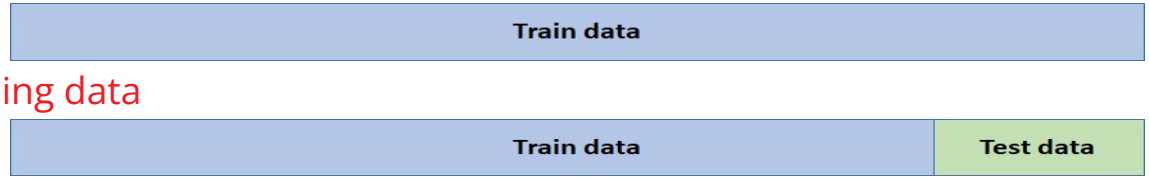
- Only train data
  - K=1 - works best for training data
- Train data & Test data

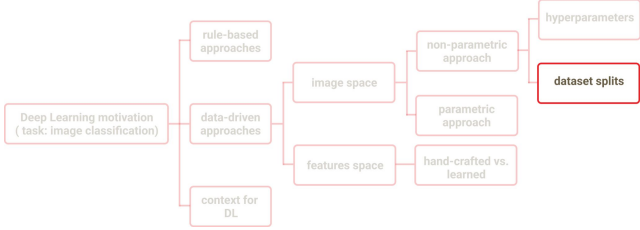




Dataset splits

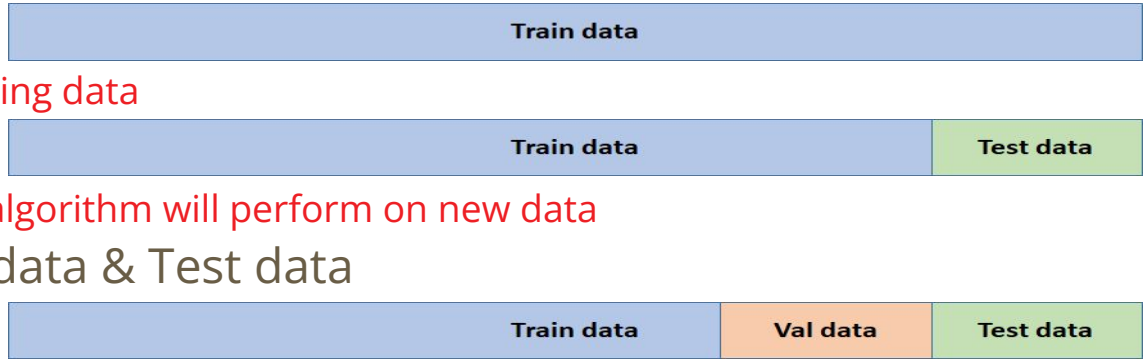
- Only train data
  - K=1 - works best for training data
- Train data & Test data
  - No intuition on how the algorithm will perform on new data

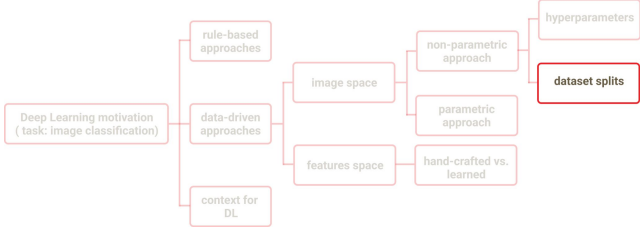




Dataset splits

- Only train data
  - K=1 - works best for training data
- Train data & Test data
  - No intuition on how the algorithm will perform on new data
- Train data & Validation data & Test data





Dataset splits

- Only train data
  - K=1 - works best for training data

- Train data & Test data
  - No intuition on how the algorithm will perform on new data

- **Train data & Validation data & Test data**

- **Cross Validation**

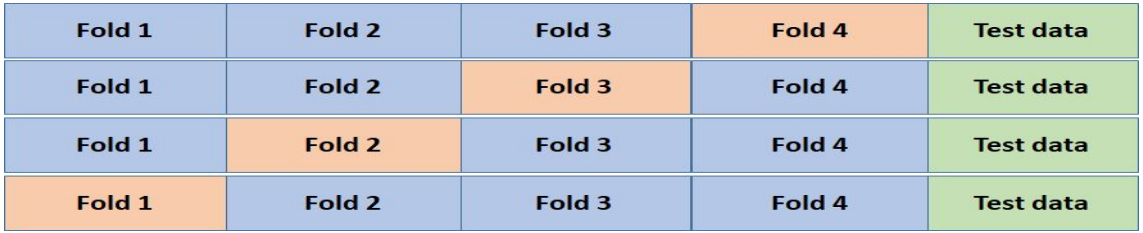
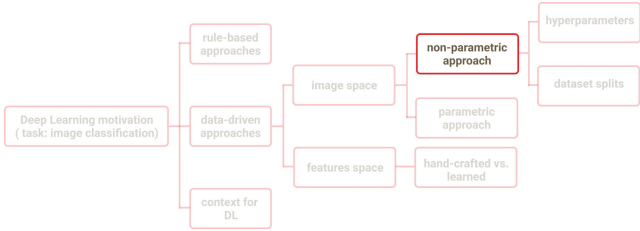
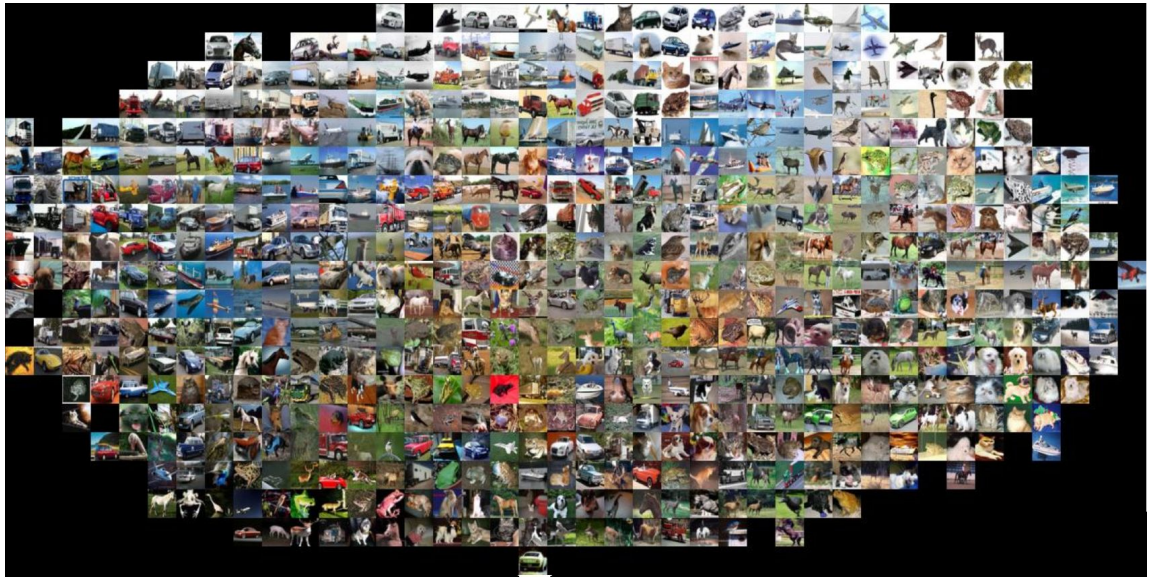


Image Space K-Nearest Neighbor



- Issues
  - Pixel-wise distances are not informative
  - Train: O(1)
  - Test: O(n)



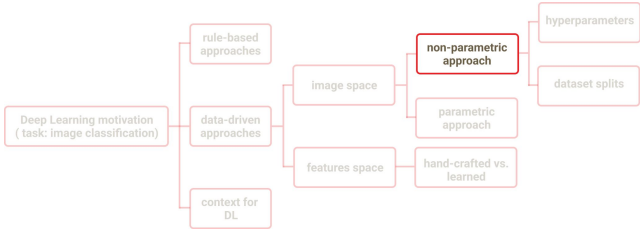
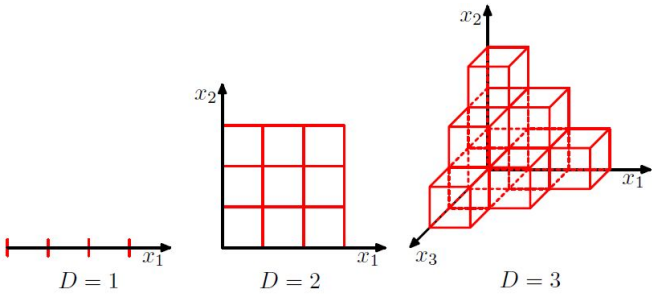
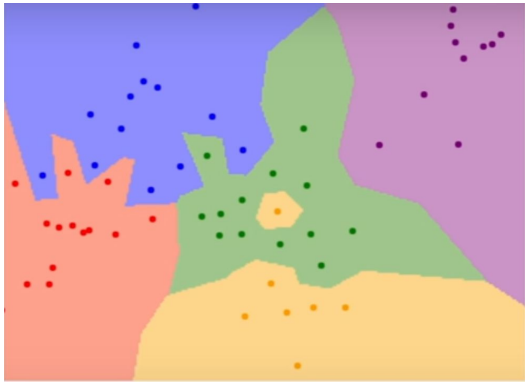


Image Space K-Nearest Neighbor

- Issues
  - Curse of dimensionality





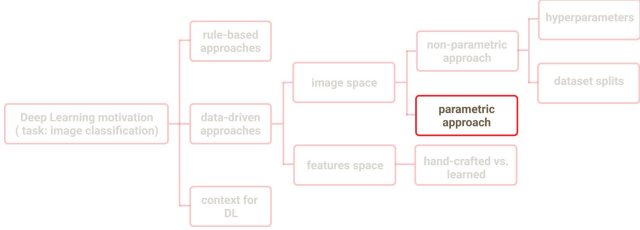
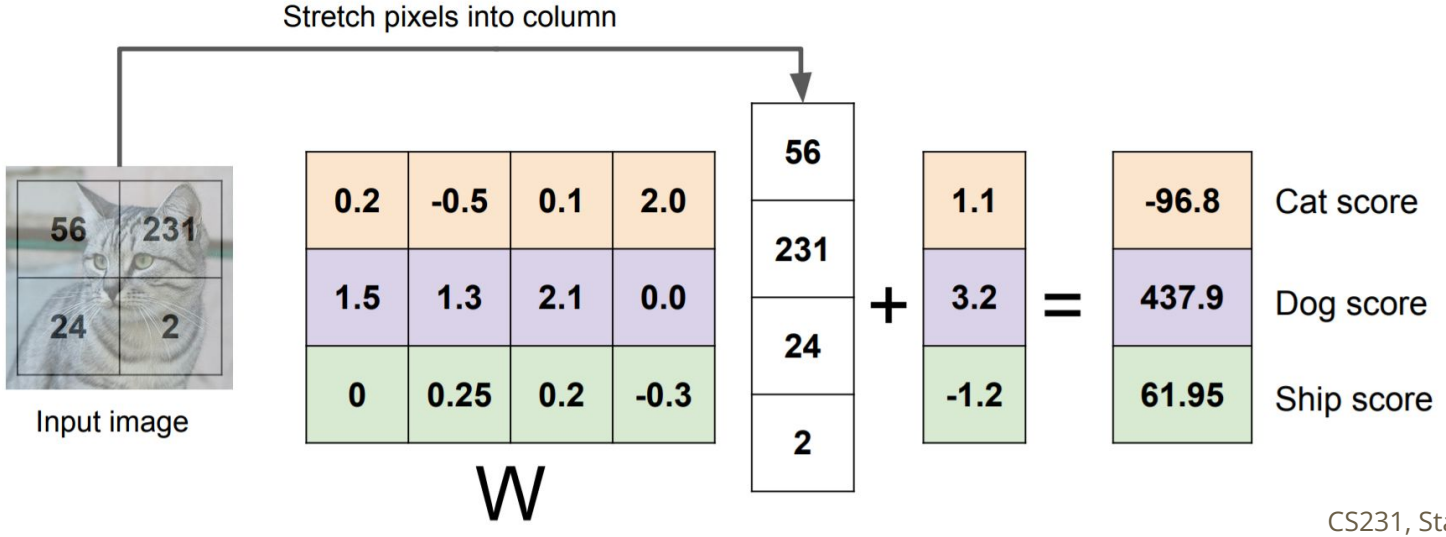
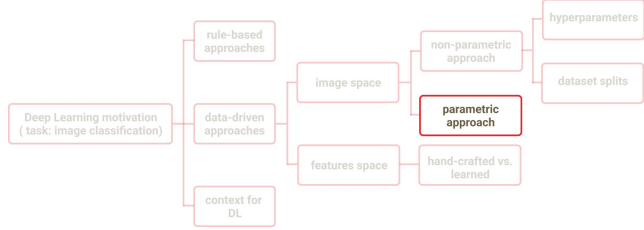


