Advanced Machine Learning

Course plan:

1. Machine Learning fundamentals
2. Classification problem revisited: decision trees
3. Neural network architectures (part one: deep NN & CNN)
4. Basics of signal and image processing
5. Application of NN for image processing: (classification, segmentation, frameworks)
6. Neural network architectures (part two, advanced ideas: RNNs, attention, autoencoders, GANs)
7. DETR, weakly supervised learning, graph NNs

machine learning fundamentals

Materials for this section:

- I. Goodfellow , Y. Bengio, and A. Courville, *Deep Learning, Chapter 5: Machine Learning Basics*, https://www.deeplearningbook.org
- Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, Nature **521,** 436 (2015).
- https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf
- https://tomaszgolan.github.io/introduction_to_machine_learning

What is machine learning?

Family of computer methods that improve automatically through experience
- without being *explicitly* programmed to

<u>What kind of problems we would like (to be able) to solve?</u>
- the early days of AI: problems that are intellectually difficult for humans but straightforward for computers
- nowadays, the *true* challenge to AI are tasks easy for people to perform but **hard to be described formally**
  - like recognizing spoken words or faces on images

The machine learning (ML) solution to these problems:
- allow computers to learn from experience and understand the world in terms of a hierarchy of concepts

this approach avoids the need for human to specify all the knowledge that the computer needs.

The hierarchy of concepts enables the computer to learn complicated concepts by building them out of simpler ones
- representations of data with multiple (**deep**) levels or *layers* of abstraction

What is machine learning?

If we draw a graph showing how these concepts are built on top of each other, the graph is *deep*, with many layers. For this reason, we call this approach to AI **deep learning**

Here each layer is automatically trained to learn and detect different features on the input image

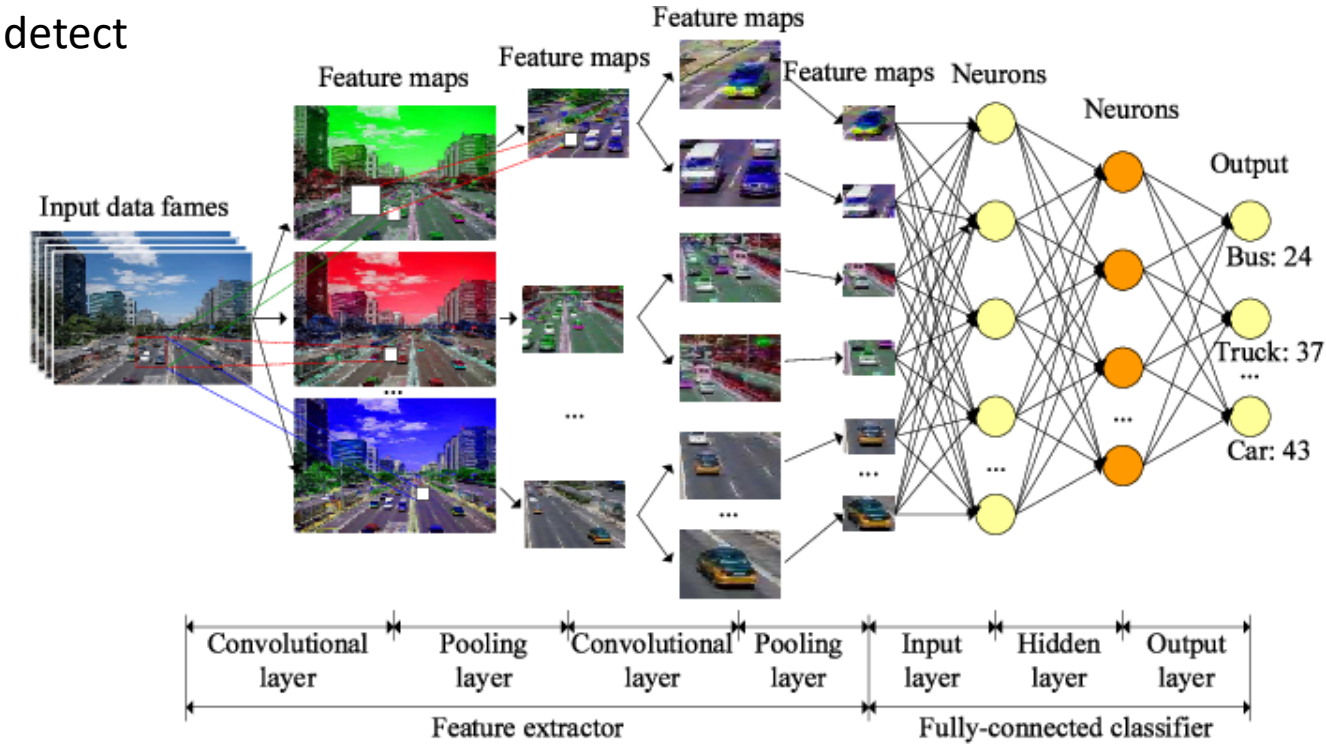


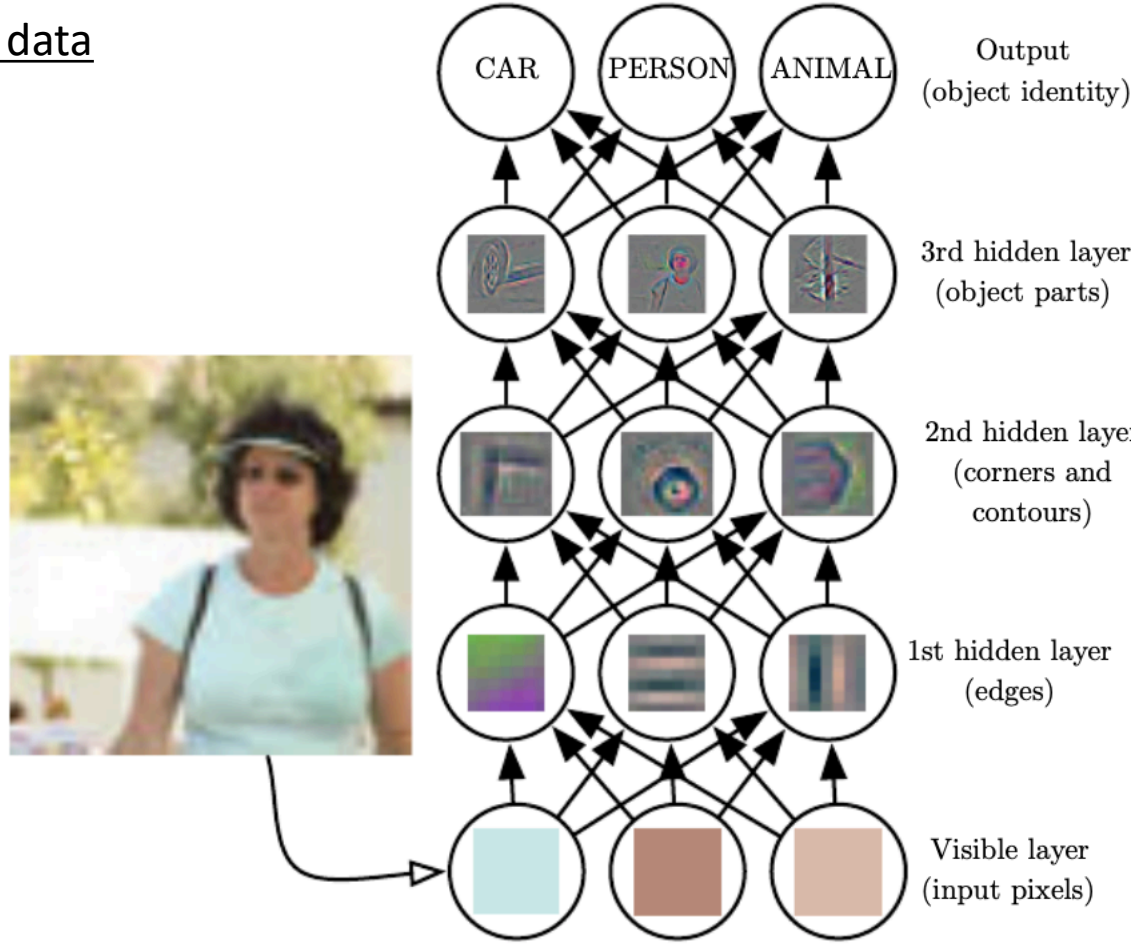
Fig. 3. Example of the CNN model for vehicle classification.

doi:10.1109/TII.2019.2909473

What is machine learning?

