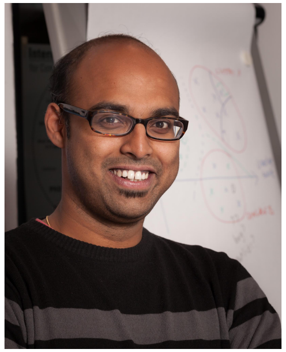
Deep Learning - 2019

Introduction

Prof. Avishek Anand

Who are we ?

- Knowledge-based Systems (KBS)
  - L3S Research Center
- Cutting edge research on
  - Information retrieval
  - Machine learning
  - Data mining
  - Deep learning
- We ....
  - Build systems and do large projects
  - Publish papers and travel the world
  - Are always looking for motivated students


Prof. Avishek Anand

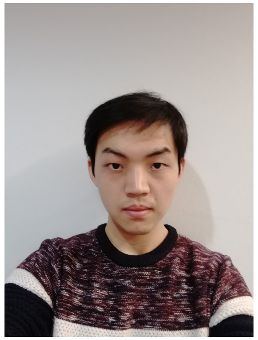

Mateusz Malinowski

DeepMind


Dr. Megha Khosla

Teaching Assistants


Jurek **Leonhardt**


Zijian **Zhang**

Maximillian **Idahl**

What is Deep Learning ?

Artificial Intelligence

Any technique that enables computers to mimic human behavior

Machine Learning

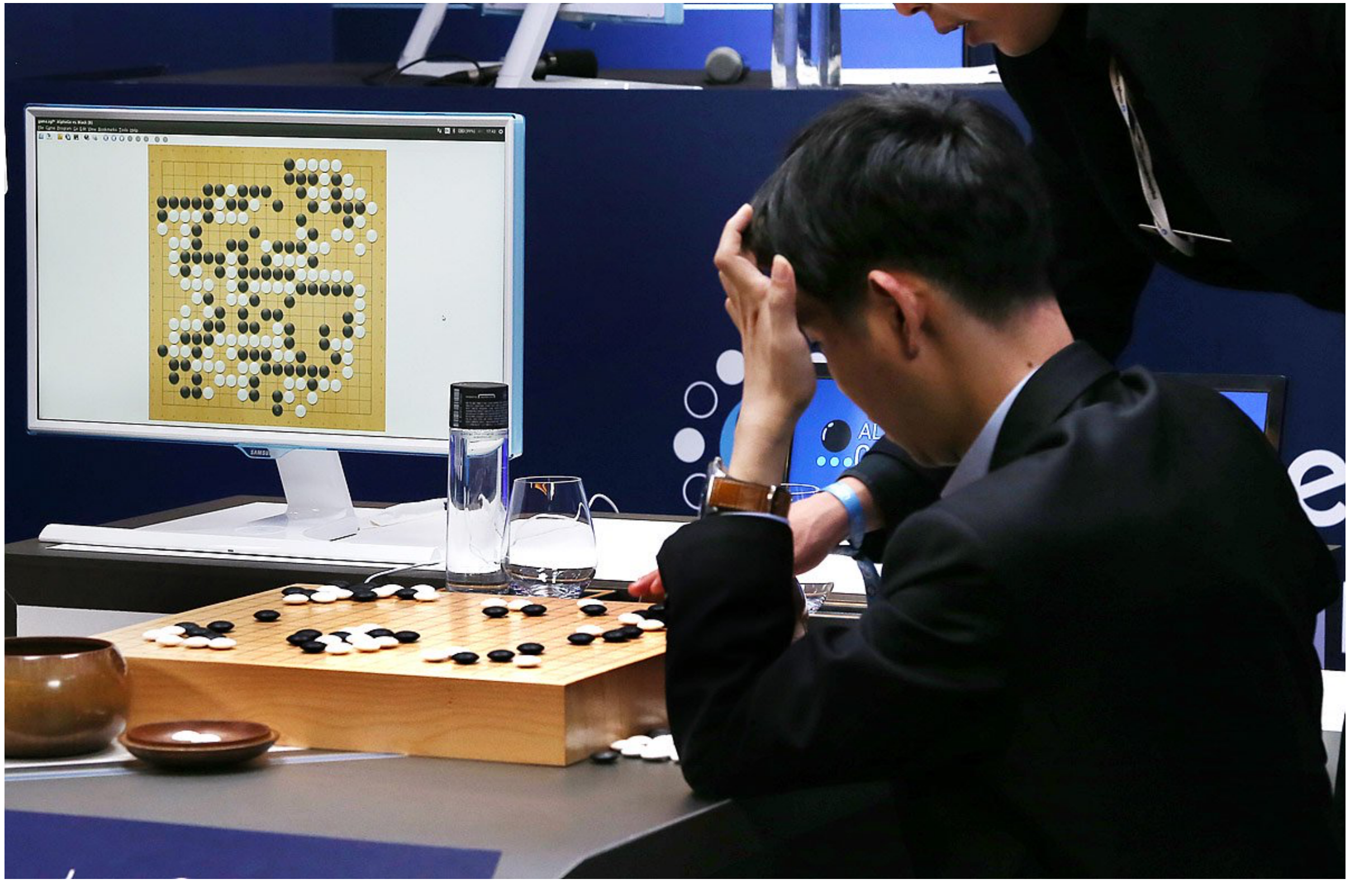
Ability to learn without explicitly being programmed

