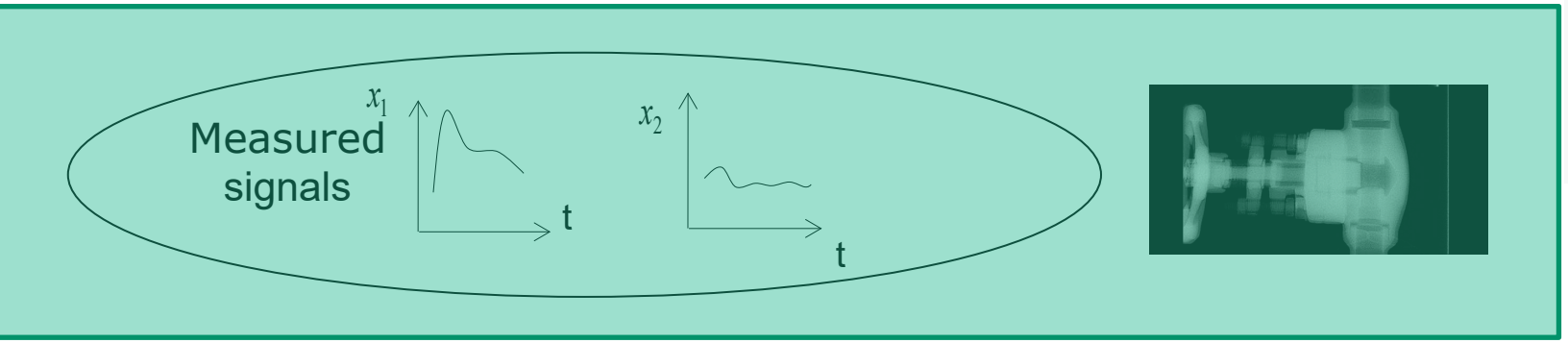
Advanced Topics in Fault Detection and Diagnostics

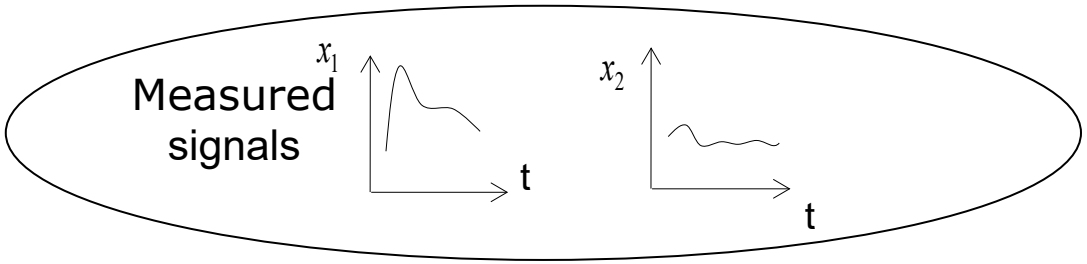
Prof. Piero Baraldi,

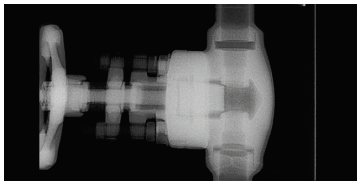
Equipment (System, Structure or Component)

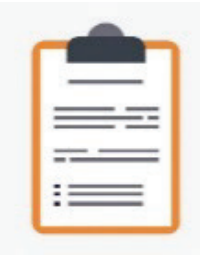


In This Lecture:
Advanced Topics in Fault Detection and Diagnostics

Equipment (System, Structure or Component)







$$\frac{dx}{dt} = A\exp\left(-\frac{Q}{RT}\right)\varphi^n$$

Detect

Anomalous operation

Normal operation

Diagnose

c_1 c_2 c_3

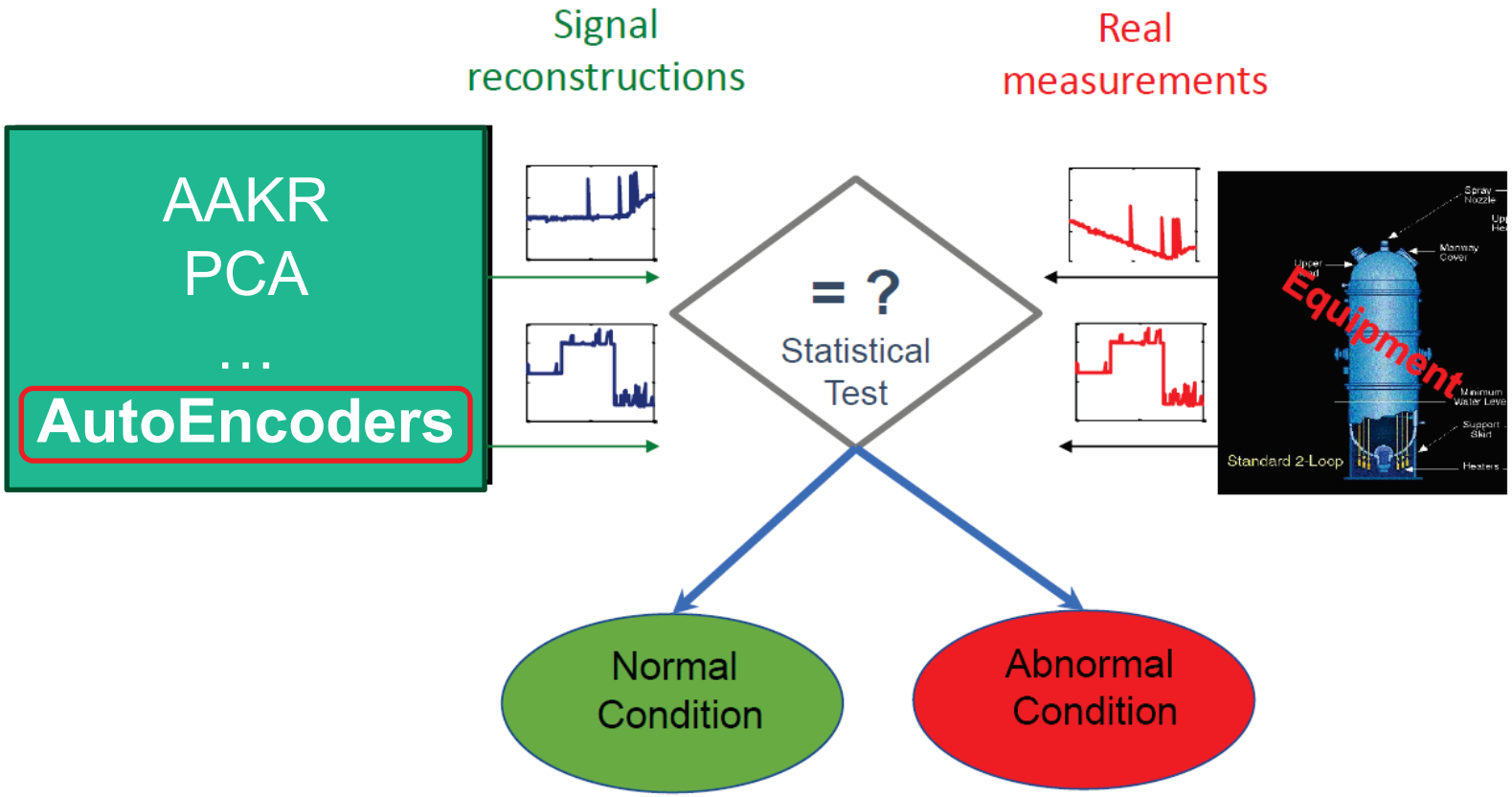
Malfunctioning type (class)

Predict

Remaining Useful Life (RUL)

PART 1: FAULT DETECTION

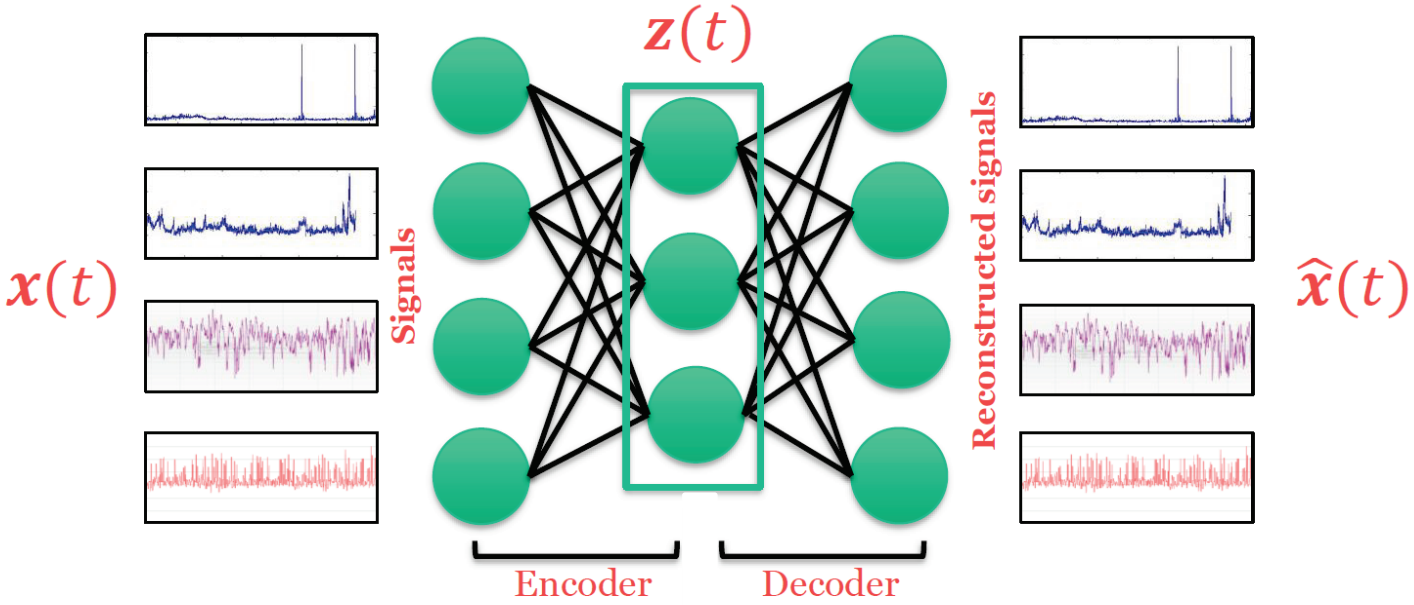
Fault Detection Methods



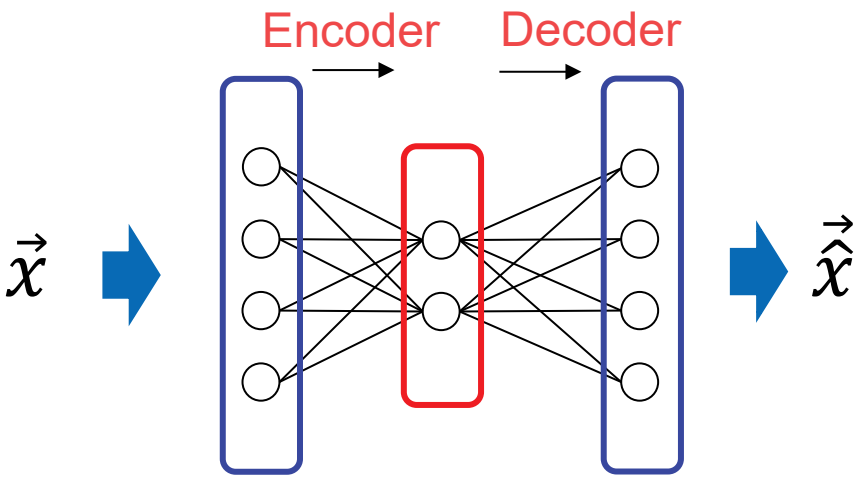
AUTOENCODERS

Autoencoder: What is it?

Autoencoder (AE):



Autoencoder Training (1)



- **Training Set**

$\boldsymbol{x}^{(n_p)}$	…	$\boldsymbol{x}^{(2)}$	$\boldsymbol{x}^{(1)}$
$x_1^{(n_p)}$		$x_1^{(2)}$	$x_1^{(1)}$
$x_2^{(n_p)}$	…	$x_2^{(2)}$	$x_2^{(1)}$
$x_3^{(n_p)}$		$x_3^{(2)}$	$x_3^{(1)}$
$x_4^{(n_p)}$		$x_4^{(2)}$	$x_4^{(1)}$

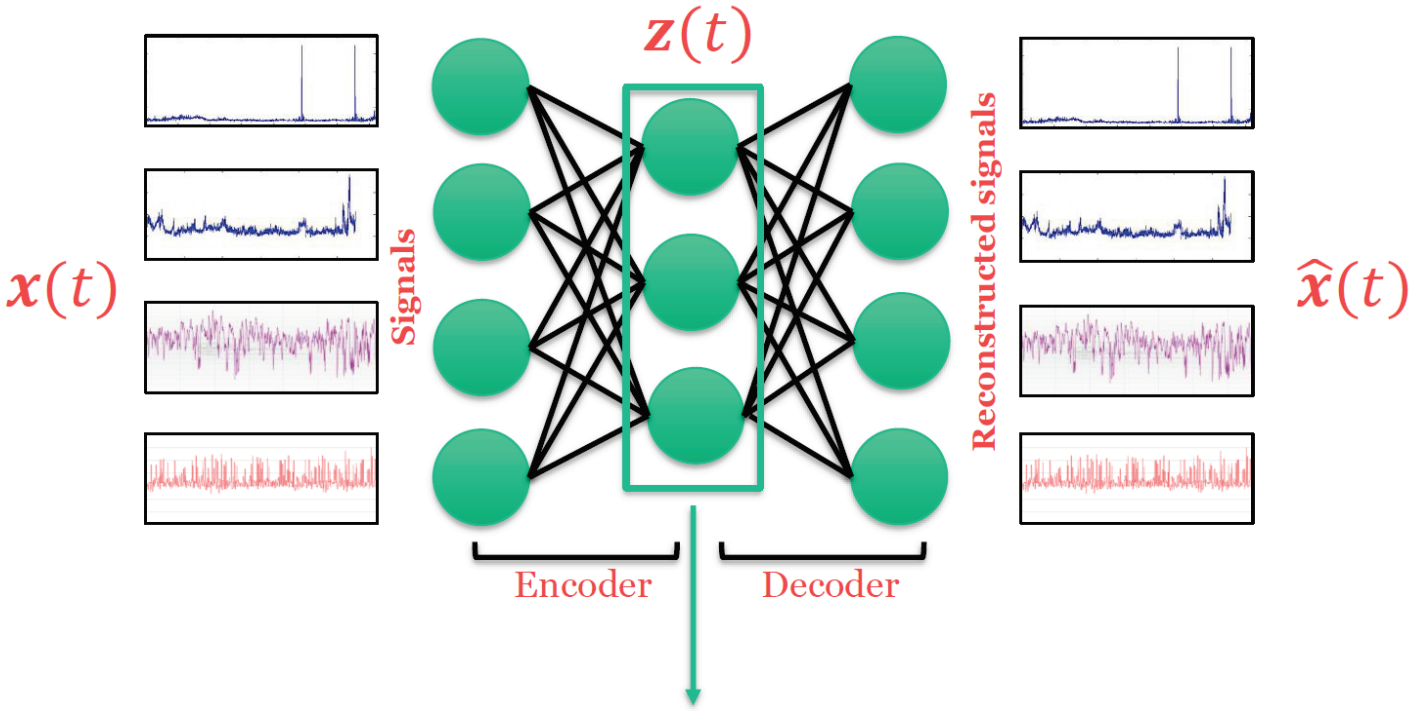
- **Cost function**

$$E = \frac{1}{n_p}\sum_{p=1}^{n_p}\left\|\boldsymbol{x}^{(p)} - \hat{\boldsymbol{x}}^{(p)}\right\|^2$$

Reconstruction Error

Autoencoder: What is it?

Autoencoder (AE):



Latent representation of $\boldsymbol{x}(t)$

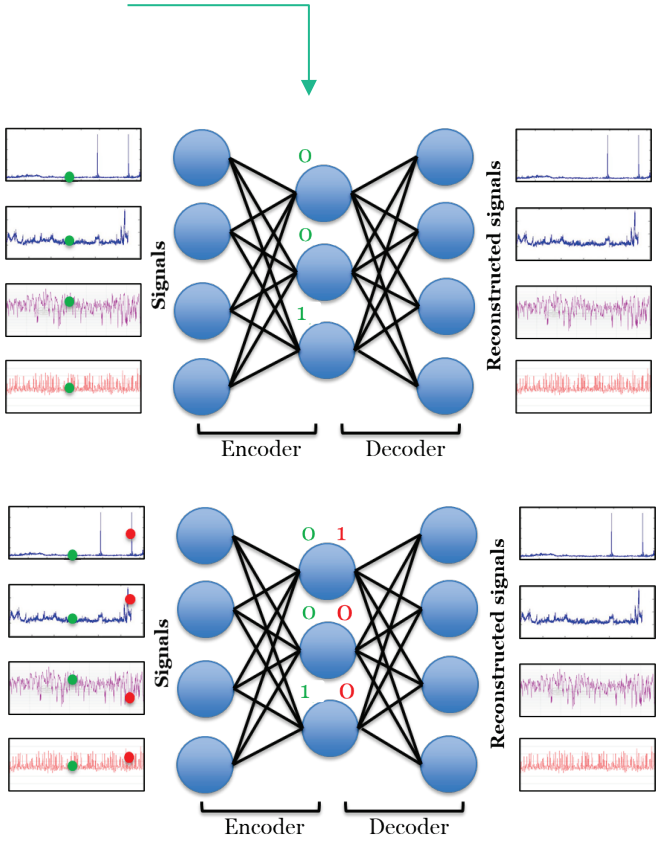
Sparse Autoencoders

Other desiderata:

- Sparsity (forcing the output of the neurons of the hidden layer to be zero most of the time)
  - Leads to better feature extraction

Focus on most relevant aspects of the data

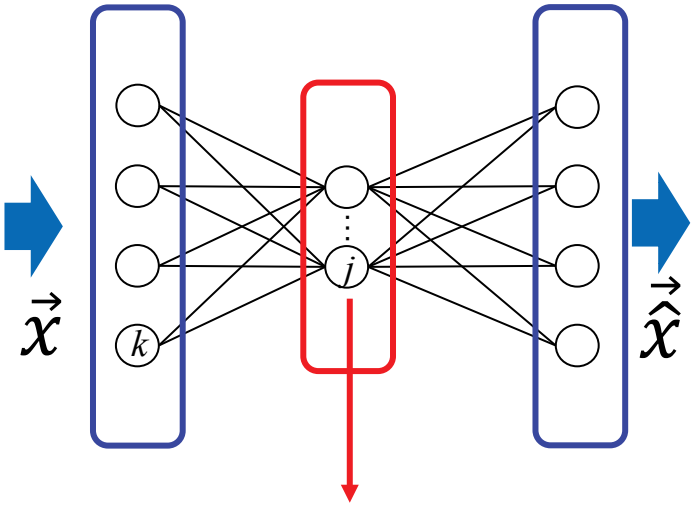
Disentangled representation



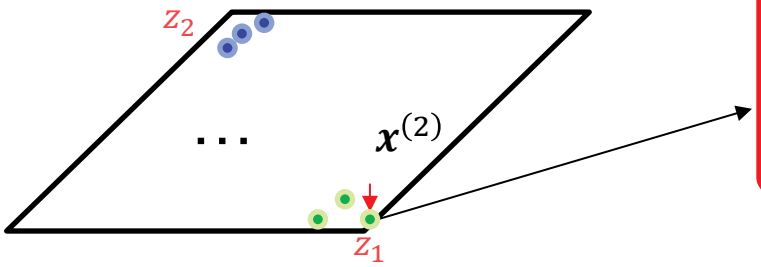
Sparsity: How to obtain it? (1)

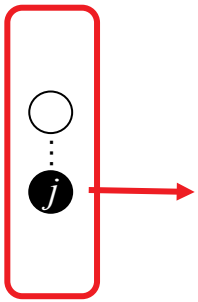
- Let us consider hidden neuron *j*

$\boldsymbol{x}^{(n_p)}$	...	$\boldsymbol{x}^{(2)}$	$\boldsymbol{x}^{(1)}$
$x_1^{(n_p)}$		$x_1^{(2)}$	$x_1^{(1)}$
$x_2^{(n_p)}$	...	$x_2^{(2)}$	$x_2^{(1)}$
$x_3^{(n_p)}$		$x_3^{(2)}$	$x_3^{(1)}$
$x_4^{(n_p)}$		$x_4^{(2)}$	$x_4^{(1)}$



Input samples
(many different patterns)



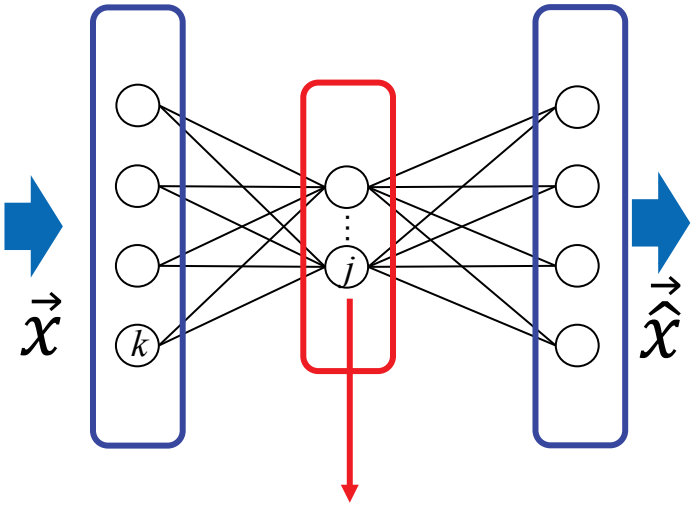


$f^h(\boldsymbol{x}^{(1)})=0, f^h(\boldsymbol{x}^{(2)})=1, \ \ldots \ f^h(\boldsymbol{x}^{(n_p)})=0$