How to efficiently solve these problems – *automatically* learn from data

A machine learning algorithm is an algorithm that is able to efficiently learn from data



To sum up:

- deep learning solves problems by introducing representations that are expressed in terms of other, simpler representations
- deep learning enables the computer to automatically learn complex concepts out of simpler ones

https://www.deeplearningbook.org/contents/intro.html

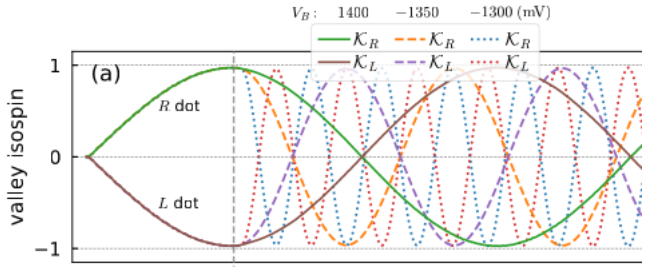
What is machine learning?

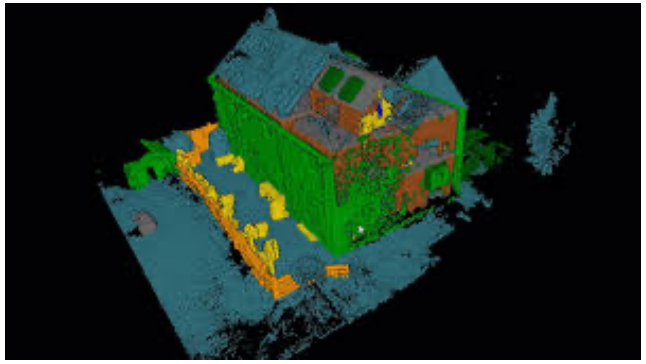
Typical ML tasks:

- classification $f : \mathbb{R}^n \to \{1, \ldots, k\}$.
- regression $f : \mathbb{R}^n \to \mathbb{R}$
- transcription (OCR/ASR), translation
- extract word and whole sentences from written or spoken data
- anomality detection
- detect events or objects that are unusual in a given context
- synthesis and sampling
- generate new examples that are similar to those in the training data
- denoising
- predict clean sample x from the corrupted one $\tilde{x}$

Example data dimensionality:

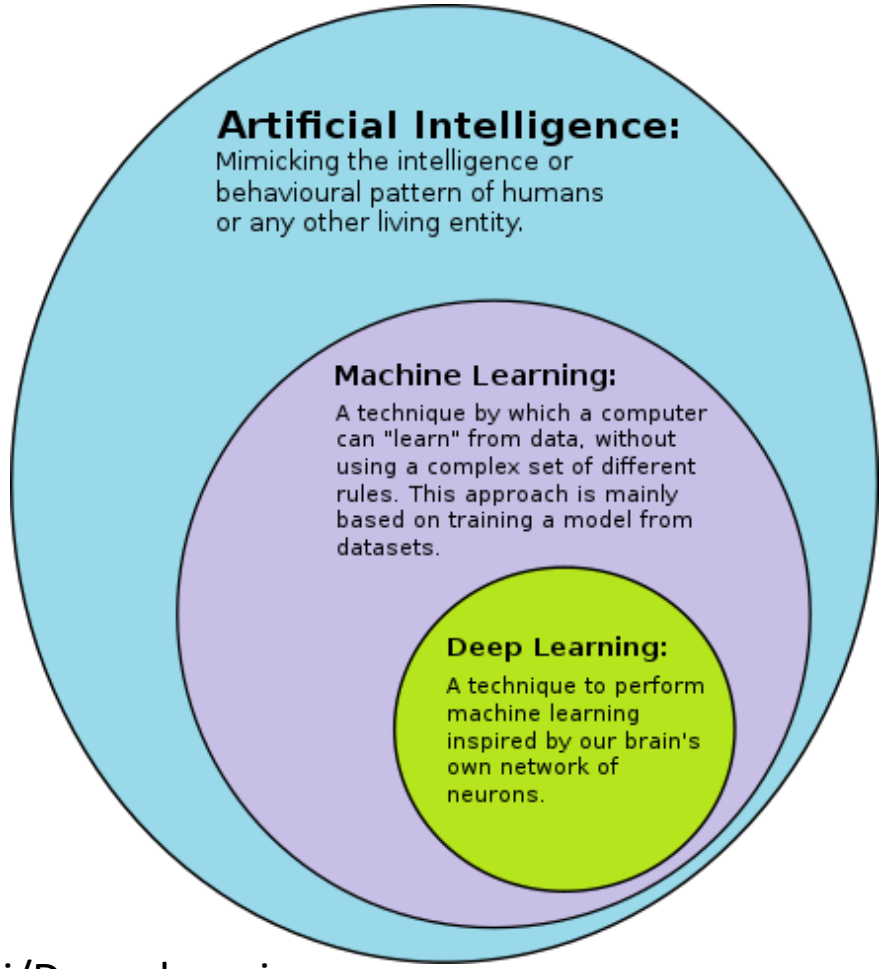
- 1d signal
- 2d image
- 3d point cloud

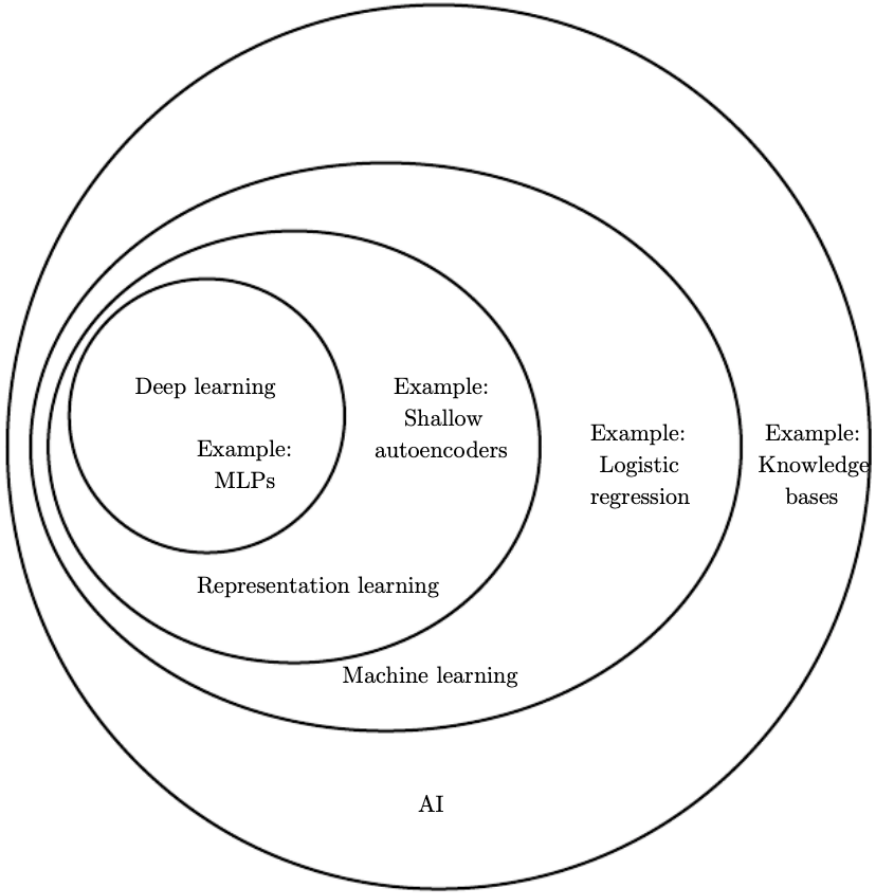




What is machine learning?

To summarize AI technologies

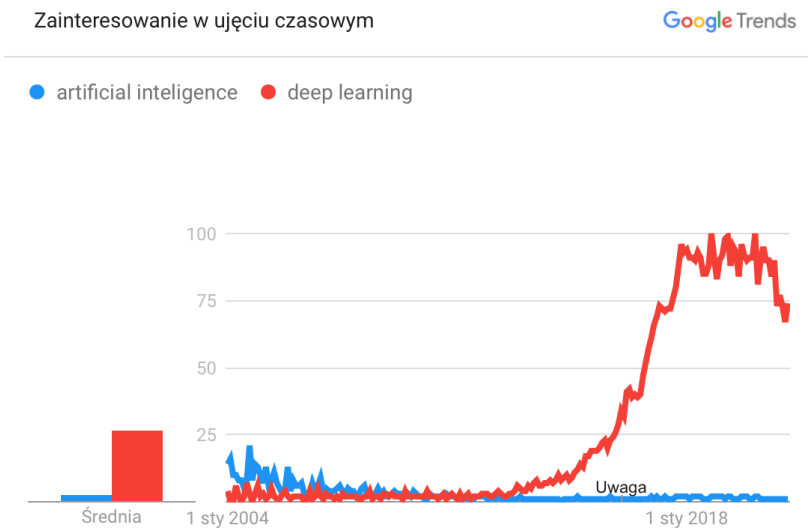




https://en.wikipedia.org/wiki/Deep_learning

Why deep learning became so popular?

Why deep learning became a crucial technology in last few years?



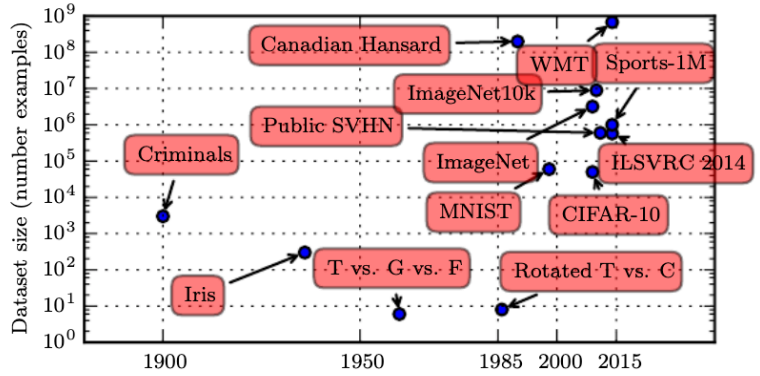


Figure 1.8: Increasing dataset size over time. In the early 1900s, statisticians studied datasets using hundreds or thousands of manually compiled measurements (Garson, 1900;

- Increasing dataset sizes
- a dumb algorithm with lots of data beats a clever algorithm with a modest amount of data

- is connected with increasing size of models

MNIST Dataset

https://www.deeplearningbook.org/contents/intro.html

- 60,000 examples, 10 classes
- features: 28x28x1
- http://yann.lecun.com/exdb/mnist/

ImageNet Dataset

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *arXiv preprint arXiv:1409.0575*. [web]

3

image-net.org: 14 M images, 1000 categories

What is COCO?

COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features:

- Object segmentation
- Recognition in context
- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 people with keypoints

cocodataset.org

Why deep learning became so popular?

How to train such a big models?

- big means that they can't be trained using CPU in a reasonable time

CPU vs GPU

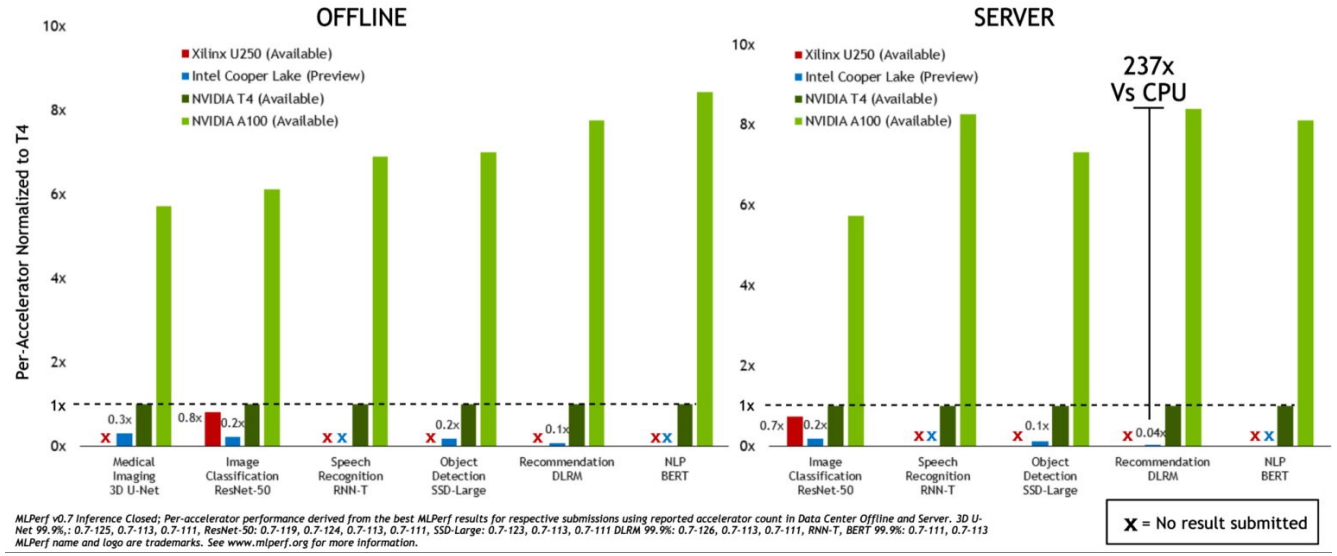
	Cores	Clock Speed	Memory	Price	Speed
CPU (Intel Core i7-7700k)	4 (8 threads with hyperthreading)	4.2 GHz	System RAM	$339	~540 GFLOPs FP32
GPU (NVIDIA GTX 1080 Ti)	3584	1.6 GHz	11 GB GDDR5 X	$699	~11.4 TFLOPs FP32
TPU NVIDIA TITAN V	5120 CUDA, 640 Tensor	1.5 GHz	12GB HBM2	$2999	~14 TFLOPs FP32 ~112 TFLOP FP16
TPU Google Cloud TPU	?	?	64 GB HBM	$6.50 per hour	~180 TFLOP

CPU: Fewer cores, but each core is much faster and much more capable; great at sequential tasks

GPU: More cores, but each core is much slower and "dumber"; great for parallel tasks

TPU: Specialized hardware for deep learning

CPU GPU TPU



ASICs specifically designed for deep learning

source: nvidia.com

https://medium.com/dataseries/cpu-gpu-and-tpu-machine-learning-5033e25b8a0f

Why deep learning became so popular?

