Machine Learning in Communication Networks

Sandeep Kumar Singh
German Aerospace Center (DLR)
Institute of Communications and Navigation, Germany

November 10th, 2020

Outline

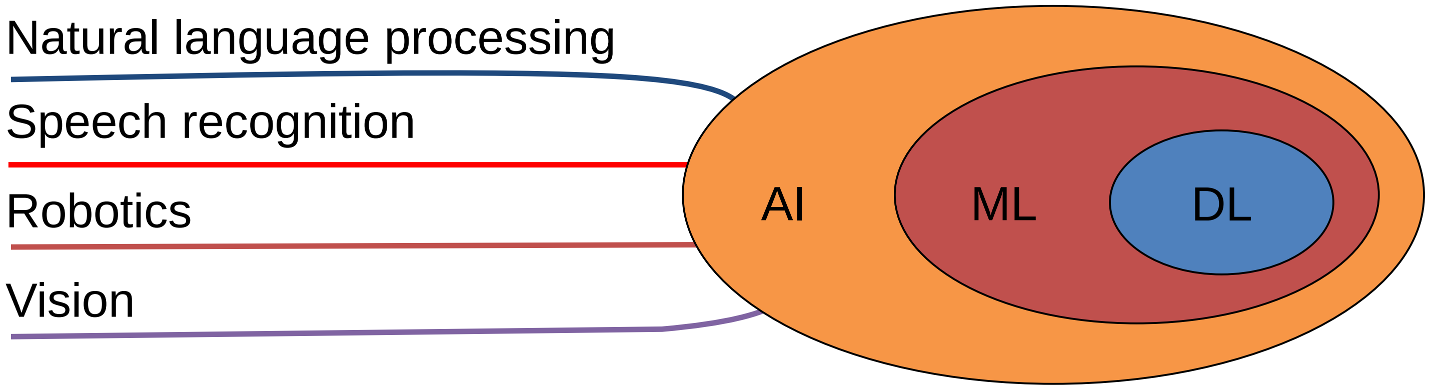
- What is Machine Learning (ML)?
- Applications and challenges
- Basic concepts and Its types
- Algorithms
- Anomaly detection
- Time-series forecasting

Unless otherwise stated, some slides are taken from M. Tornatore: Tutorial on Machine Learning

What is Machine Learning (ML)?

- *“Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed.”*- Arthur Samuel (1959):
- *“Teaching a computer to automatically learn concepts through data observation”*

- For our purposes (in the context of communication networks):
  - A math/statistical instrument to make decisions by inferring statistical properties of monitored data



AI Techniques, Applications and Challenges in Communication Networks

Artificial Intelligence

Applications

- Network Planning
- Control & Management
- Resource Management
- Security and Privacy
- Attack Detection
- Low Power Consumption
- Transmission

Challenges

- Multiple Admin Domains
- Multi-vendor Devices
- Privacy
- Low Cost IoT Devices

Techniques

- Machine learning
- Statistical learning
- Genetic Algorithms
- Swarm Intelligence
- Fuzzy logic

Models/Algorithms

- Neural networks (A/C/R)
- Support Vector Machines
- Decision Tree
- Random Forest
- K-means
- K-Nearest Neighbors

Models/Algorithms

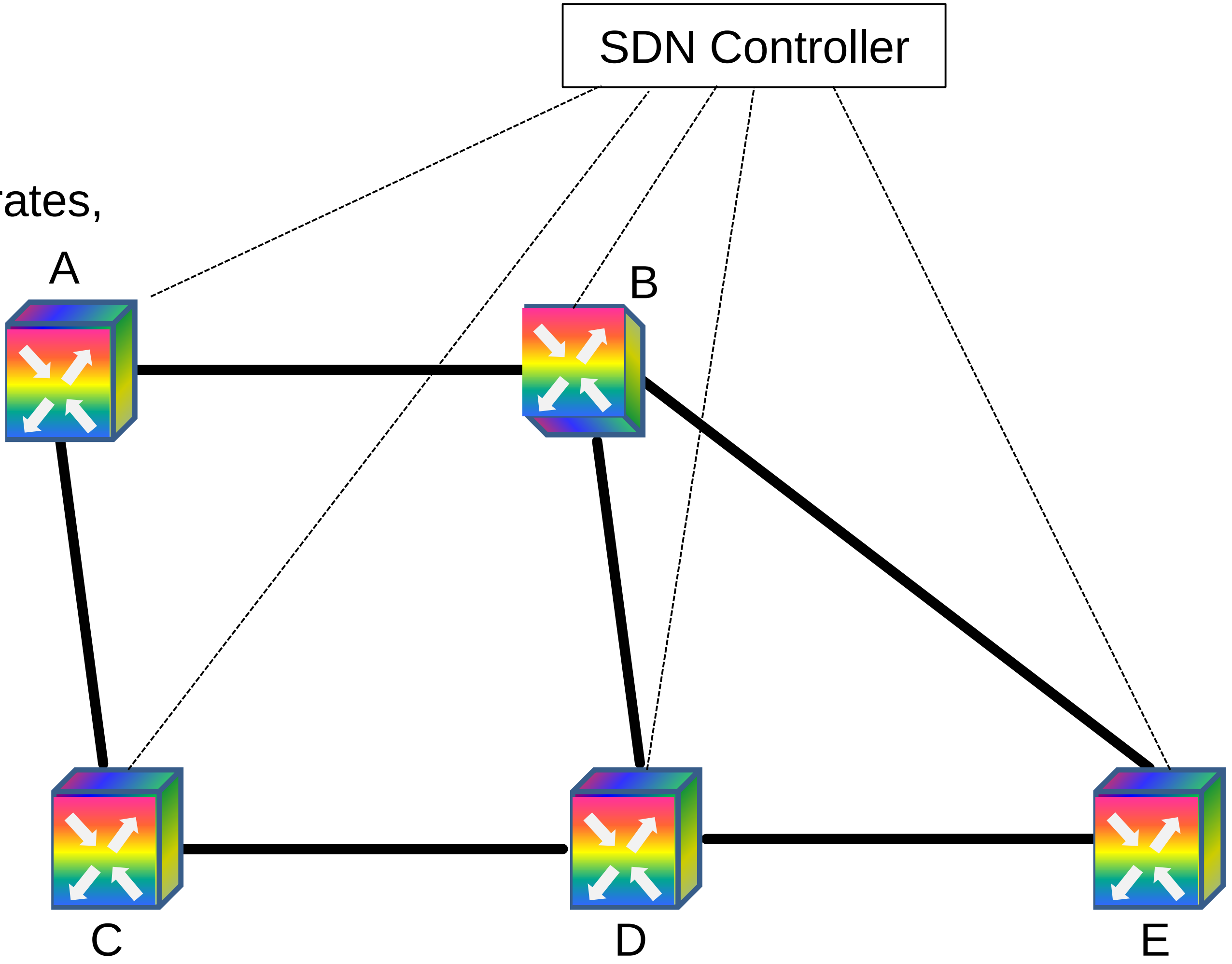
- Bayesian Methods
- Hidden Markov Models
- Kalman Filtering

*J. Mata, I. de Miguel, R. J Duran, and N. Merayo, S. K Singh, A. Jukan, and M. Chamania, "Articial intelligence (AI) methods in optical networks: A comprehensive survey," Optical Switching and Networking, vol. 28, pp.43-57, 2018

Why Only Now in Networking?