Image Space
Parametric approach - Linear Classifier

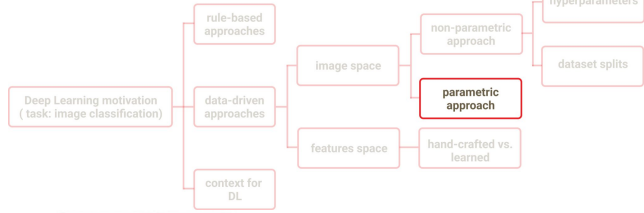
- f(x,**W**)=**W**x + b, **W** - parameters to be learned

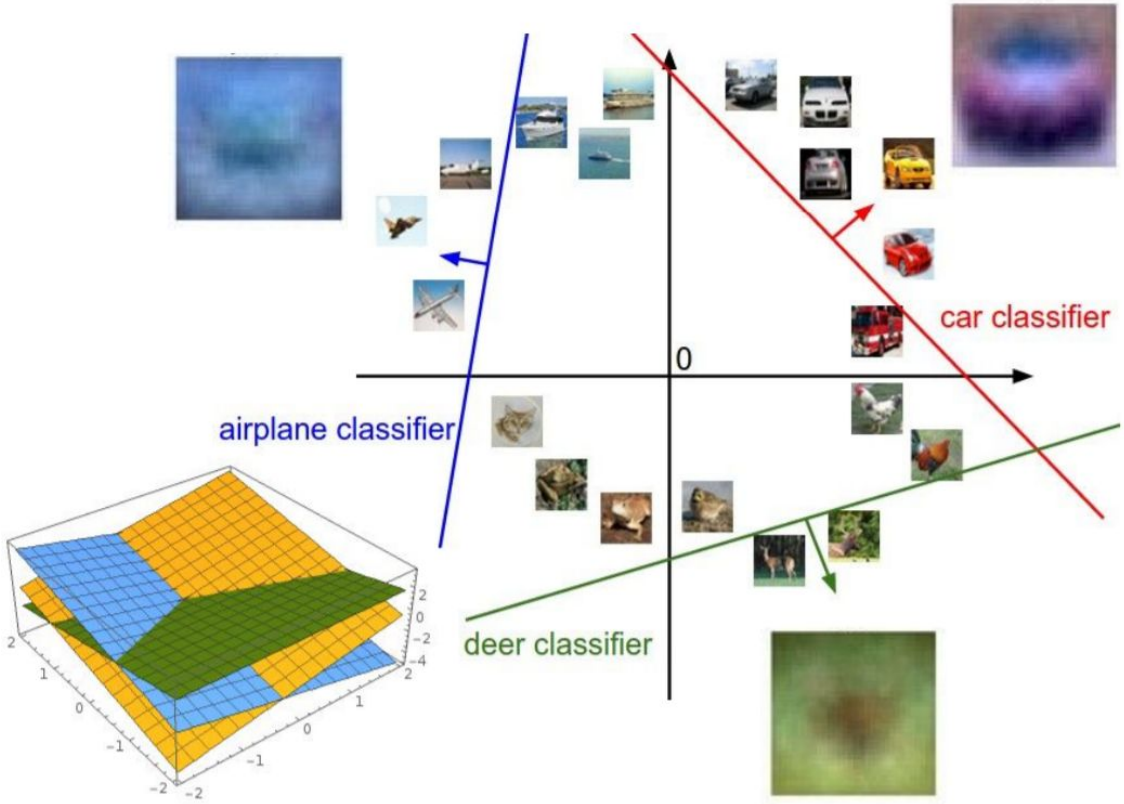


Linear Classifier - Interpretation

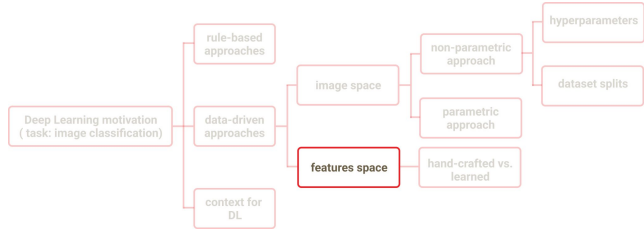


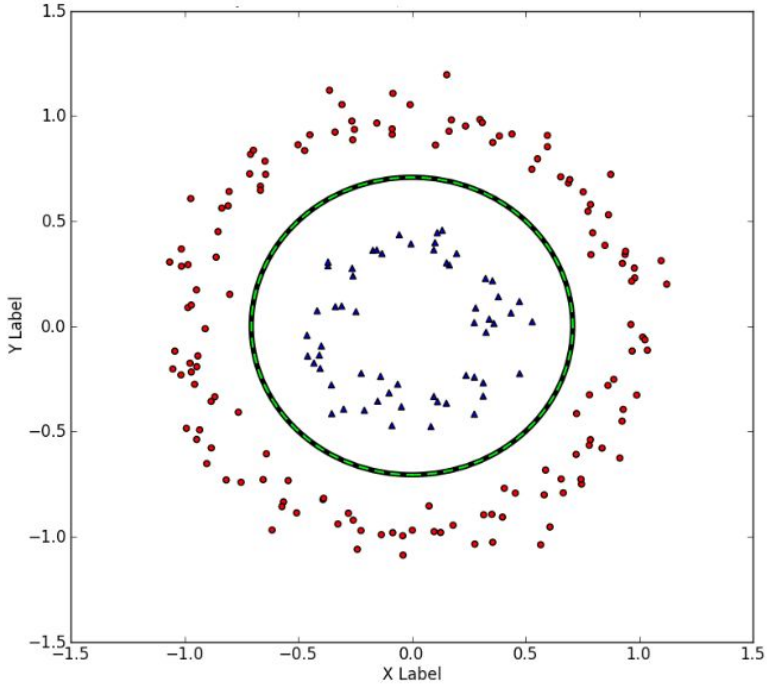
Linear Classifier - Interpretation

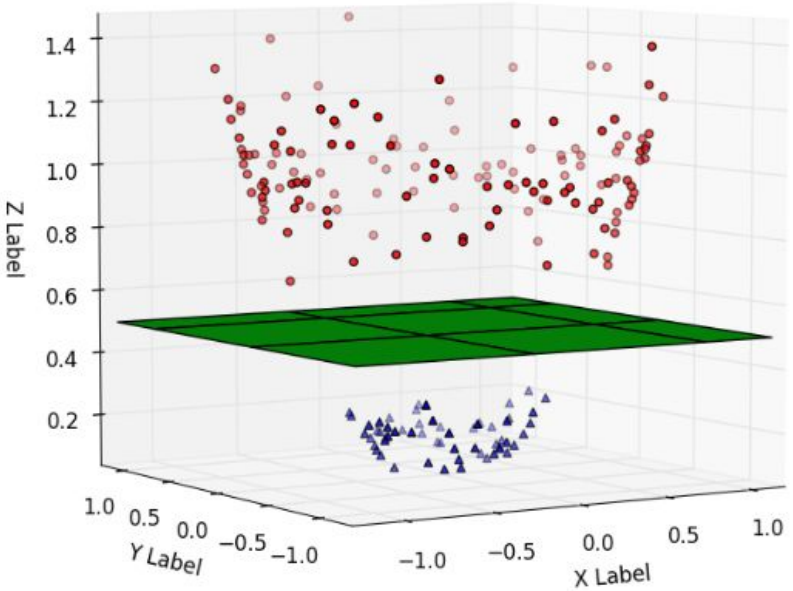




Features Space - Motivation







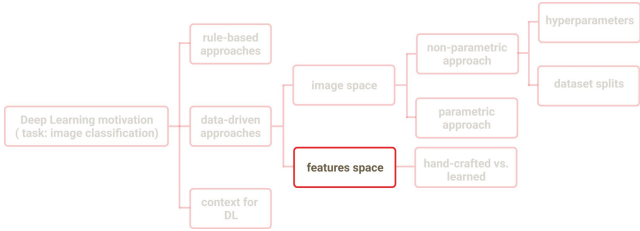
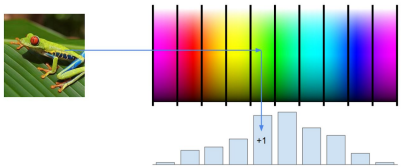


Image Space => Features Space

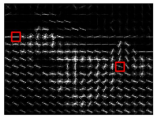
- Hand-crafted

Example: Color Histogram



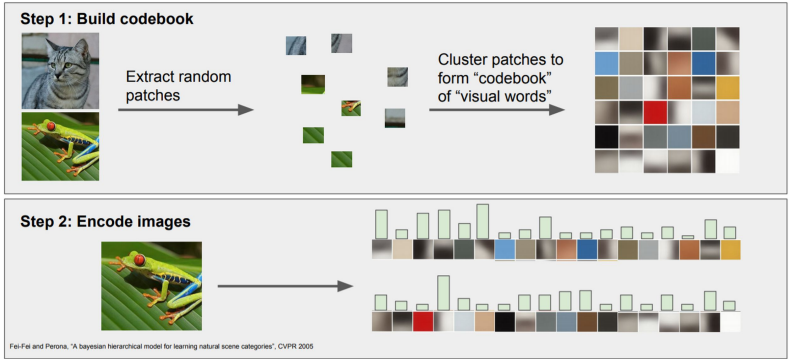
Example: Histogram of Oriented Gradients (HoG)

Divide image into 8x8 pixel regions
Within each region quantize edge direction into 9 bins

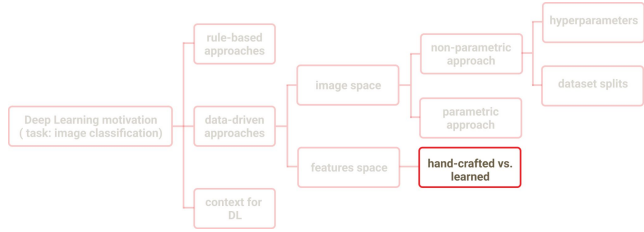


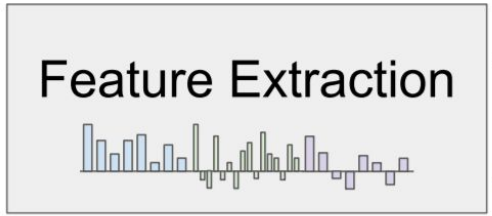
Example: 320x240 image gets divided into 40x30 bins; in each bin there are 9 numbers so feature vector has 30*40*9 = 10,800 numbers