Sparsity: How to obtain it? (1)

- Let us consider hidden neuron *j*

$\boldsymbol{x}^{(n_p)}$	...	$\boldsymbol{x}^{(2)}$	$\boldsymbol{x}^{(1)}$
$x_1^{(n_p)}$		$x_1^{(2)}$	$x_1^{(1)}$
$x_2^{(n_p)}$	...	$x_2^{(2)}$	$x_2^{(1)}$
$x_3^{(n_p)}$		$x_3^{(2)}$	$x_3^{(1)}$
$x_4^{(n_p)}$		$x_4^{(2)}$	$x_4^{(1)}$

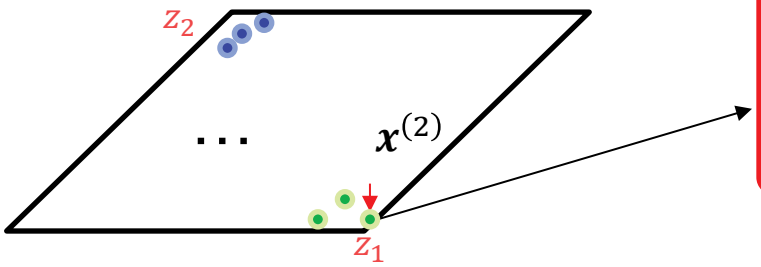


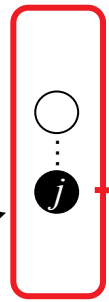
Average output of hidden neuron *j* over the training set

$$\hat{\rho}_j = \frac{1}{n_p}\sum_{p=1}^{n_p} f^h\left(\sum_{k=1}^{4} x_k^{(p)} w_{jk} + w_{j0}\right)$$

Output of hidden neuron *j* given input $\boldsymbol{x}^{(p)}$

Input samples
(many different patterns)





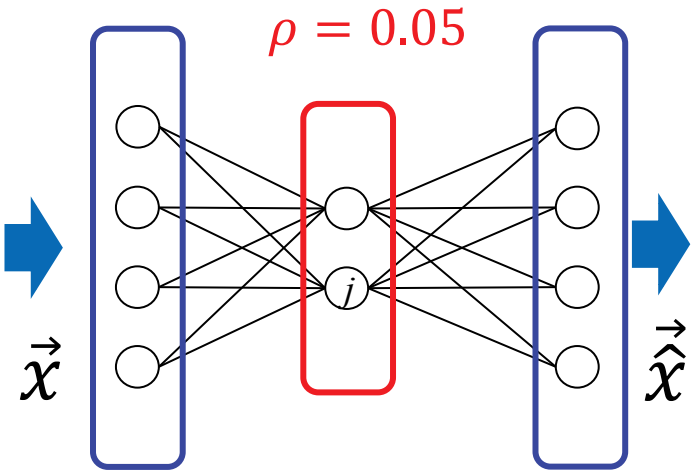
$$f^h(\boldsymbol{x}^{(1)})=0, f^h(\boldsymbol{x}^{(2)})=1, \ \cdots \ f^h\left(\boldsymbol{x}^{(n_p)}\right)=0$$

$$\hat{\rho}_j = \frac{n}{n_p} \qquad n = 3, \text{the number of } \bullet$$

Small number when $n \ll n_p$

Sparsity: How to obtain it? (2)

- The desired output of the hidden neuron has the distribution of a Bernulli random variable with expected value ρ.



$$R_{sparse} = \sum_{j}^{n_h} \mathrm{KL}\left(\rho \,\middle\|\, \hat{\rho}_j\right) = \sum_{j}^{n_h} \left[\rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho) \log\left(\frac{1-\rho}{1-\hat{\rho}_j}\right)\right]$$

Kullback–Leibler (KL) divergence:
It measure the distance between the Bernulli distribution with mean ρ and the Bernulli distribution with mean $\hat{\rho}_j$

Training objective for sparsity

$$\hat{\rho}_j = \frac{1}{n_p} \sum_{p=1}^{n_p} f_j\left(\boldsymbol{x}^{(p)}\right) \longrightarrow \rho$$

Minimize $\mathrm{KL}\left(\rho \,\middle\|\, \hat{\rho}_j\right)$

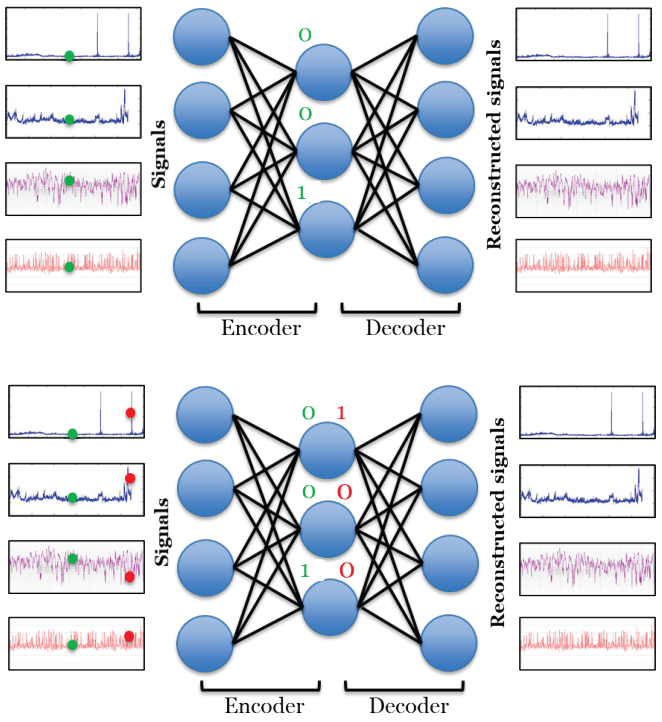
Sparce Autoencoders

Other desiderata:

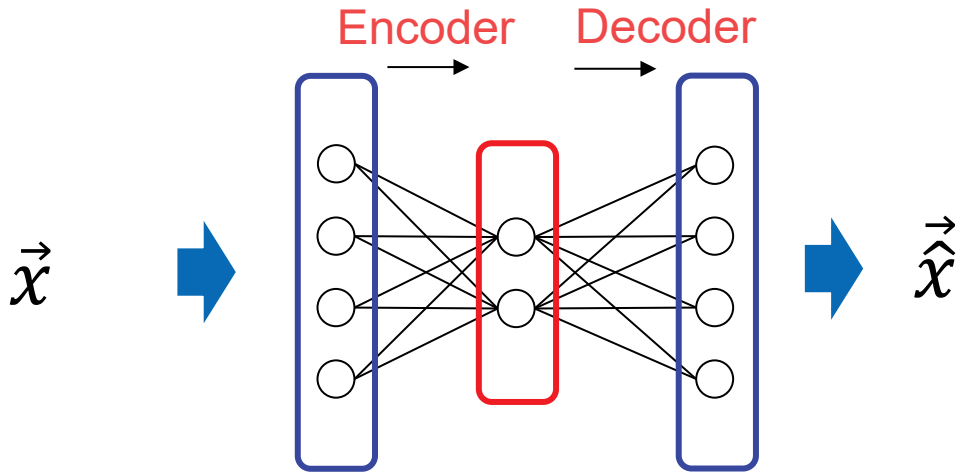
- Sparsity (forcing the output of the neurons to be zero most of the time)
  - Leads to better feature extraction
- Prevents overfitting

Reduce model complexity

Constrain the value of inner weights to be small



Sparse Autoencoder: Training



$$\boldsymbol{x}^{(p)}, p = 1, \ldots, n_p$$

- **Cost function**

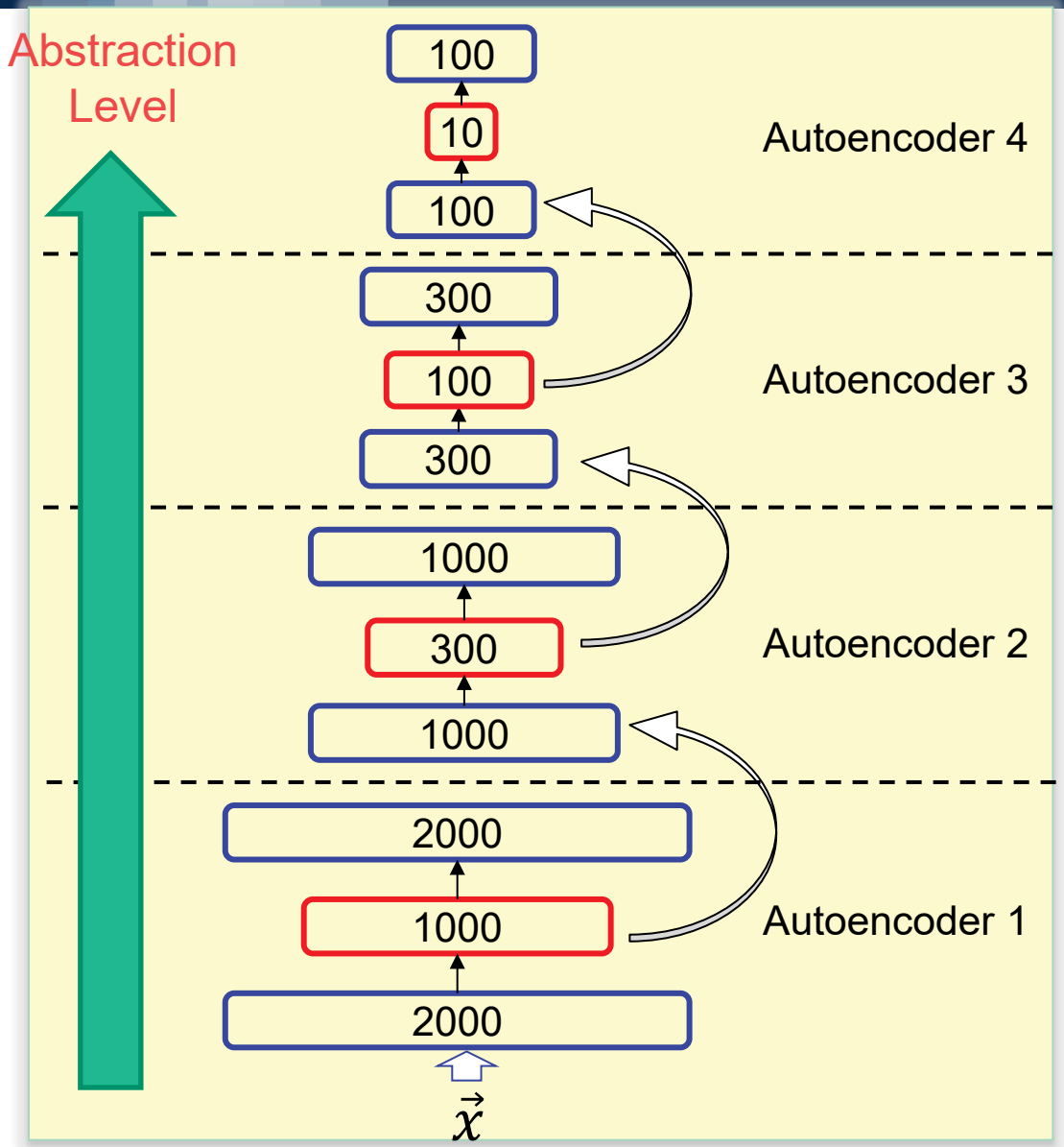
$$E = \frac{1}{n_p}\sum_{p=1}^{n_p}\left\|\boldsymbol{x}^{(p)} - \hat{\boldsymbol{x}}^{(p)}\right\|^2 + \beta * R_{sparse} + \lambda * \frac{1}{2}\|\mathbf{W}\|^2$$

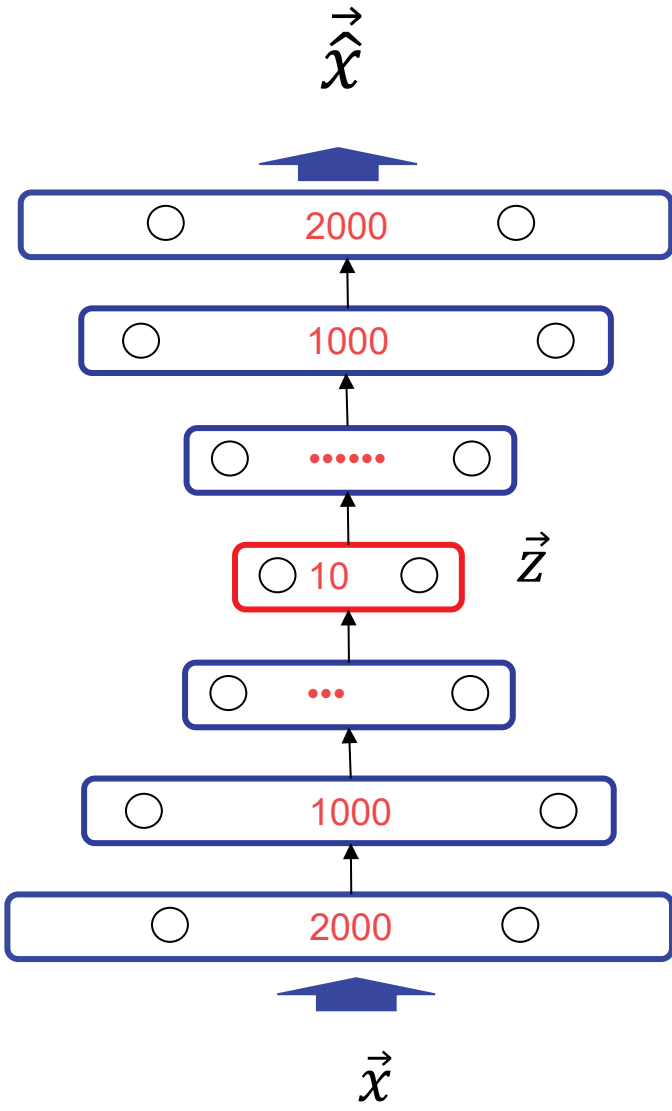
Reconstruction Error

Sparse decomposition of the input

Constrain the value of inner weights

Stacking Autoencoders for Data Dimensionality Reduction

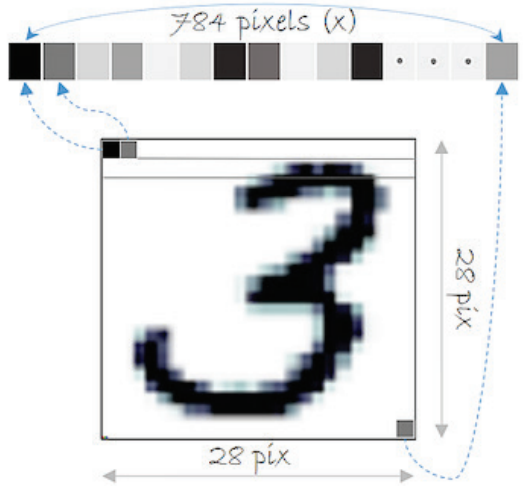




Example: MNIST database of handwritten digits [1]

MNIST database of handwritten digits [1]

- Inputs: 28x28 images (784 pixels)
- Training set: n_p =60,000 samples

$\vec{x} =$ 

Pixel values

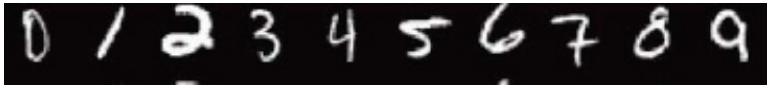
$\in [0, 255]$

Normalized

[1] http://yann.lecun.com/exdb/mnist/
[2] Simard P Y, Steinkraus D, Platt J C. Best practices for convolutional neural networks applied to visual document analysis[C]//null. IEEE, 2003: 958.
[3] Ciresan et al. Neural Computation 10, 2010 and arXiv 1003.0358, 2010

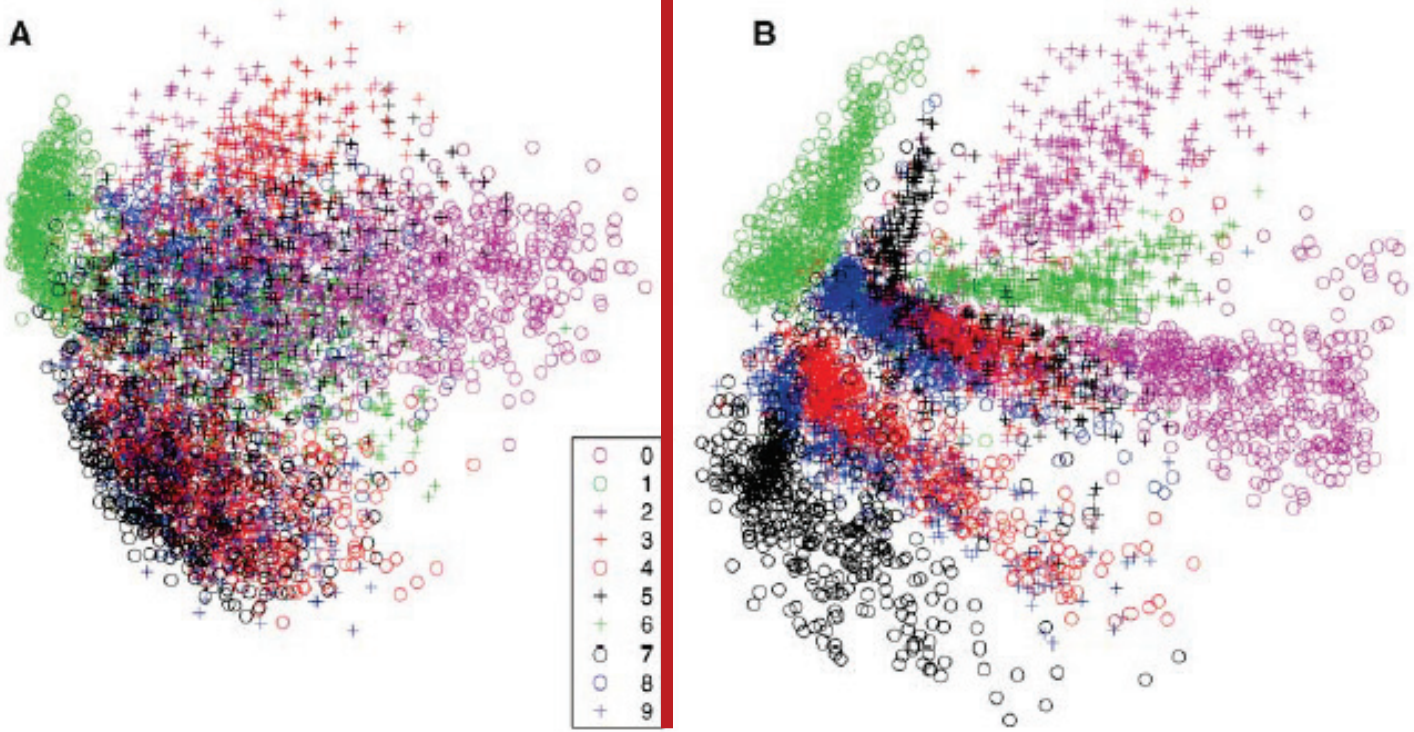
Example

- Case study: MNIST DATASET (60000 images)
- Dimensionality reduction:



PCA

Stacked Autoencoder (787;1000;500;250;30;2)



Reducing the Dimensionality of Data with Neural Networks G. E. Hinton* and R. R. Salakhutdinov, Science, Vol 313, 2006

Case study

AE for anomaly detection in rotating machines

Bearings

- Fundamental in many industries:
  - Assist in rotation
  - Reduce friction
  - Maintain correct position

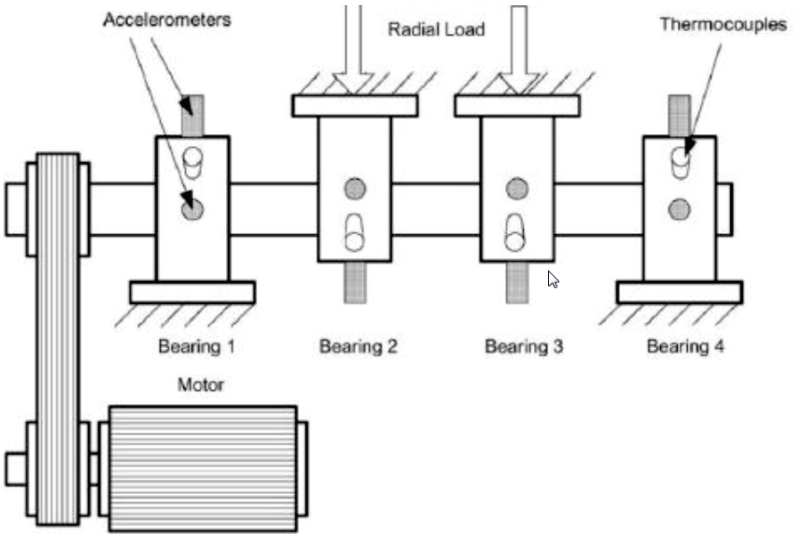
- Most critical components of rotating machinery:
  - Frequent failures
  - Several different failure causes
  - Failure consequences: large downtime, damage to adjacent components and systems, and significant repair costs