Deep Learning

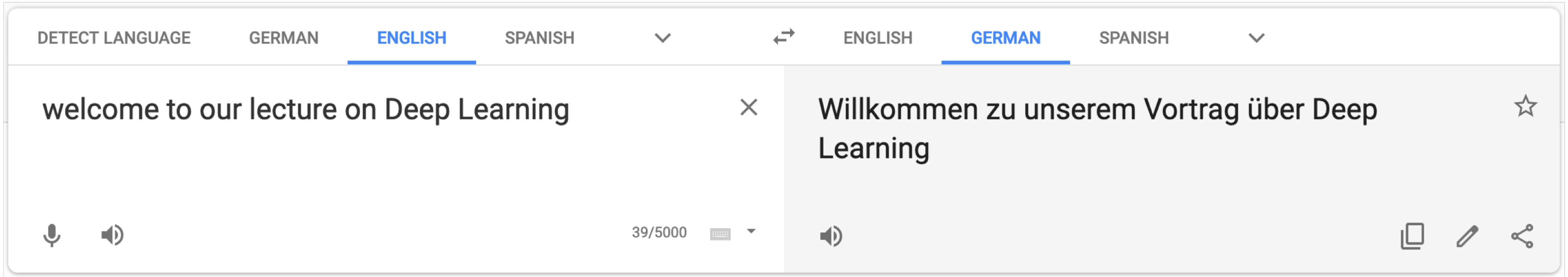
Extract patterns from data using Neural Networks

Achievements of DL

Achievements of DL

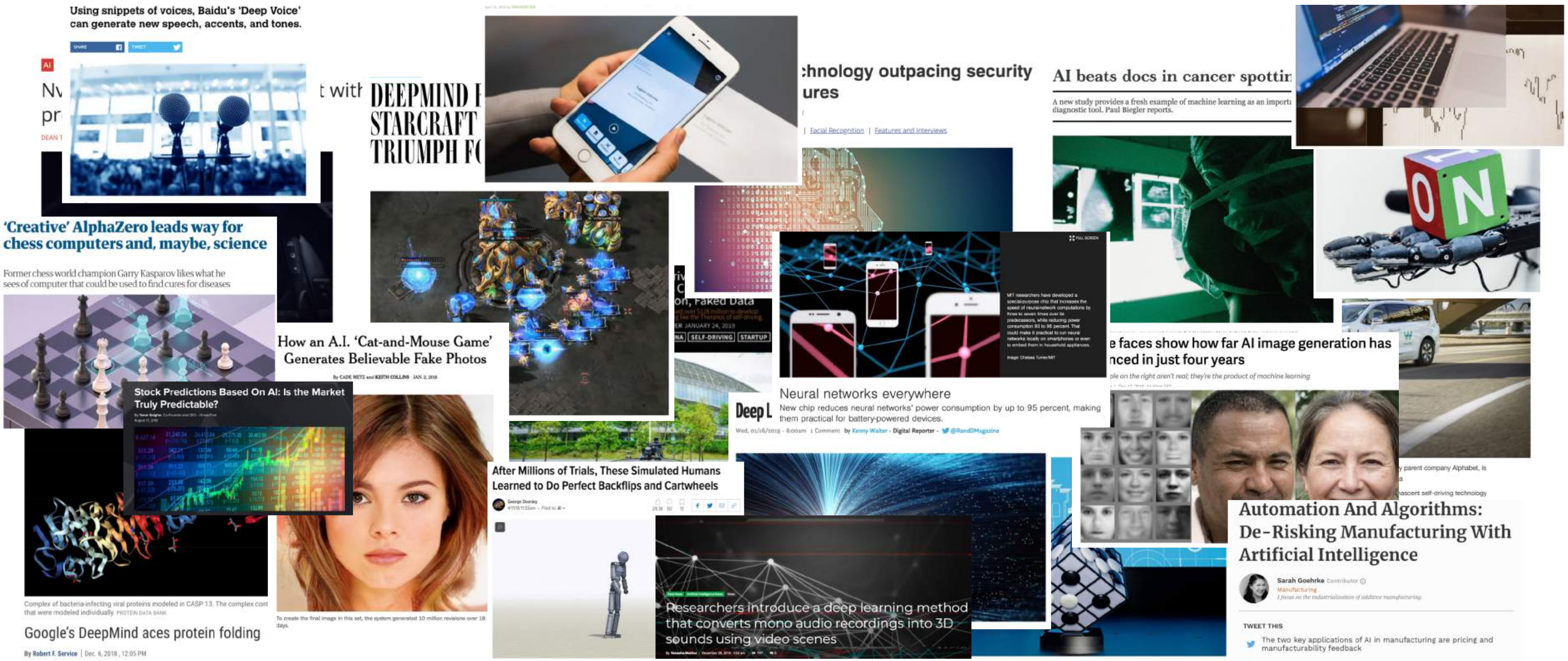


Achievements of DL



Google Translate

The Rise of Deep Learning



Why now ?

Not an old idea, but the time has come now ….