- Data Plane-Complexity increase
  - Coherent Transmission System
    - Several system parameters to choose from: modulation techniques and formats, coding rates, symbol rate…
    - DSP: Huge availability of data
  - Telecommunication or Optical Networks
    - Customizable channel width, BV-ROADM

- Control Plane-New Enablers
  - Software Defined Networking
  - Intelligence (computing capabilities) everywhere



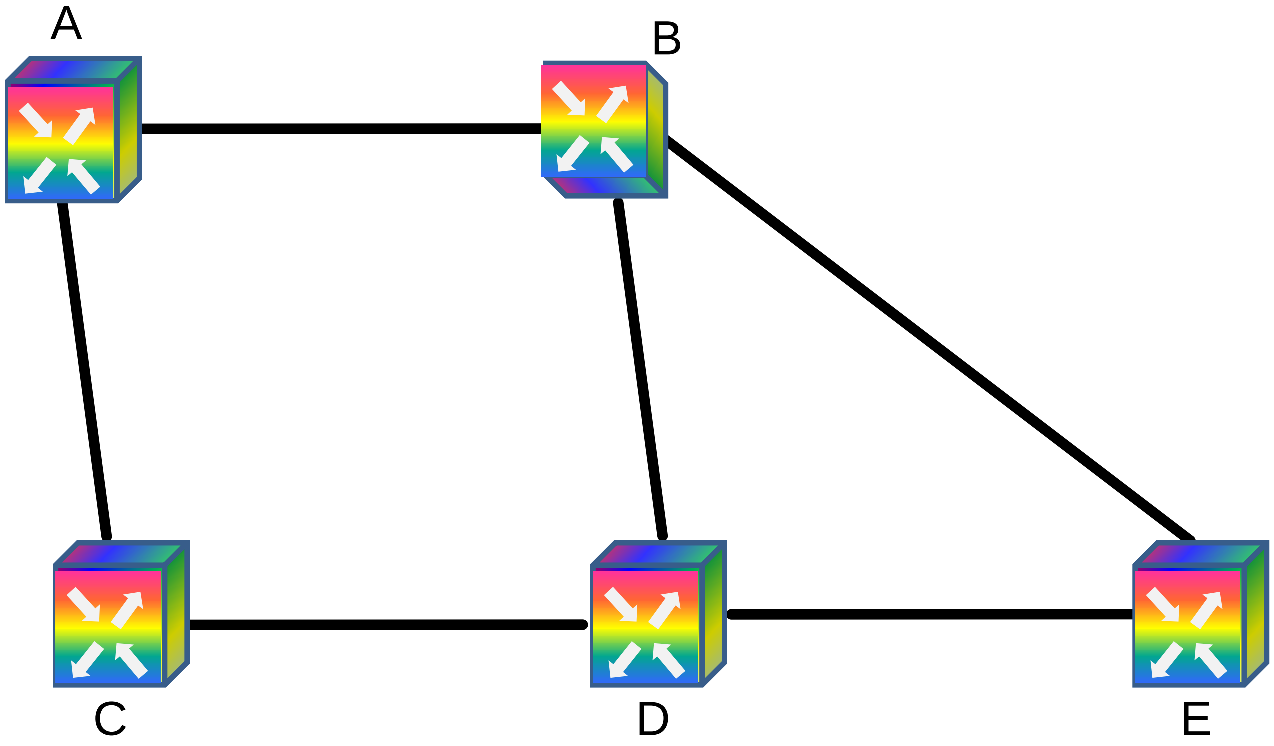
ML Categories (1)

- **Supervised-learning**
  - Training: “labelled” data
  - Main objective: given a set of “historical” input(s) predict an output
    - – Regression: output value is continuous
    - – Classification: output value is discrete or “categorical”
  - An example: Traffic forecasts
    - Given traffic during last week/month/year
    - – Predict traffic for the next period (regression)
    - – Predict if available resources will be sufficient (classification)
  - Other examples
    - – Speech/image recognition
    - – Spam classifier
    - – House prices prediction/estimation

A Supervised Learning Example

Establish a new lightpath (optical channel)

Objective: find BER of a new lightpath a priori at a particular wavelength, mod., ROADM age, on a given path



A Supervised Learning Example

Training:

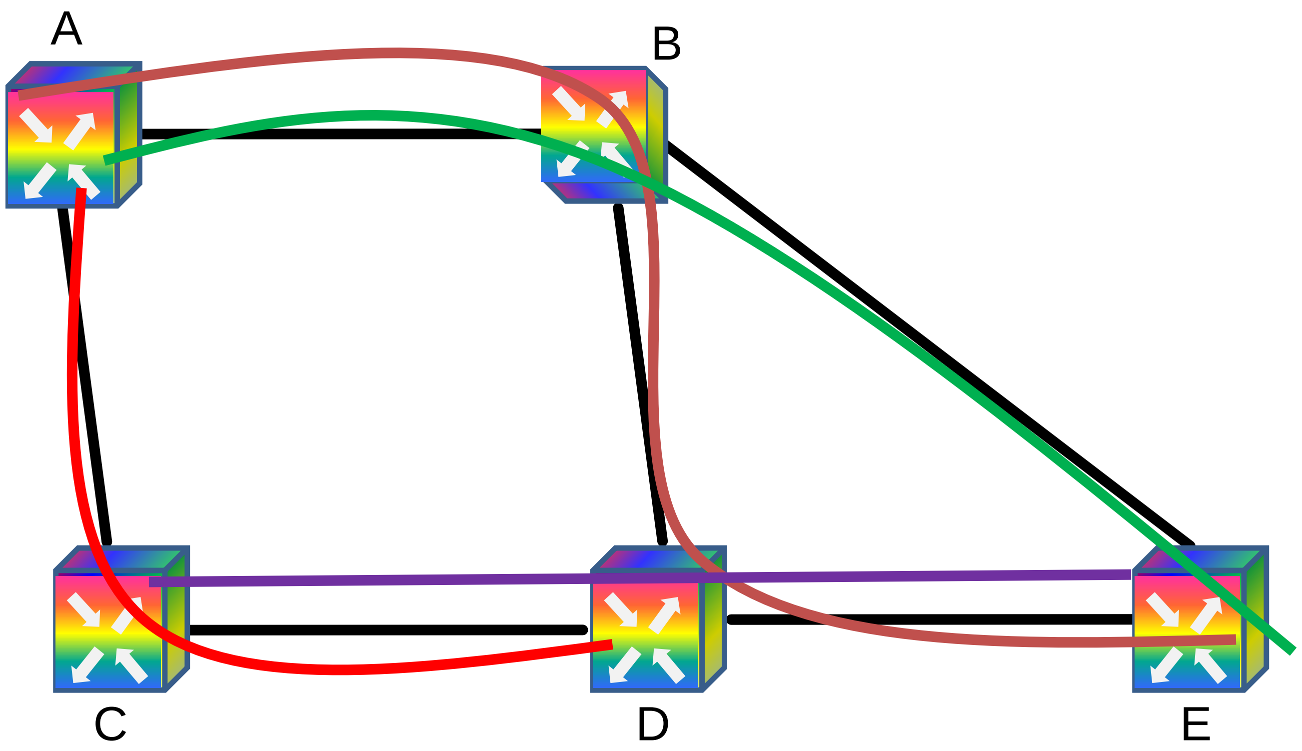
$\lambda = 1550$ nm, path = A-B-D-E, mod = QPSK, BER = 10^{-5}

$\lambda = 1553$ nm, path = A-B-E, mod = QPSK, BER = 10^{-6}

…

Testing:

$\lambda = 1555$ nm, path = C-D-E, mod = QPSK, BER = ?