Automatic detection of bearing abnormal conditions

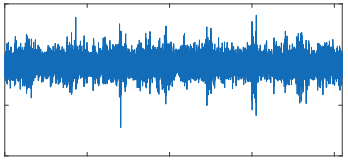
Case study

[1]

4 bearings and 1 shaft



- Available information: vibration signals collected during run-to-failure trajectories



B1

B2

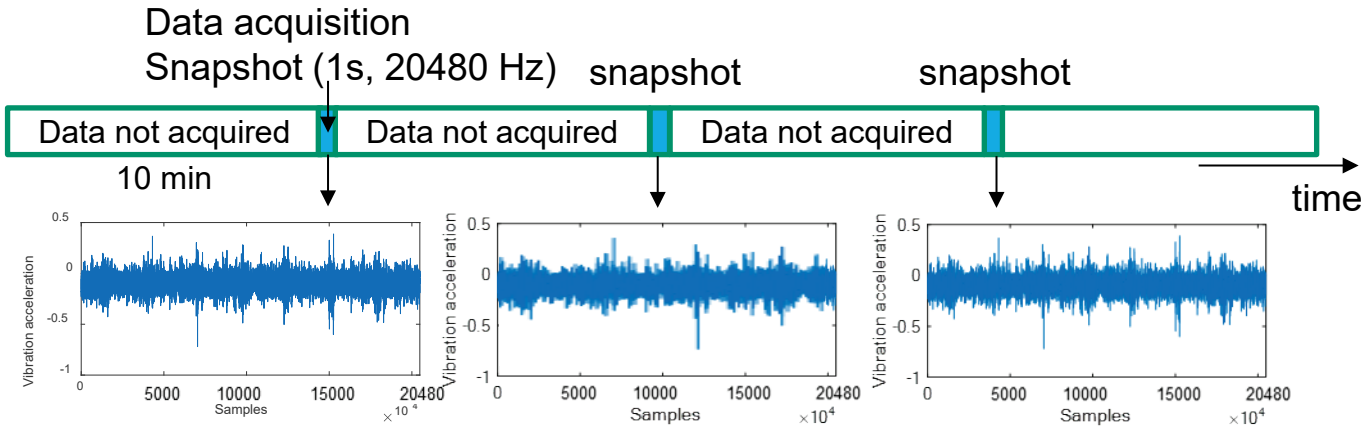
B3

B4

<u>Objective:</u>
Identify the onset of abnormal conditions

[1] Benchmark available at Nasa prognostic repository (https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/). J. Lee, H. Qiu, G. Yu, J. Lin, and Rexnord Technical Services (2007). IMS, University of Cincinnati. "Bearing Data Set", NASA Ames Prognostics Data Repository (http://ti.arc.nasa.gov/project/prognostic-data-repository), NASA Ames Research Center, Moffett Field, CA

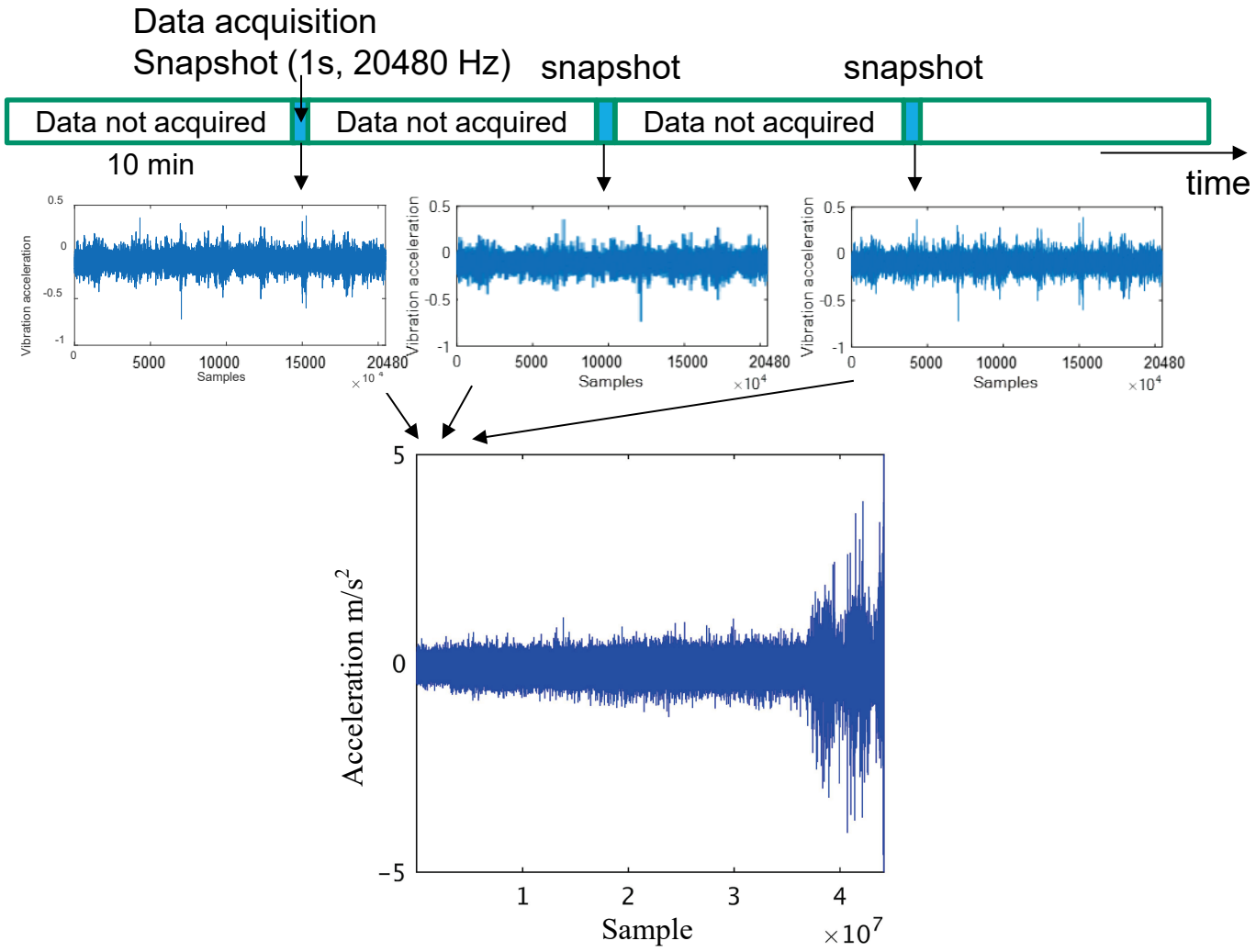
Case study- data collection



Case study- data collection

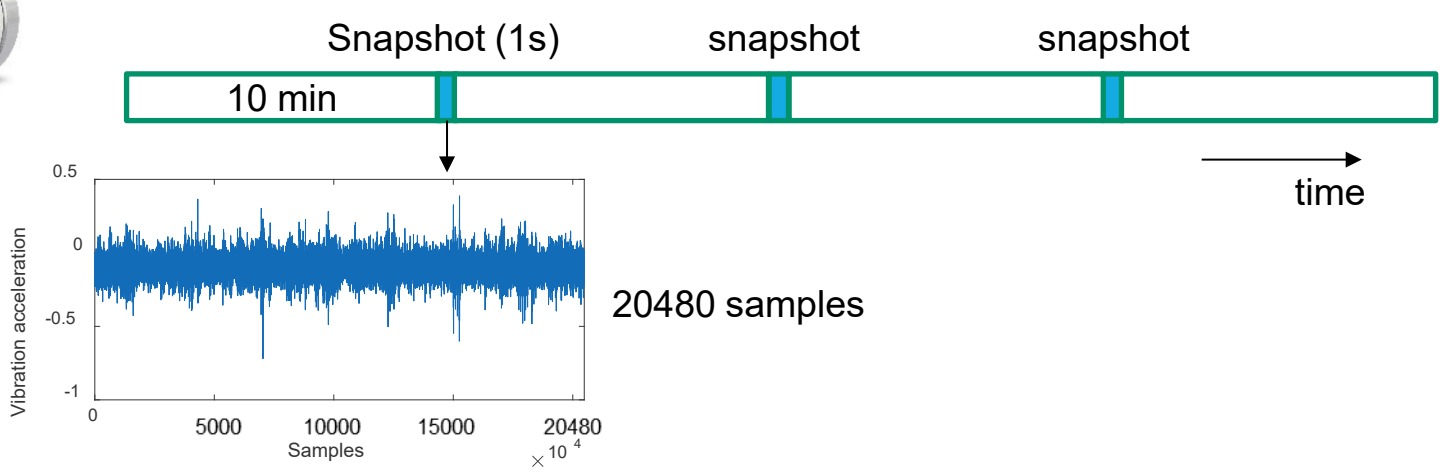
B3

Failed
(Inner race)



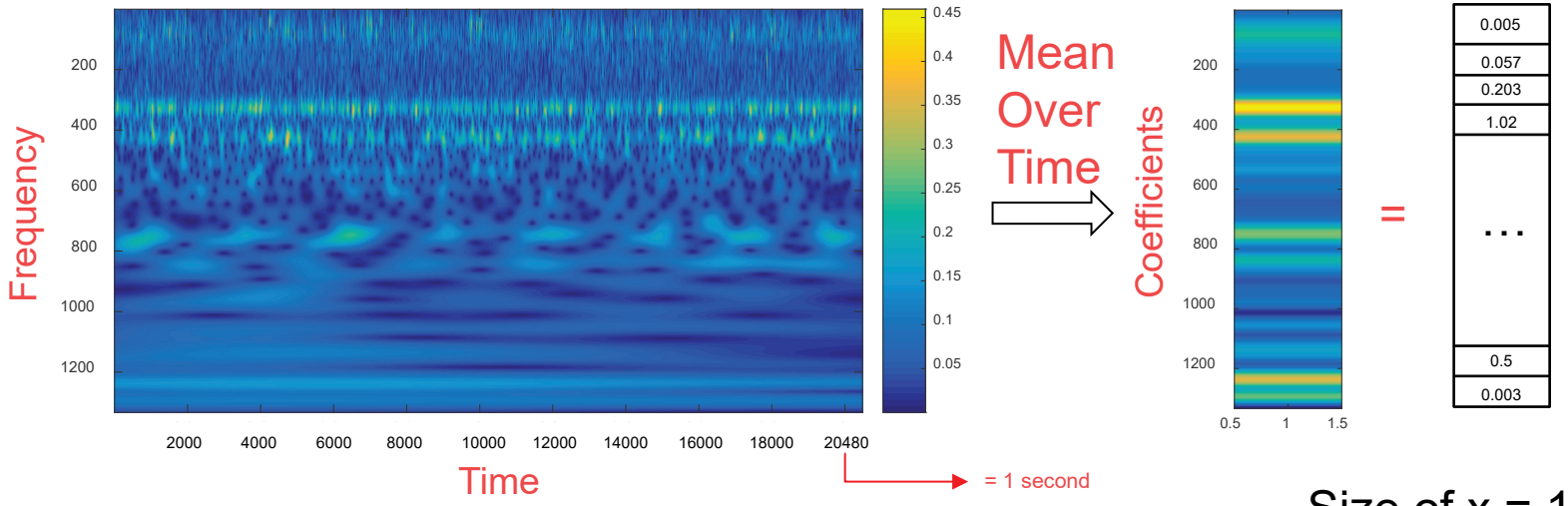
Case study: data preprocessing

B3



20480 samples

Continuous Wavelet Transform



Size of x = 1333

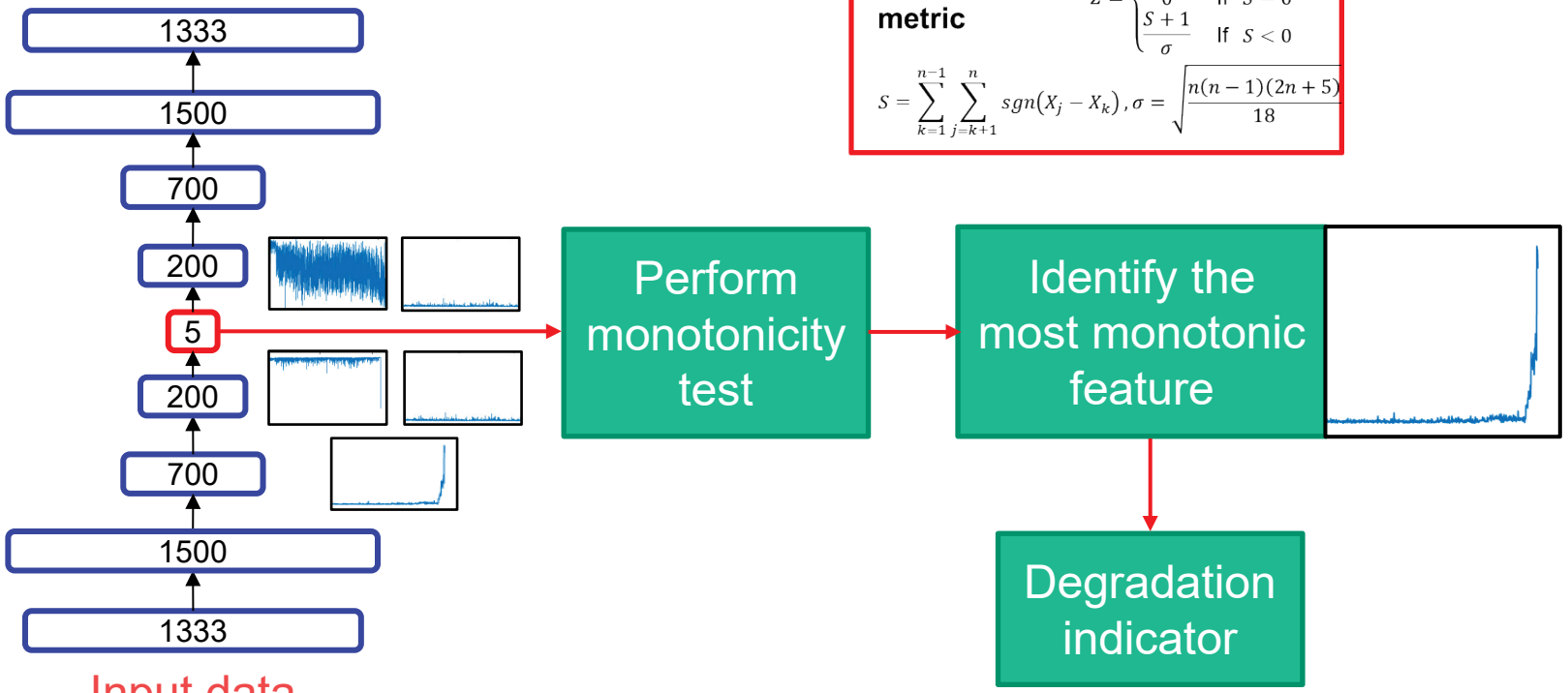
Case study – Part 1: Build a degradation indicator

Build stacked autoencoders

Test of monotonicity: Mann-Kendall metric

$$Z = \begin{cases} \frac{S-1}{\sigma} & \text{If } S > 0 \\ 0 & \text{If } S = 0 \\ \frac{S+1}{\sigma} & \text{If } S < 0 \end{cases}$$

$$S = \sum_{k=1}^{n-1} \sum_{j=k+1}^{n} sgn(X_j - X_k), \sigma = \sqrt{\frac{n(n-1)(2n+5)}{18}}$$



B3

Failed (Inner race)

Basic idea:

- The inner layer of the stacked autoencoders should contain information about degradation
- Degradation is monotonic

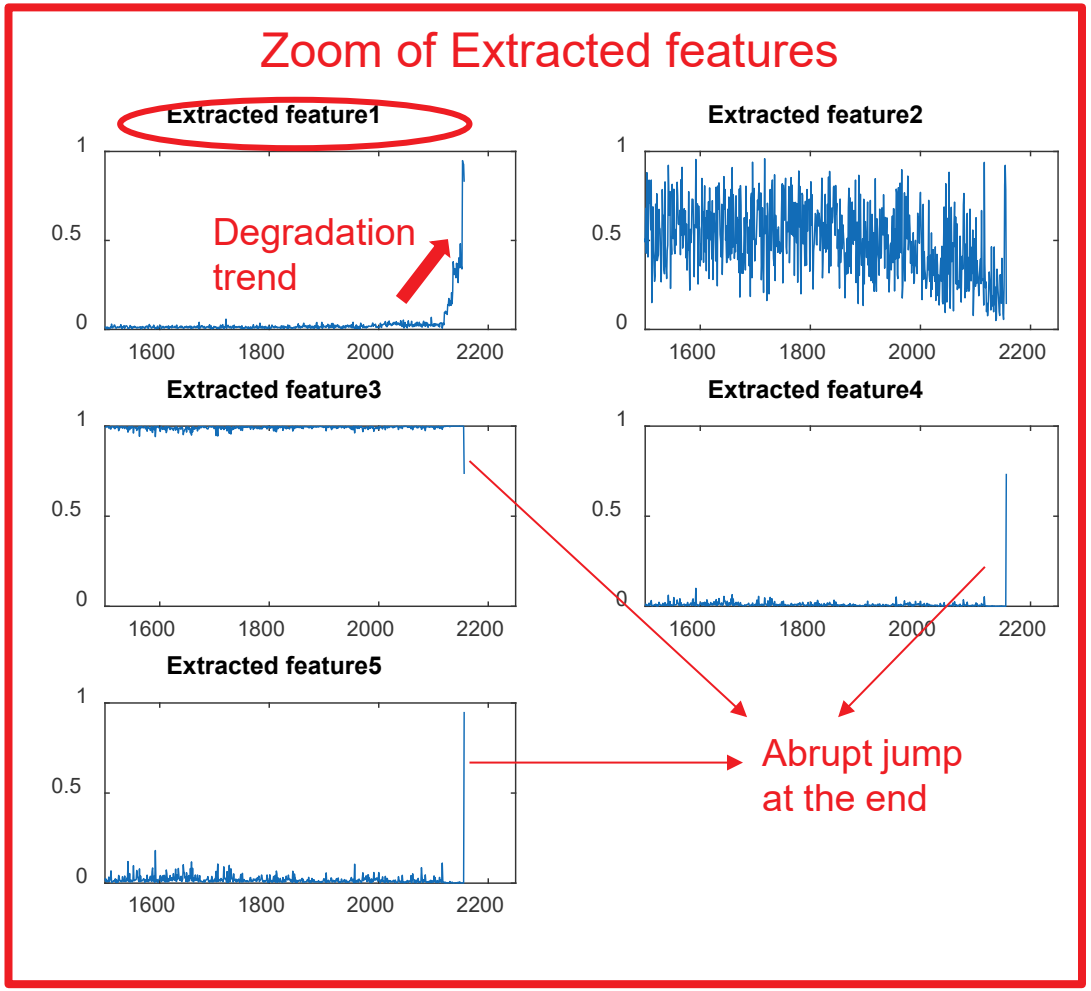
Case study: feature evaluation

Feature	Test of signal monotonicity (Mann-Kendall)
1	***27.6913***
2	-21.7529
3	9.6151
4	-13.6292
5	-11.1801

Test of monotonicity:
Mann-Kendall metric

$$Z = \begin{cases} \frac{S-1}{\sigma} & \text{If } S > 0 \\ 0 & \text{If } S = 0 \\ \frac{S+1}{\sigma} & \text{If } S < 0 \end{cases}$$

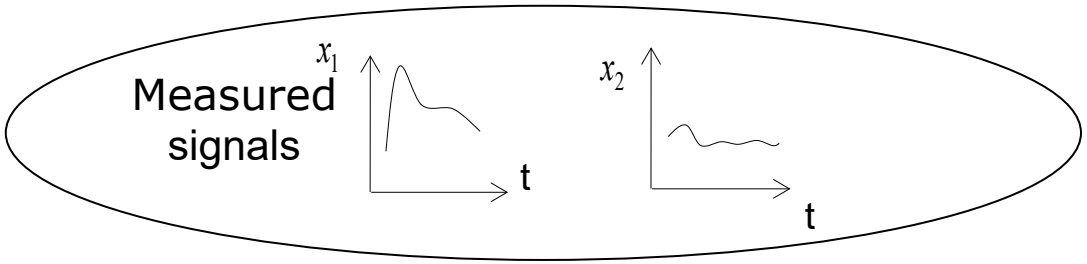
$$S = \sum_{k=1}^{n-1} \sum_{j=k+1}^{n} sgn(X_j - X_k), \sigma = \sqrt{\frac{n(n-1)(2n+5)}{18}}$$

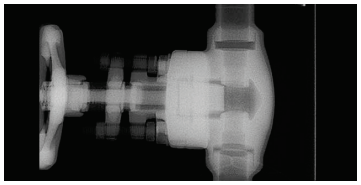


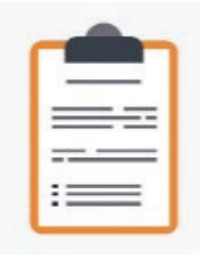
Case Study: … It will continue

PART 1: FAULT DETECTION
PART 2: FAULT DIAGNOSTICS

Equipment (System, Structure or Component)







$$\frac{dx}{dt} = A \exp\left(-\frac{Q}{RT}\right)\varphi^n$$

Detect

Anomalous operation

Normal operation

Diagnose

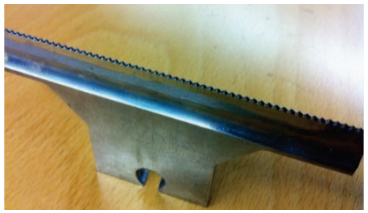
c_1 c_2 c_3

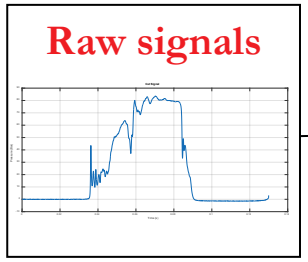
Malfunctioning type (class)