Example: Bag of Words



Hand-Crafted vs. Learned Features

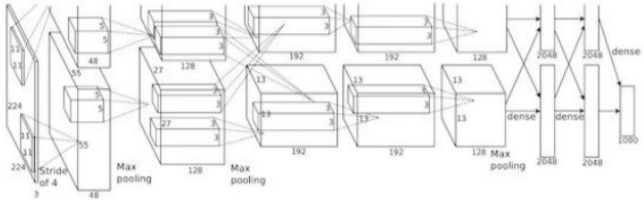




f

10 numbers giving scores for classes

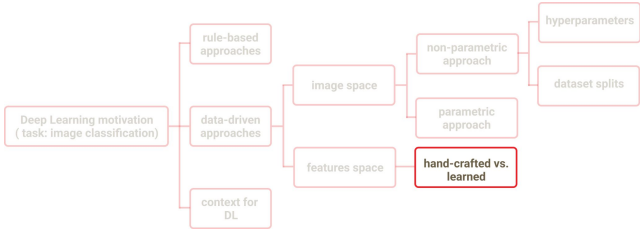
training



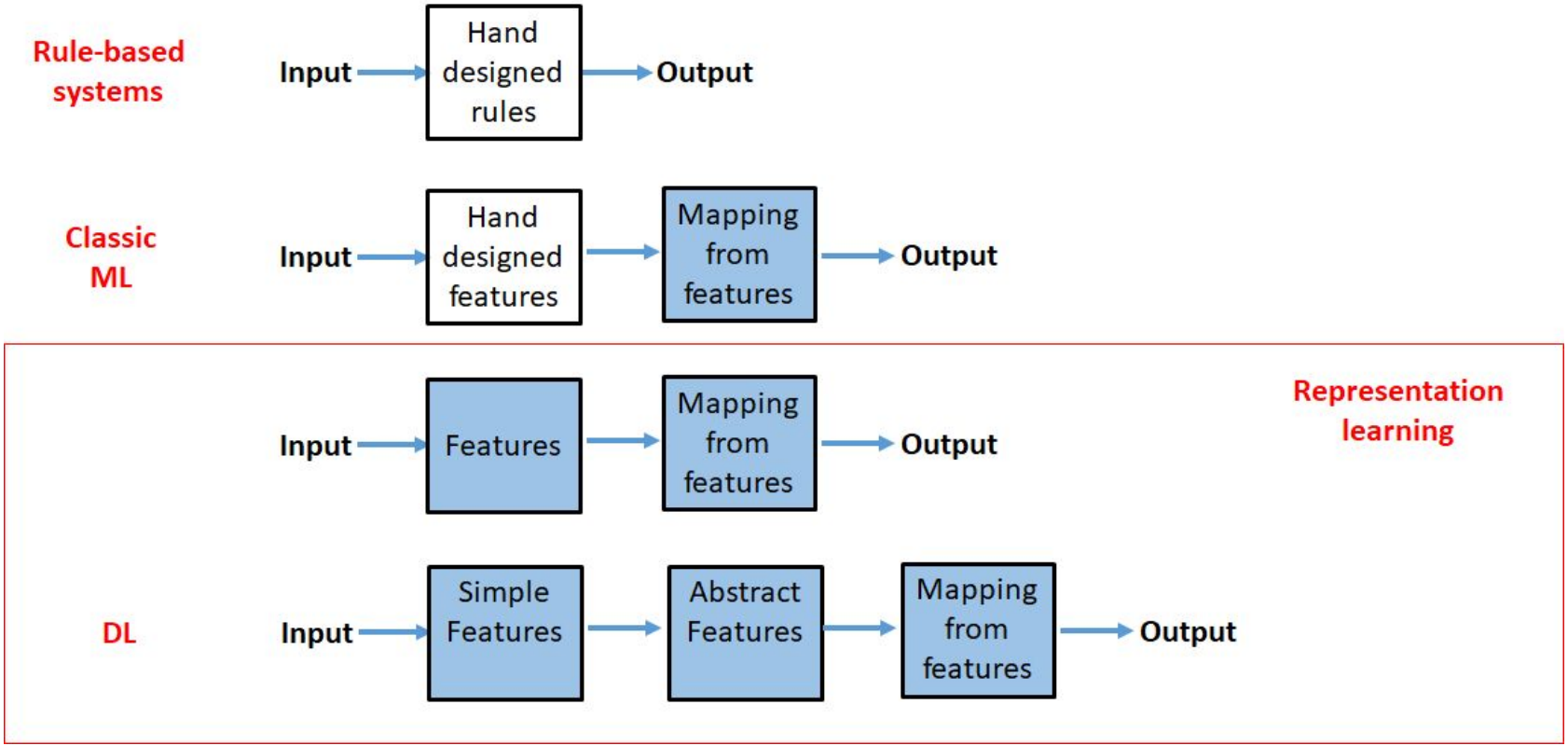
Krizhevsky, Sutskever, and Hinton, "Imagenet classification with deep convolutional neural networks", NIPS 2012. Figure copyright Krizhevsky, Sutskever, and Hinton, 2012. Reproduced with permission.

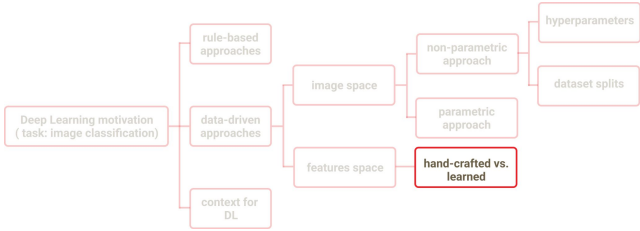
10 numbers giving scores for classes

training



Hand-Crafted vs. Learned Features

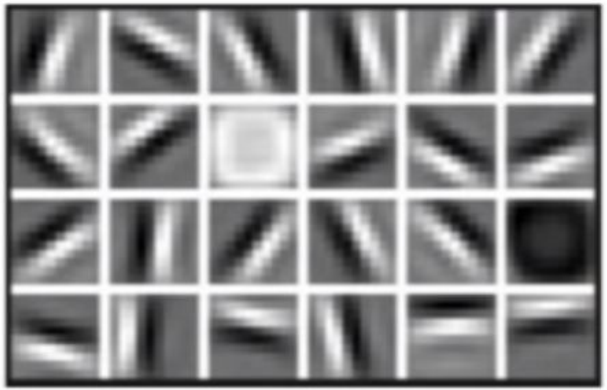




Why Deep Learning?

- Learn underlying features directly from data

Low Level Features



Lines & Edges

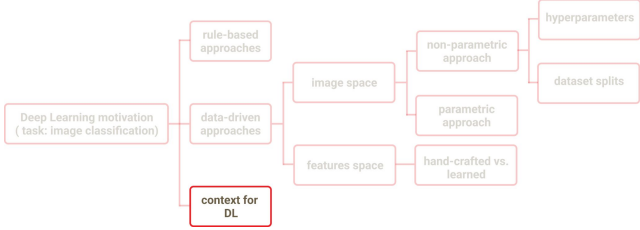
Mid Level Features

Eyes & Nose & Ears

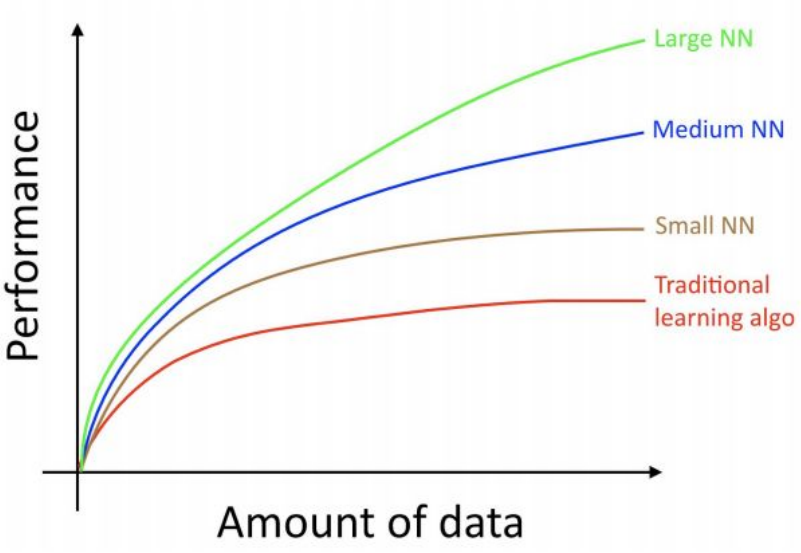
High Level Features

Facial Structure

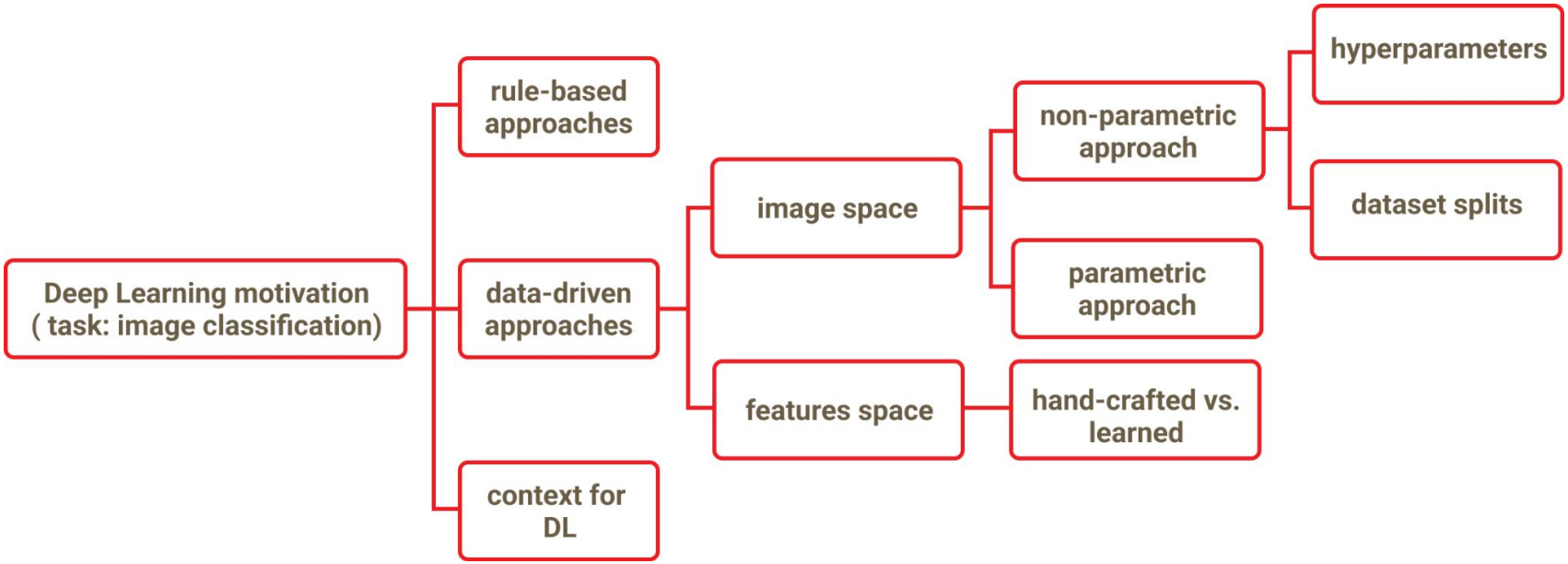
Context for Deep Learning



- Big data
  - Larger datasets
- Hardware
  - GPUs
  - Massively parallelizable algorithms
- Deep Learning Frameworks
  - PyTorch
  - TensorFlow
  - Second Layer Libs:
    - Keras - over TensorFlow
    - Fast.ai - over PyTorch