ML Categories (2)

- **Unsupervised Learning**
- Data is not “labelled”
- Main objective: derive structures (patterns) from available data
  - Clustering finding “groups” of similar data
  - Anomaly detection
- An example: cell traffic classification
- Given traffic traces
  - Understand if some cells provide similar patterns
  - Residential, business, close to theatre, cinema, stadium...
  - This information can be used to make network resources planning
- Other example
  - Group people according to their interests to improve advertisement

An Unsupervised Learning Example (1)

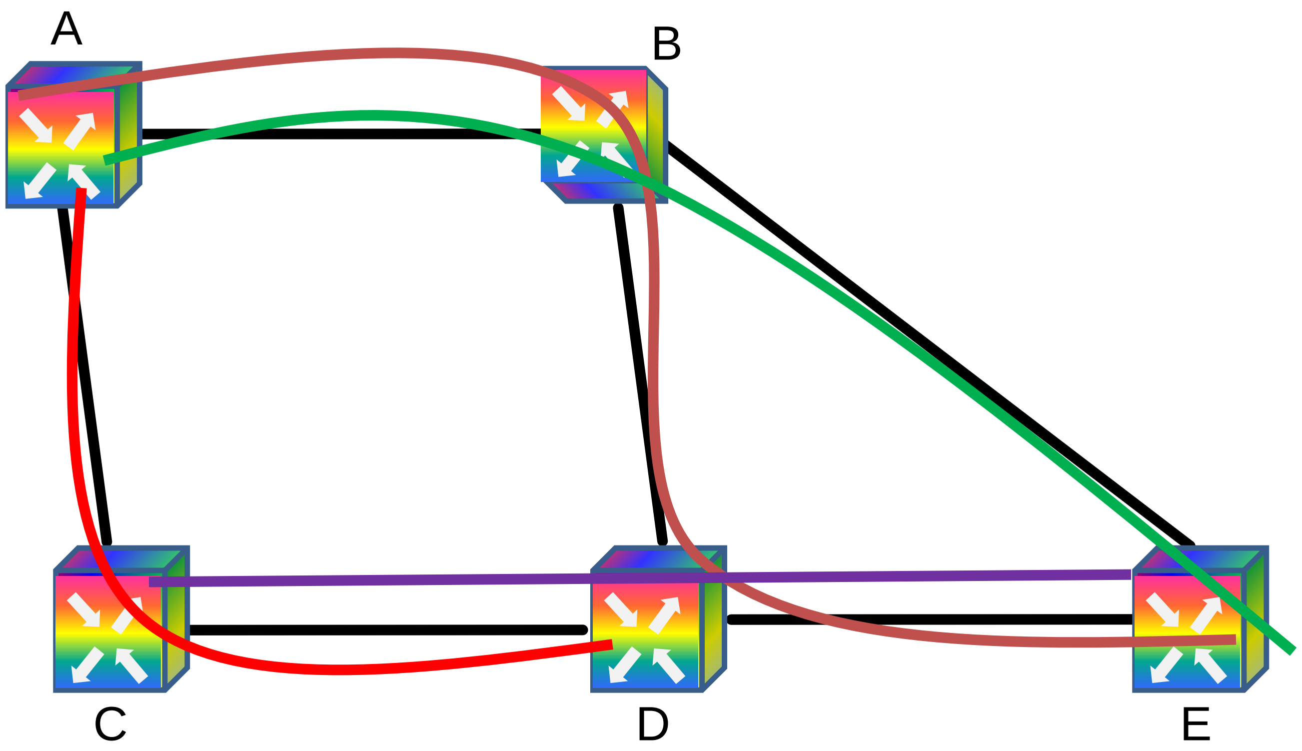
Data:

$\lambda = 1550$ nm, path = A-B-D-E, mod = QPSK, BER = 10^{-5}

$\lambda = 1553$ nm, path = A-B-E, mod = QPSK, BER = 10^{-6}

$\lambda = 1553$ nm, path = A-C-D, mod = QPSK, BER = 10^{-6}

$\lambda = 1555$ nm, path = C-D-E, mod = QPSK, BER = 10^{-5}



An Unsupervised Learning Example (2)

Data after few months:

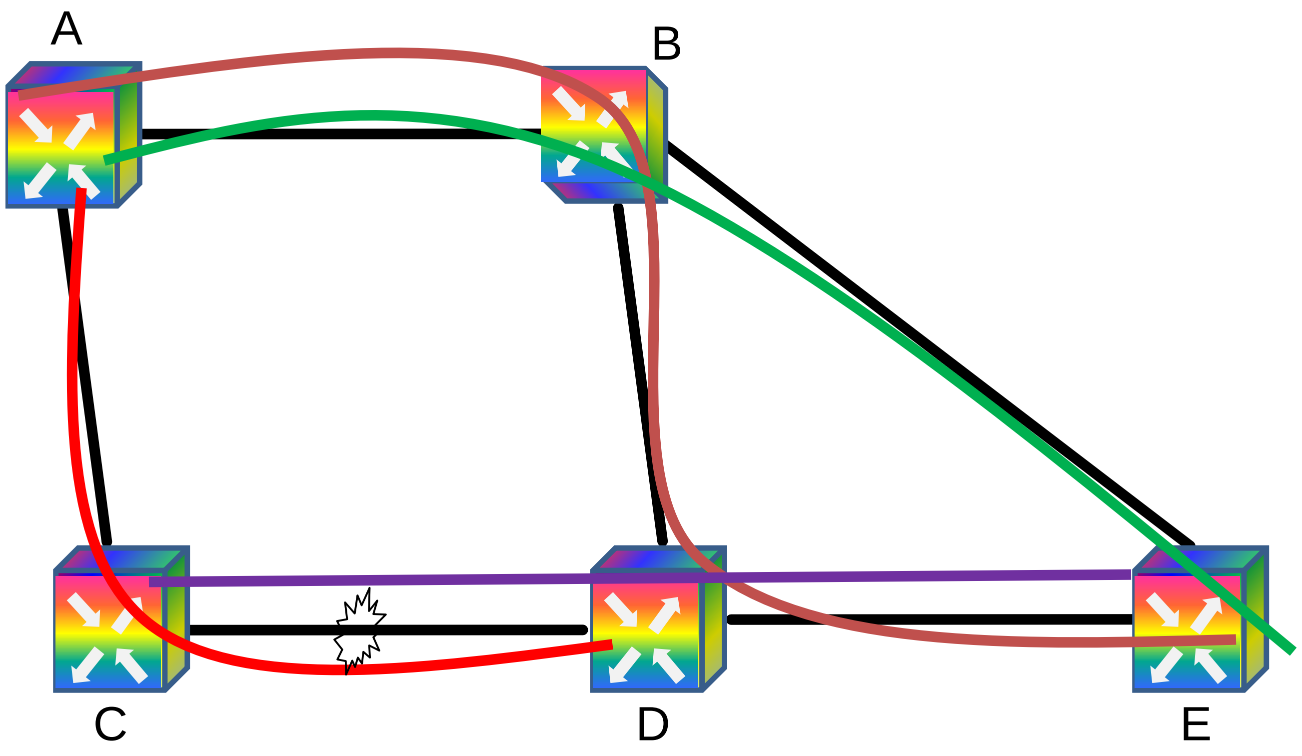
$\lambda = 1550$ nm, path = A-B-D-E, mod = QPSK, BER = 10^{-5}