Predict

Remaining Useful Life (RUL)

ANN-based Fault Diagnostics





Feature extraction
(Variance/Maximum /Minimum/Skewness/Kurtosis/Peak/Wavelet…)

f_1 f_2 f_3 ⋮ f_N

Feature selection
(Lasso/ Filter/wrapper method…)

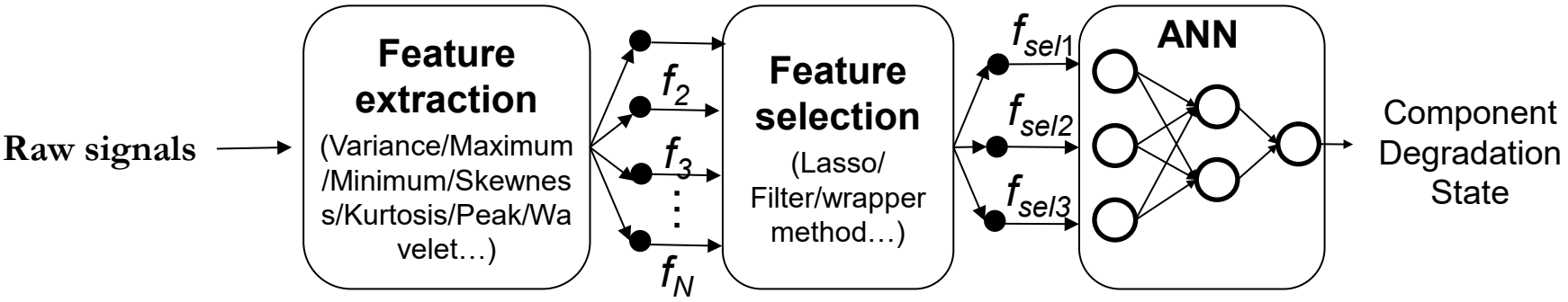
f_{sel1} f_{sel2} f_{sel3}

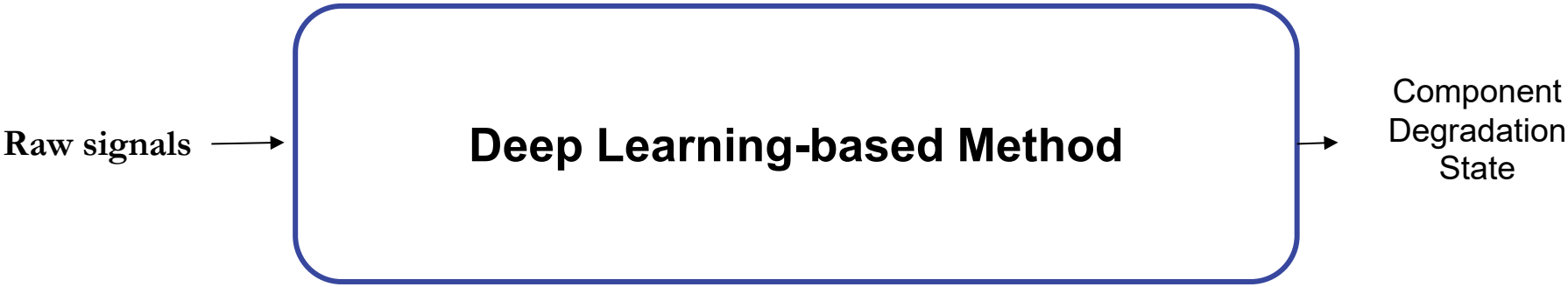
ANN

• **Fault class**

Expert intervention!

Deep Learning-based Fault Diagnostics





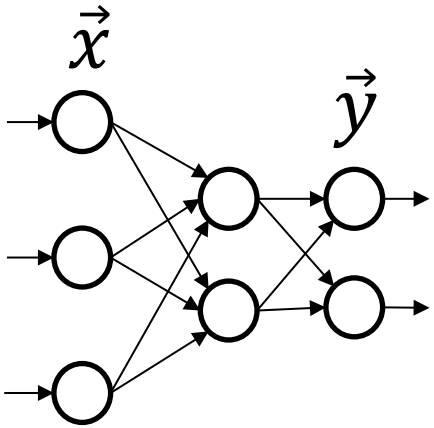
Extract & select useful features automatically!

Part 1: Deep Neural Networks

Part 2: Convolutional Neural Networks

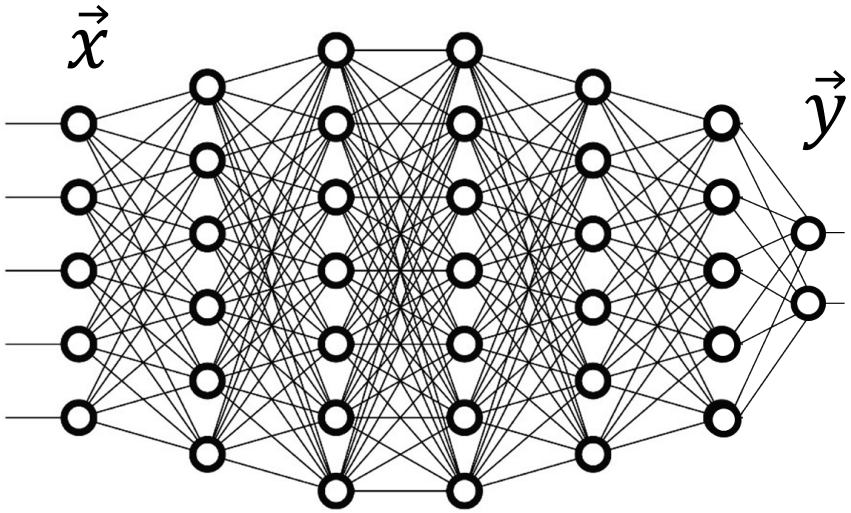
Shallow vs. Deep Neural Network Architecture

Shallow Neural Network



Number of hidden layers: typically <3

Deep Neural Network (DNN)

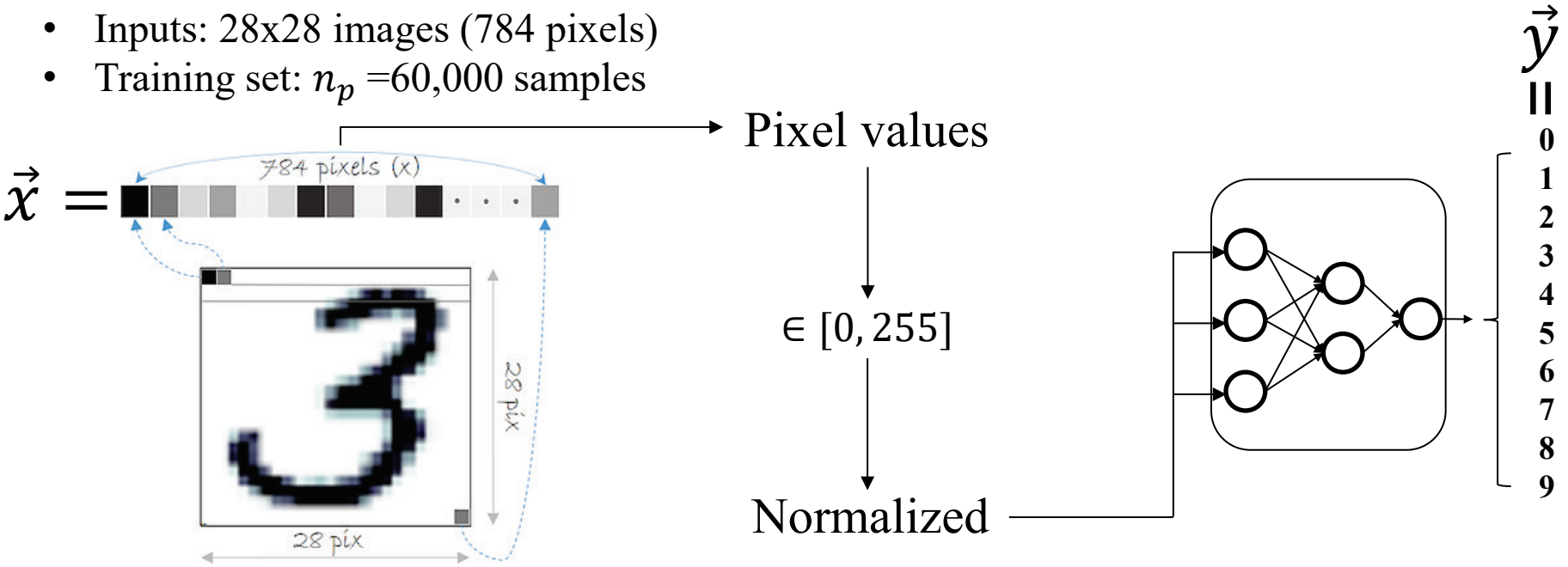


Number of hidden layers: >=3

ANN Learning capability: An upper limit

MNIST database of handwritten digits [1]

- Inputs: 28x28 images (784 pixels)
- Training set: n_p =60,000 samples



Test set: 10,000 samples

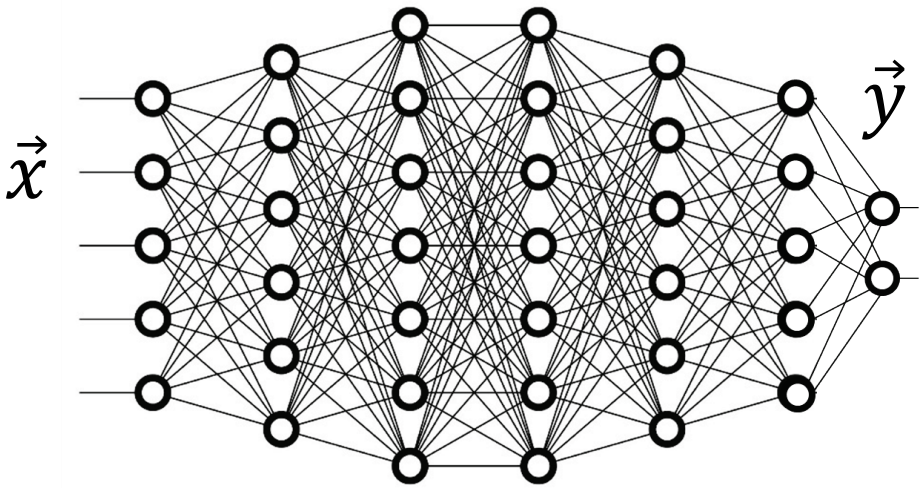
	Accuracy (%)
ANN	99.3
DNN	99.65

[1] http://yann.lecun.com/exdb/mnist/
[2] Simard P Y, Steinkraus D, Platt J C. Best practices for convolutional neural networks applied to visual document analysis[C]//null. IEEE, 2003: 958.
[3] Ciresan et al. Neural Computation 10, 2010 and arXiv 1003.0358, 2010

Training DNNs: the Objective

Training set:
(input-output examples)

$$\begin{bmatrix} \overrightarrow{x^1} & \overrightarrow{y^1} \\ \overrightarrow{x^2} & \overrightarrow{y^2} \\ \dots & \dots \\ \overrightarrow{x^{n_p}} & \overrightarrow{y^{n_p}} \end{bmatrix}$$



Training DNNs: the Challenge

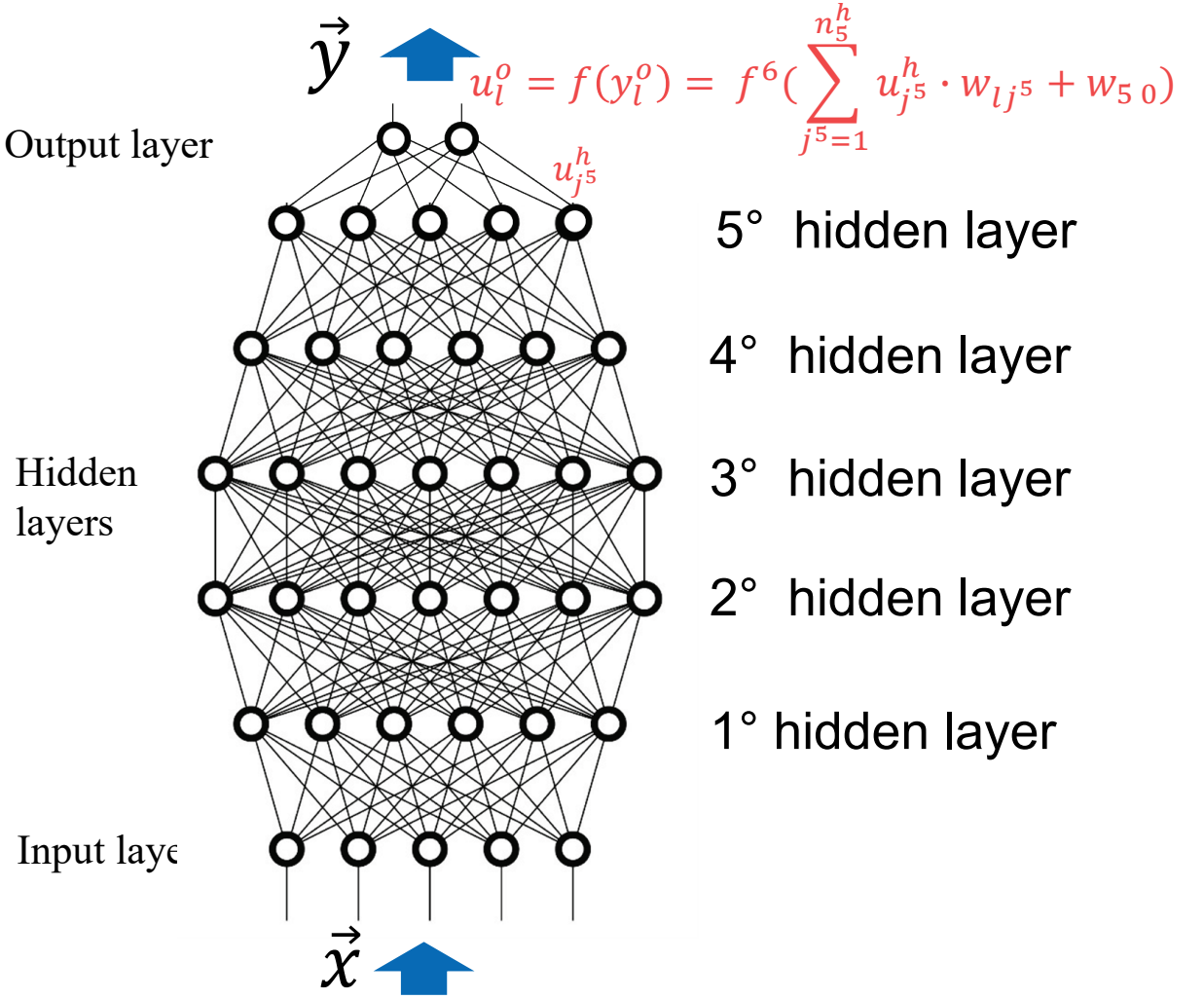
- Error back propagation: $w^{(i+1)} = w^{(i)} - \eta \frac{\partial E}{\partial w}$

$$E = \frac{1}{2n_o} \sum_{l=1}^{n_o} (u_l^o - t_l)^2$$

Number of output neurons

TRUE *l*-th output

ANN *l*-th output



Training DNNs: the Challenge

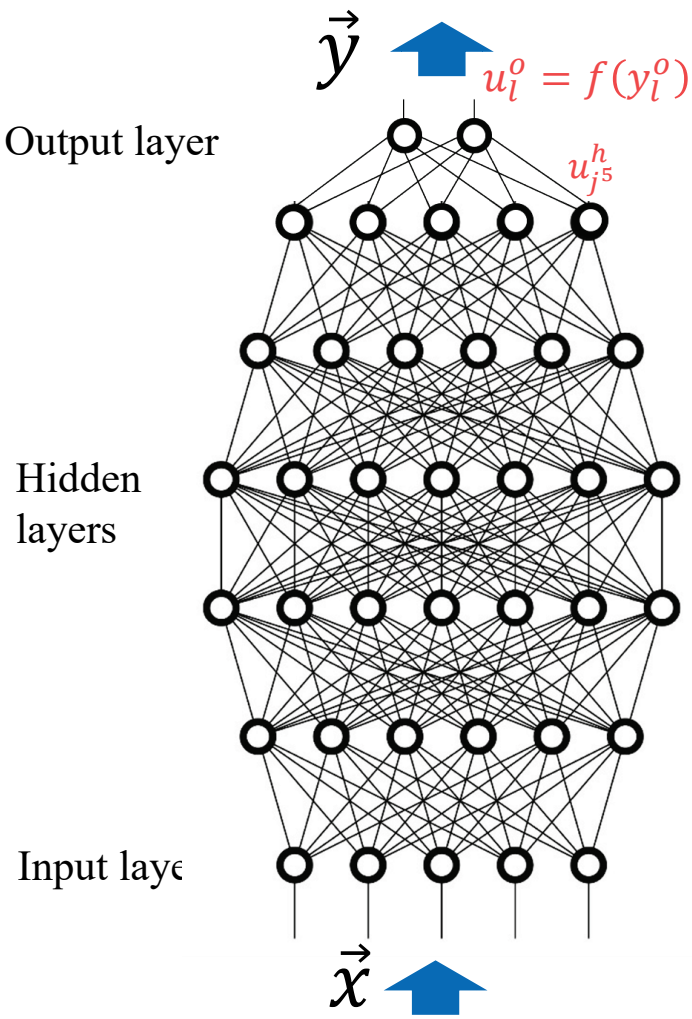
- Error back propagation: $w^{(i+1)} = w^{(i)} - \eta \frac{\partial E}{\partial w}$

$$E = \frac{1}{2n_o} \sum_{l=1}^{n_o} (u_l^o - t_l)^2$$

Number of output neurons

TRUE l-th output

ANN l-th output



$$w_{lj5}^{(i+1)} = w_{lj5}^{(i)} - \eta \frac{\partial E}{\partial u_l^o} \frac{\partial u_l^o}{\partial y_l^o} \frac{\partial y_l^o}{\partial w_{lj5}}$$

Product of 3 partial derivatives

Training DNNs: the Challenge

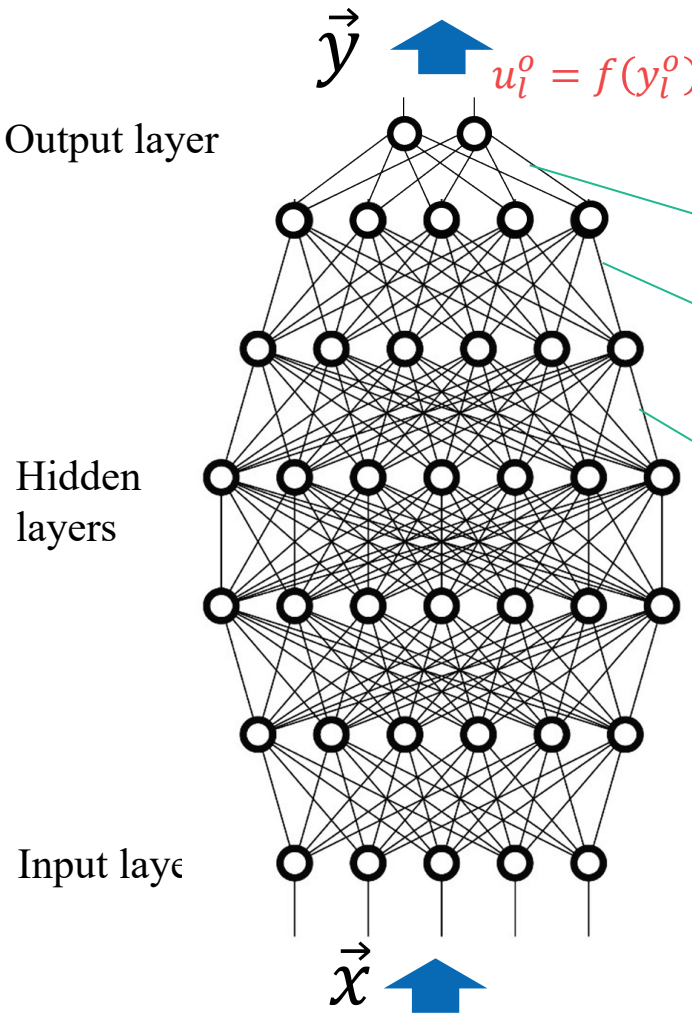
- Error back propagation: $w^{(i+1)} = w^{(i)} - \eta \frac{\partial E}{\partial w}$

$$E = \frac{1}{2n_o} \sum_{l=1}^{n_o} (u_l^o - t_l)^2$$

Number of output neurons

TRUE *l*-th output

ANN *l*-th output



$$w_{lj4}^{(i+1)} = w_{lj4}^{(i)} - \eta \frac{\partial E}{\partial u_l^o} \frac{\partial u_l^o}{\partial y_l^o} \frac{\partial y_l^o}{\partial w_{lj4}}$$ Product of 3 partial derivatives

$$w_{j4j3}^{(i+1)} = w_{j4j3}^{(i)} - \eta \frac{\partial E}{\partial u_l^o} \frac{\partial u_l^o}{\partial y_l^o} \frac{\partial y_l^o}{\partial u_{j4}^h} \frac{\partial u_{j4}^h}{\partial y_{j3}^h} \frac{\partial y_{j3}^h}{\partial w_{j4j3}}$$ Product of 5 partial derivatives

$$w_{j3j2}^{(i+1)} = w_{j3j2}^{(i)} - \eta \frac{\partial E}{\partial u_l^o} \frac{\partial u_l^o}{\partial y_l^o} \frac{\partial y_l^o}{\partial u_{j4}^h} \frac{\partial u_{j4}^h}{\partial y_{j3}^h} \frac{\partial y_{j3}^h}{\partial u_{j2}^h} \frac{\partial u_{j2}^h}{\partial y_{j2}^h} \frac{\partial y_{j2}^h}{\partial w_{j3j2}}$$ Product of 7 partial derivatives