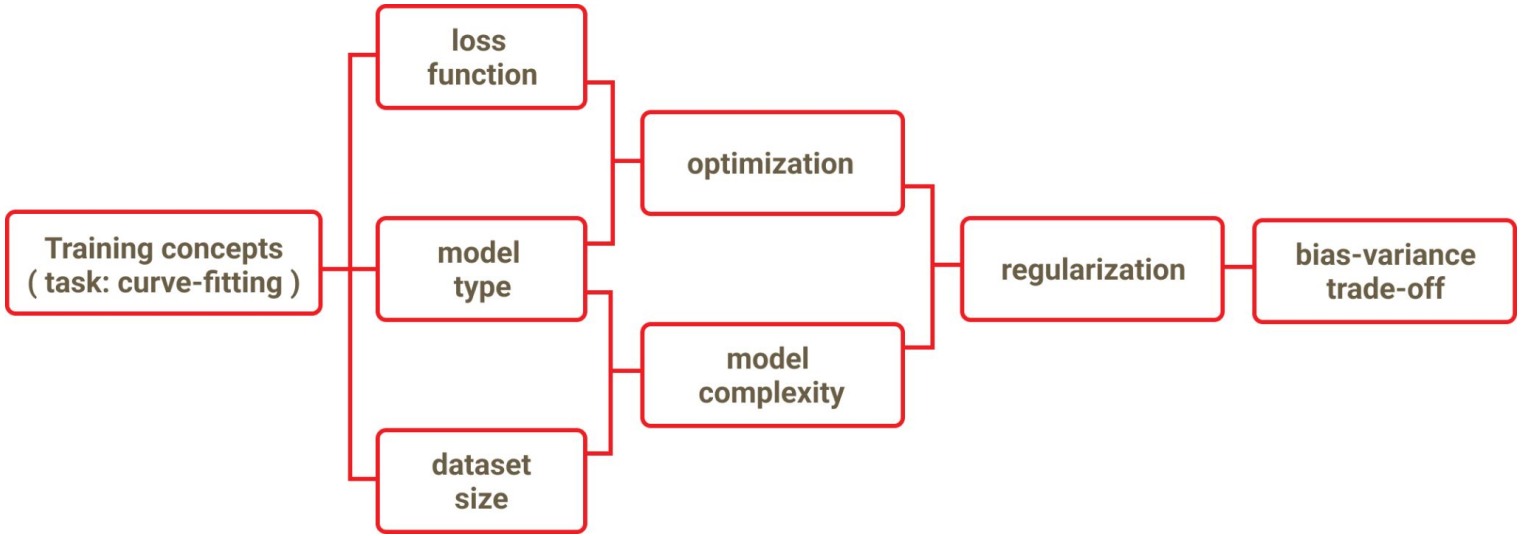


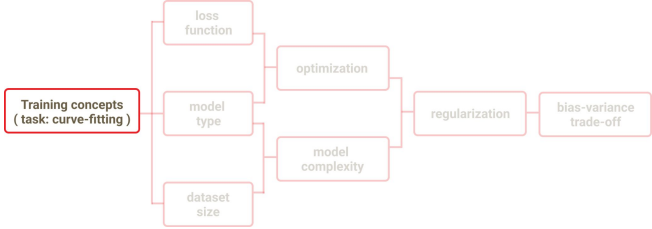
Recap - Deep Learning Motivation



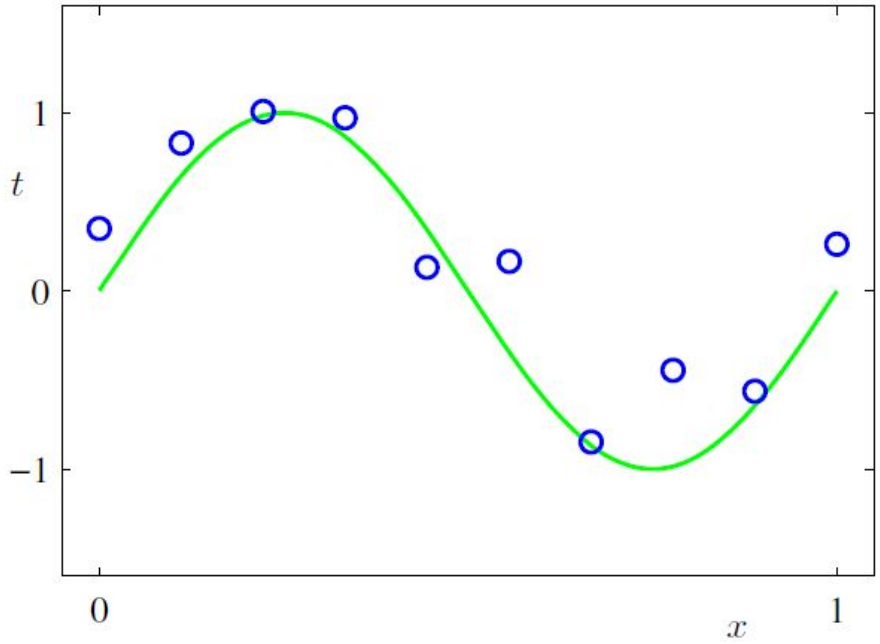
II. Training Concepts

Training Concepts





Task: Curve-Fitting



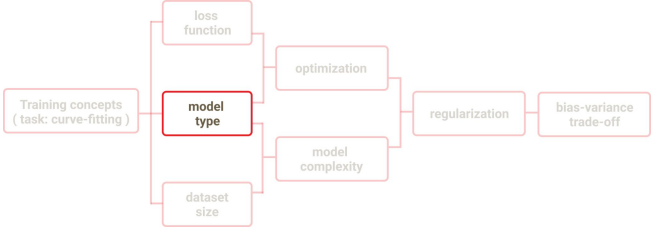
- Training set: $(x_1, t_1), (x_2, t_2), \ldots, (x_N, t_N)$
  - $x_i,\ t_i\ \in \mathbb{R}$

Data generating process: $\sin(2\pi x) + noise$

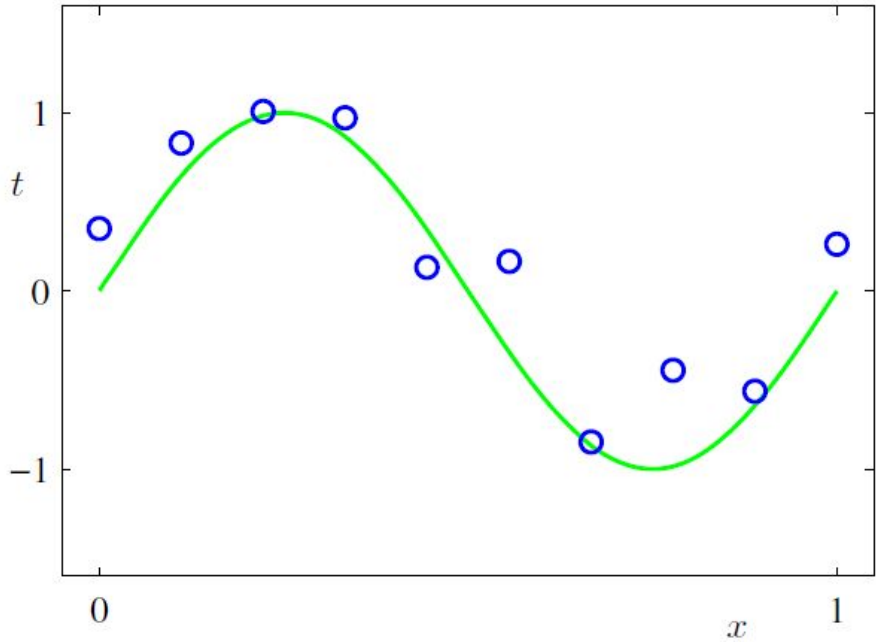
- Underlying regularity which we wish to learn
- Individual observations are corrupted by random noise

- **Goal**: predict $\hat{t}$ for some new $\hat{x}$
  - regression problem

Pattern Recognition, Christopher M. Bishop



Polynomial Curve Fitting



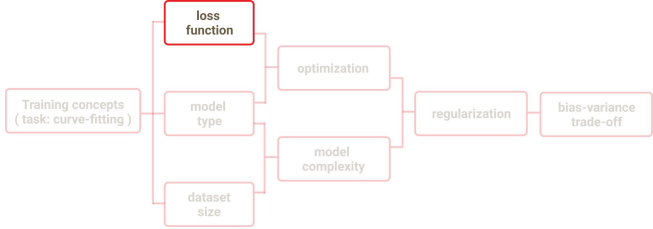
- Fit the data using a polynomial function

$$y(x, \mathrm{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M$$

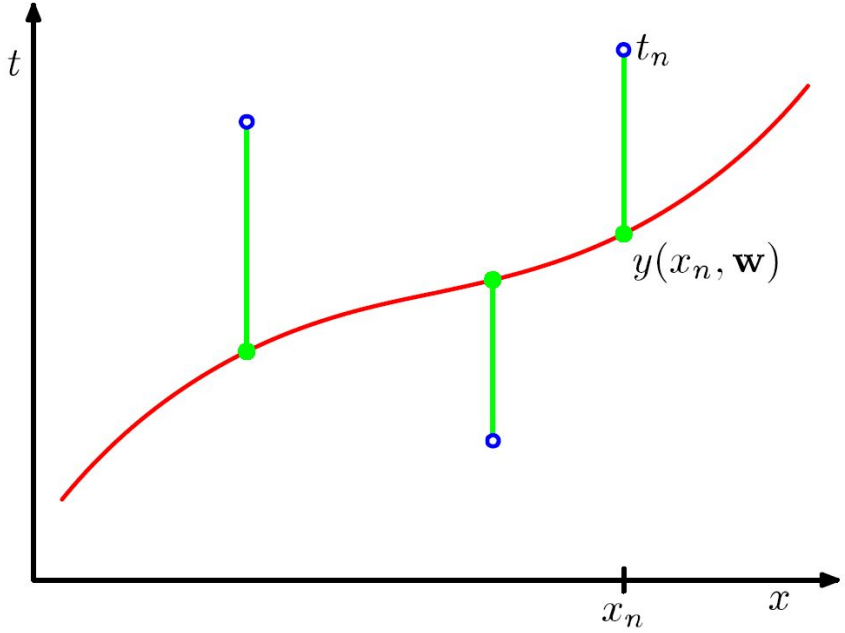
- M - order of the polynomial - model complexity
- w - parameters to be learned

- y - nonlinear function of x
- y - linear function of the parameters w

Pattern Recognition, Christopher M. Bishop



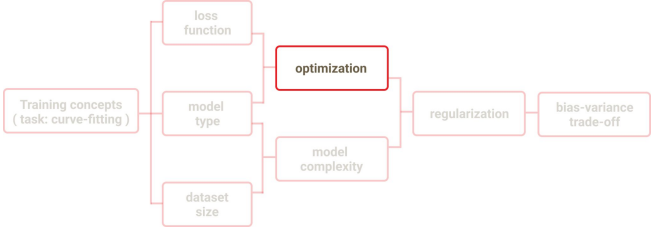
Loss Function



- Sum of squares error

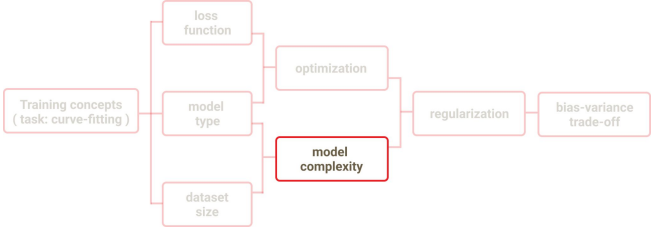
$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\left(y(x_n, \mathbf{w}) - t_n\right)^2$$

Closed Form Solution

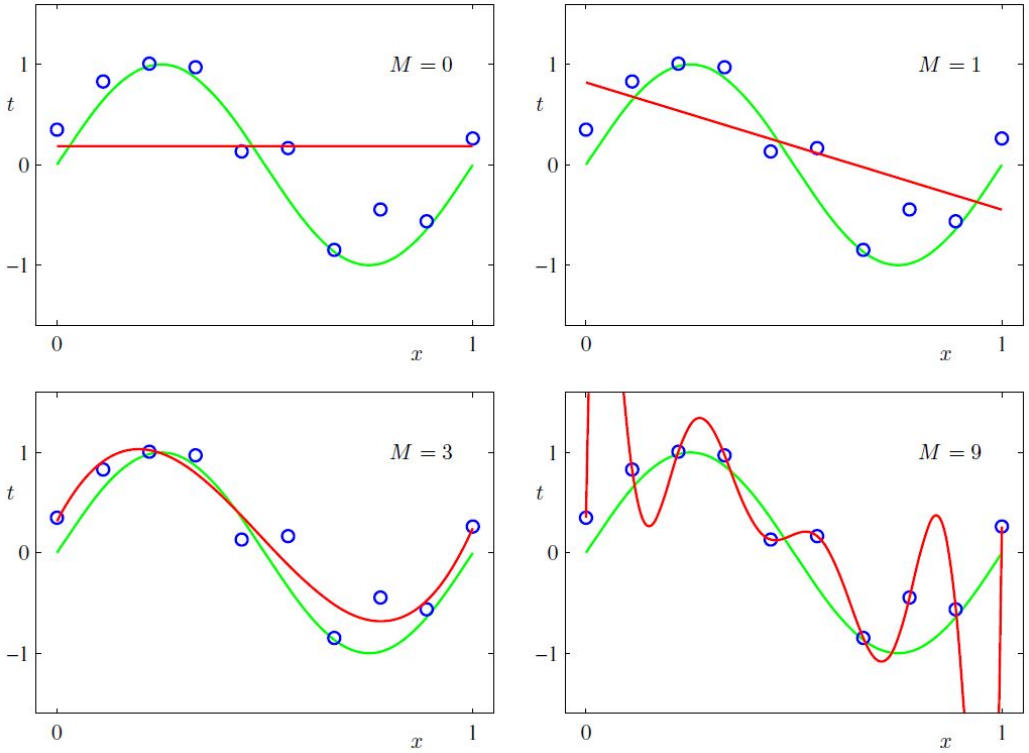


- Goal is to minimize $E(\mathbf{w})$

- $E(\mathbf{w})$ quadratic function of the parameters
  - Derivative w.r.t. parameters is linear in the parameters
  - Unique solution $\mathbf{w}^*$ can be found in closed form

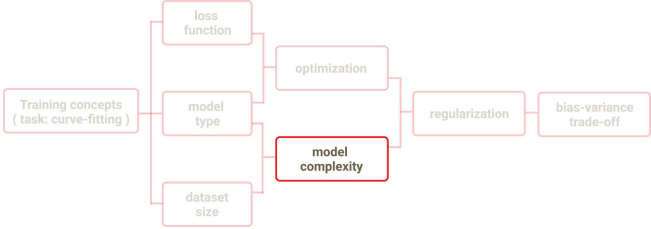


Model Complexity



$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M$$

Pattern Recognition, Christopher M. Bishop

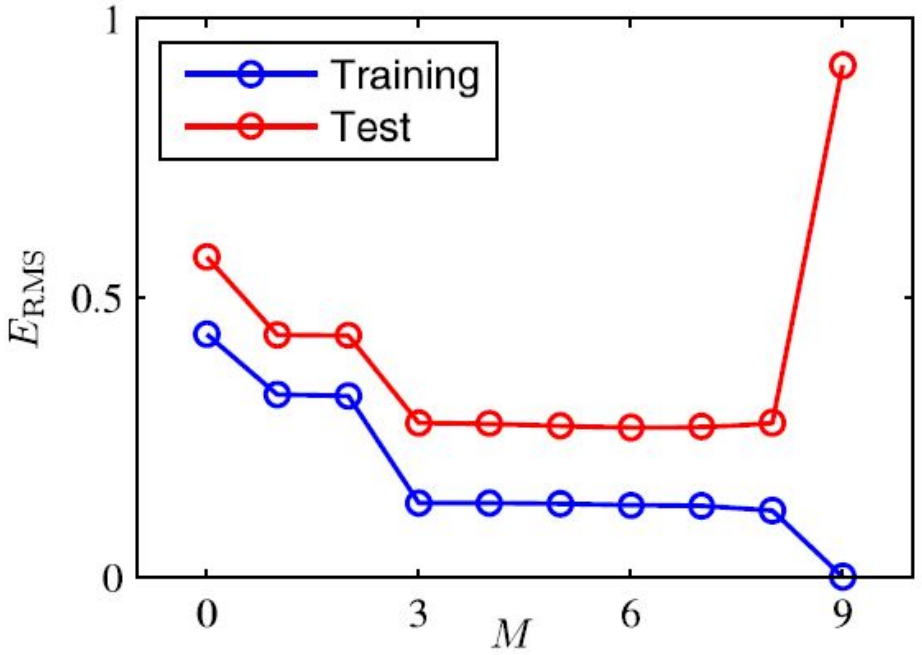


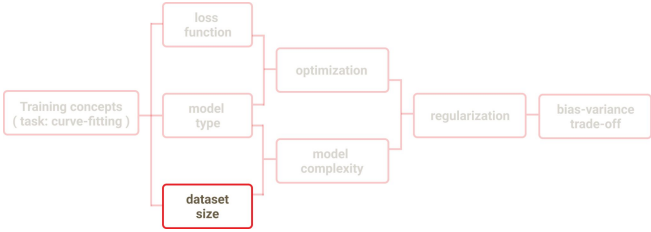
Model Complexity

	$M=0$	$M=1$	$M=6$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

Root-Mean-Square Error:

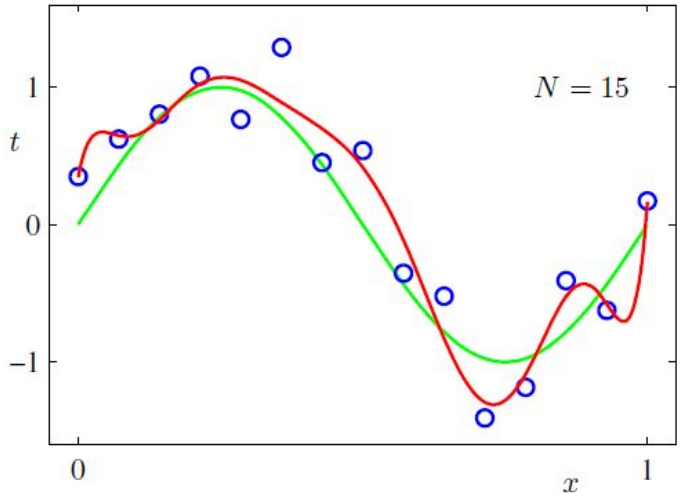
$$E_{RMS} = \sqrt{\frac{2E(\mathbf{w}^*)}{N}}$$

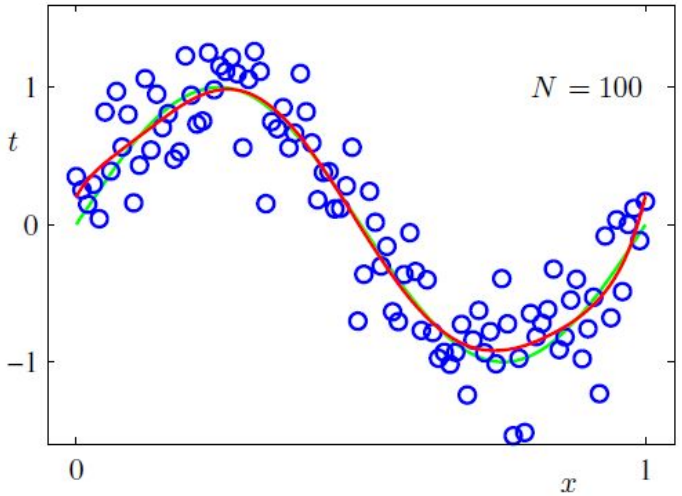




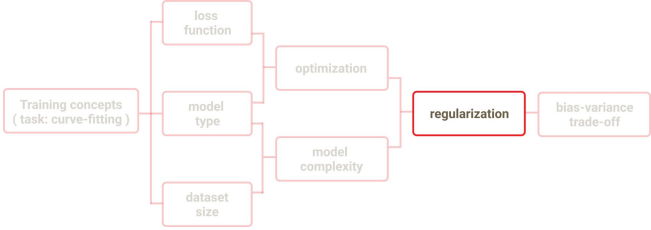
Increase training set

- M = 9





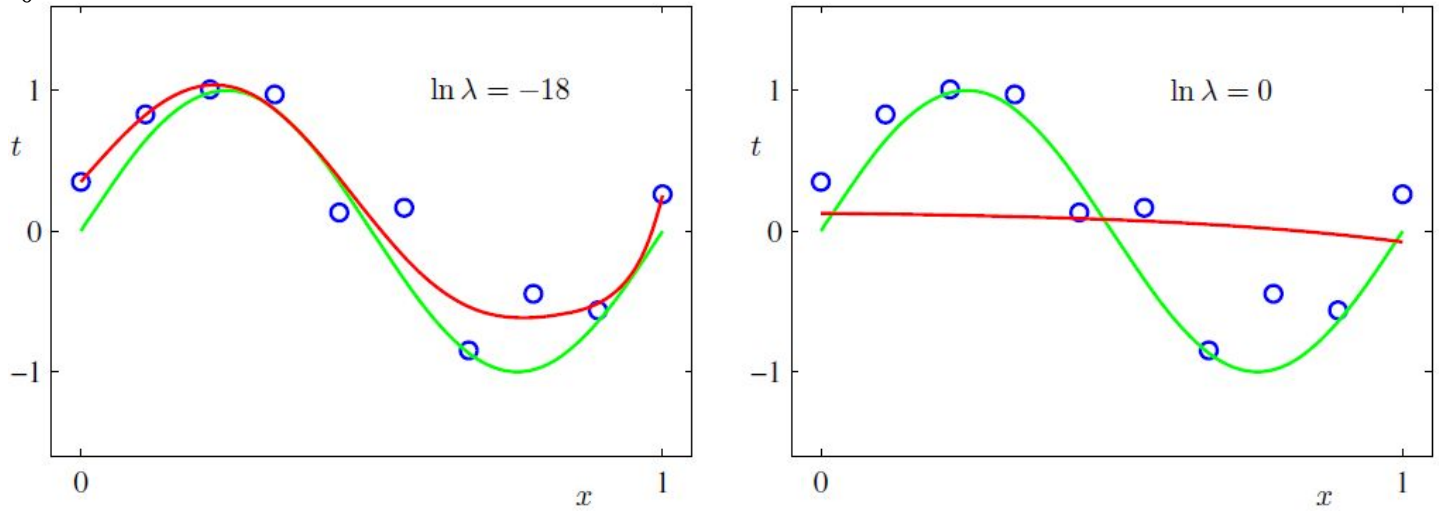
Pattern Recognition, Christopher M. Bishop



Regularization

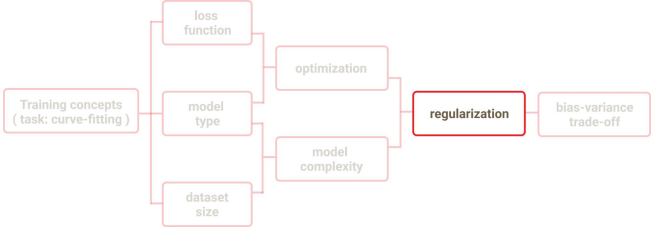
$$\tilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(y(x_n,\mathbf{w}) - t_n)^2 + \frac{\lambda}{2}||\mathbf{w}||^2)$$

$$||\mathbf{w}||^2 = \sum_{i=0}^{M} w_i^2$$

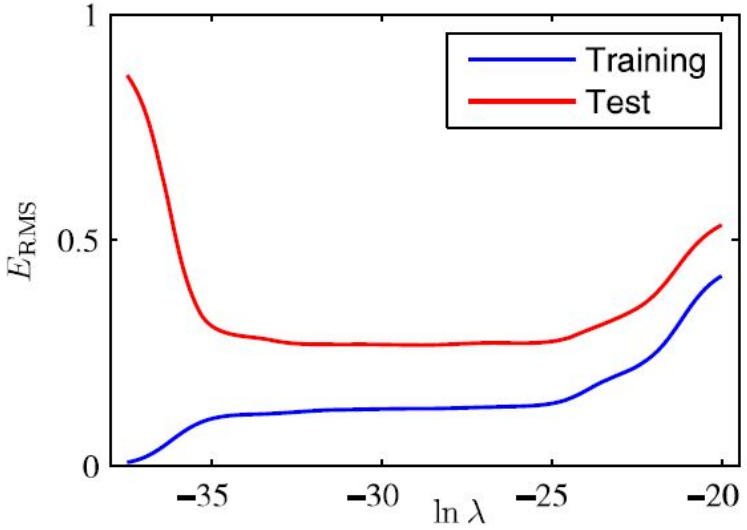


Pattern Recognition, Christopher M. Bishop

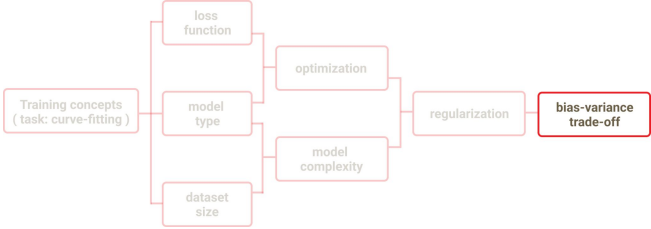
Regularization



	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01



Pattern Recognition, Christopher M. Bishop

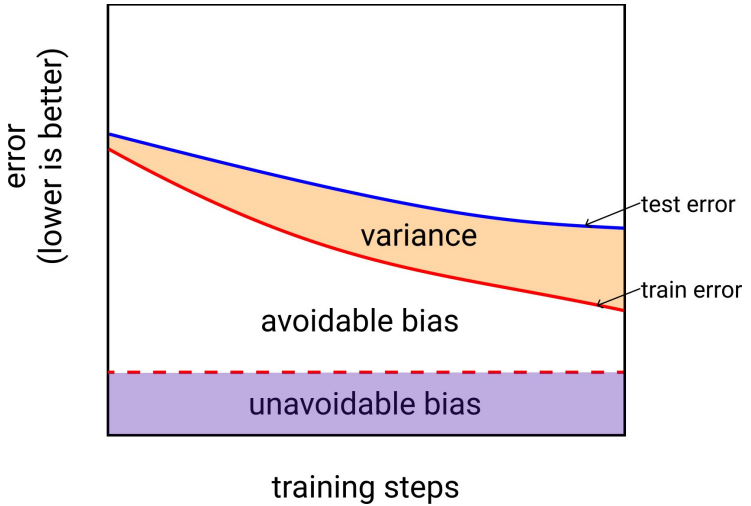


Bias and Variance

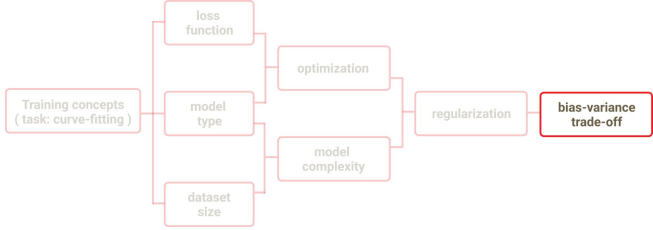
- Algorithm's Bias
  - Error on training set
- Algorithm's Variance
  - Performance decrease between train set and test set

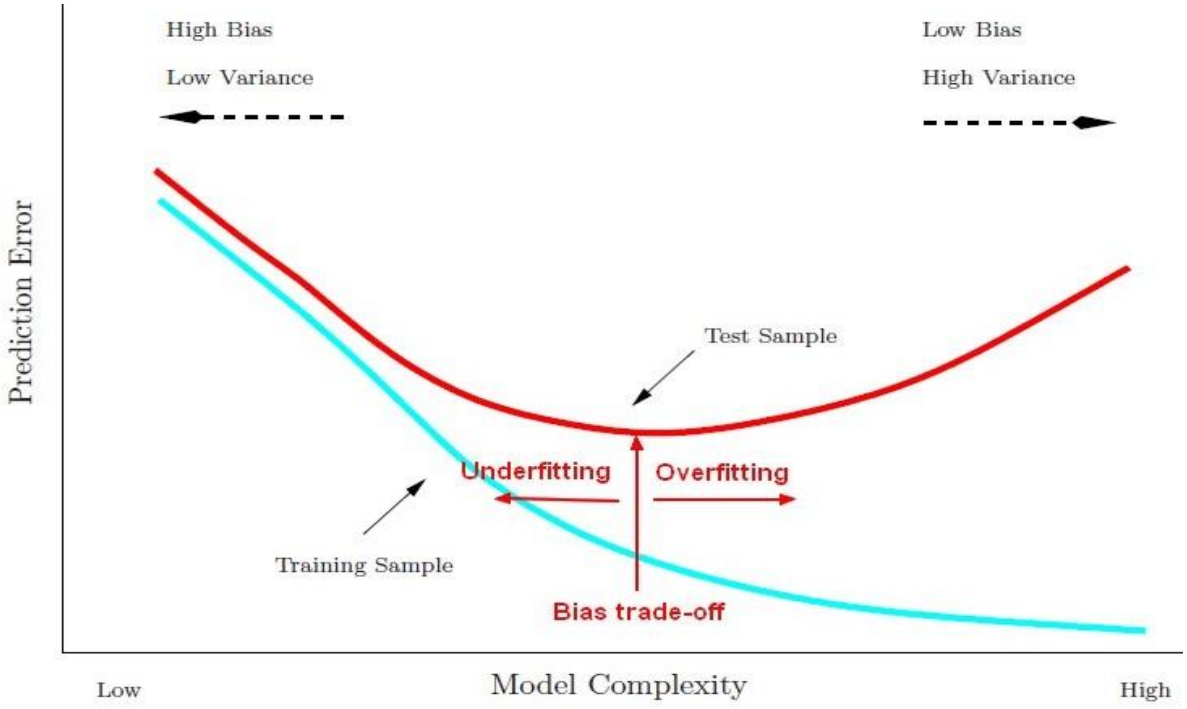
- High Variance => Overfitting
- High Bias => Underfitting