$\lambda = 1553$ nm, path = A-B-E, mod = QPSK, BER = 10^{-6}

$\lambda = 1553$ nm, path = A-C-D, mod = QPSK, BER = 10^{-2}

$\lambda = 1555$ nm, path = C-D-E, mod = QPSK, BER = 10^{-2}

Anomaly detection



ML Categories (3)

- **Semi-Supervised learning**
  - Hybrid of previous two categories
  - **Main objective**: most of the training samples are unlabelled, only few are labelled.
    - Common when labelled data are scarce or expensive
  - Self-training: start with labelled data, then label unlabelled data based on first phase

ML Categories (4)

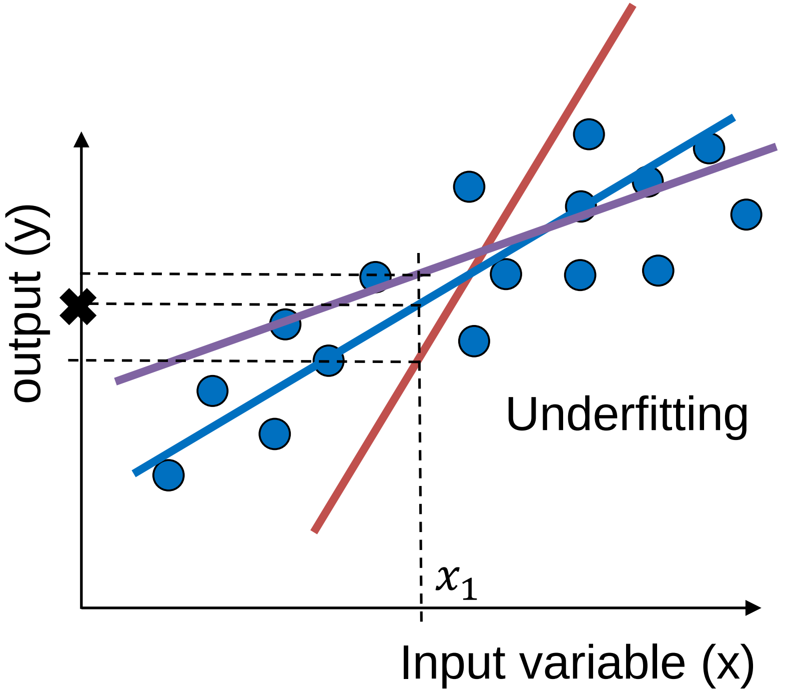
- **Semi-Supervised learning**
  - Hybrid of previous two categories
  - **Main objective**: most of the training samples are unlabelled, only few are labelled.
    - Common when labelled data are scarce or expensive
  - Self-training: start with labelled data, then label unlabelled data based on first phase

- **Reinforcement learning**
  - Available data is not “labelled”
  - **Main objective**: learn a policy, i.e., a mapping between in inputs/states and actions. Behavior is refined through rewards
  - Methodologically similar to «optimal control theory» or «dynamic programming»

Supervised Learning : Regression

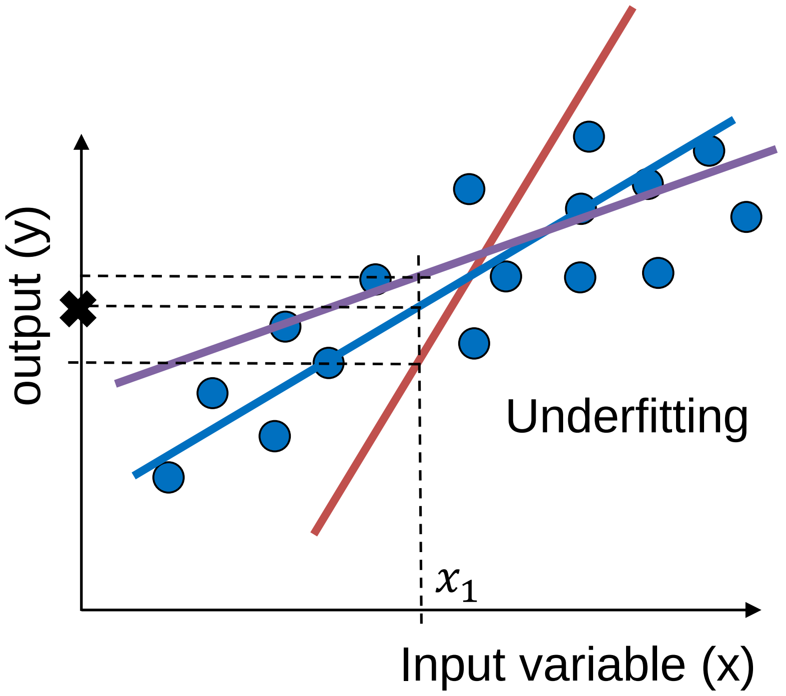
- $\mathbf{y} = h(\boldsymbol{x})$: relationship between one dependent variable (Y) and other variables independent variables
- y takes infinite real values
- Univariate: $h(x = x_1) = \theta_0 + \theta_1 x_1$
- Multivariate: $h(x = (x_1, x_2, \ldots, x_n) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n$
- Nonlinear regression: $h(x = (x_1, x_2, \ldots, x_n) = \theta_0$

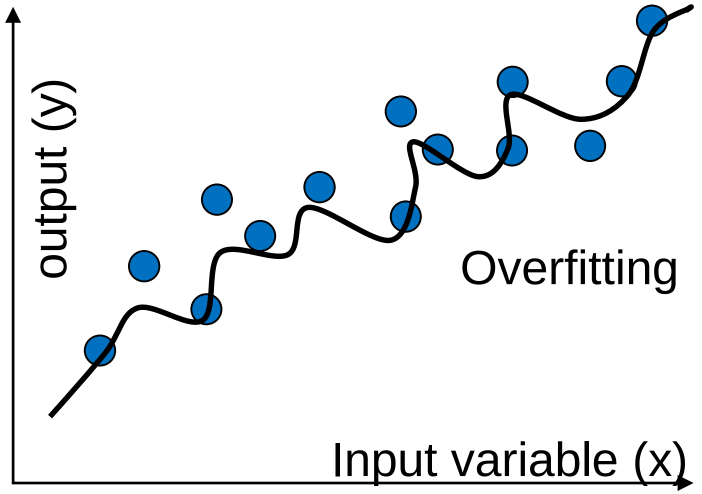
$$+\theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n$$
$$+\theta_{11} x_1 x_1 + \theta_{12} x_1 x_2 + \ldots + \theta_{1n} x_1 x_n$$
$$\ldots$$