$w_{j2j1}^{(i+1)} = w_{j2j1}^{(i)} - 0$

$w_{j1k}^{(i+1)} = w_{j1k}^{(i)} - 0$

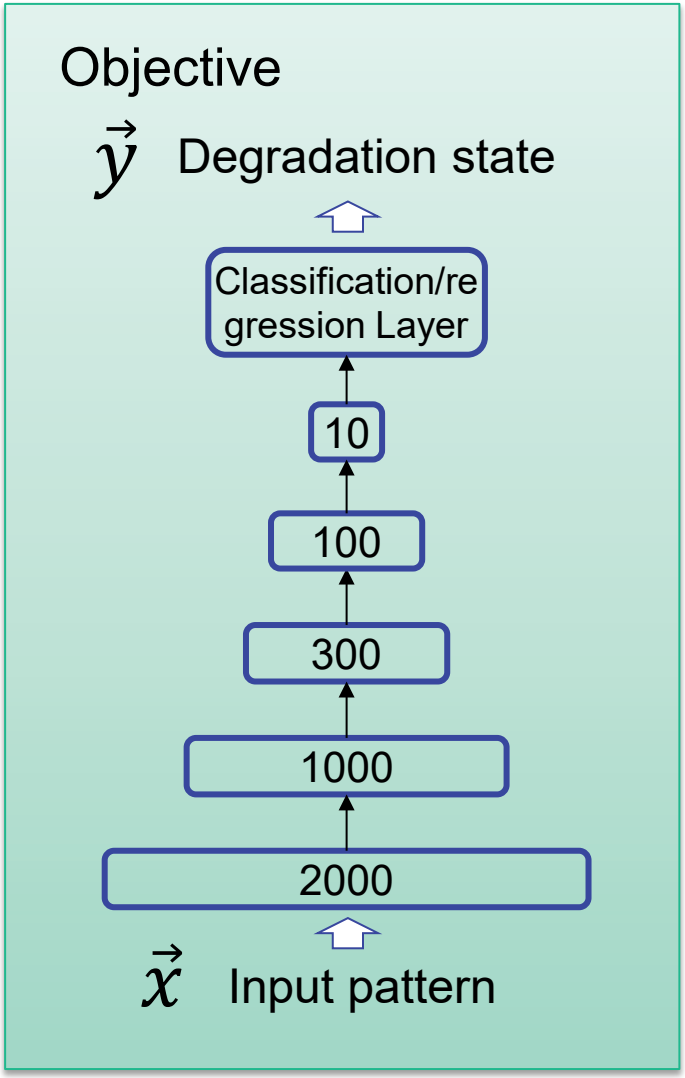
Partial derivatives are small numbers (<1), the product of many partial derivatives is ≈0

Problem: Error backpropagation fails to tune weights!

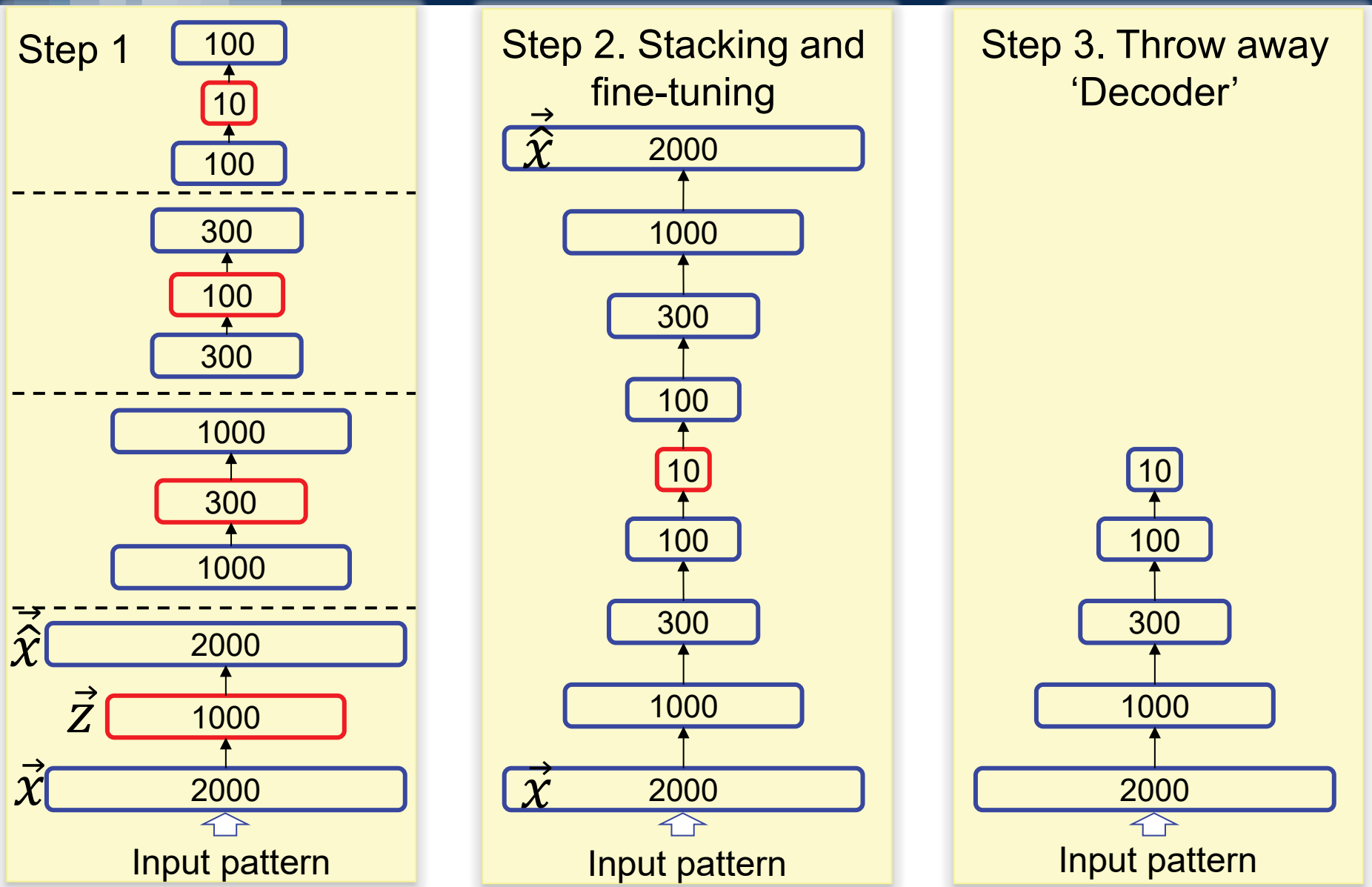
Gradient vanishing

DNN: ARCHITECTURE & TRAINING

Build a DNN using autoencoders

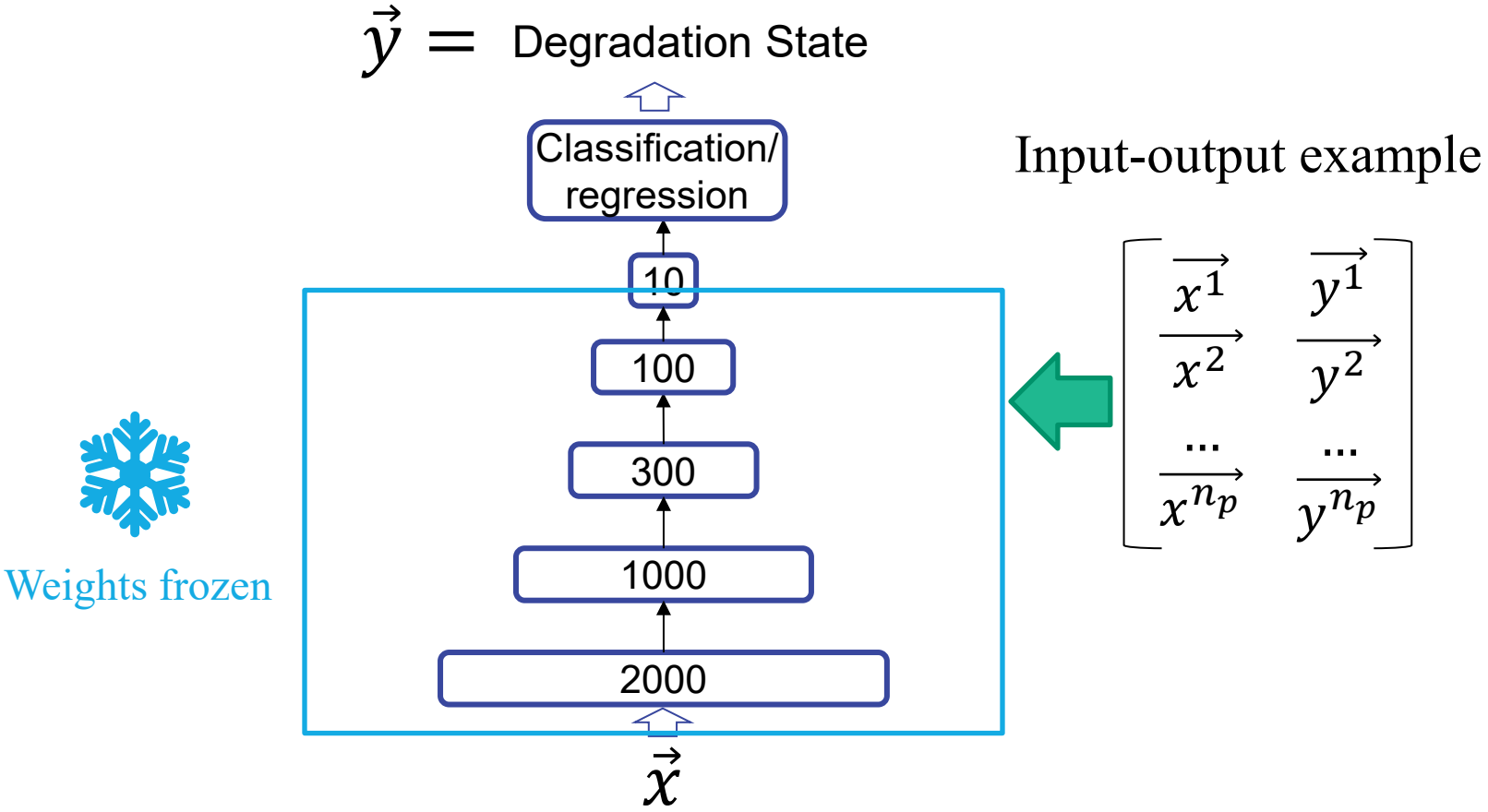


Build a DNN using autoencoders

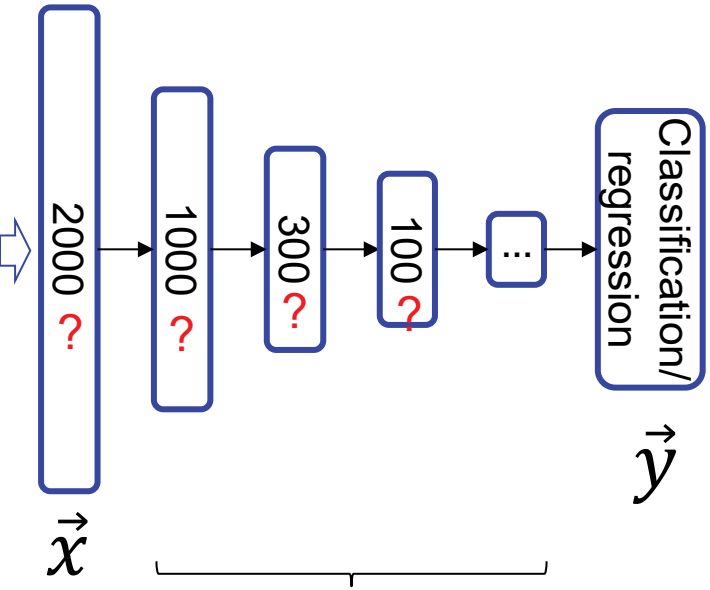


Build a DNN using autoencoders

Step 4: add a classification/regression layer and fine tuning

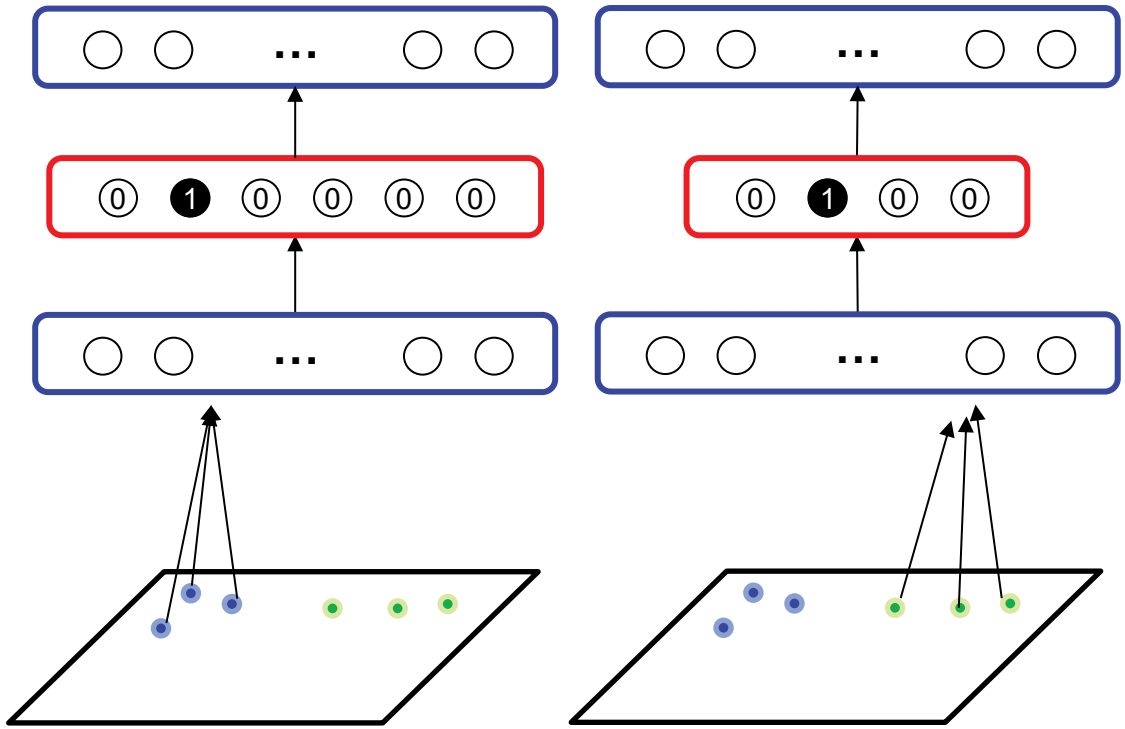


DNN: How to set the architecture?



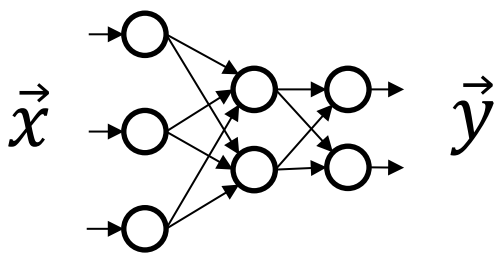
How many layers?

Sparse autoencoders are able to extract interesting features in the data relatively independently from the number of layers and neurons



Comparison: Shallow and Deep Neural Network

Shallow Neural Network



Number of hidden layers: typically <3

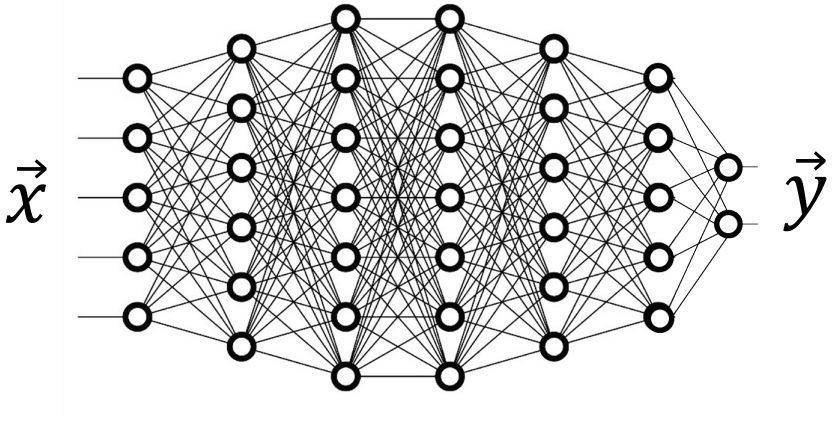
Training **Back propagation**

Accuracy ×

Raw data ×

Big data ×

Deep Neural Network (DNN)



Number of hidden layers: >=3

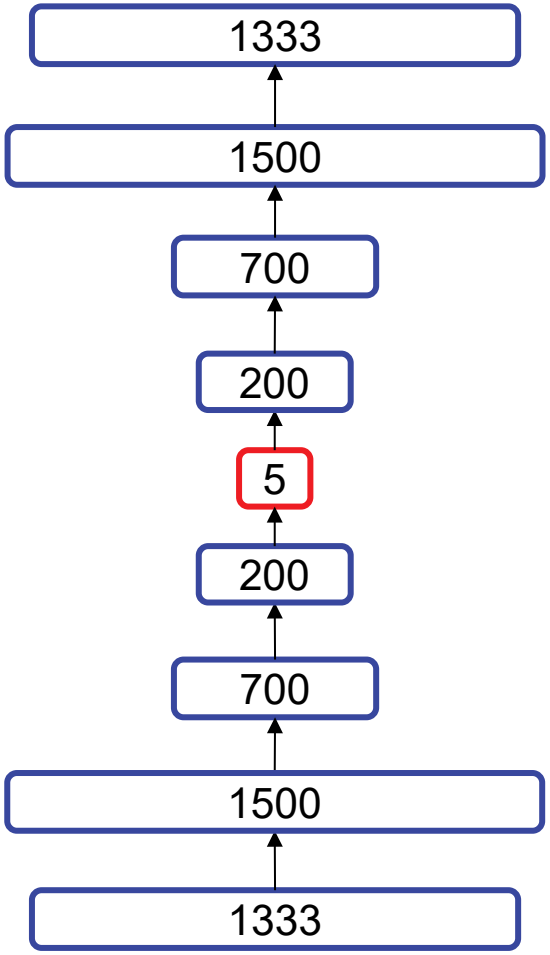
Training **Back propagation: Layer-wise, Fine-tuning**

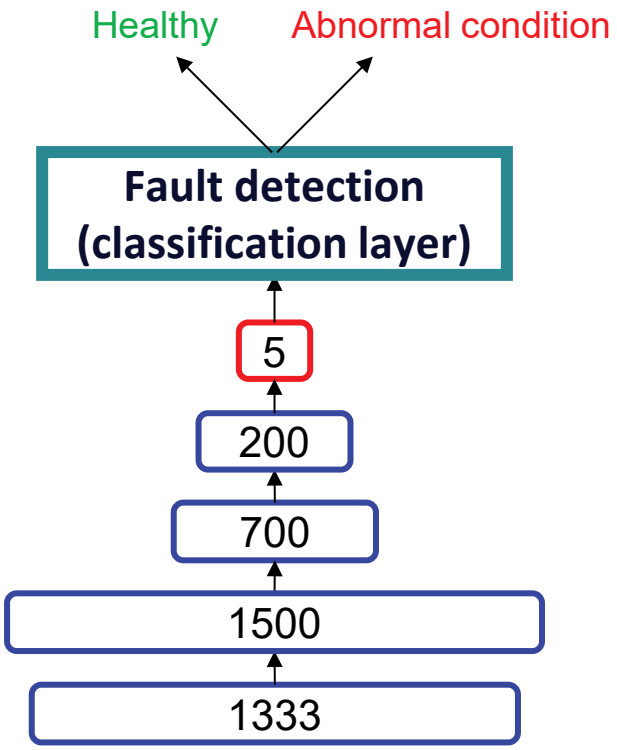
Accuracy √

Raw data √

Big data √

Case study – Part II: training a DNN for anomaly detection

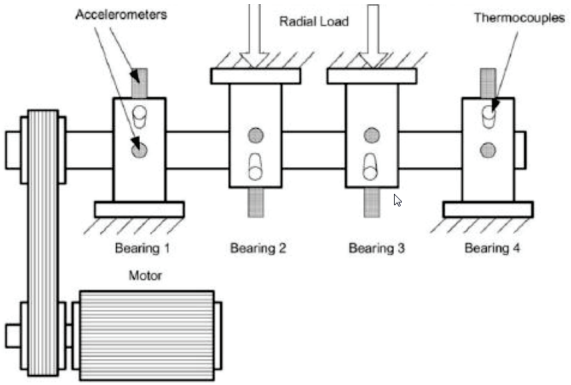




AE for anomaly detection in rotating machines

Case study: available data

2 run-to-failure trajectories (experiments)