Deep learning models size vs human brain

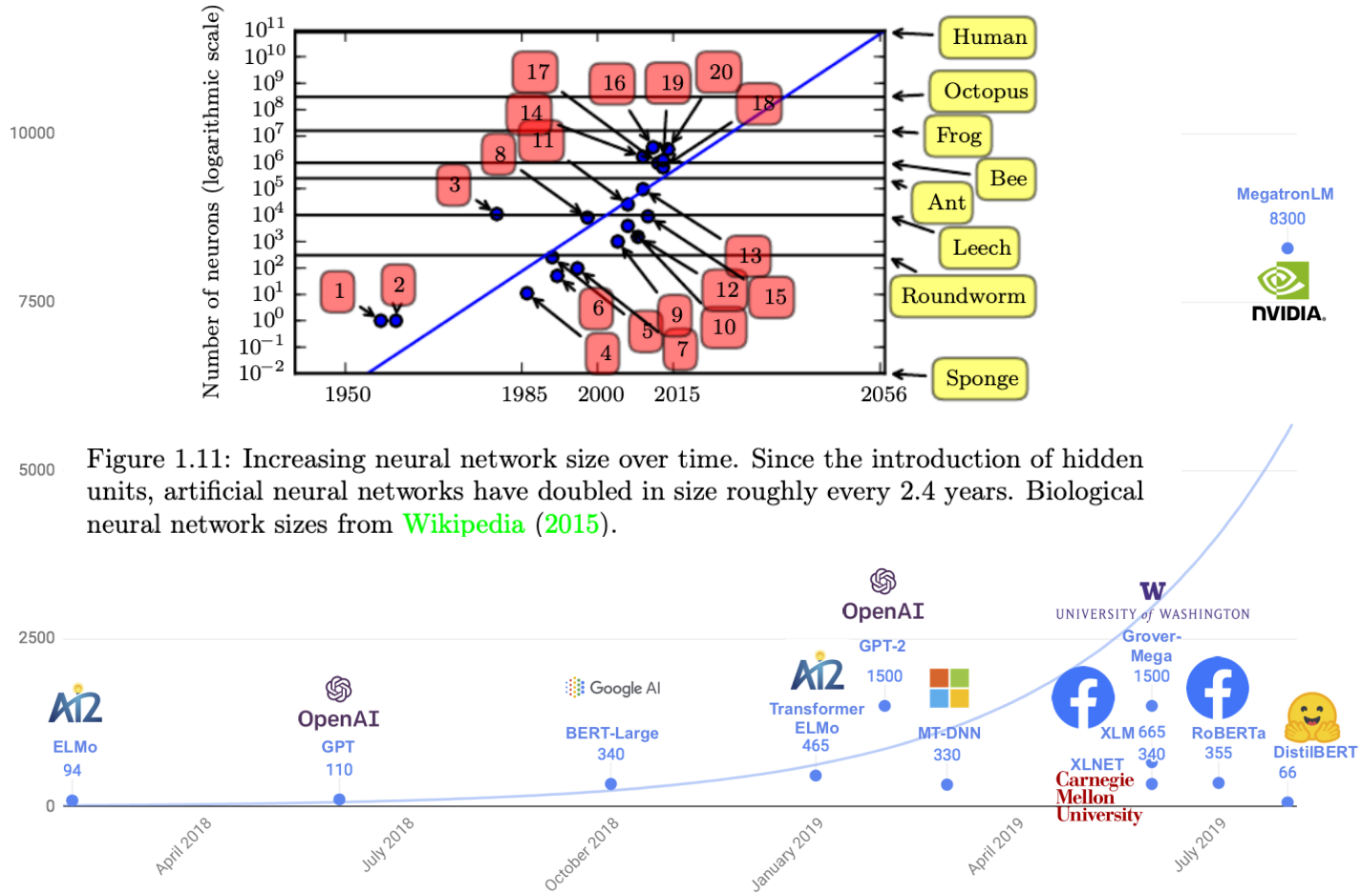


Figure 1.11: Increasing neural network size over time. Since the introduction of hidden units, artificial neural networks have doubled in size roughly every 2.4 years. Biological neural network sizes from Wikipedia (2015).

https://www.deeplearningbook.org/contents/intro.html
https://medium.com/huggingface/distilbert-8cf3380435b5

artificial neural network vs (human) brain
- differences:

- number of connections: human brain has 10^5 times more connections in total
- topology: connections are randomly distributed, artificial networks for layered feedforward structure
- human brain is much faster in generating outputs
- brain structure is much more stable an resilient to errors (artificial networks are susceptible to attacks)
- brains are being in training mode continuously

Image Source: https://timedotcom.files.wordpress.com/2014/05/brain.jpg?w=1100&quality=85

Brain with 1.6 x 10^{10} neurons

10^4 -10^5 connections per neuron

approx. 10^{15} connections in total

Why deep learning became so popular?

NLP models are the biggest

10^{12}

SWITCH TRANSFORMERS: SCALING TO TRILLION PARAMETER MODELS WITH SIMPLE AND EFFICIENT SPARSITY

William Fedus*
Google Brain
liamfedus@google.com

Barret Zoph*
Google Brain
barretzoph@google.com

Noam Shazeer
Google Brain
noam@google.com

ABSTRACT

In deep learning, models typically reuse the same parameters for all inputs. Mixture of Experts (MoE) models defy this and instead select *different* parameters for each incoming example. The result is a sparsely-activated model – with an outrageous number of parameters – but a constant computational cost. However, despite several notable successes of MoE, widespread adoption has been hindered by complexity, communication costs, and training instability. We address these with the Switch Transformer. We simplify the MoE routing algorithm and design intuitive improved models with reduced communication and computational costs. Our proposed training techniques mitigate the instabilities, and we show large sparse models may be trained, for the first time, with lower precision (bfloat16) formats. We design models based off T5-Base and T5-Large (Raffel et al., 2019) to obtain up to 7x increases in pre-training speed with the same computational resources. These improvements extend into multilingual settings where we measure gains over the mT5-Base version across all 101 languages. Finally, we advance the current scale of language models by pre-training up to trillion parameter models on the "Colossal Clean Crawled Corpus", and achieve a 4x speedup over the T5-XXL model.[1]

https://arxiv.org/pdf/2101.03961.pdf

Machine learning approaches

Supervised learning:

- data is organized in pairs (input, expected output)
- problems: classification, regression
- SVM, decision trees, neural networks
- gaussian discriminant analysis, naive Bayes

"training examples"

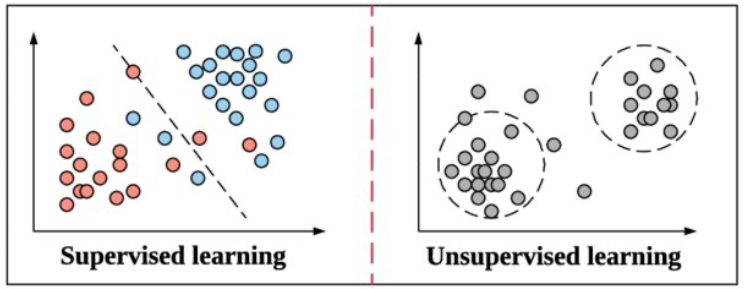
Training set: $\mathcal{D} = \{\langle \mathbf{x}^{[i]}, y^{[i]} \rangle, i = 1, \ldots, n\}$,