1952 Stochastic Gradient Descent

1958 Perceptron
- Learnable Weights

⋮

1986 Backpropagation
- Multi-Layer Perceptron

1995 Deep Convolutional NN
- Digit Recognition

⋮

1. Big Data

- Larger Datasets
- Easier Collection & Storage

IMAGENET

WIKIPEDIA
The Free Encyclopedia

2. Hardware

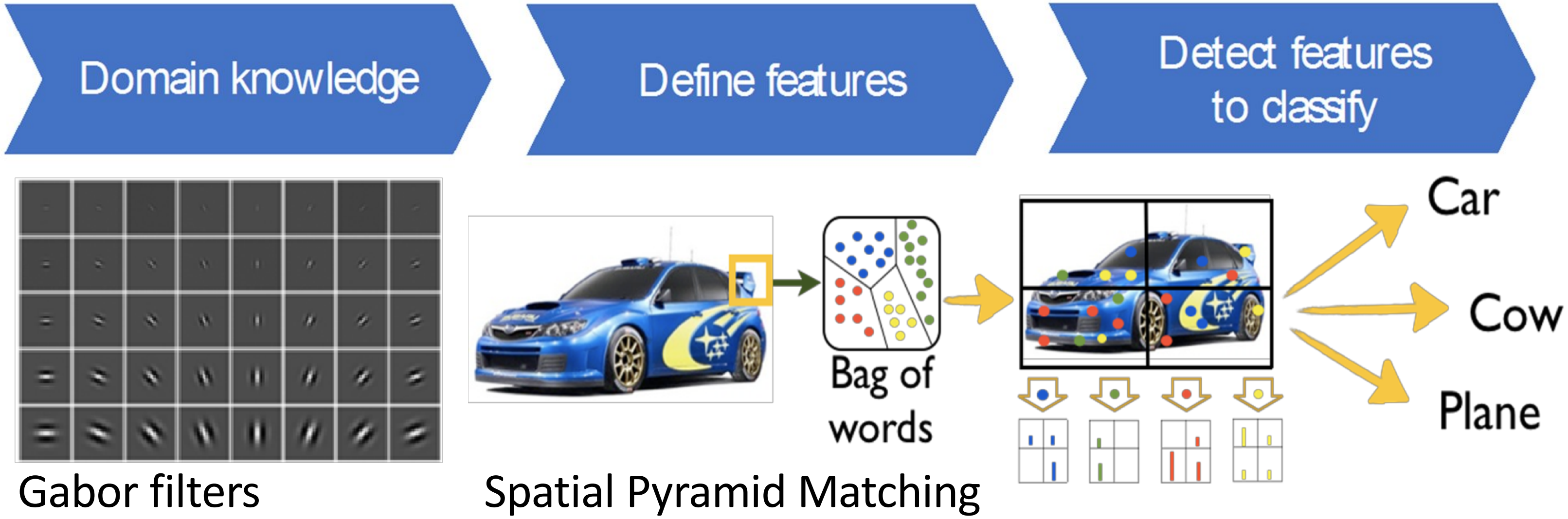
- Graphics Processing Units (GPUs)
- Massively Parallelizable

3. Software

- Improved Techniques
- New Models
- Toolboxes

TensorFlow

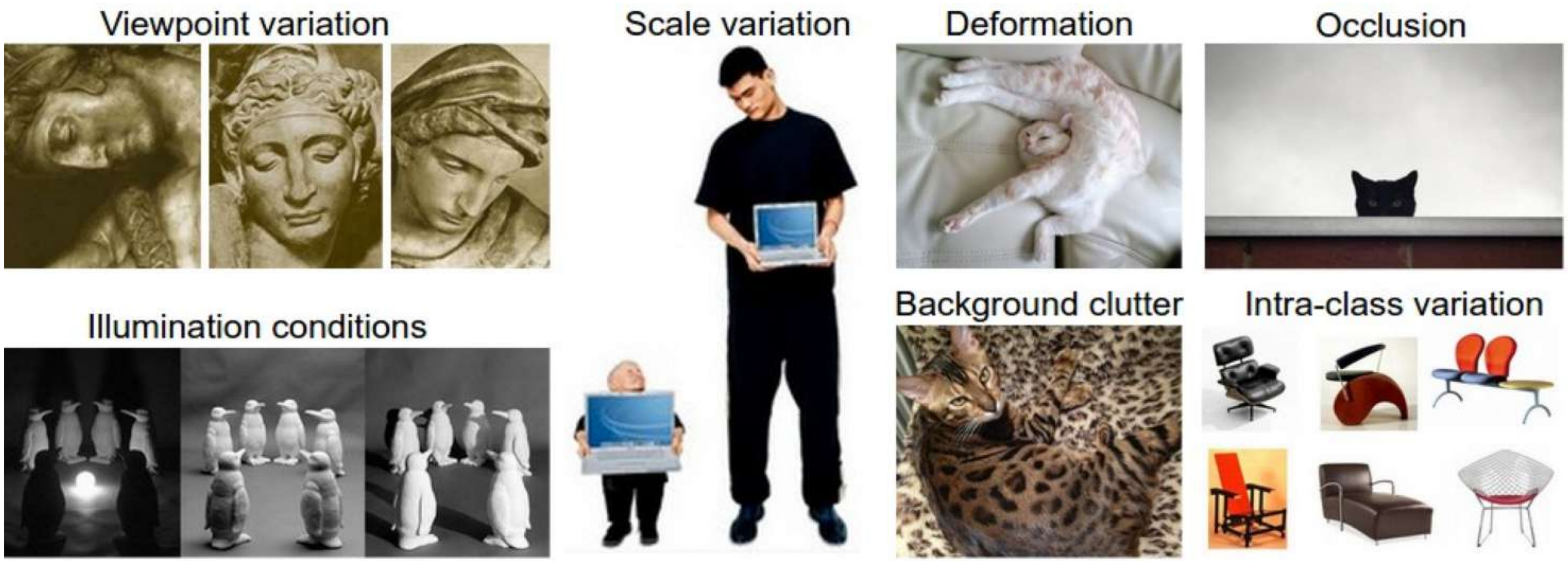
Classical Machine Learning Pipeline



- Input Data → Engineer Features → Build Model
  - Needs domain knowledge
  - Time consuming
  - Good Features make good models, but what are good features?
- What if one could automatically learn features from the input data ?