- Bias = Unavoidable Bias + Avoidable Bias

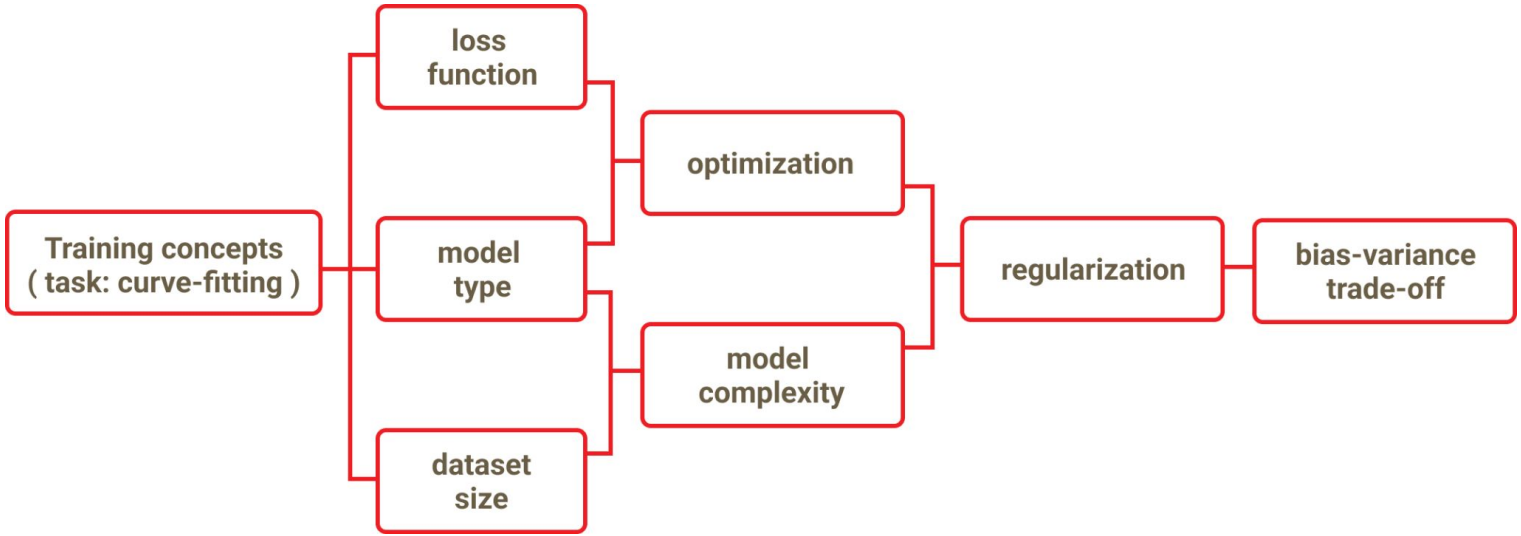


Bias-Variance Trade-Off



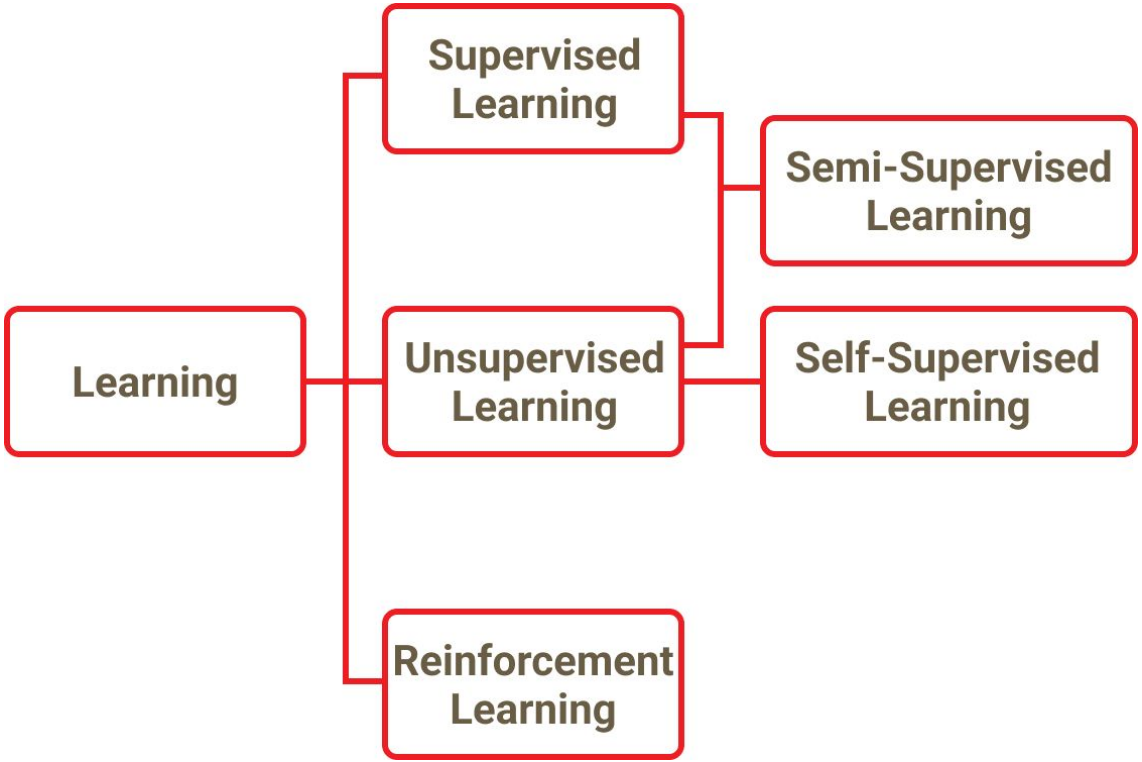


Recap - Training Concepts

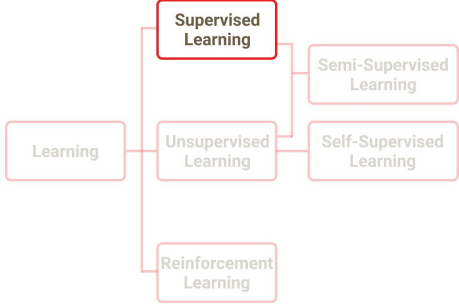


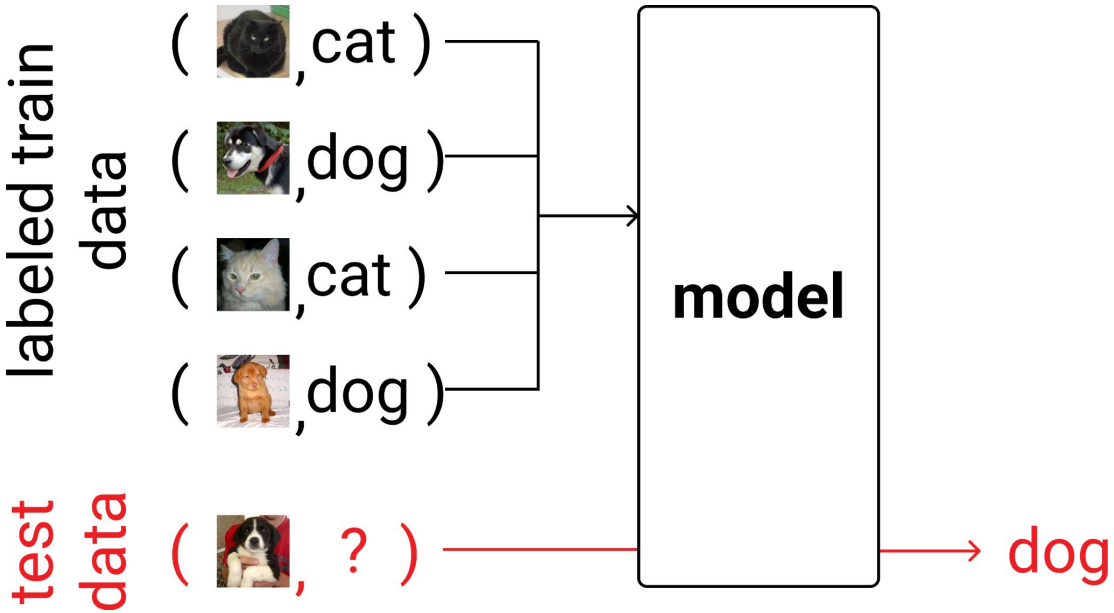
III. Types of Learning

Types of Learning



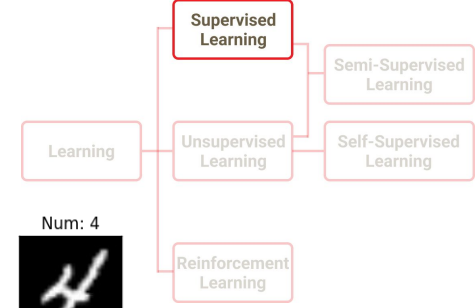
Supervised Learning

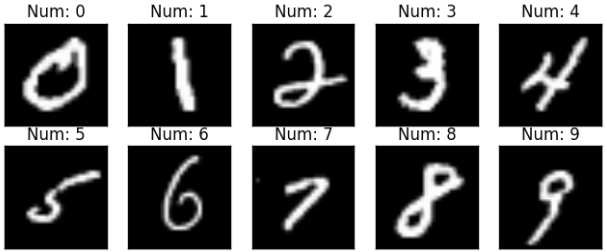


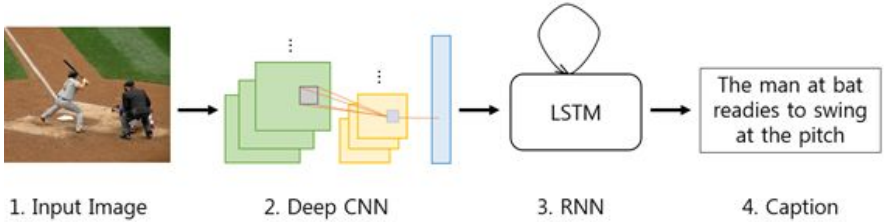


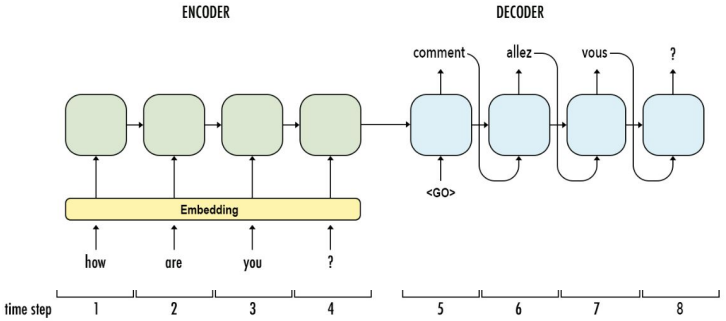
Supervised Learning

- Labeled examples
- Classification or regression tasks

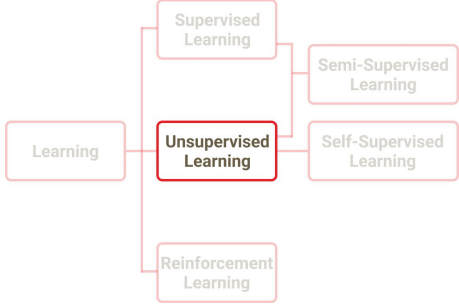








Unsupervised Learning



unlabeled train data

test data ( , ?)