Unknown function: $f(\mathbf{x}) = y$

Hypothesis: $h(\mathbf{x}) = \hat{y}$ sometimes

Classification: $h : \mathbb{R}^m \to \mathcal{Y}, \quad \mathcal{Y} = \{1, \ldots, k\}$

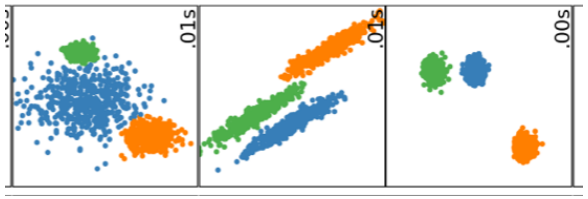
Regression: $h : \mathbb{R}^m \to \mathbb{R}$



www.researchgate.net/project/software-defined-network-3

Unsupervised learning:

- in opposite to supervised learning, data is not labeled
- problems: clustering, dimensionality reduction
- PCA, LDA, k-means clustering



https://scikit-learn.org

Reinforcement learning

Example: Supervised vs Unsupervised

- Having N photos of different animals
- Supervised task (requires labeled data)

> Train an algorithm to recognise given species on a photo.
>
> Output: There is X on a photo.

- Unsupervised task

> Train an algorithm to group animals with similar features.
>
> Output: No idea what it is, but it looks similar to these animals.

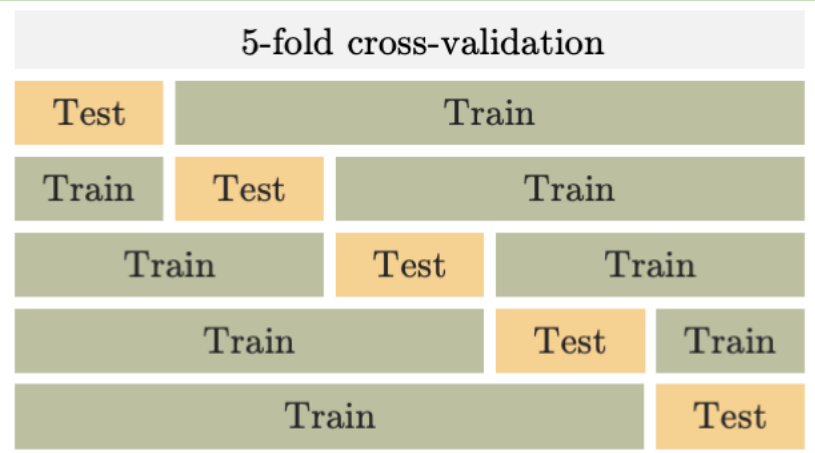
source: tomaszgolan.github.io/introduction_to_machine_learning

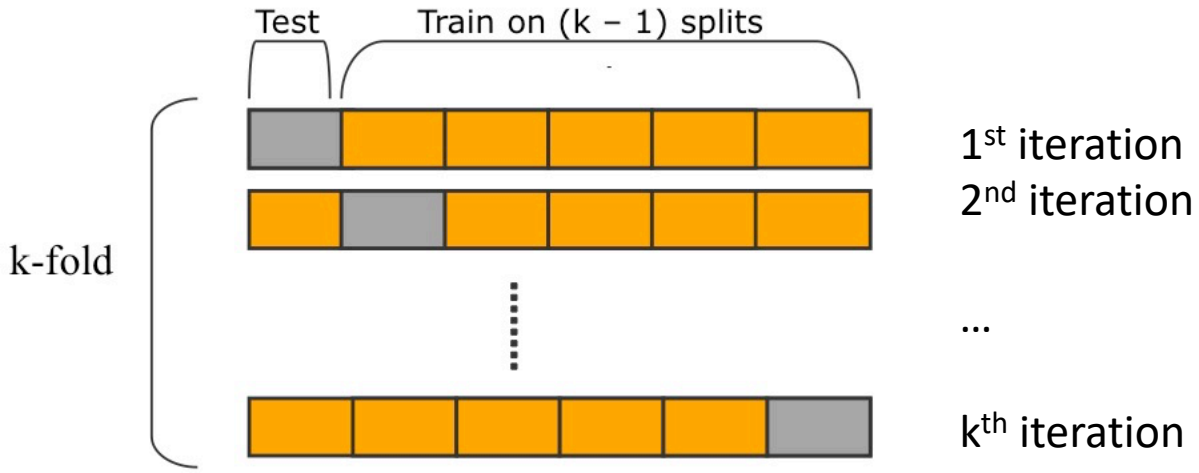
Learning scheme

How to perform learning and evaluate model performance

- divide dataset into **train** and **test** subsets
  - test should be evaluated on data *unseen* during training

- *k*-fold cross-validation





1st iteration
2nd iteration

…

kth iteration

https://aszokalski.github.io/AI/Cross-Validation.html

- in practice, while testing big models,
  we limit ourselves to only one *k*-fold iteration

Bias-variance tradeoff

The ability to perform well on previously unobserved inputs is called **generalization**:

- during learning we optimize model to have lowest **training error**,
- we want the generalization error, also called the **test error** to be low as well.

Training and testing error correspond to the two central challenges in machine learning: (model) **underfitting** and **overfitting.**

strongly recommended paper: https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf

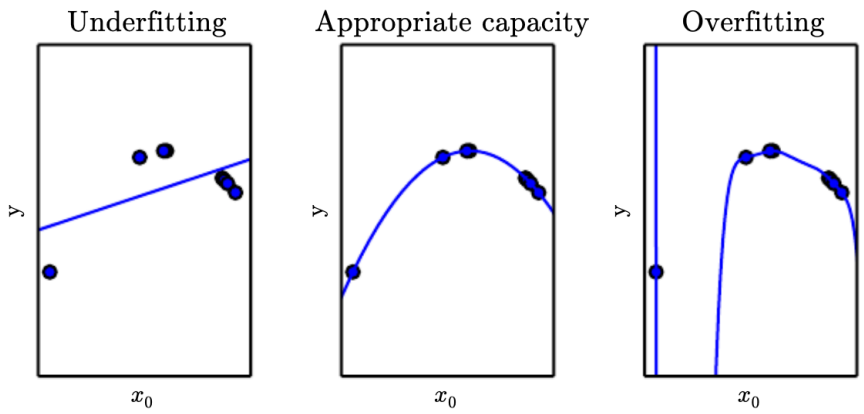
Bias-variance tradeoff

Underfitting: model is not able to obtain a sufficiently low error on the training set

Overfitting occurs when the distance between the training and test error is too large