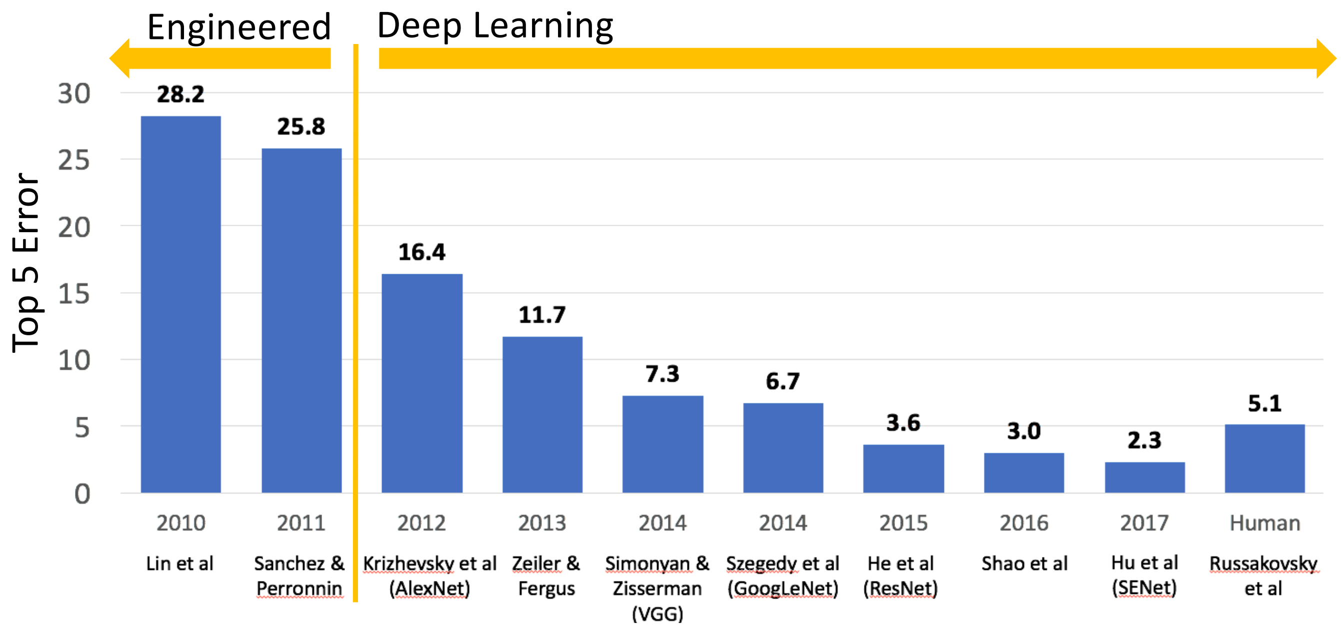
Challenges in designing features

- Our perception works well in different circumstances
  - E.g., we can recognize our friends in the morning or in the afternoon



Deep Learning as Representation Learning

- New pipeline advertised by DL:
  - input data -> learn representation -> Build Models
  - learn representation together with classification **(end-to-end training)**



ImageNet

DL as Representation Learning

- A lot of time spent on creating new architectures
  - Still better performance than engineering features
  - Still less time than engineering features
  - there is still hope that with progress we will need less and less engineering of the architectures, and this will be either automated or we find the great ones

- Great progress on problems with unclear engineered features
  - Multimodal representation -- how to model language and visual signals at once ?

Course Logistics

Email for Assignment Submissions: pir-assignments@l3s.de
Email for Lecture related queries : leonhardt@l3s.de, zzhang@l3s.de
Use [DL2019] in the subject line

Course Logistics

- Lecture Slides and Assignments will be available in Stud.IP a day before the lecture

- Mini-Projects (25% of the total credit)
  - Work in groups of max 3
  - Discuss progress with the tutors in the exercise sessions
  - Submit a writeup (Sample latex code will be provided)
  - Presentations in the last lecture + last tutorial

- Final written exam (75% of the total credit)
  - August 8
  - Topics released in May 7 :
    - DL for Language and Retrieval (Question Answering)
    - DL for Graphs (Social networks, knowledge graphs, Biological Networks)
    - Interpretability of DL approaches

Tutorials and Exercises

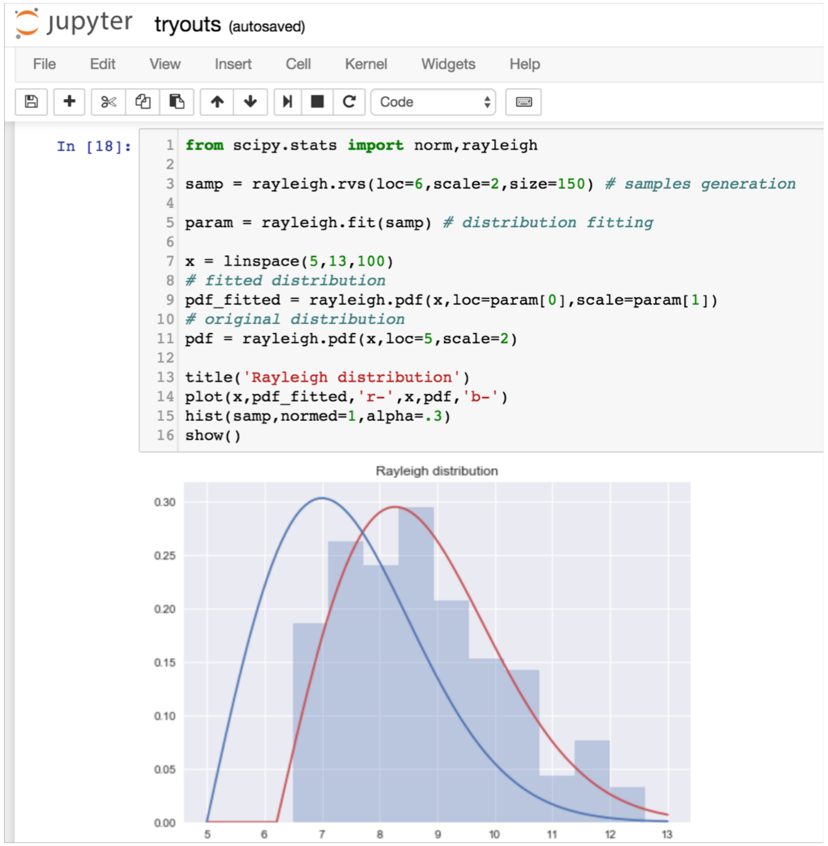
- Tutorials and Recitations
  - 2 Recitations (In the exercise session)
    - Machine learning pipeline
    - Introduction to Tensor flow
  - 10 tutorials
- Exercises accompanied with each lecture to be discussed in 2 weeks from the release date
  - Tutorial Assignments to be turned in by 9 am Tuesday (In python notebooks)