Supervised Learning : Regression

- $\mathbf{y} = h(\boldsymbol{x})$: relationship between one dependent variable (Y) and other variables independent variables
- y takes infinite real values
- Univariate: $h(x = x_1) = \theta_0 + \theta_1 x_1$
- Multivariate: $h(x = (x_1, x_2, \dots, x_n) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$
- Nonlinear regression: $h(x = (x_1, x_2, \dots, x_n) = \theta_0$
  $+\theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$
  $+\theta_{11} x_1 x_1 + \theta_{12} x_1 x_2 + \dots + \theta_{1n} x_1 x_n$
  ...
- How to choose parameters (weights): $\boldsymbol{\theta}$ $(\theta_0,\ \theta_1,\ \theta_2, \dots)$?
- Minimize training loss: mean square error $min_{\boldsymbol{\theta}} \left\{ \mathrm{J}(\boldsymbol{\theta}) = \frac{1}{2m} \sum_{i=1}^{m} \left(h(\boldsymbol{x}^{\boldsymbol{i}}) - \boldsymbol{y}^{\boldsymbol{i}} \right)^2 \right\}$
- Gradient descent algo: Partial derivative $\frac{d}{d\boldsymbol{\theta}_j} \mathrm{J}(\boldsymbol{\theta}),\ \text{for j} = 0, 1, 2, \dots$
- Not enough data/too many features => overfitting





Supervised Learning: Logistic Regression/Classification

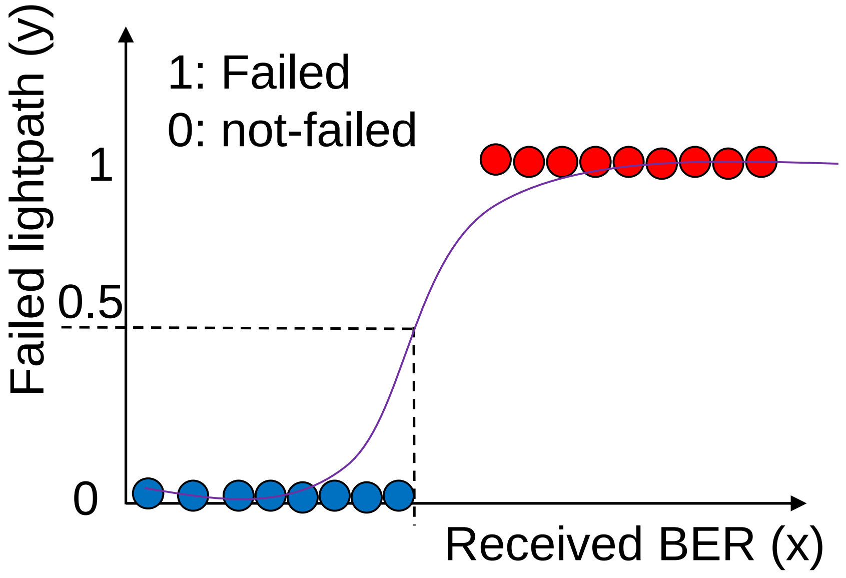
- Output $\mathrm{y} = h(\boldsymbol{x})$ takes finite discrete values
  - Binary classification: $\mathrm{y}\ \epsilon\ \{0, 1\}$, True/False, anomaly/normal
  - Multiclass classifier: $\mathrm{y}\ \epsilon\ \{0, 1, 2, 3, \ldots K\}$

$$h(\boldsymbol{x}) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n + \ldots)$$
$$= g(\theta^T \boldsymbol{x})$$

Where $g(.) - \boldsymbol{logistic\ function}$

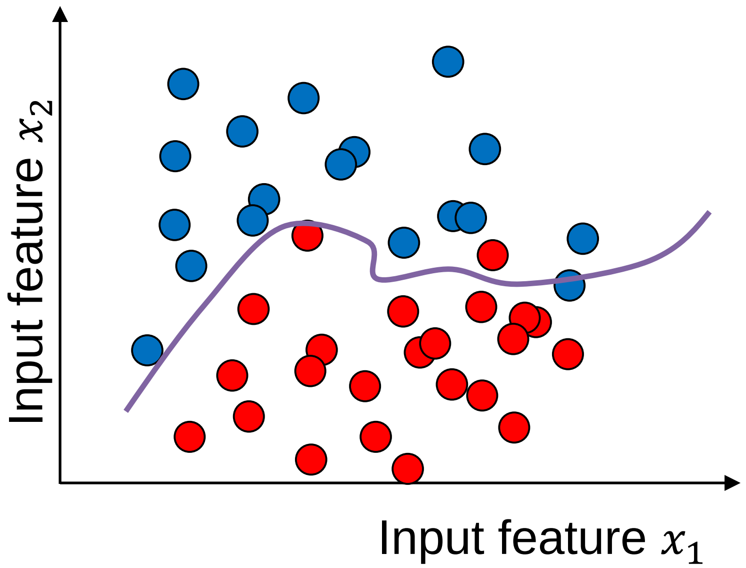
$$h(\boldsymbol{x}) = \frac{1}{1+e^{-\theta^T x}}$$ => Sigmoid function

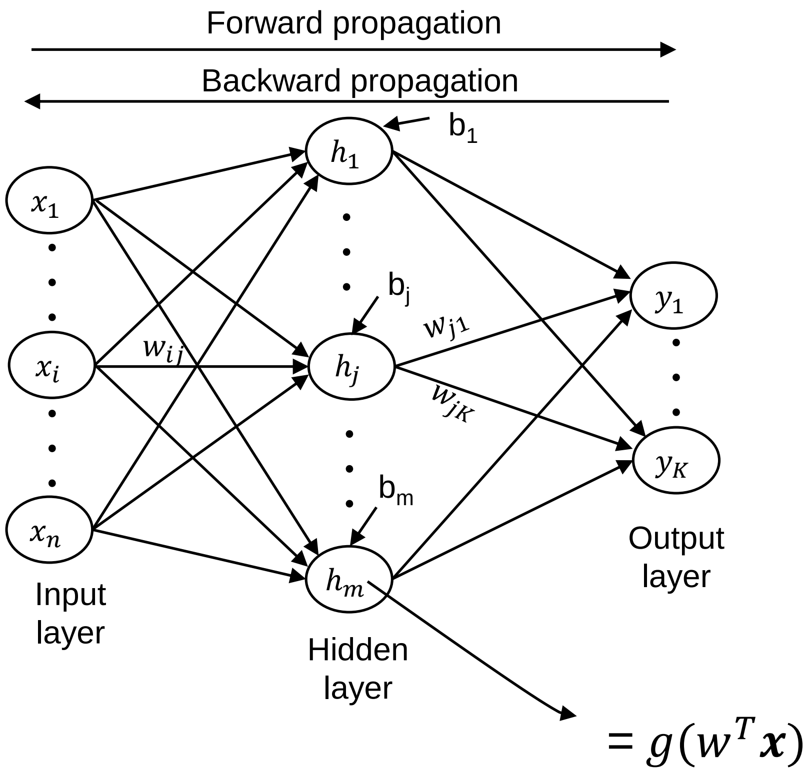
$\rightarrow p(y = k|\boldsymbol{x}, \theta)$



Neural Networks

- Why do we need a new algorithm?
  - Some problems are just too complex
  - Many features play a role in deciding boundaries among classes => Increased feature space
  - Even difficult for human