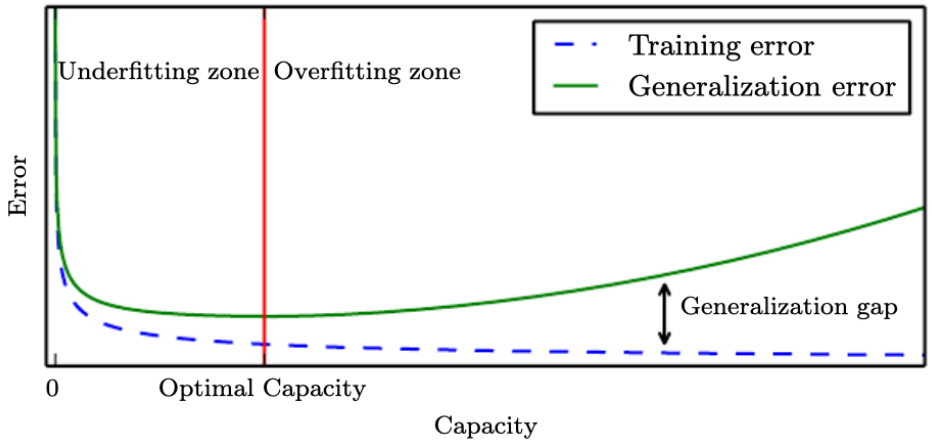
We can control whether a model is more likely to overfit or underfit by altering its **capacity**

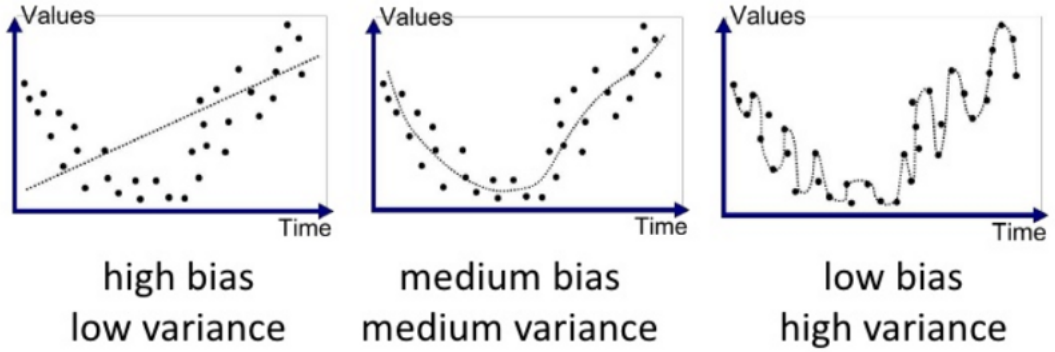
Here capacity means degree of the fitted polynomial:



https://www.deeplearningbook.org/contents/ml.html

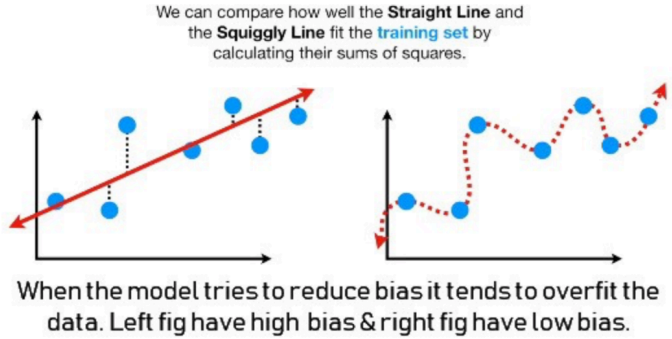
Bias-variance tradeoff





$$\mathrm{MSE} = \mathbb{E}\left[\left(\hat{x}(t) - x(t)\right)^2\right] = \mathrm{Bias}\left(\hat{x}(t)\right)^2 + \mathrm{Var}\left(\hat{x}(t)\right),$$
$$\text{with } \mathrm{Bias}\left(\hat{x}(t)\right) = \mathbb{E}\left(\hat{x}(t)\right) - x(t)$$

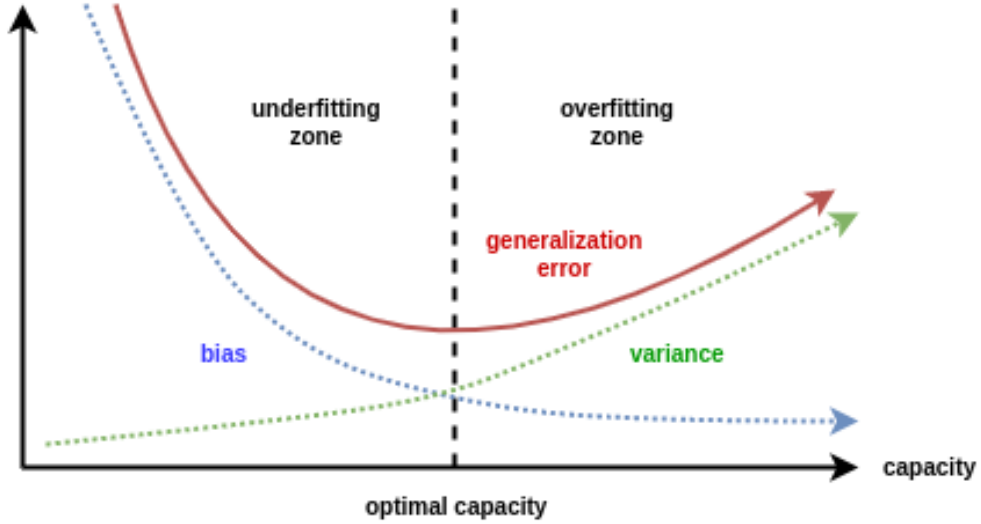
- High bias can cause an algorithm to miss the relevant relations in data (underfitting).
- High variance can cause an algorithm to model the random noise in the training data (overfitting).

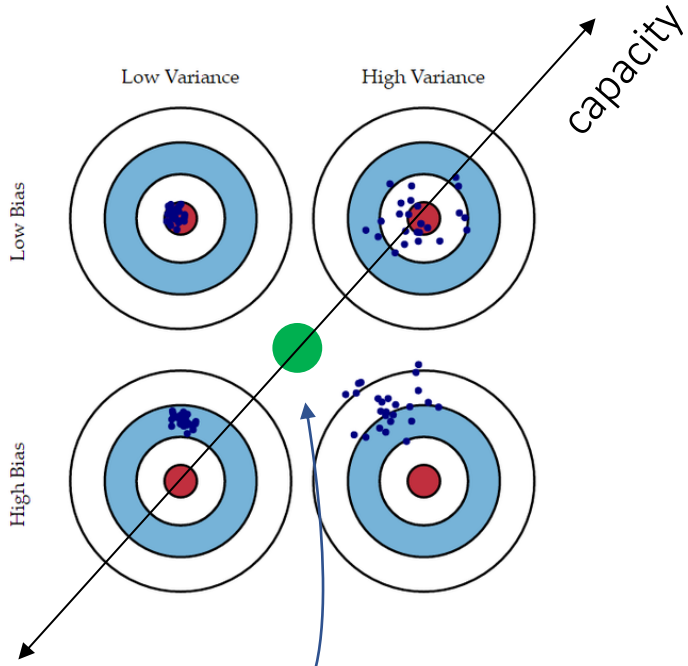


https://www.deeplearningbook.org/contents/ml.html
https://towardsdatascience.com/tradeoffs-how-to-aim-for-the-sweet-spot-c20b40d5e6b6

Bias-variance tradeoff

To low bias will result in to high variance and vice versa





optimization
“sweet spot”

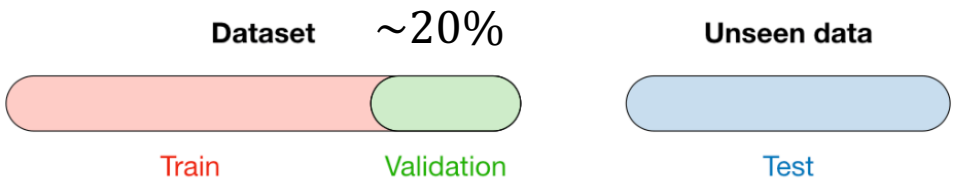
https://towardsdatascience.com/tradeoffs-how-to-aim-for-the-sweet-spot-c20b40d5e6b6