- Composition of Exercises
  - Approx. 60% exercise questions based on formal matter
  - Approx. 40% exercise questions based on programming assignments

Python Notebooks

- Most/All lectures will be accompanied with python notebooks
  - Helps you see directly how the concepts translate into tangible value
  - Allows you play around for better understanding
  - Provides stub code for easier entry to code

- Programming Language
  - Python (v 3.0) – fast, easy, small code footprint, large online community
  - Tensorflow
  - Google/Stack Overflow is your friend but drive with caution



Bonus Structure

- Bonus to improve the final grade

- Students who submit correct solutions to the exercise are chosen by the tutors
  - Chosen students present their solution in the tutorials

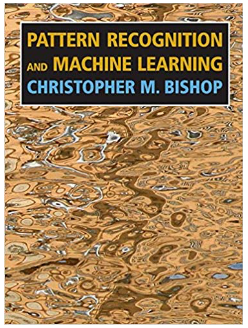
- Each successful presentation = 1 badge
- 3 badges = 1 bonus point = 0.3 grade improvement on the final grade
  - We can decide to bump it up 1 badge = 0.5 improvement depending on the response

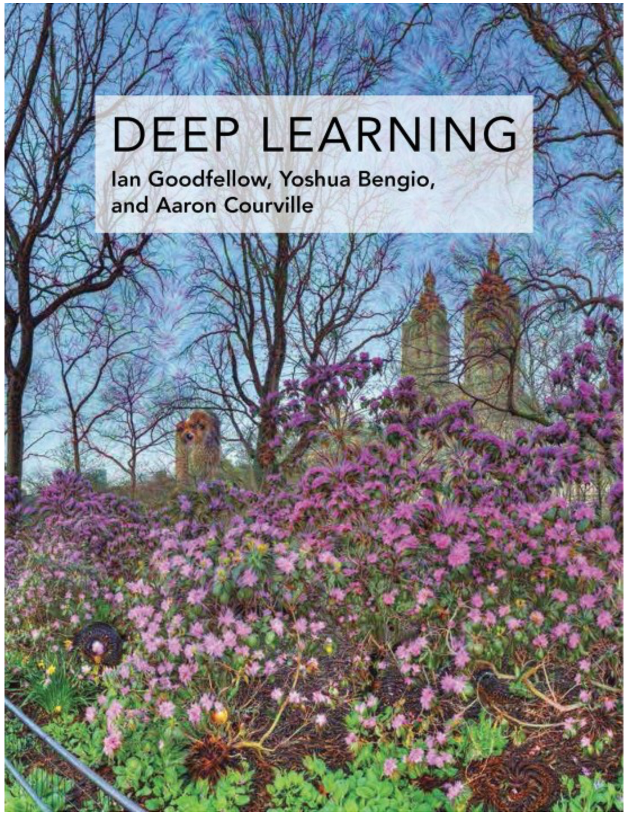
0.3333... Grade improvement

- Bonus in the Mini project (After reaching 100%)
  - Great job = 1 bonus point
  - Blow my mind = 2 bonus points (publication quality)

Literature and code

- Books to be followed
  - Deep Learning (Goodfellow et al.)
  - Pattern Recognition and machine learning





- References to papers to be included with the slides

- Programming language: **Python + Tensorflow**
- Google Colab : https://colab.research.google.com/

Pre-requisites

- Linear Algebra
- Calculus (Practice your derivatives, needed in exercises)
- Machine Learning basics

- Ability to work hard
- Interest in setting up experiments
- Being self critical

Goals of the lecture

- Get a solid understanding of the nuts and bolts of Supervised Neural Networks
  - Conv Nets, RNNs, GANs, Attention Mechanism
  - How they are applied to specific tasks

- A general understanding of optimization strategies to guide training Deep Architectures
  - The ability to design from scratch, and train novel deep architectures
  - Pick up the basics of a general purpose Neural Networks toolbox

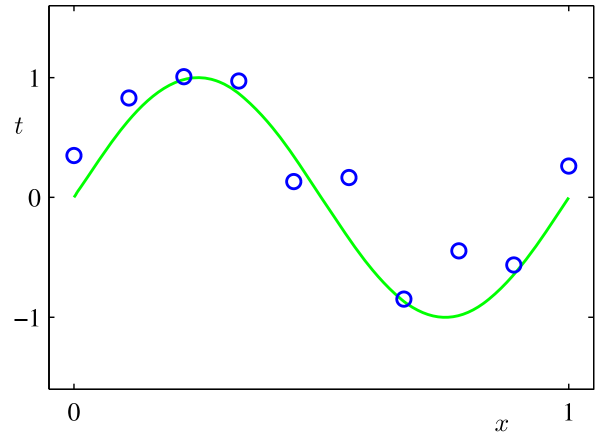
- Ability to translate an architecture to code and into a fully functioning solution

Is Machine Learning curve fitting ?

Simple Regression

- Assume that data is generated from a known function + random noise

$$sin(2\pi x) + \epsilon$$



- You are only given some sample of observations (training data)
  - A finite sample of input and targets

$$\mathbf{x} \equiv (x_1, \ldots, x_N)^{\mathrm{T}}$$

$$\mathbf{t} \equiv (t_1, \ldots, t_N)^{\mathrm{T}}$$

- Goal: to model the underlying distribution that generates data sand to make predictions

- First thought: Lets try curve fitting

Polynomial curve fitting

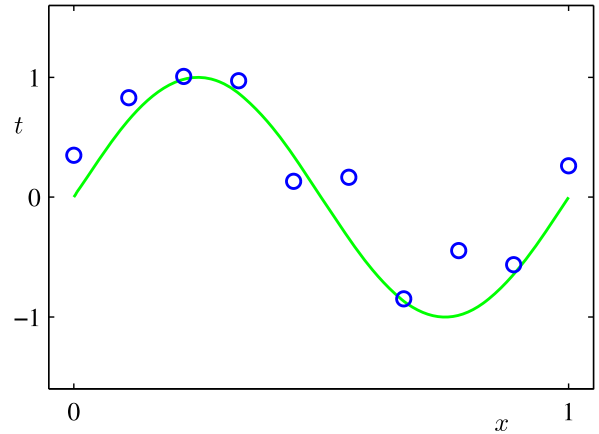
Co-efficients , weights, parameters

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

- Given a polynomial of degree M, goal is to find w that fits my data
- The best w is found with respect to some measure of goodness
  - When is one set of params better than another ?