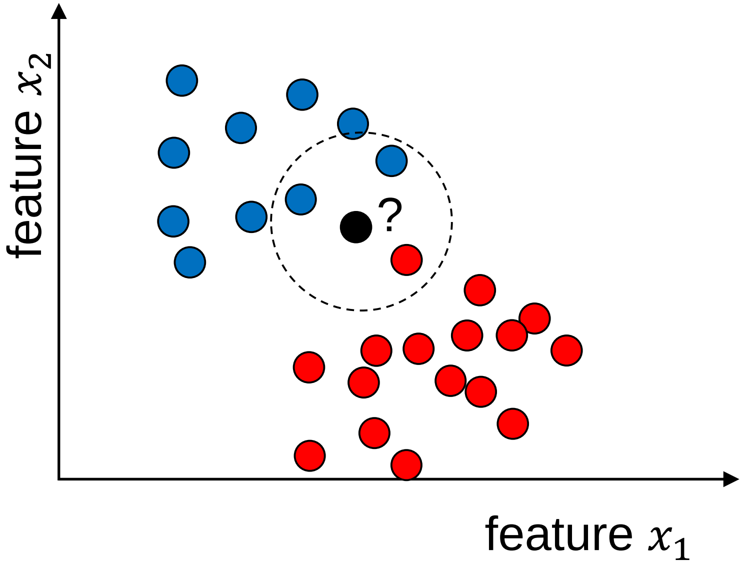
Backpropagation algorithm uses **Gradient descent** optimization to minimize error:

$$W^{i+1} \leftarrow W^i - \epsilon \nabla e_W^i$$

Learning rate

K-Nearest Neighbors (KNN)

- Used for classification and regression
- Decision based on the K nearest points in the training sets
  - Need to choose K
- Example 1: classification (K=3)
  - Choose the most frequent class among the KNN -> predict class 1
  - Changing the value of K (e.g. K=5) may affect the result



Performance Metrics of ML Algorithms

- Regression: minimum prediction error
- Classification:
  - Accuracy: Fraction of test instances correctly classified

Predicted class

True class	Normal (-)	Anomaly (+)
Normal	a	b
Anomaly	c	d

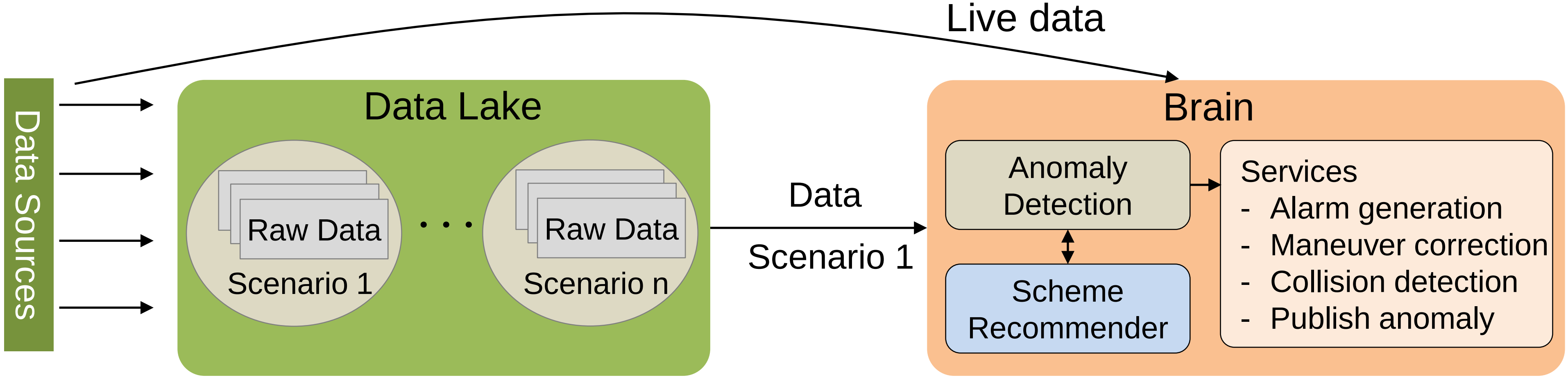
Total samples = a+b+c+d

True negative (**TN**) = a
True positive (**TP**) = d
False negative (**FN**) = c
False positive (**FP**) = b
Objective: minimum FP, FN

- Precision = **TP/(TP+FP):** What proportion of positive identifications was actually correct?
- Recall = **TP/(TP+FN)** : What proportion of actual positives was identified correctly?
- F1-score = **2PR/(P+R)**

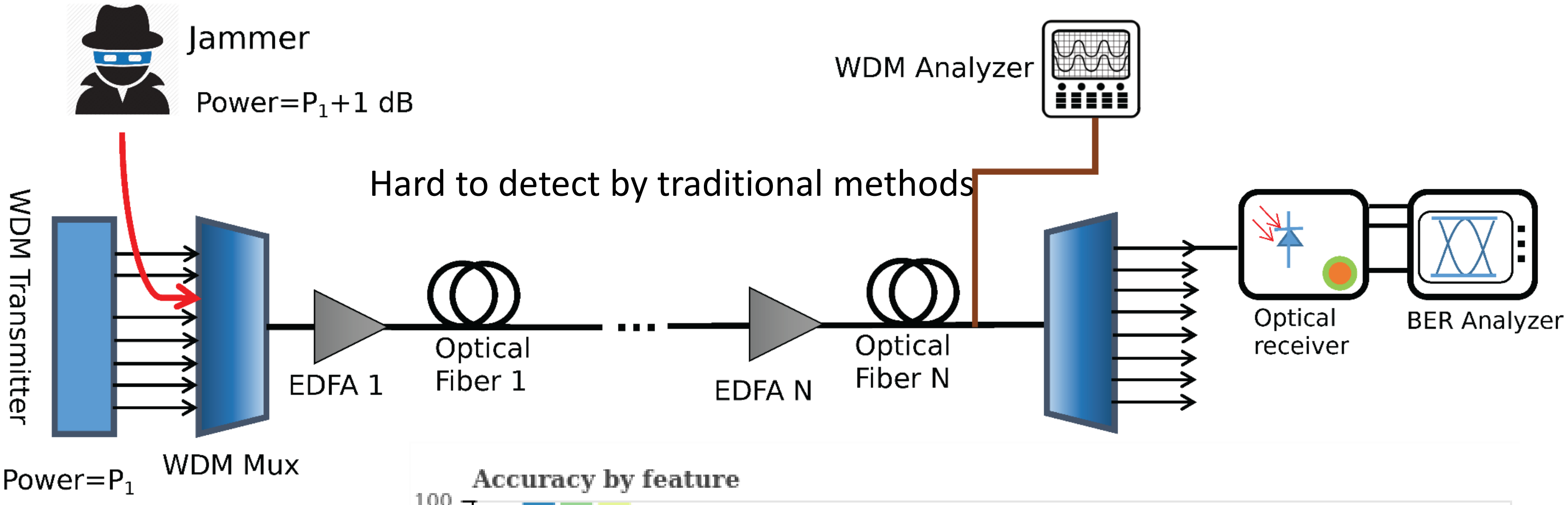
How to Detect Anomalies?