Avoiding overfitting: validation set

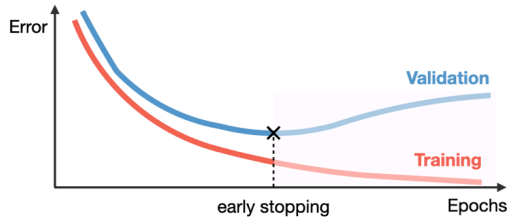
How to avoid overfitting

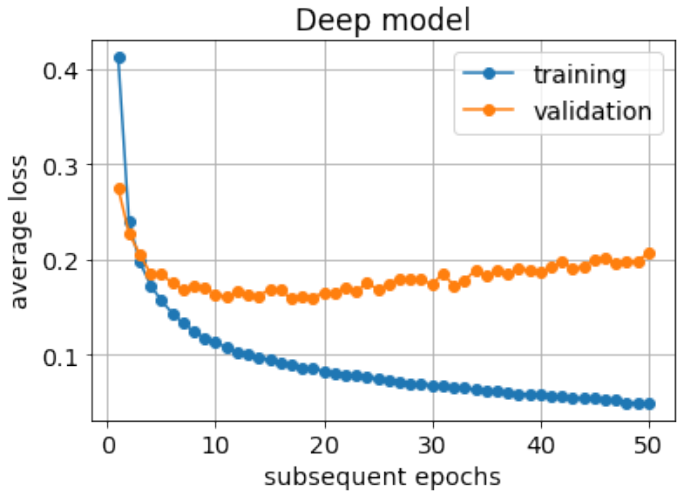
- The model should chosen/tuned (*hyperparameters*) to be as simple as possible but not simpler (Occam's razor), i.e. with appropriate capacity.

We need another subset (**validation** set) to estimate the generalization error during training, allowing for the model hyperparameters to be updated accordingly.



□ **Early stopping** — This regularization technique stops the training process as soon as the validation loss reaches a plateau or starts to increase.





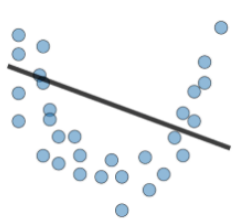
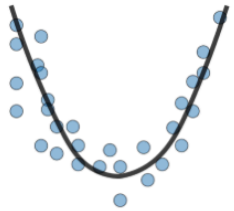
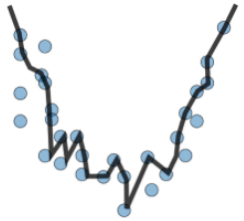
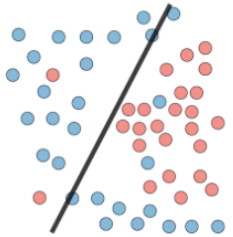
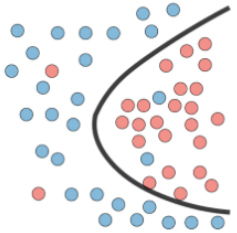
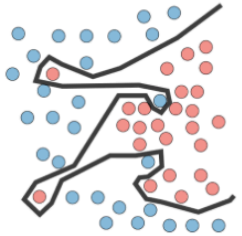
training stopping moment is one of the hyperparameters to be tuned

https://stanford.edu/~shervine/teaching/cs-229/cheatsheet-machine-learning-tips-and-tricks

Avoiding overfitting

How to avoid overfitting

- Do not train to long!
- Regularization: adding weight to loss function to prefer one types of solutions over other (e.g. with lower polynomial degree)
- Augmentation
- Increase dataset size (but new data should be coherent)
  …a dumb algorithm with lots of data beats a clever algorithm with a modest amount of data…

	Underfitting	Just right	Overfitting
Symptoms	• High training error • Training error close to test error • High bias	• Training error slightly lower than test error	• Very low training error • Training error much lower than test error • High variance
Regression illustration			
Classification illustration			
Deep learning illustration			
Possible remedies	• Complexify model • Add more features • Train longer		• Perform regularization • Get more data

https://stanford.edu/~shervine/teaching/cs-229/cheatsheet-machine-learning-tips-and-tricks

No free lunch theorem

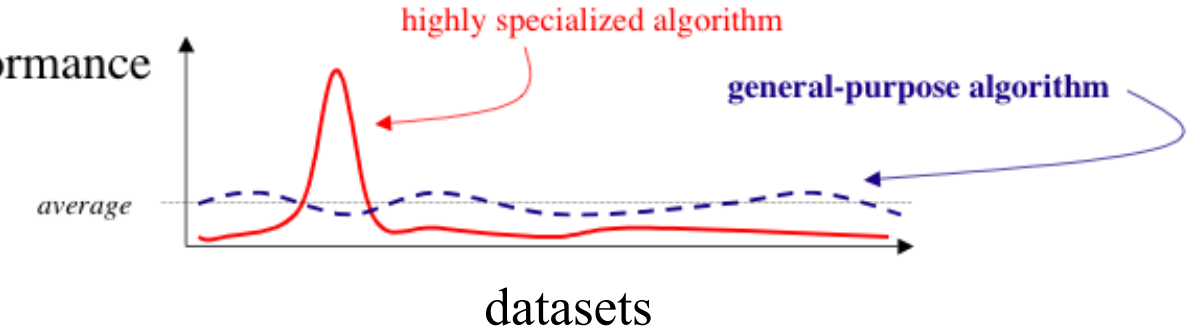
Suppose that we have all possible domains (datasets) for a given problem (e.g. image classification) and need to compare two methods A and B:

No free lunch theorem states that:

- mean performance for algorithms A and B averaged over the all possible datasets is the same,
- this holds true even when one of the algorithms is just random guessing.

There is no such thing as a single, universally-best machine learning algorithm, and there are no *a priori* reasons to favor one algorithm over all others.

- we can't get good machine learning for *free*.
- we must use knowledge about out data and the context, to select an appropriate machine learning model.