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

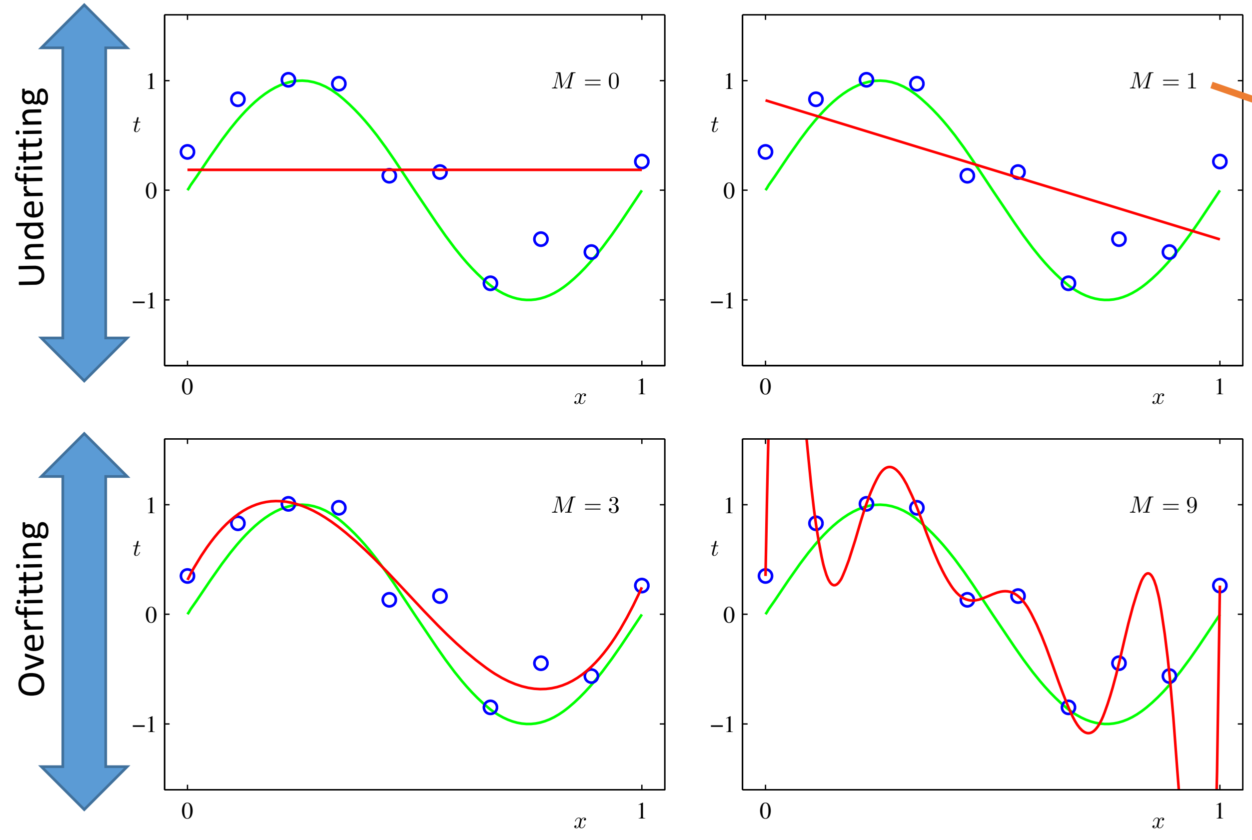
Loss function



- The best set of params are the ones with lowest loss; formally set-up as an optimization problem

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

Which degree should you use ?

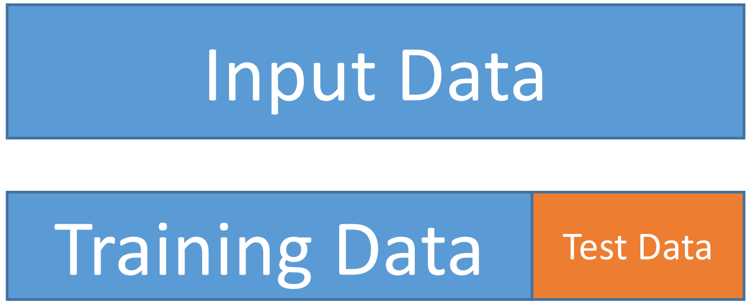


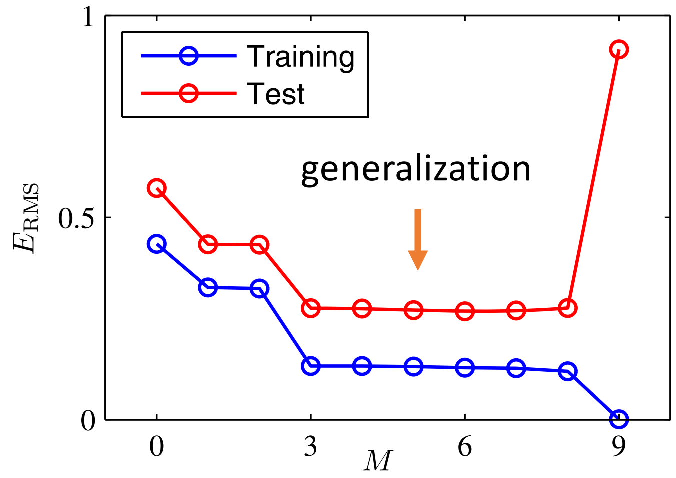
Hyperparameter = degree of the polynomial