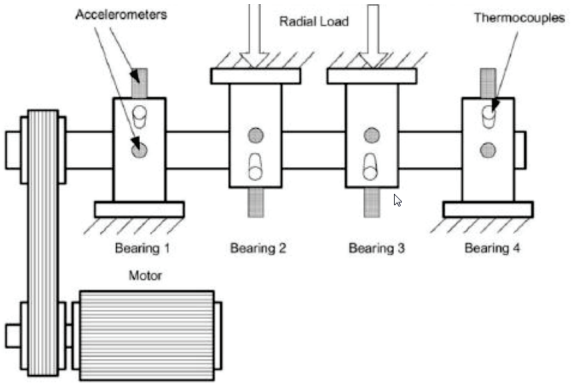


	Experiment 1	Experiment 2
Bearing 1	Healthy	Failed
Bearing 2	Healthy	Healthy
Bearing 3	Failed (inner race)	Healthy
Bearing 4	Failed (roller element)	Healthy

[1] Benchmark available at Nasa prognostic repository (https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/). J. Lee, H. Qiu, G. Yu, J. Lin, and Rexnord Technical Services (2007). IMS, University of Cincinnati. "Bearing Data Set", NASA Ames Prognostics Data Repository (http://ti.arc.nasa.gov/project/prognostic-data-repository), NASA Ames Research Center, Moffett Field, CA

Case study: Training Data - AE

2 run-to-failure trajectories (experiments)

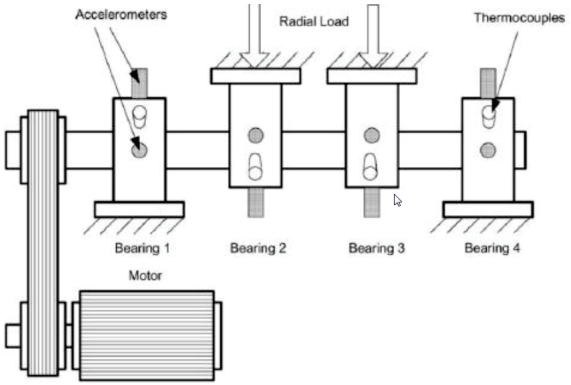


	Experiment 1	Experiment 2
Bearing 1	Healthy	Failed
Bearing 2	Healthy	Healthy
Bearing 3	Failed (inner race)	Healthy
Bearing 4	Failed (roller element)	Healthy

[1] Benchmark available at Nasa prognostic repository (https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/). J. Lee, H. Qiu, G. Yu, J. Lin, and Rexnord Technical Services (2007). IMS, University of Cincinnati. "Bearing Data Set", NASA Ames Prognostics Data Repository (http://ti.arc.nasa.gov/project/prognostic-data-repository), NASA Ames Research Center, Moffett Field, CA

Case study: Trainin Data - Deep NN

2 run-to-failure trajectories (experiments)



	Experiment 1	Experiment 2
Bearing 1	Healthy	Failed
Bearing 2	Healthy	Healthy
Bearing 3	Failed (inner race)	Healthy
Bearing 4	Failed (roller element)	Healthy

[1] Benchmark available at Nasa prognostic repository (https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/). J. Lee, H. Qiu, G. Yu, J. Lin, and Rexnord Technical Services (2007). IMS, University of Cincinnati. "Bearing Data Set", NASA Ames Prognostics Data Repository (http://ti.arc.nasa.gov/project/prognostic-data-repository), NASA Ames Research Center, Moffett Field, CA

Case study: Training Data - Deep NN

B1

Onset of failure: 539 [1], 546 [2]

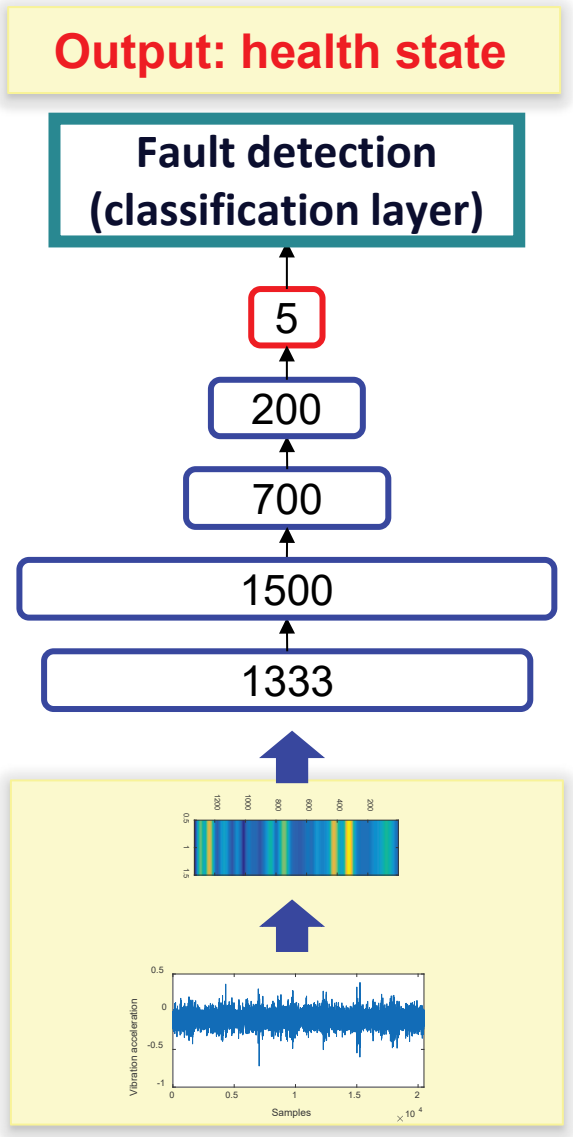
Snapshot	Label	Health State
1~538	0	Healthy
539~546	0.5	Possibly Failed
547~984	1	Failed

[1] Qiu, H., J. Lee, J. Lin, and G. Yu, *Wavelet filter-based weak signature detection method and its application on rolling element bearing prognostics.* Journal of sound and vibration, 2006. **289**(4): p. 1066-1090.

[2] Hasani, R.M., G. Wang, and R. Grosu, *An Automated Auto-encoder Correlation-based Health-Monitoring and Prognostic Method for Machine Bearings.* arXiv preprint arXiv:1703.06272, 2017.

	Experiment 1	Experiment 2
Bearing 1	Healthy	Failed
Bearing 2	Healthy	Healthy
Bearing 3	Failed (inner race)	Healthy
Bearing 4	Failed (roller element)	Healthy

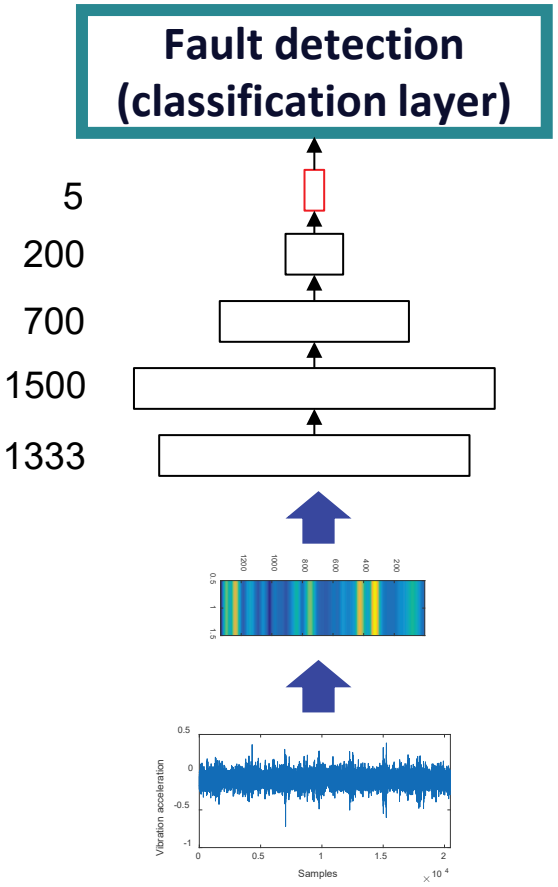
Case study: Developed Architecture



Case study: results

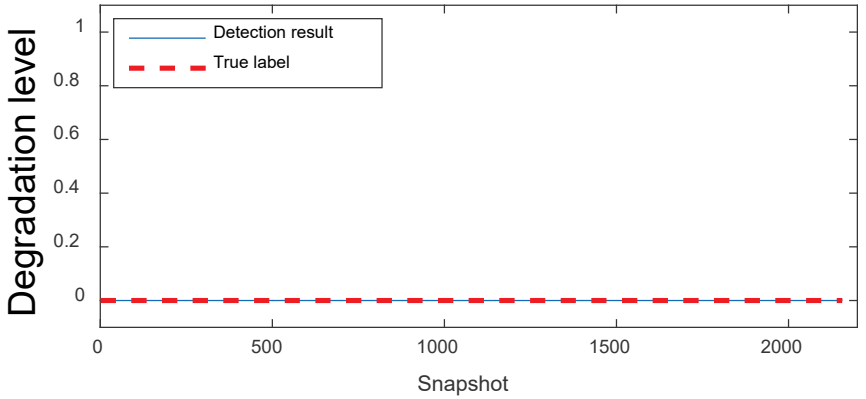
- **Developed model**

TEST SET = EXPERIMENT 1



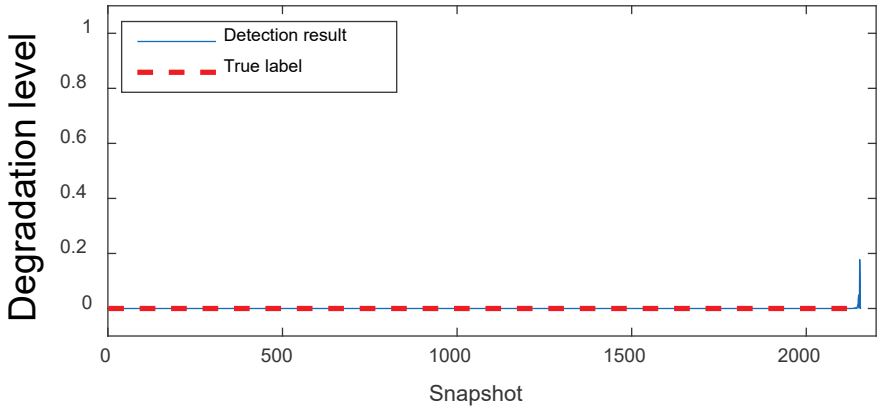
Healthy

B1



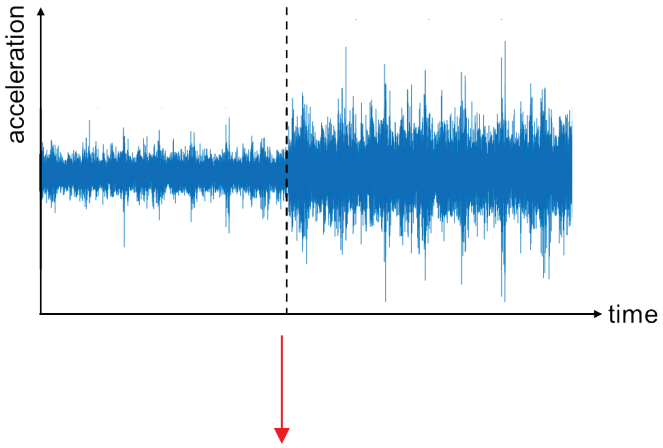
Healthy

B2



Case study: results

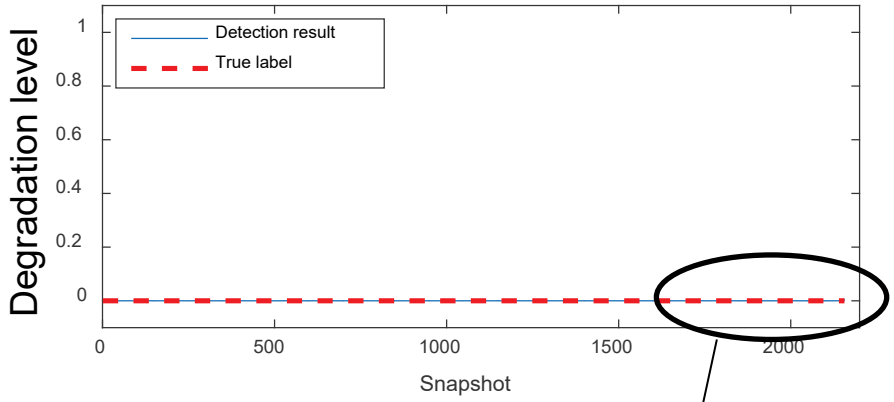
- **Input signals:**



Failure of bearing 3

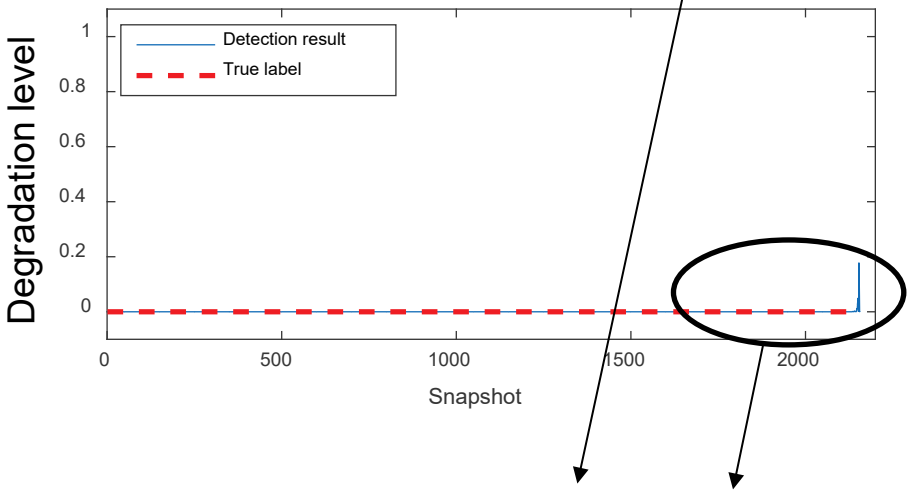
Healthy

B1



Healthy

B2



Robust: No false alarms, even when measured signals are influenced by failure of bearings B3 and B4

Case study: results

Ground Truth:

B4 

Failed
(Roller element)

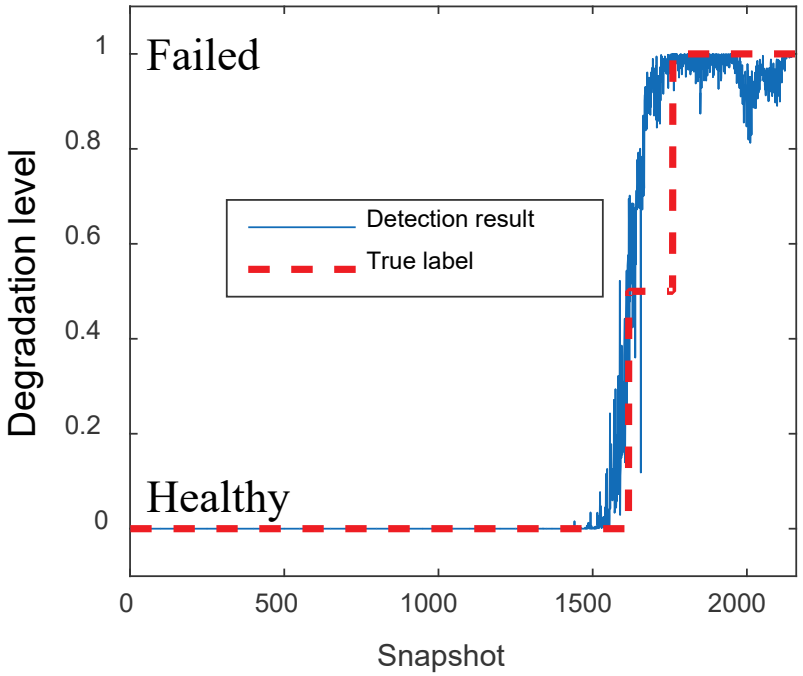
Snapshot	Label	Health State
1~1616	0	Healthy
1617~1760	0.5	Possibly Failed
1761~2156	1	Failed

[1] Qiu, H., J. Lee, J. Lin, and G. Yu, *Wavelet filter-based weak signature detection method and its application on rolling element bearing prognostics.* Journal of sound and vibration, 2006. **289**(4): p. 1066-1090.

[2] Hasani, R.M., G. Wang, and R. Grosu, *An Automated Auto-encoder Correlation-based Health-Monitoring and Prognostic Method for Machine Bearings.* arXiv preprint arXiv:1703.06272, 2017.

[3] Yu, J., *Health condition monitoring of machines based on hidden Markov model and contribution analysis.* IEEE Transactions on Instrumentation and Measurement, 2012. **61**(8): p. 2200-2211.

➢ Result



The onset is identified at snapshot 1680 with threshold *Th*=0.5, which lies in the possibly failed range [1617, 1760]

Part 1: Deep Neural Networks

Part 2: Convolutional Neural Networks

Industry 4.0 – New sources of data

Data Types

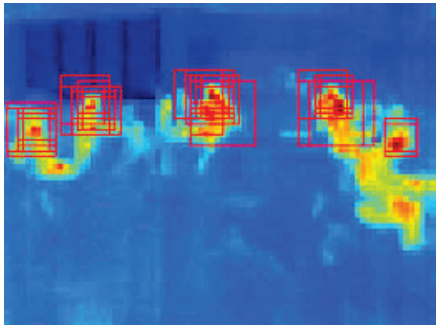
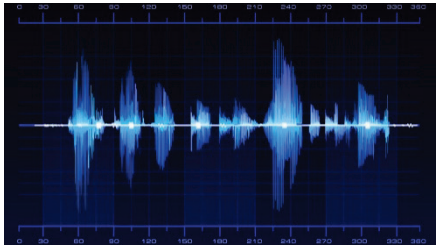


Image data

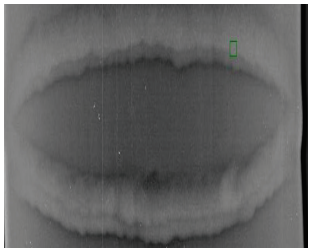
Text data



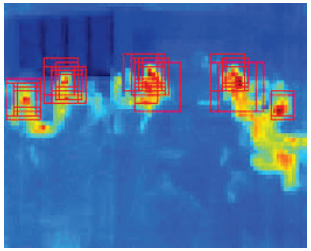
Sound data

Image data for condition monitoring

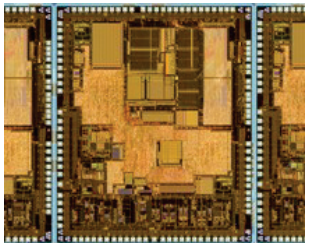
X-ray images



Thermal images



Electron microscope images



- Slow & Time consuming
- Not accurate
- Subjective
- Expensive

Objective:

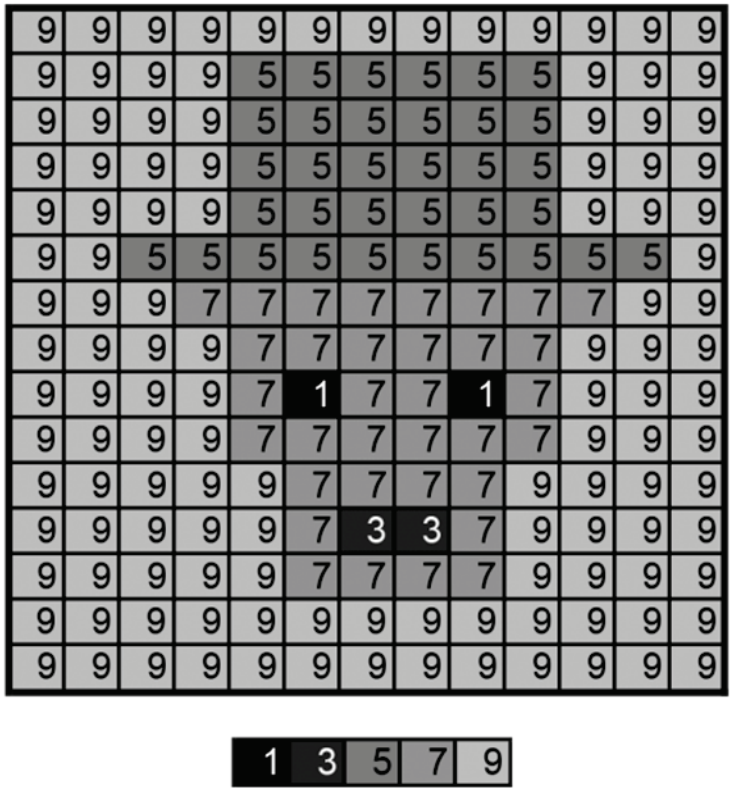
AUTOMATIC DEFECT DETECTOR

- Healthy
- Defective

Images as matrices

9	9	9	9	9	9	9	9	9	9	9	9	9
9	9	9	9	5	5	5	5	5	5	9	9	9
9	9	9	9	5	5	5	5	5	5	9	9	9
9	9	9	9	5	5	5	5	5	5	9	9	9
9	9	9	9	5	5	5	5	5	5	9	9	9
9	9	5	5	5	5	5	5	5	5	5	5	9
9	9	9	7	7	7	7	7	7	7	7	9	9
9	9	9	9	7	7	7	7	7	7	9	9	9
9	9	9	9	7	1	7	7	1	7	9	9	9
9	9	9	9	7	7	7	7	7	7	9	9	9
9	9	9	9	9	7	7	7	7	9	9	9	9
9	9	9	9	9	7	3	3	7	9	9	9	9
9	9	9	9	9	7	7	7	7	9	9	9	9
9	9	9	9	9	9	9	9	9	9	9	9	9
9	9	9	9	9	9	9	9	9	9	9	9	9