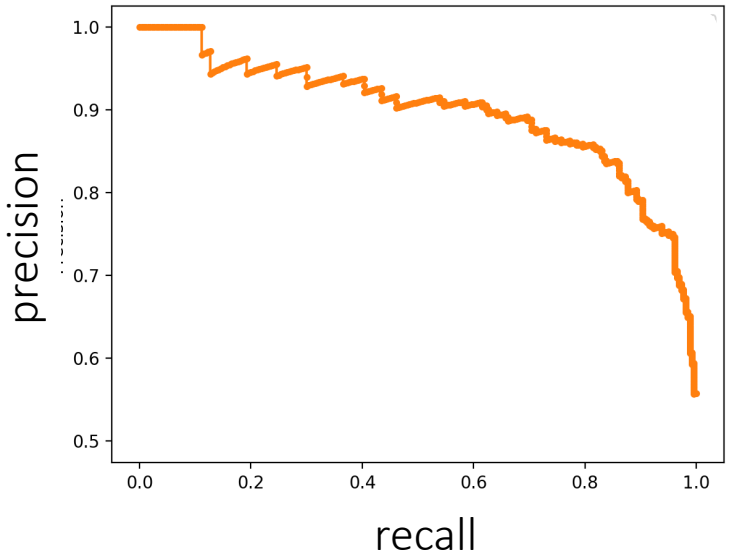


https://medium.com/@LeonFedden/the-no-free-lunch-theorem-62ae2c3ed10c

Precision-recall tradeoff

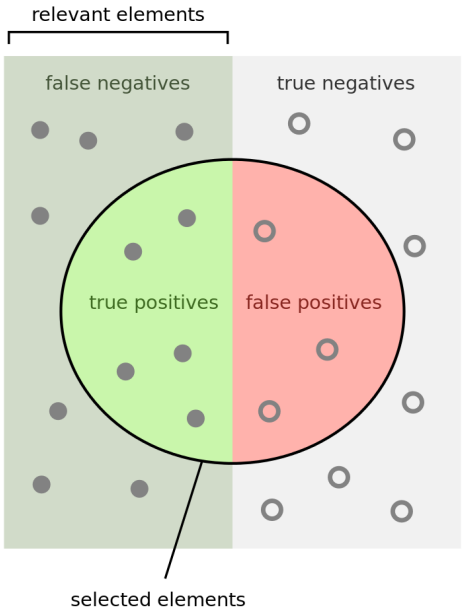
classification and detection metrics:
- information stored in *confusion matrix* are used to calculate:
- accuracy,
- precision, recall,
- F1 score

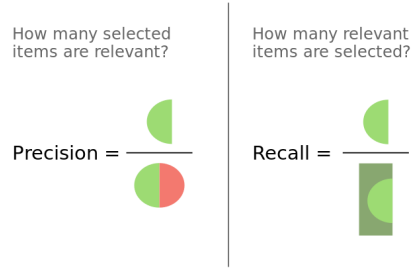
precision-recall tradeoff:
- if you increase precision (by model tuning), it will reduce recall, and vice versa.



		Predicted class +	**Predicted** class -
Actual class	+	**TP** True Positives	**FN** False Negatives Type II error
	-	**FP** False Positives Type I error	**TN** True Negatives

Metric	Formula	Interpretation
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$	Overall performance of model
Precision	$\frac{TP}{TP + FP}$	How accurate the positive predictions are
Recall Sensitivity	$\frac{TP}{TP + FN}$	Coverage of actual positive sample
Specificity	$\frac{TN}{TN + FP}$	Coverage of actual negative sample
F1 score	$\frac{2TP}{2TP + FP + FN}$	Hybrid metric useful for unbalanced classes





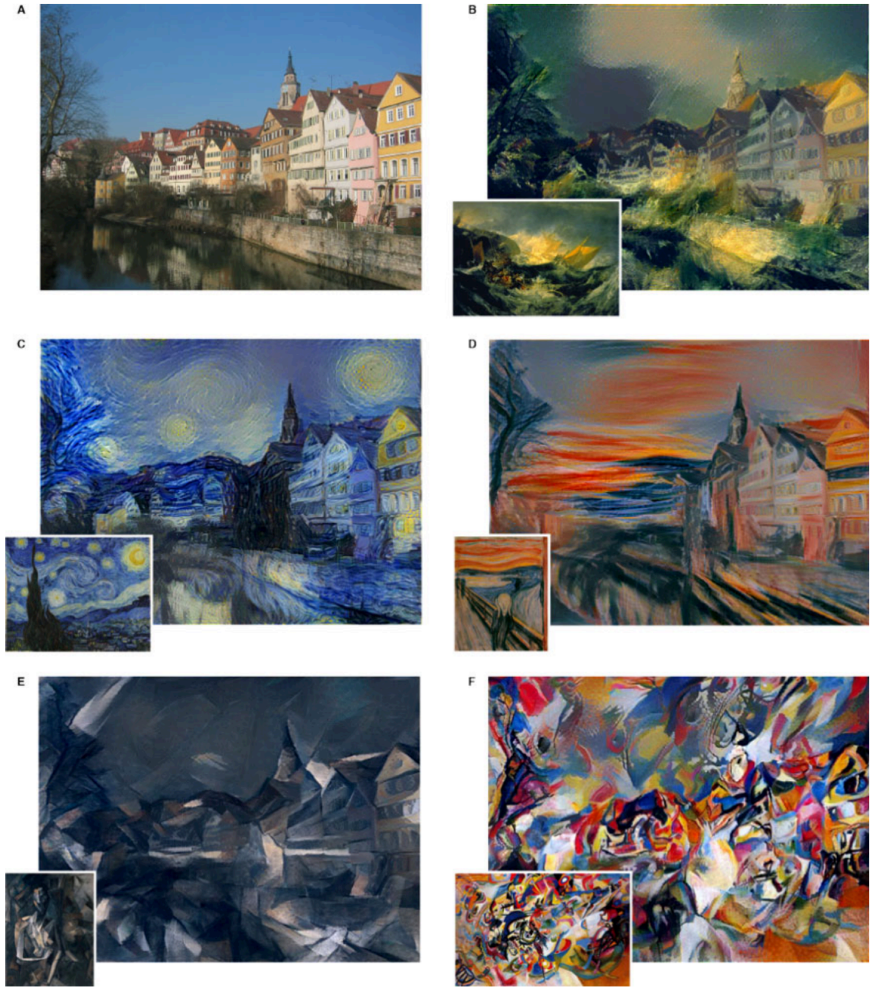
https://en.wikipedia.org/wiki/Precision_and_recall

https://stanford.edu/~shervine/teaching/cs-229/cheatsheet-machine-learning-tips-and-tricks

ML applications

ML applications

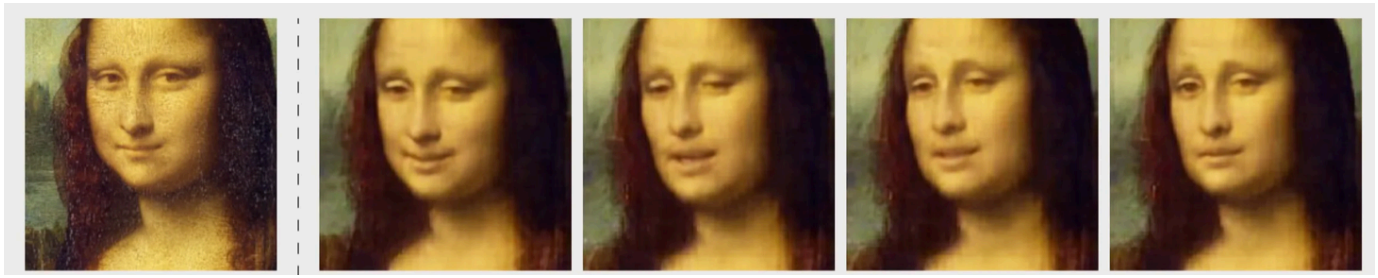
- Image recognition
  - Google Maps - finding licence plates and faces; extracting street names and building numbers
  - Facebook - recognising similar faces
- Speech recognition
  - Microsoft - Cortana
  - Apple - Siri
- Natural Language Processing
  - Google Translate - machine translation
  - Next Game of Thrones Book - language modeling
- Misc
  - PayPal - fraud alert
  - Netflix, Amazon - recommendation system



https://tomaszgolan.github.io/introduction_to_machine_learning

https://arxiv.org/abs/1508.06576

ML applications: deep fake



Deep ML: potential dangers

Deep fake wars:
Facebook is actively seeking for (ML) tools to detect/eliminate deep fake information

https://miroslawmamczur.pl/

ML frameworks

ML solution are nowadays extremely easy to implement

- we are able to build quite a complex deep neural network model in a 5 minutes
  (of course learning itself can take 5 days ;-)