Training, Validate and Test

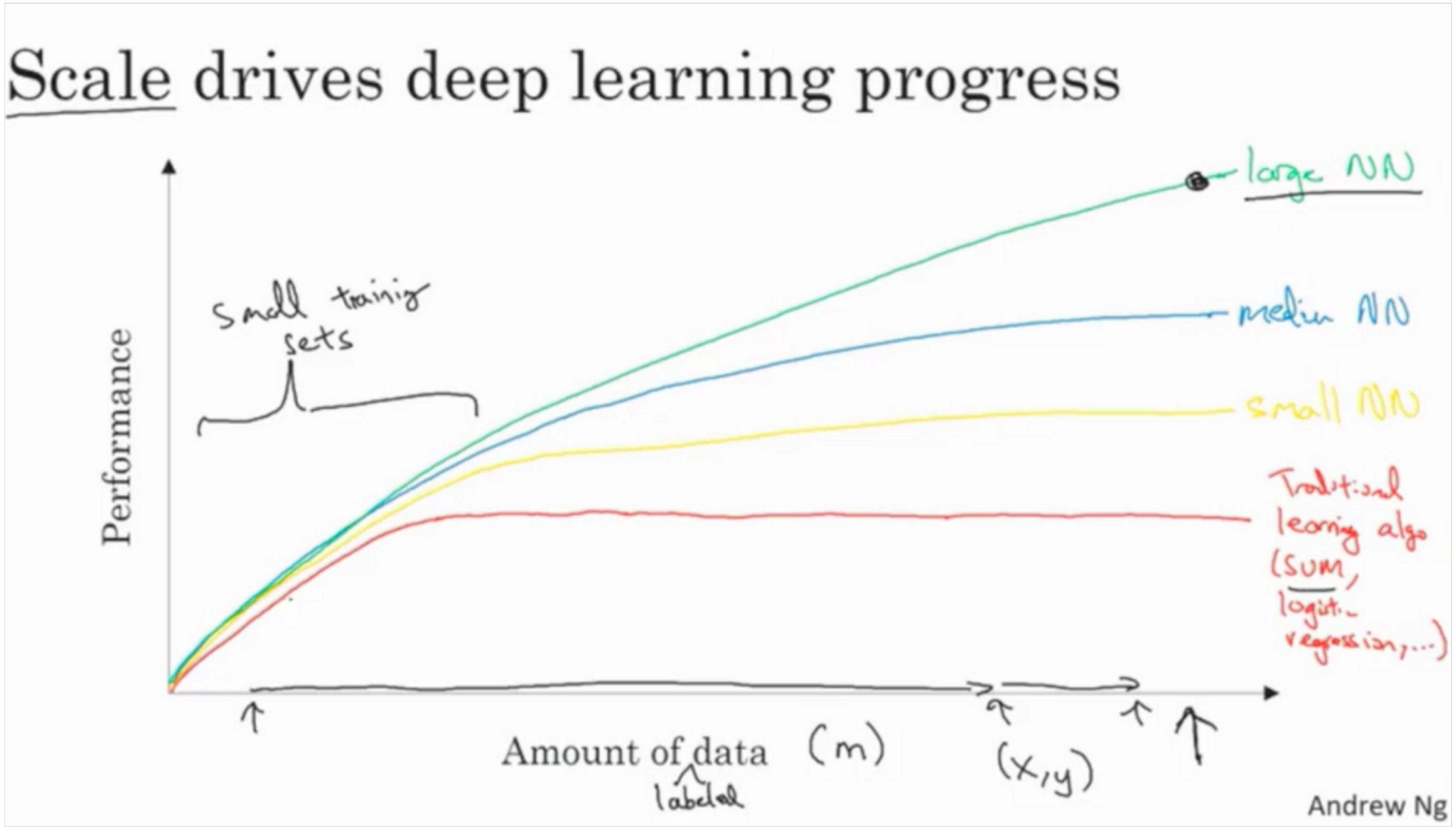
- Generalization: How well does your model perform on unseen data ?
- Set aside a test set to check for generalization
- In principle, we use a validation set/dev set for model selection