Pixels



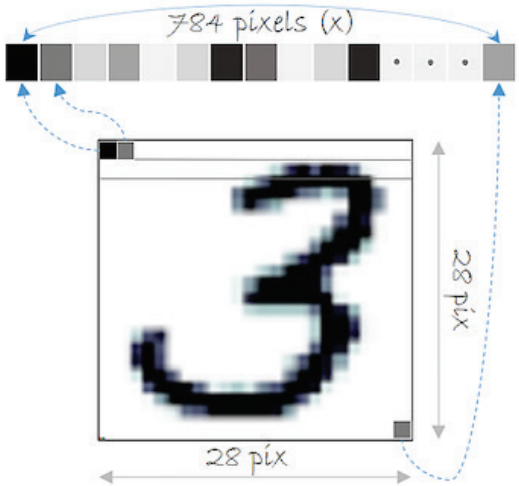
In black & white images: $\in [0, 255]$

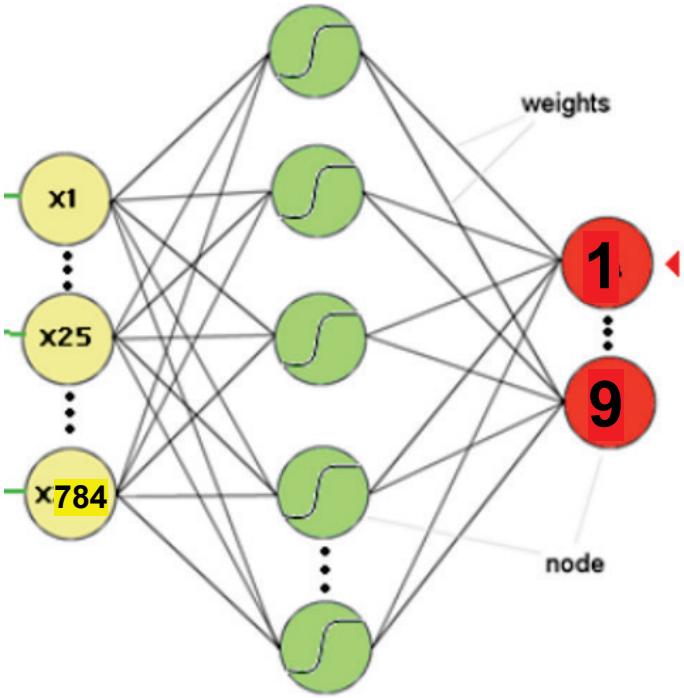
Black White

In colored images (E.g., RGB= Red Green Blue): Three matrices, one for each color

ANN for Image Recognition

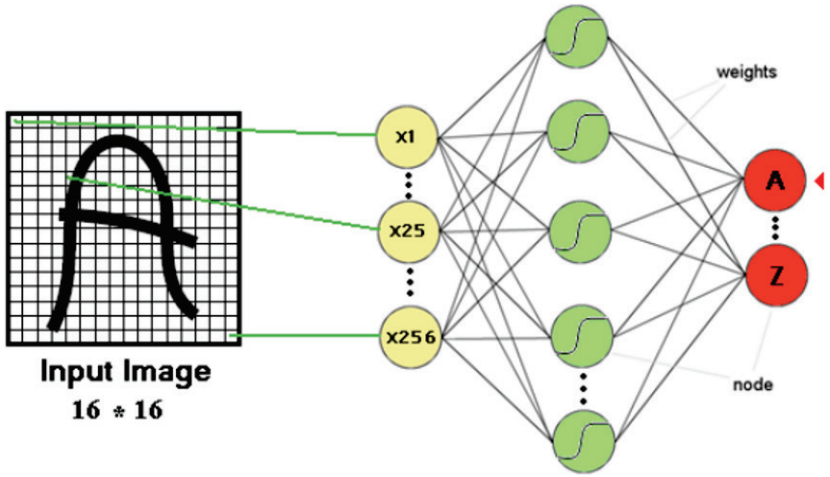
$\vec{x} =$

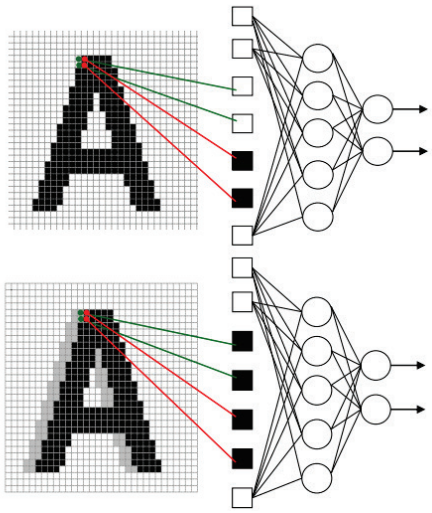




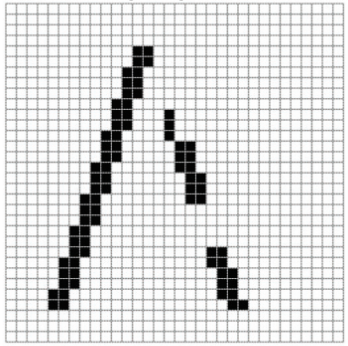
ANN for Image Recognition

- Too hard to train
- Too sensitive to noise
- No invariance to shifting, scaling, etc.



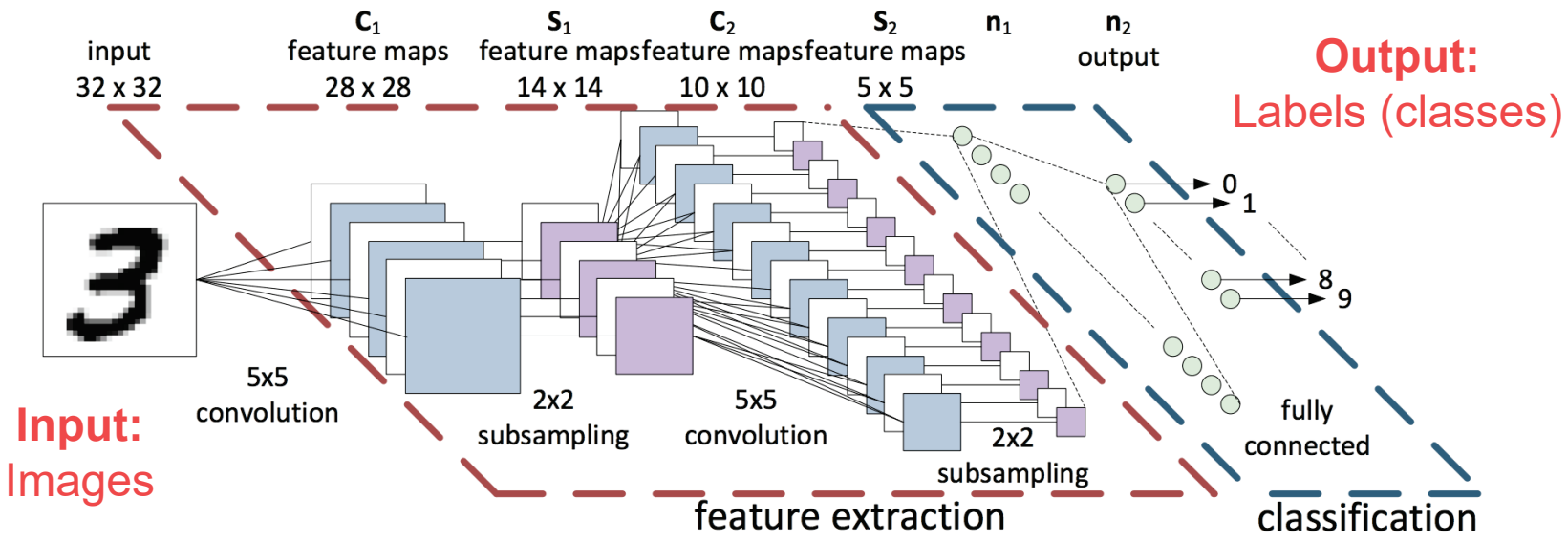


Changing pixels:

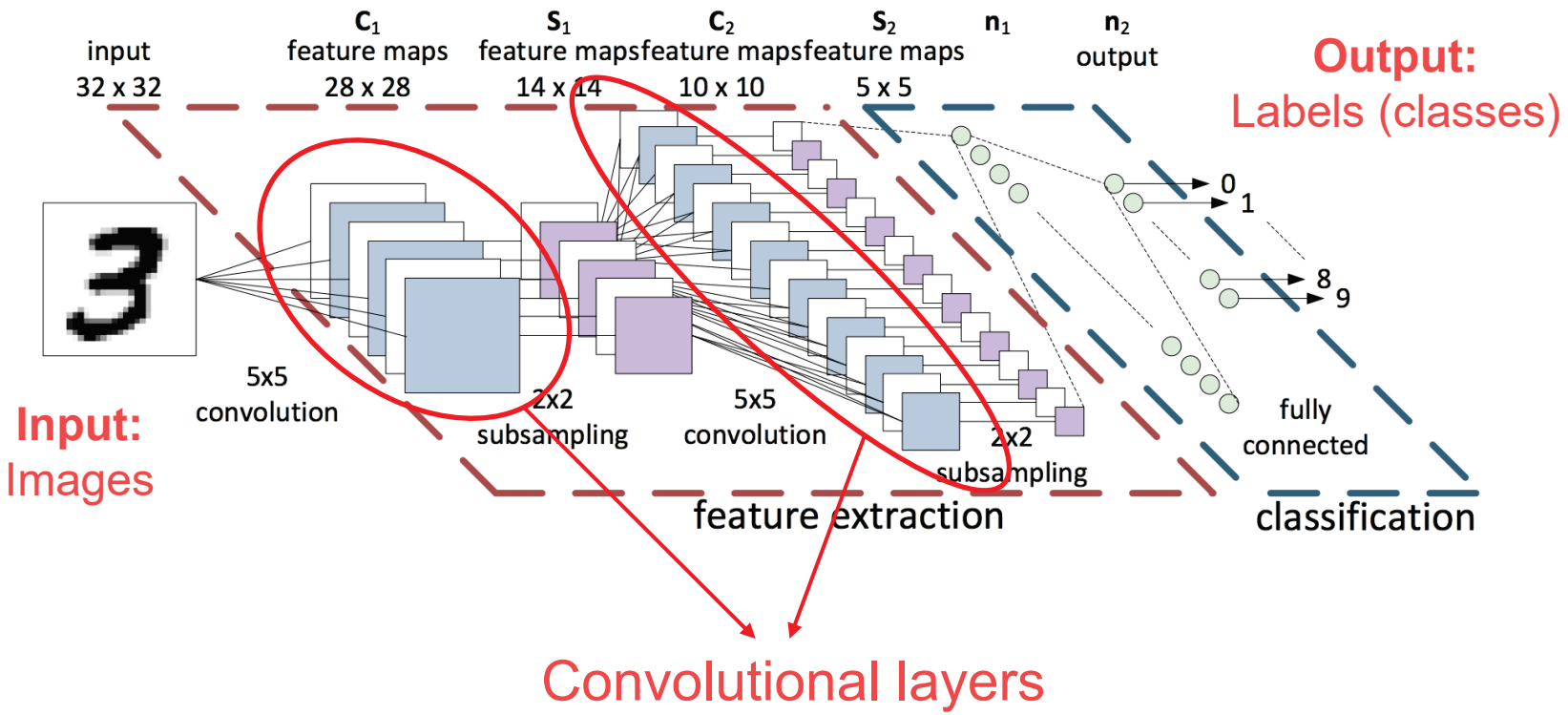


Several inputs are changing due to 2 pixel shift

Convolutional Neural Networks (CNN): Architecture



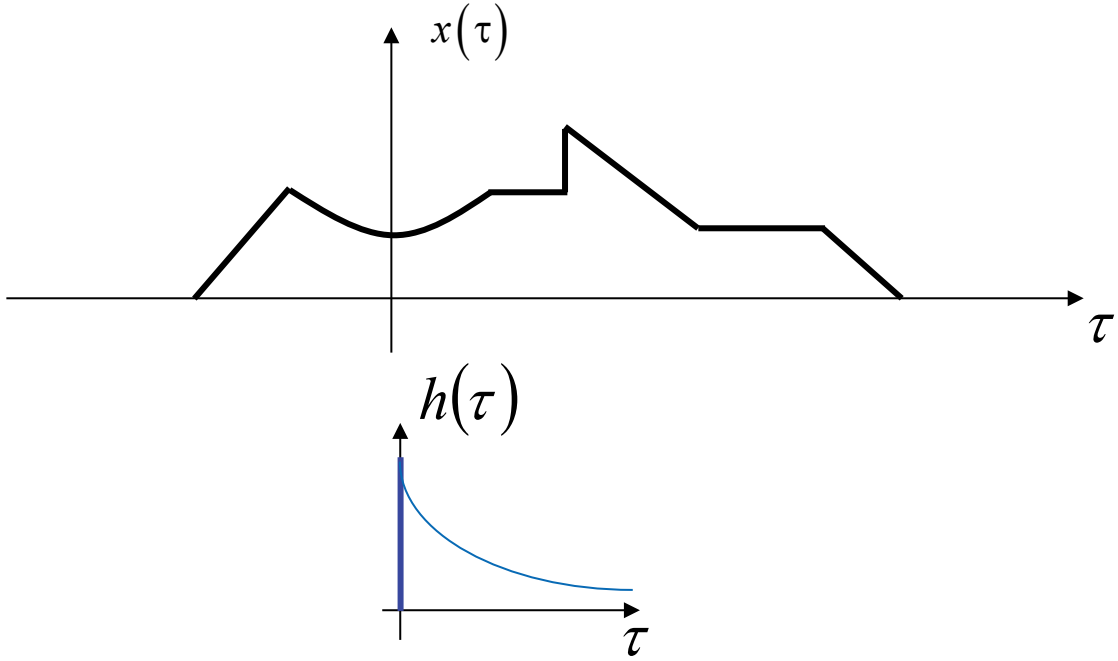
Convolutional layer



Convolution: Preliminaries

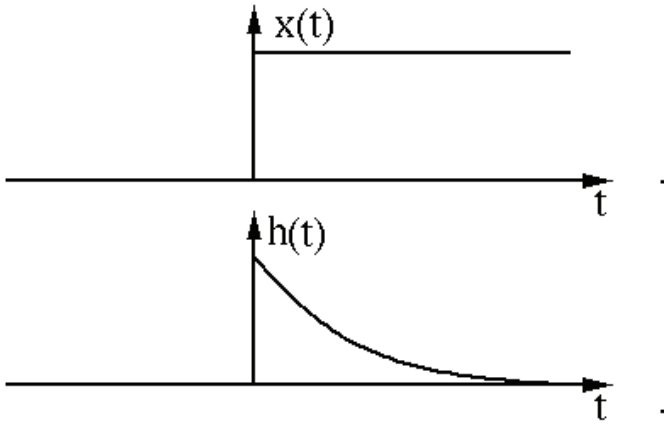
Let h and x two functions defined on $\mathbb{R}$. The **convolution** between h and x is the function $h * x$:

$$(h * x)(t) = \int_{-\infty}^{+\infty} x(\tau)h(t-\tau)d\tau$$



Convolution: Pictorial representation

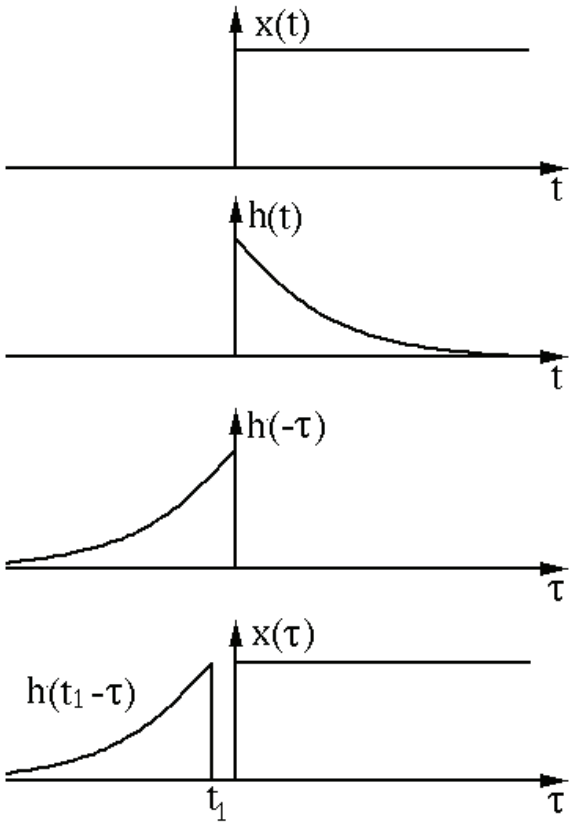
$$(h * x)(t) = \int_{-\infty}^{+\infty} x(\tau)h(t-\tau)d\tau$$



Convolution at t = weighted average of a local region around t of $x(t)$

Convolution: Pictorial representation

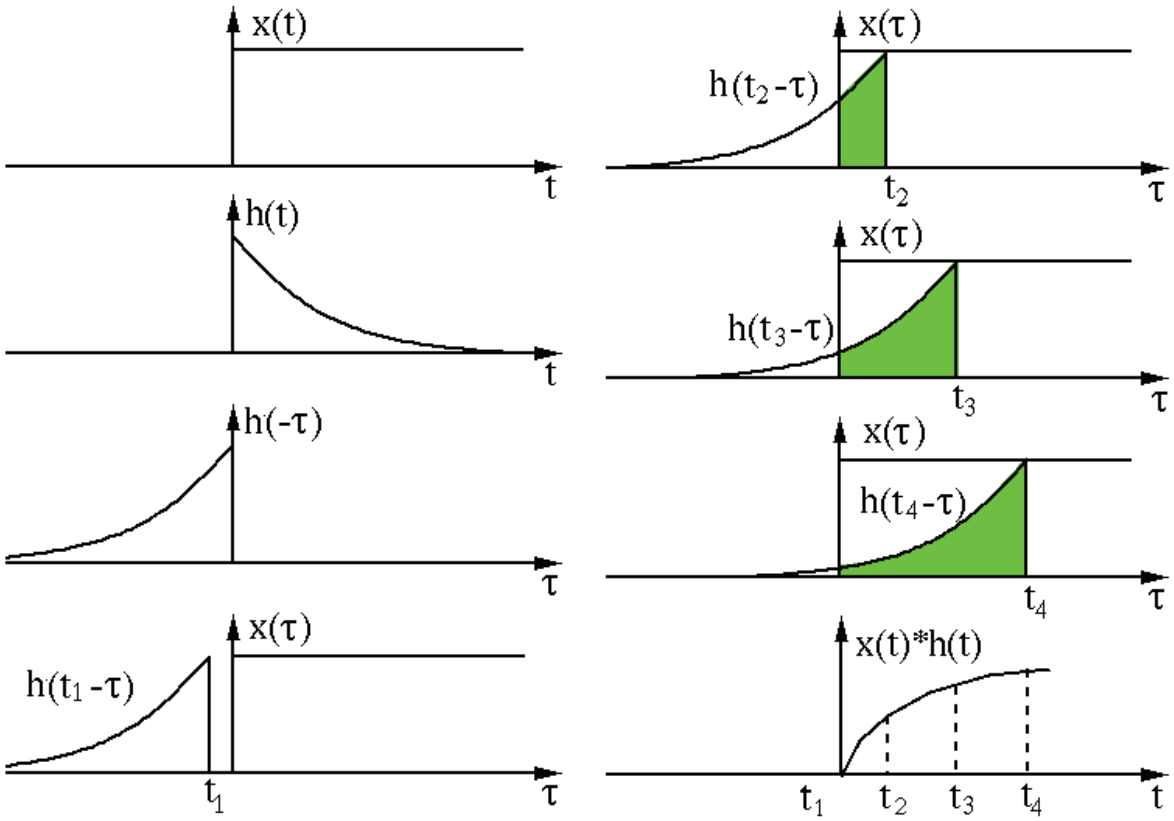
$$(h * x)(t_1) = \int_{-\infty}^{+\infty} x(\tau)h(t_1 - \tau)d\tau = 0$$



Convolution at t = weighted average of a local region around t of $x(t)$

Convolution: Pictorial representation

$$(h * x)(t) = \int_{-\infty}^{+\infty} x(\tau)h(t-\tau)d\tau$$



Convolution at t = weighted average of a local region around t of $x(t)$

1-D Discrete Convolution Example

$$\vec{x} = [1 \;\; 3 \;\; 2 \;\; 6 \;\; 5]$$
$$\vec{h} = [2 \;\; 1 \;\; 3]$$

t=1

$$\begin{matrix}[1 & 3 & 2 & 6 & 5] \\ \updownarrow & \updownarrow & \updownarrow & & \\ [2 & 1 & 3] & & \end{matrix}$$

t=2

$$\begin{matrix}[1 & 3 & 2 & 6 & 5] \\ & \updownarrow & \updownarrow & \updownarrow & \\ & [2 & 1 & 3] & \end{matrix}$$