- Manual
  -- Cumbersome, time-consuming, error-prone
- Statistical methods
  -- Limited by data distribution assumptions
- Machine learning
  -- High computation, benefit from big data, ability to learn complex functions

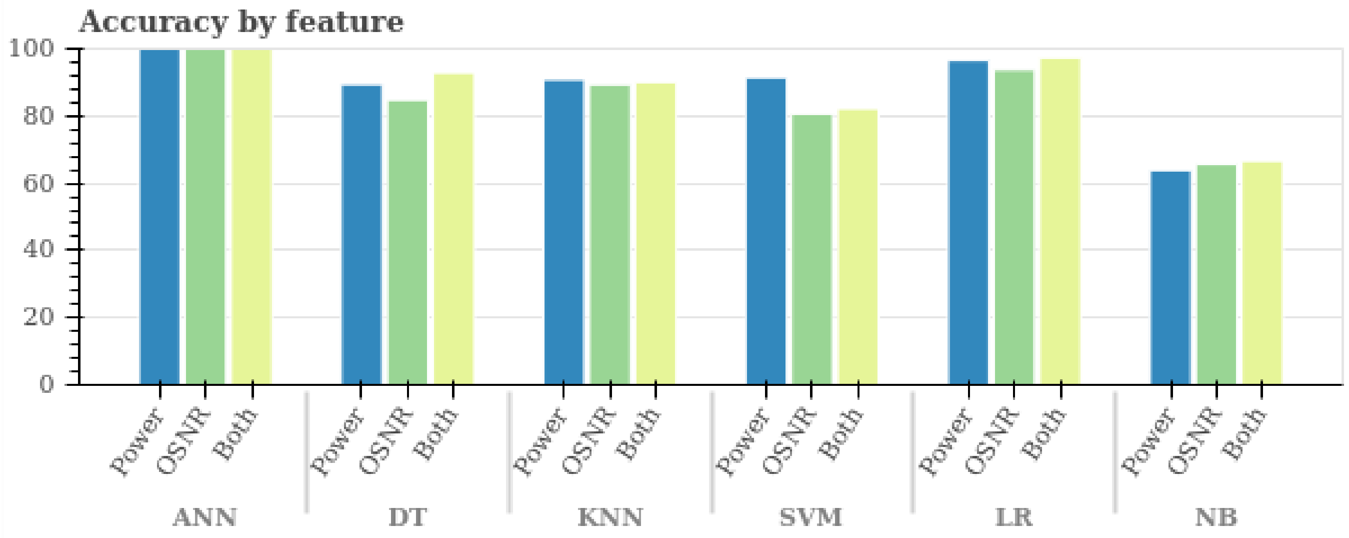


Anomaly detection framework

An Anomaly Detection Example



500 data samples
50-50% (Un)jammed



* M. Bensalem, S. K. Singh, and A. Jukan, "Machine Learning Techniques to Detecting and Preventing Jamming Attacks in Optical Networks," under submission in IEEE Globecom 2019.

How to Predict Time-Series Data (e.g. Traffic, Bandwidth Requirement)?

- Problem Statement:

$$\text{Input sequence } \mathbf{X}^v_{t_i} = (x^v_{t_{i-l+1}}, \ldots, x^v_{t_{i-1}}, x^v_{t_i})$$

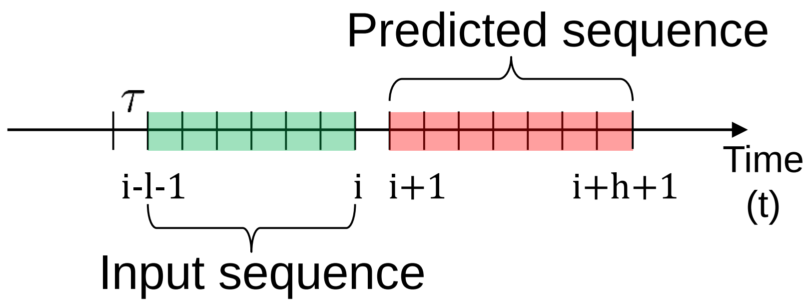
$$\text{Predicted sequence } \tilde{\mathbf{Y}}^v_{t_i} = (\tilde{x}^v_{t_{i+1}}, \tilde{x}^v_{t_{i+2}}, \ldots, \tilde{x}^v_{t_{i+h+1}})$$

$$\text{Feature vector } x^v_{t_i} \equiv (lat, lon, COG, SOG)^v_{t_i}$$

$$l : \text{input sequence length}$$

$$h : \text{output sequence length}$$

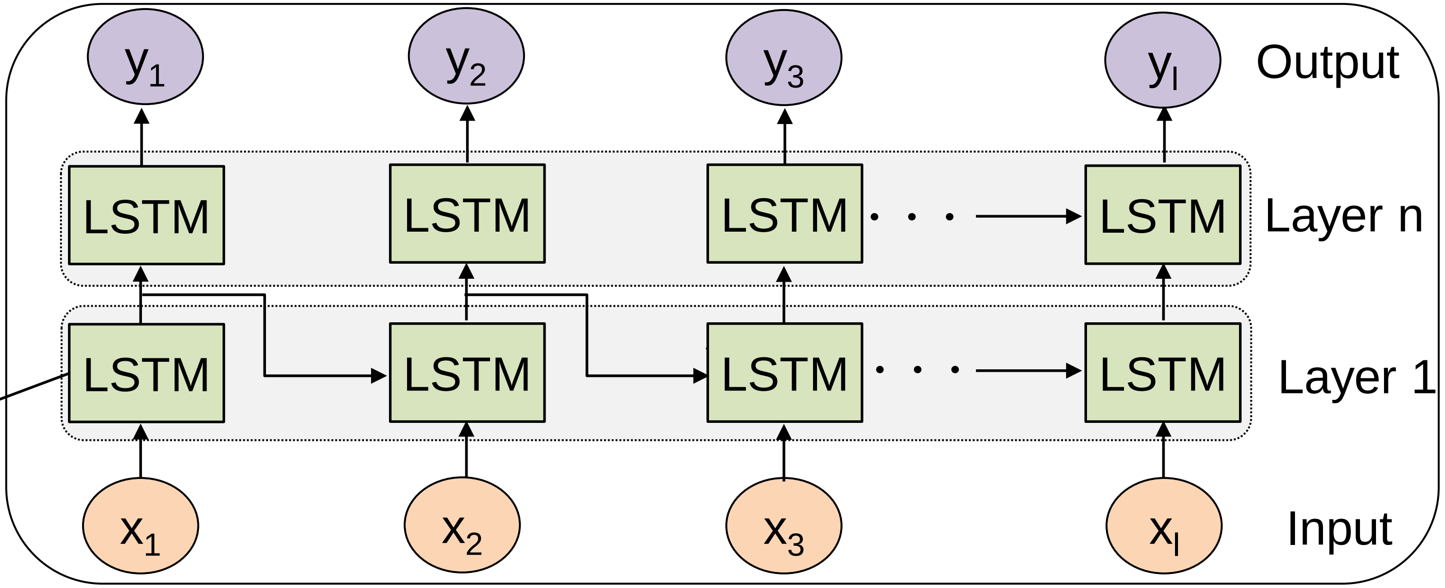
$$\tilde{\mathbf{Y}}^v_{t_i} = \phi_{l,h}(\mathbf{X}^v_{t_i})$$



- Statistical method: most popular Kalman Filter (KF)
- Machine learning: Long Short-Term Memory (LSTM, 1997), Sequence-to-Sequence (Seq2Seq, 2016), etc.