Generalization is the key to ML



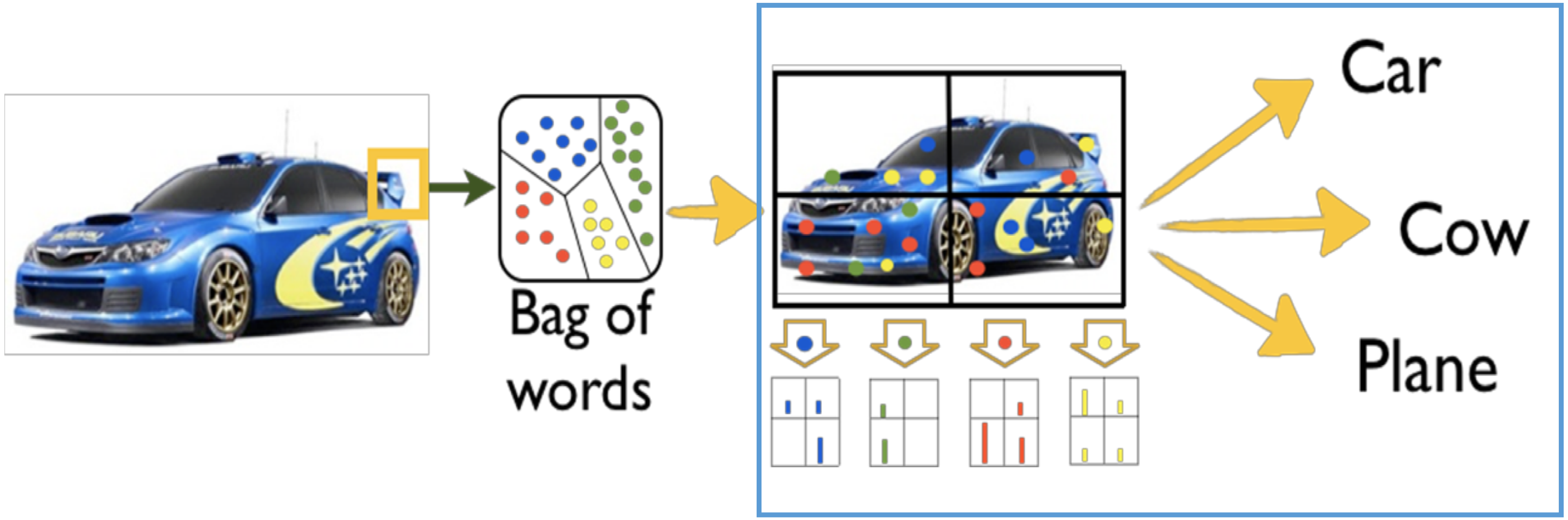


Why Deep Learning ? (revisited)



Basic Notions and Intuitions

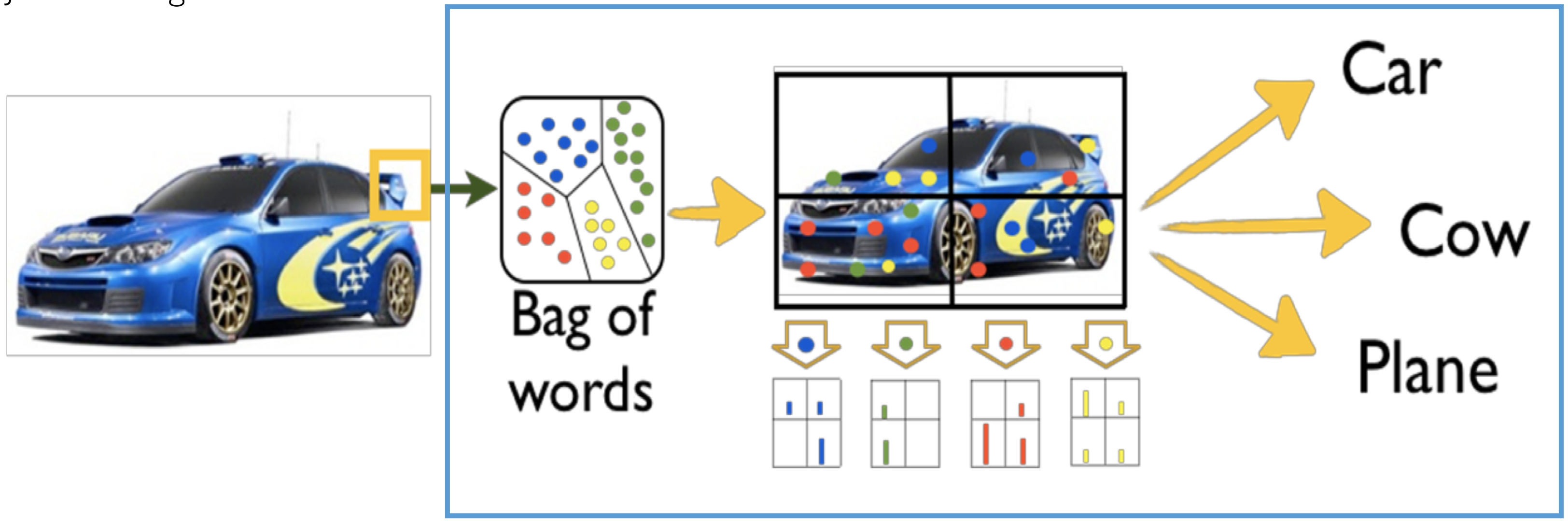
- Joint training
  - Training at least two modules together
  - Pre-DL era: hand-engineer all the modules, train only the last one (e.g., classifier)



Train together classifier and 'high-level' features

Basic Notions and Intuitions

- Joint training
  - Training at least two modules together
  - Pre-DL era: hand-engineer all the modules, train only the last one (e.g., classifier)
- End-to-end training (e2e training)
  - Join training of all the modules



Train together classifier and 'high-level' and 'low-level' features

Basic Notions and Intuitions

- Joint training
  - Training at least two modules together
  - Pre-DL era: hand-engineer all the modules, train only the last one (e.g., classifier)
- End-to-end training (e2e training)
  - Join training of all the modules
- Back-propagation (Backprop)
  - Technique for training Neural Networks
    - Compositions of functions
  - Technically it computes a Jacobian matrix by using a chain rule
  - But it does it efficiently

$$\frac{\partial}{w} f(g(w)) = \frac{\partial}{g(w)} f(g(w)) \cdot \frac{\partial}{w} g(w)$$

Basic Notions and Intuitions

- Joint training
  - Training at least two modules together
  - Pre-DL era: hand-engineer all the modules, train only the last one (e.g., classifier)
- End-to-end training (e2e training)
  - Join training of all the modules
- Back-propagation (Backprop)
  - Technique for training Neural Networks (compositions of functions)
  - Technically it computes a Jacobian matrix by using a chain rule
  - But it does it efficiently
- Optimization
  - We often try to find the best parameters that 'explains data' under the current model (e.g., neural network)
  - We did it in the curve-fitting example
  - There are, however, other alternatives like sampling

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

Thank You (TA's Needed)

Email for Assignment Submissions: pir-assignments@l3s.de
Email for Lecture related queries : leonhardt@l3s.de, zzhang@l3s.de
Use [DL2019] in the subject line