model

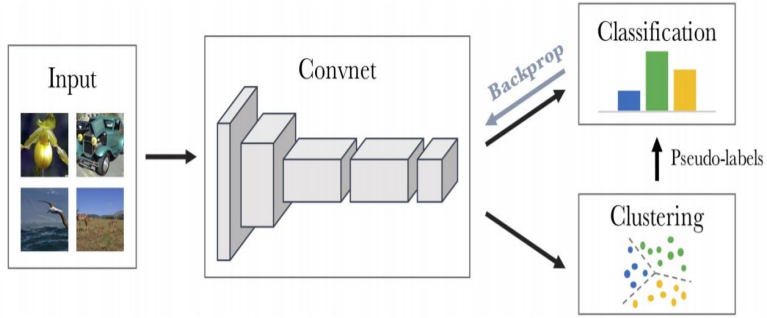
cluster 1

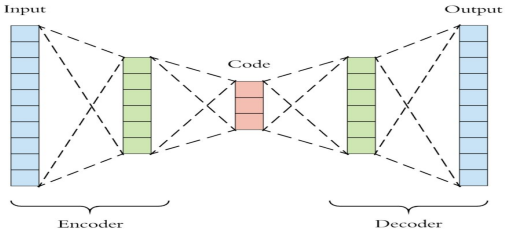
cluster 2

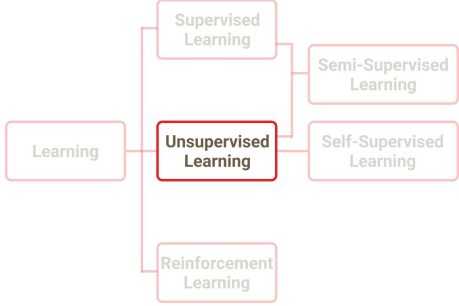
cluster 2

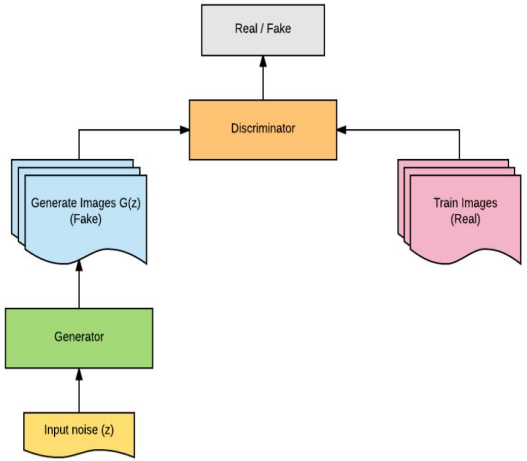
Unsupervised Learning

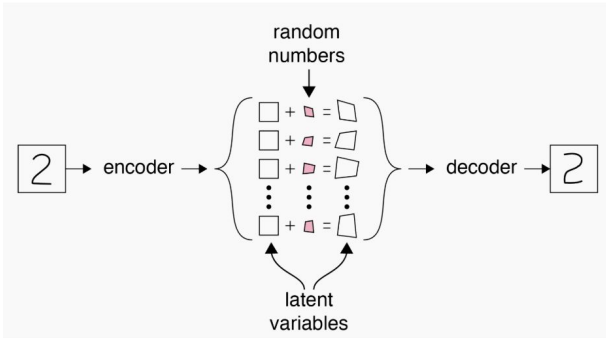
- Unlabeled data
- Unsupervised features learning



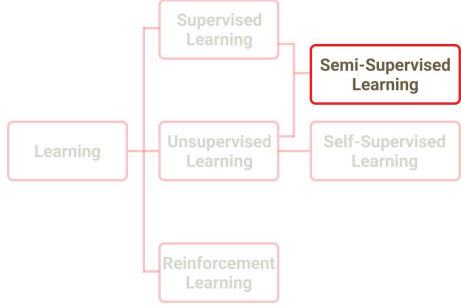


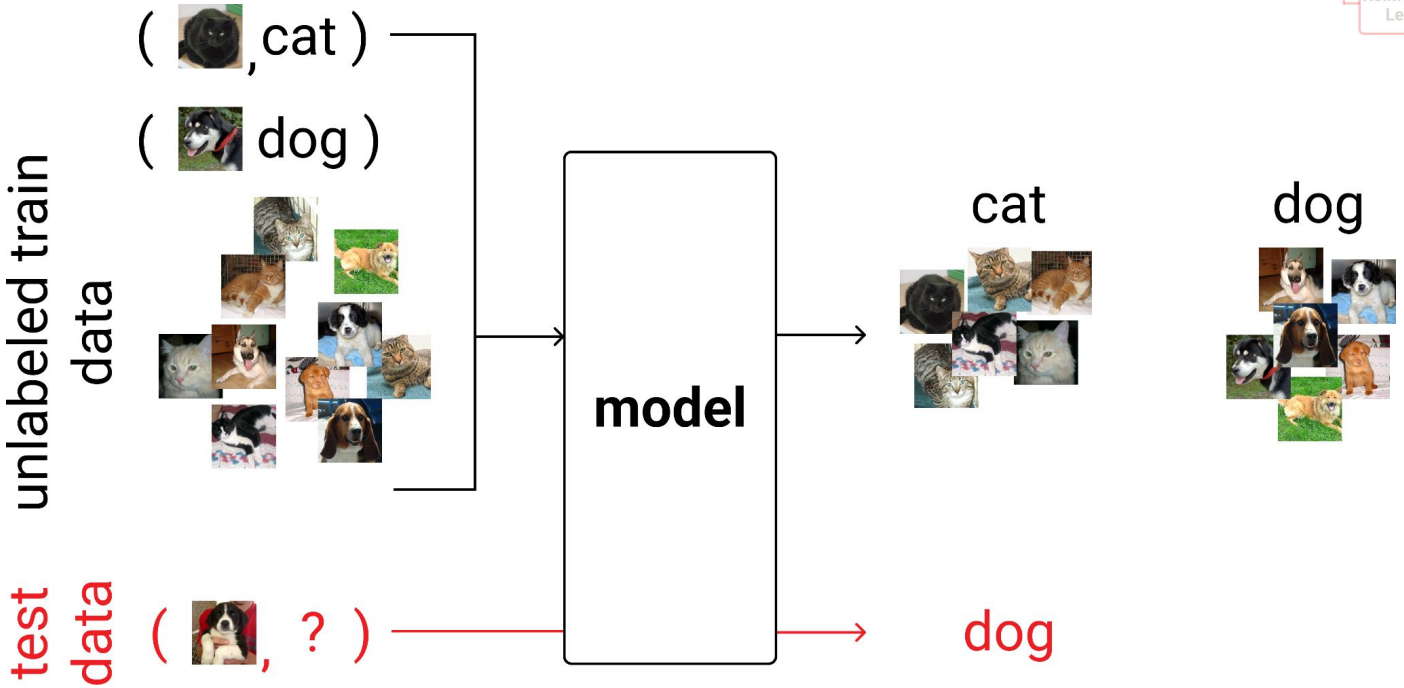




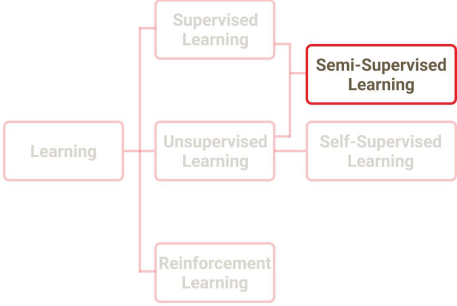


Semi-supervised learning



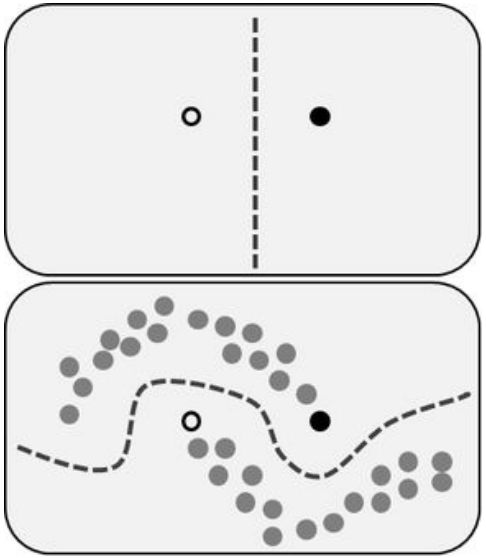


Semi-supervised learning

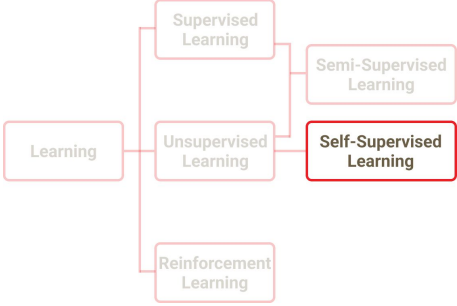


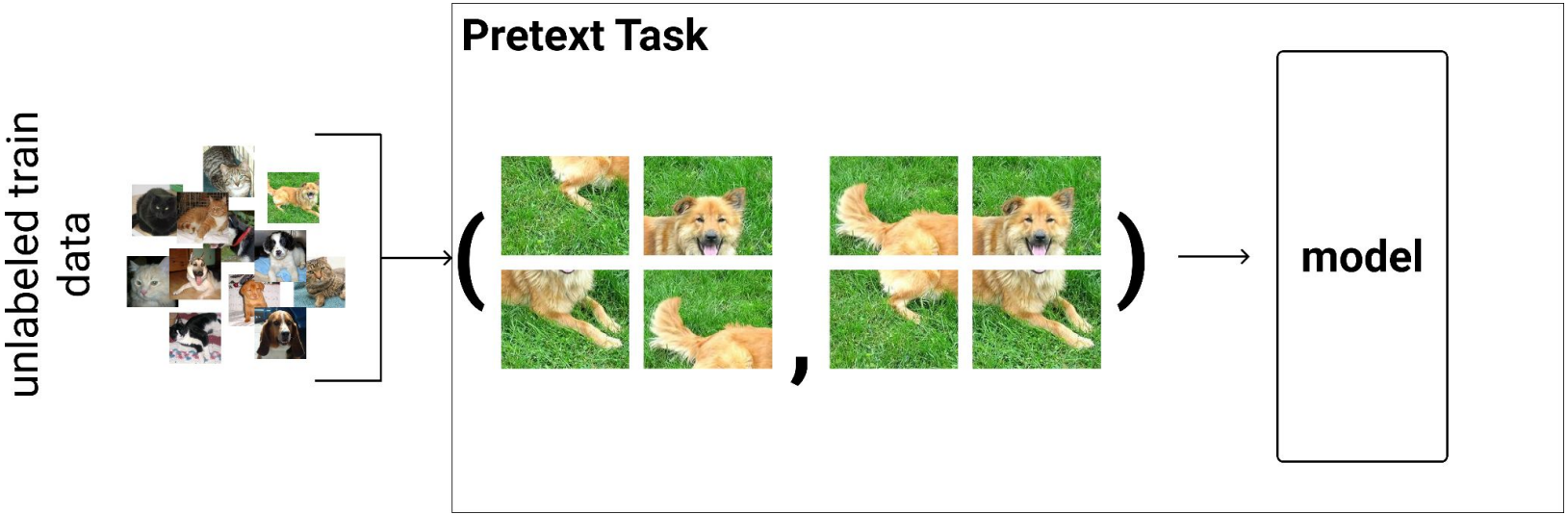
- Small fraction of training examples are labeled

- Tasks:
  - Speech Analysis
  - Webpage Classification
  - Genetic Sequencing

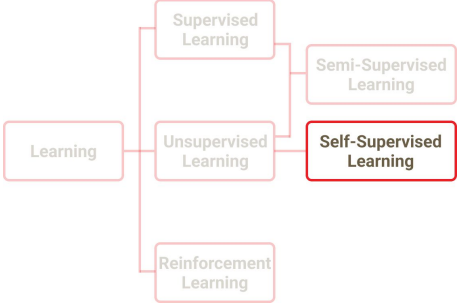


Self-Supervised Learning

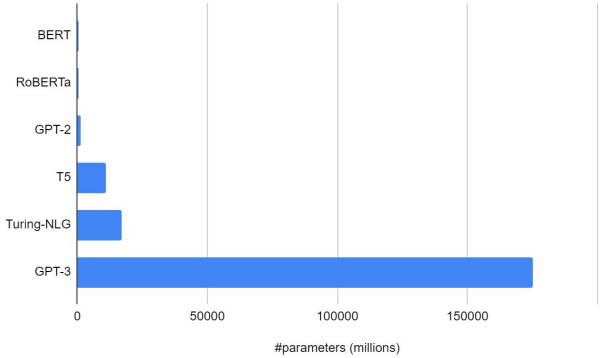




Self-Supervised Learning



- GPT-3
  - self-supervised language model - next word prediction
  - 175 billion parameters
  - trained on 10 000 years of continuous speech (at 100 words per minute)



Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1  Translate English to French:        ← task description
2  cheese =>                           ← prompt
```

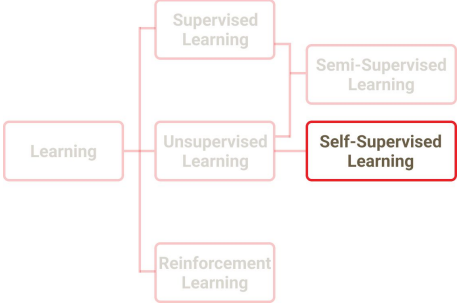
Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← examples
3  peppermint => menthe poivrée
4  plush girafe => girafe peluche
5  cheese =>                           ← prompt
```

Brown et al. Language Models are Few-Shot Learners. NeurIPS 2020, https://openai.com/blog/customized-gpt-3/

Self-Supervised Learning



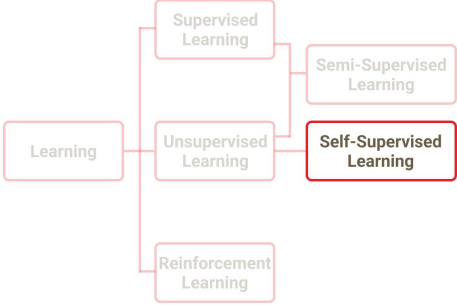
- DALL-E

TEXT PROMPT an armchair in the shape of an avocado. . . .