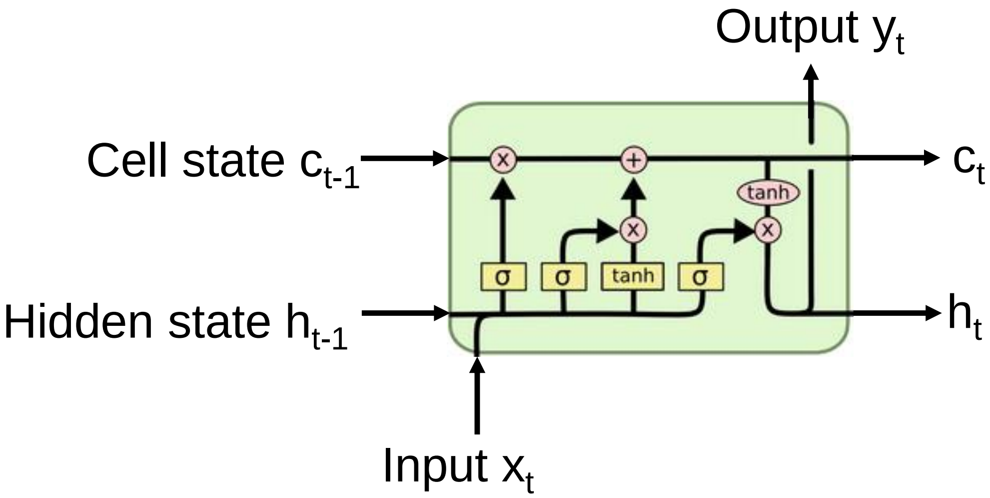
LSTM Network

- LSTM Network:
  Input seq. len (l) = output seq. len (h)



LSTM Network

LSTM cell or unit

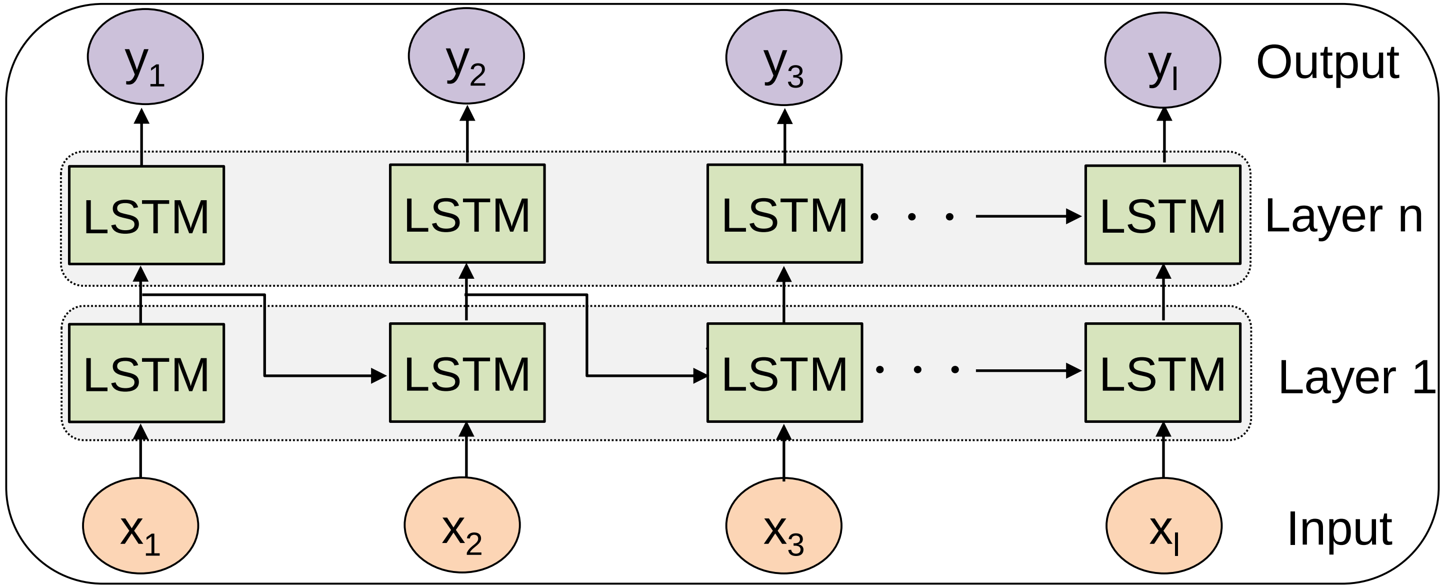


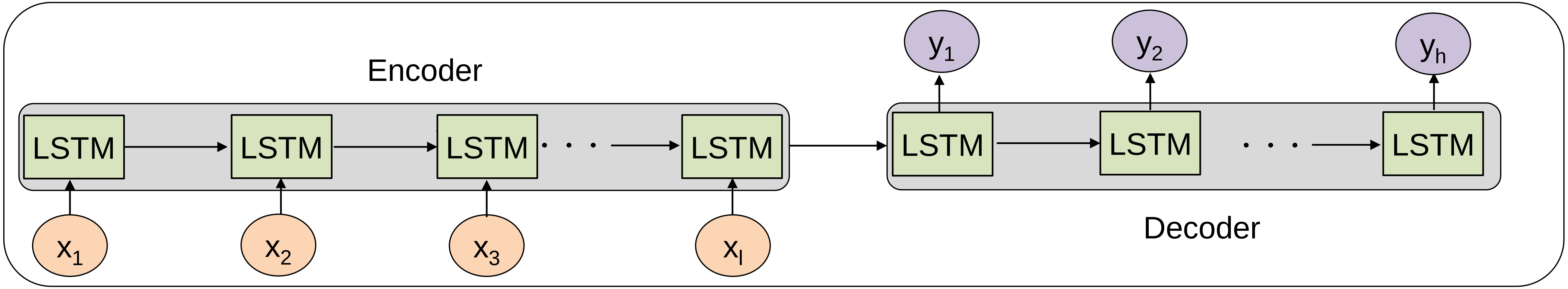
LSTM Network and Seq2Seq

- LSTM Network:
  Input seq. len (l) = output seq. len (h)

- Seq2Seq:
  LSTM encoder (l) + LSTM decoder (h)

  Many to one — One to many





Questions to be addressed

- Which ML algorithm best describes our problem?
- Which data/features should we consider to make predictions?
- Is it worth collecting as much data as possible?
- Is there any irrelevant parameter we can (or should) neglect?
- What is the performance of our learning algorithm?
- And what is its complexity?
- Do we want ML as a black-box?
- Hybrid learning (statistical+ML)?
- …

Questions?

Thanks for your attention!

Long Short-Term Memory (LSTM)

- Mapping from input to output

x_t: input vector at time step t

$h_t = f_W(h_{t-1}, x_t)$, e.g., $tanh(W_{hh}h_{t-1} + W_{xh}x_t)$

h_t: new hidden state

$f_W(\cdot)$: some function with parameter (Weight) W

y_t : Output $= W_{hy}h_t$

- LSTM solves vanishing gradient problem in (Vanilla) RNN