AI-GENERATED IMAGES

Ramesh et al. Zero-Shot Text-to-Image Generation. ICML 2021, https://openai.com/blog/dall-e/

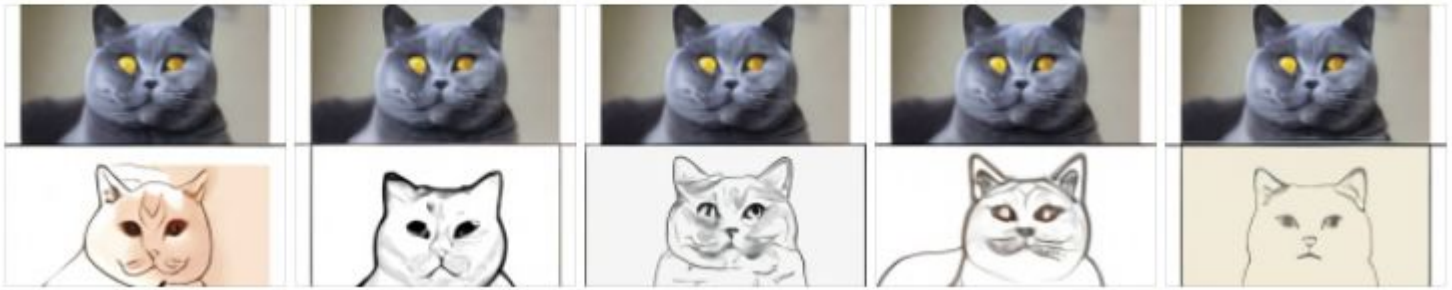
Self-Supervised Learning



- DALL-E

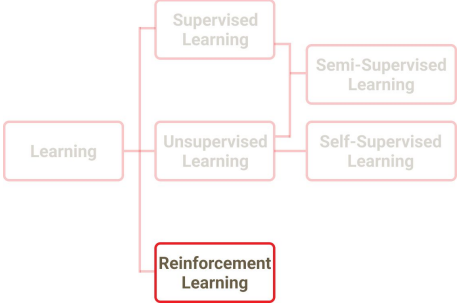
TEXT & IMAGE PROMPT: the exact same cat on the top as a sketch on the bottom

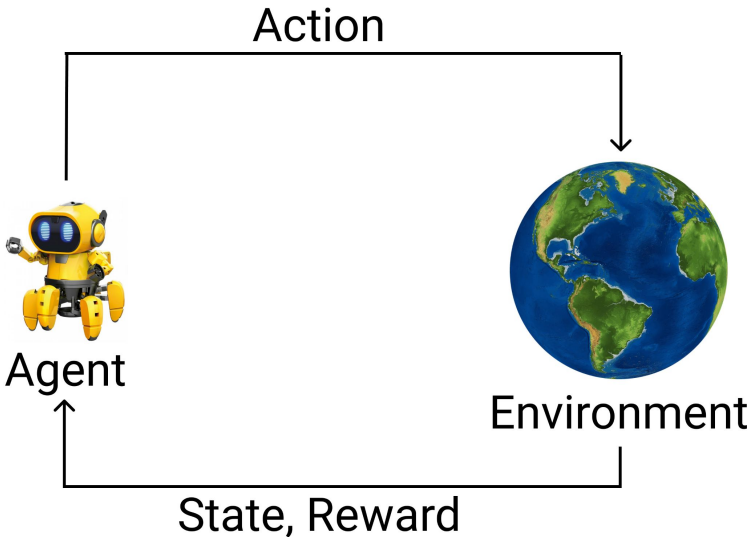
AI-GENERATED IMAGES:



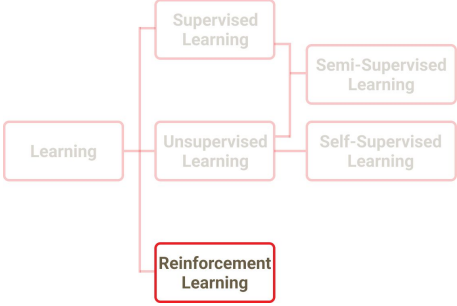
Ramesh et al. Zero-Shot Text-to-Image Generation. ICML 2021, https://openai.com/blog/dall-e/

Reinforcement Learning



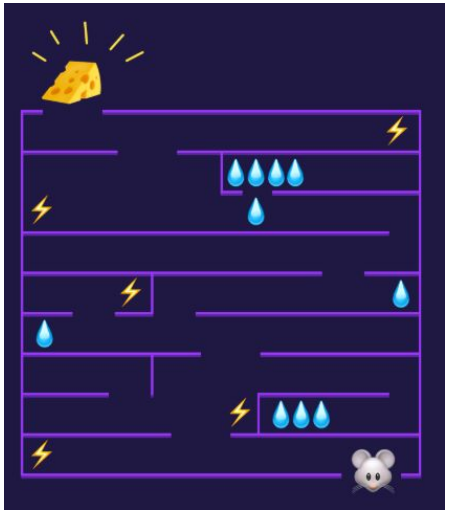


Reinforcement Learning

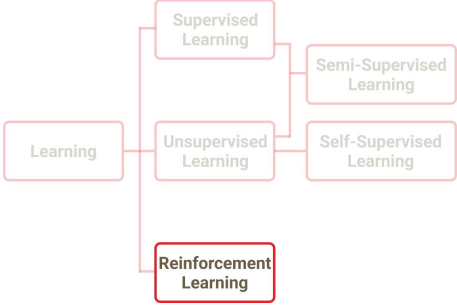


- Optimal behavior for maximal reward

- Applications
  - Robotics
  - Complex games: e.g. AlphaGO
  - Autonomous Driving

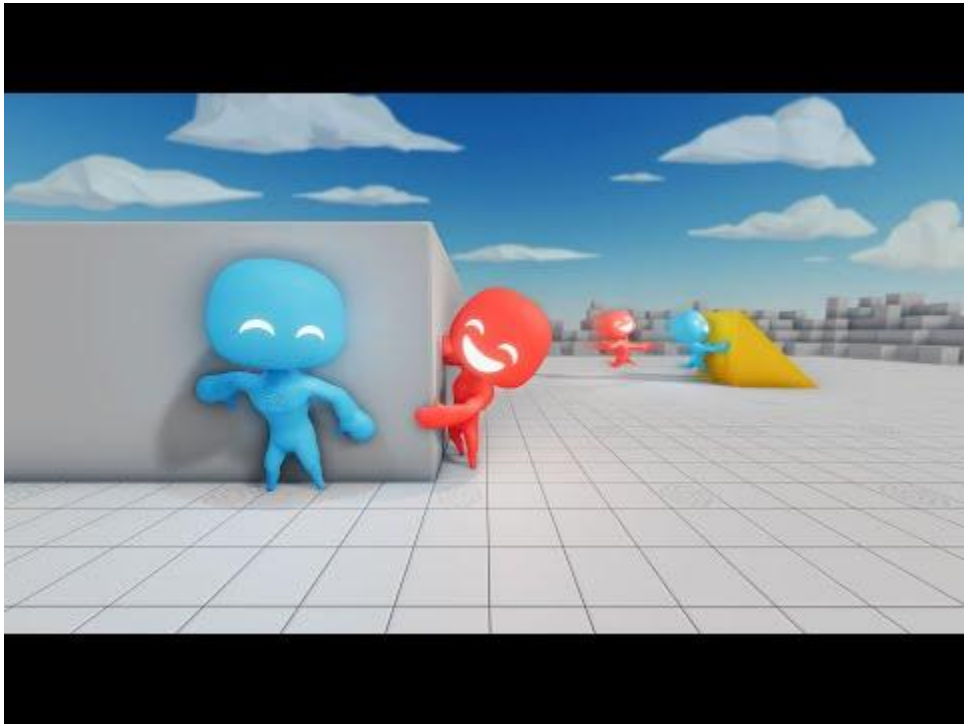


Reinforcement Learning



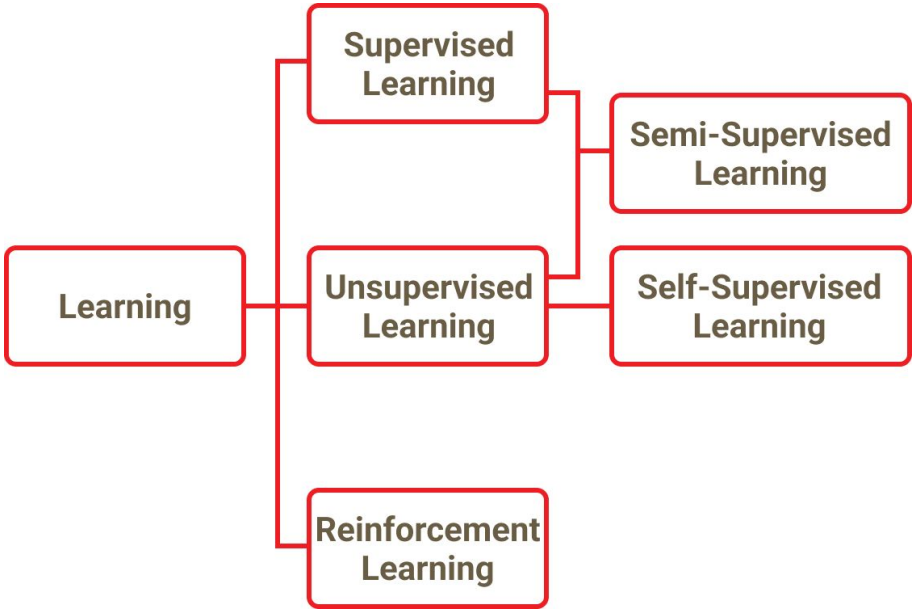
Multi-Agent

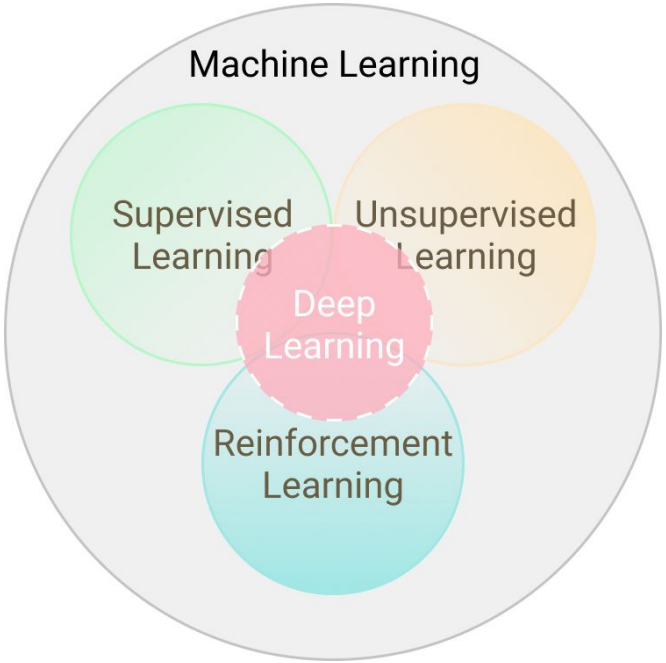
Hide and Seek



Baker et al. Emergent Tool Use From Multi-Agent Autocurricula. ICLR 2020, https://openai.com/blog/emergent-tool-use/

Recap - Types of Learning





Next: Neural Networks

Thank you!

References

- Deep Learning - Ian Goodfellow, Yoshua Bengio, Aaron Courville
  - Chapter 1
- Pattern Recognition and Machine Learning - Christopher Bishop
  - Chapter 1
- CS231n: Convolutional Neural Networks for Visual Recognition
  - Module 1