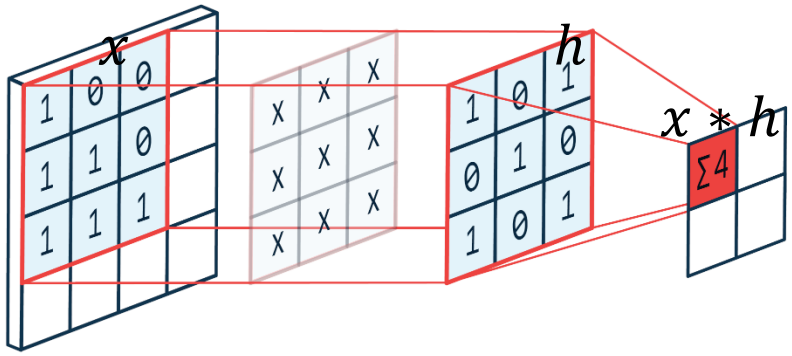
t=3

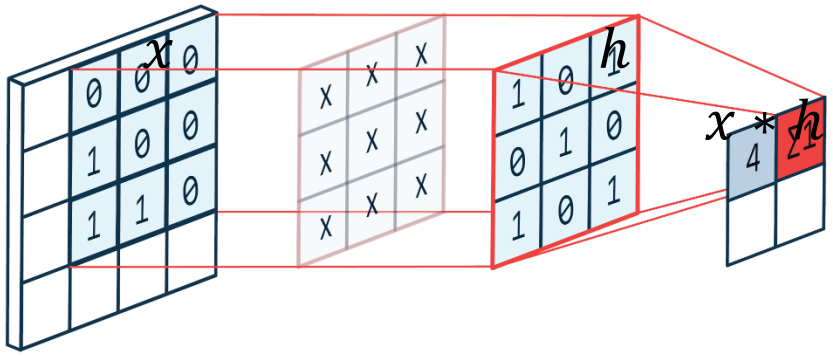
$$\begin{matrix}[1 & 3 & 2 & 6 & 5] \\ & & \updownarrow & \updownarrow & \updownarrow \\ & & [2 & 1 & 3] \end{matrix}$$

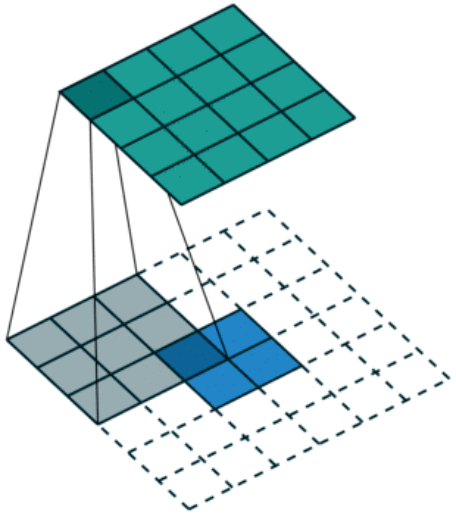
$$\vec{h} * \vec{x} = [\big((2 \times 1) + (1 \times 3) + (3 \times 2)\big),$$
$$\big((2 \times 3) + (1 \times 2) + (3 \times 6)\big),$$
$$((2 \times 2) + (1 \times 6) + (3 \times 5))]$$

Convolution on Images: Discrete convolution

- Like the 1-D case, we slide $\vec{h}$ on $\vec{x}$, this time in two dimensions (row by row)





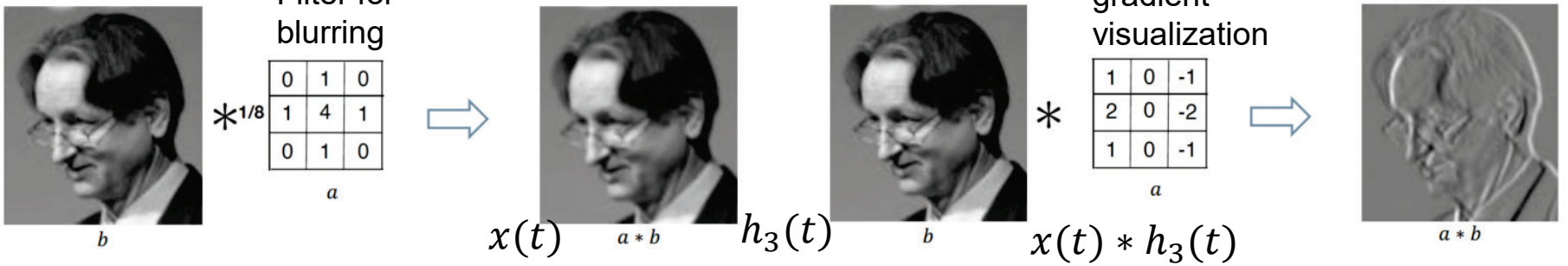


A Filter (Kernel)
e.g., 3 × 3

Filters revel the key features

Why convolution?

- Convolution → Filtering one function by another function
- Applications in image processing:
  - Sharpening
  - Blurring
  - Edge detection
  - Etc

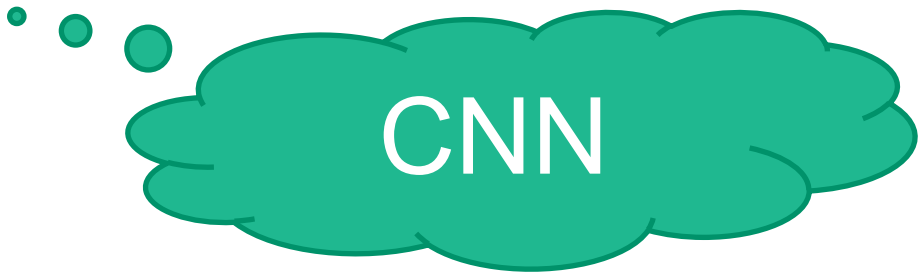


Problem definition

- Problem: classification of images (E.g., of oranges and pencils)

- Possible solution: Designing a filter for edge detection and defining rules based on the geometry of detected edges for classification of images
  - Problems:
    - not robust
    - might not work for different angles and lighting conditions
    - will not work if the images contain other objects besides oranges and pencils
    - …

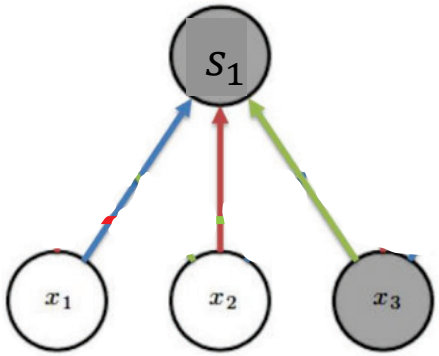
- Better solution: A neural network trained on images of oranges and pencils which automatizes the process of obtaining filters and performing classification



Convolutional layer (1-D case)

$$\vec{x} = [x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; x_5]$$

$$h = [\;\boxed{W1}\boxed{W2}\boxed{W3}\;]$$

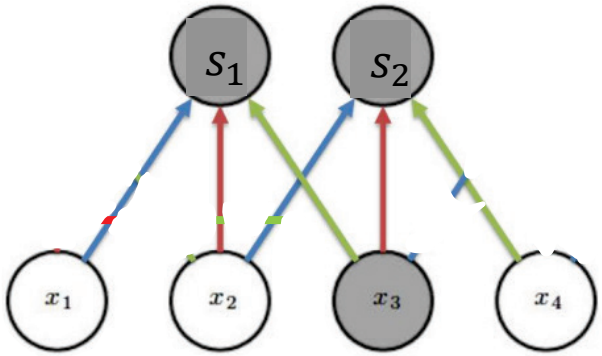


$$s_1 = x_1 \cdot w_1 + x_2 \cdot w_2 + x_3 \cdot w_3 = (h * x)\,(1)$$

Convolutional layer (1-D case)

$$\vec{x} = [x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; x_5]$$

$$h = [\; \boxed{W1}\boxed{W2}\boxed{W3} \;]$$

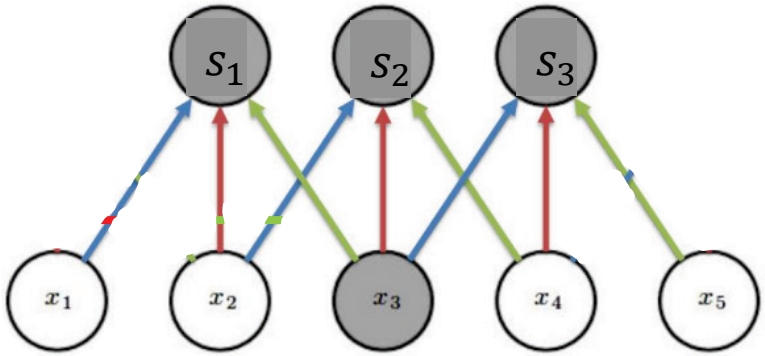


$$s_1 = x_1 \cdot w_1 + x_2 \cdot w_2 + x_3 \cdot w_3 = (h * x)\ (1)$$
$$s_2 = x_2 \cdot w_1 + x_3 \cdot w_2 + x_4 \cdot w_3 = (h * x)\ (2)$$

Convolutional layer (1-D case)

$$\vec{x} = [x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; x_5]$$

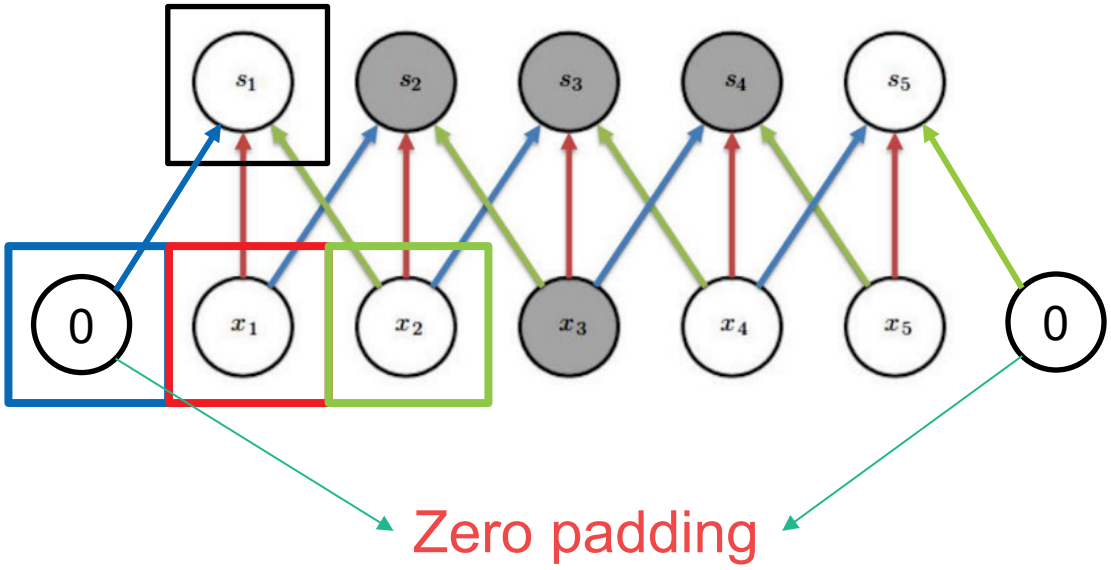
$$h = [\; \boxed{W1}\boxed{W2}\boxed{W3} \;]$$



$$s_1 = x_1 \cdot w_1 + x_2 \cdot w_2 + x_3 \cdot w_3 = (h * x) \; (1)$$
$$s_2 = x_2 \cdot w_1 + x_3 \cdot w_2 + x_4 \cdot w_3 = (h * x) \; (2)$$
$$s_3 = x_3 \cdot w_1 + x_4 \cdot w_2 + x_5 \cdot w_3 = (h * x) \; (3)$$

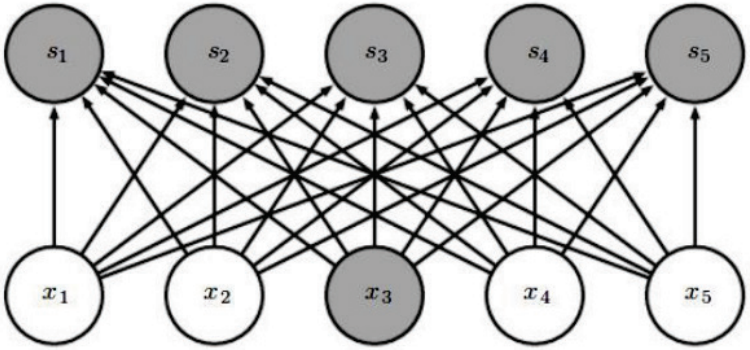
Convolutional layer (1-D case) with zero padding

$$h = [\ \boxed{W1}\ \boxed{W2}\ \boxed{W3}\ ]$$



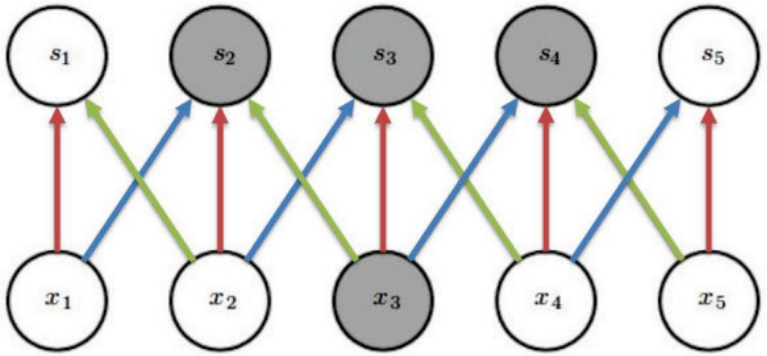
Convolutional vs. Fully-connected Layers

Fully connected



25 weights

3x1 Convolution



3 weights

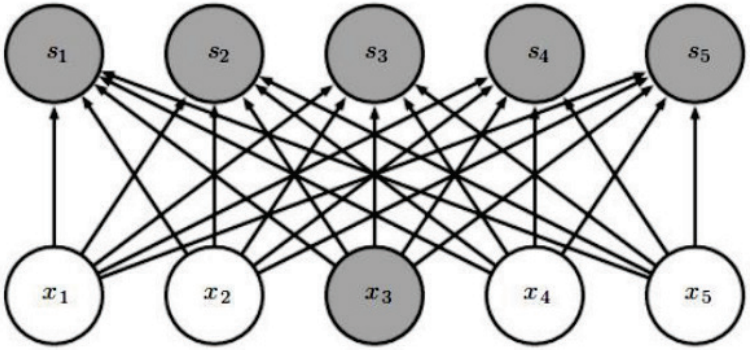
Each neuron is connected to only three neurons

CNN Layers:

- Sparse connectivity
- Parameter sharing
- Translation invariance

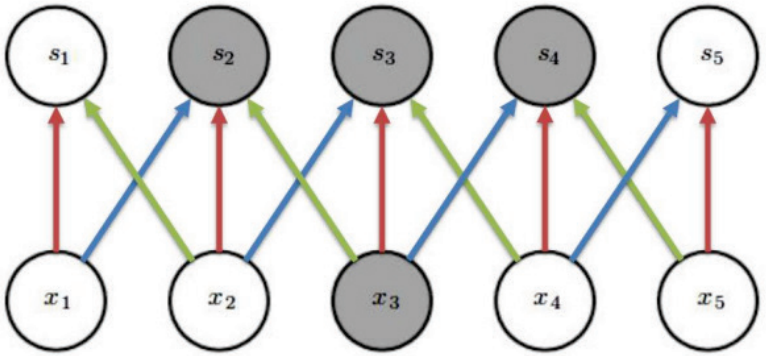
Convolutional vs. Fully-connected Layers

Fully connected



25 weights

3x1 Convolution

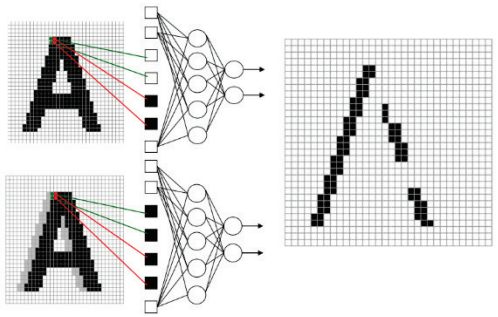


3 weights

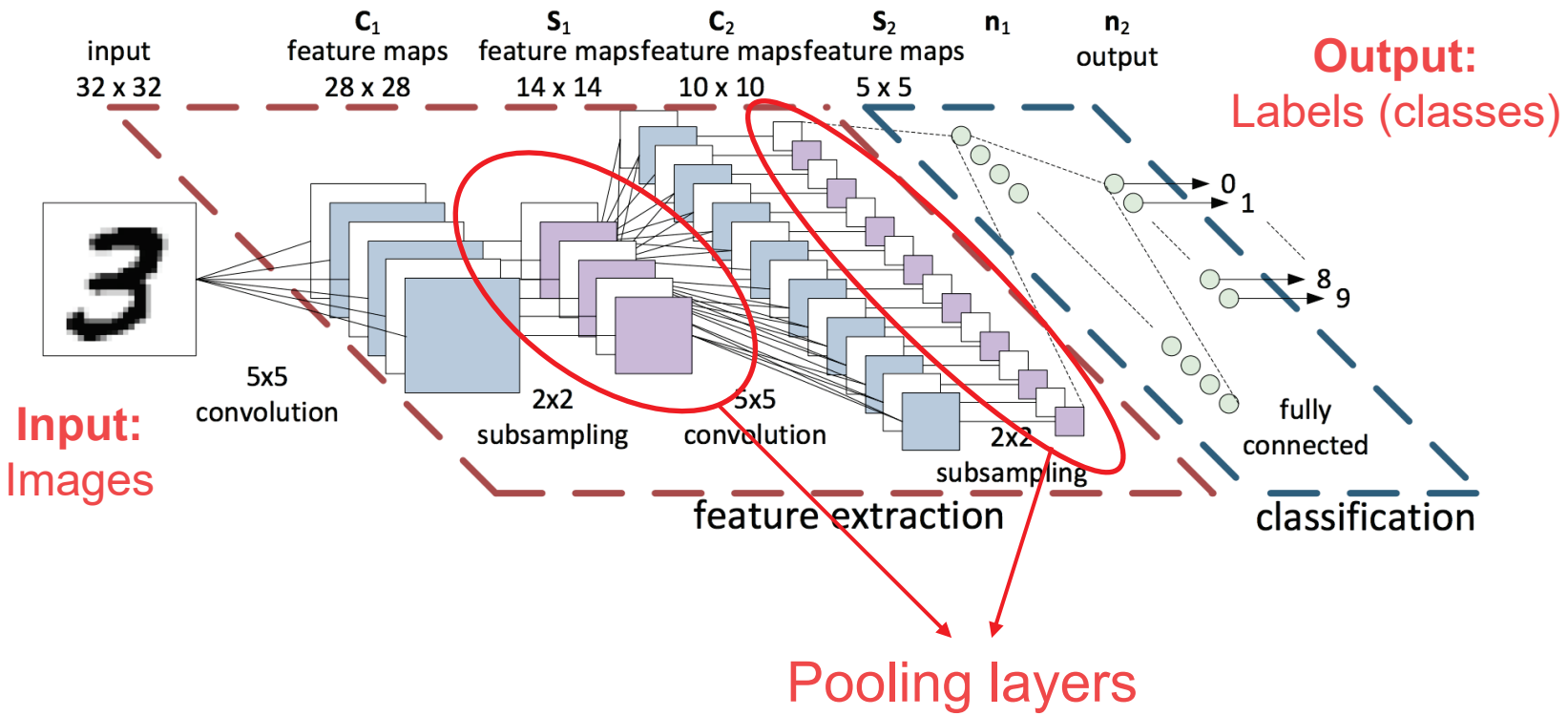
Each neuron is connected to only three neurons

CNN Layers:

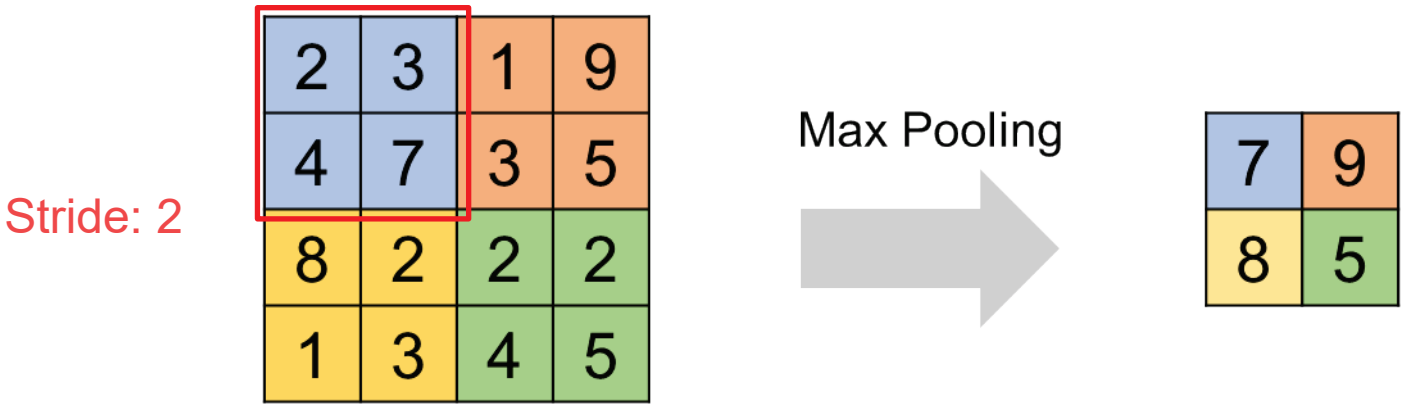
- Sparse connectivity
- Parameter sharing
- Translation invariance



Pooling layer



CNN: Pooling or Subsampling



- Translation invariance
- Reducing the dimension of input image
- Reducing the computation cost
- Avoiding the problem of overfitting

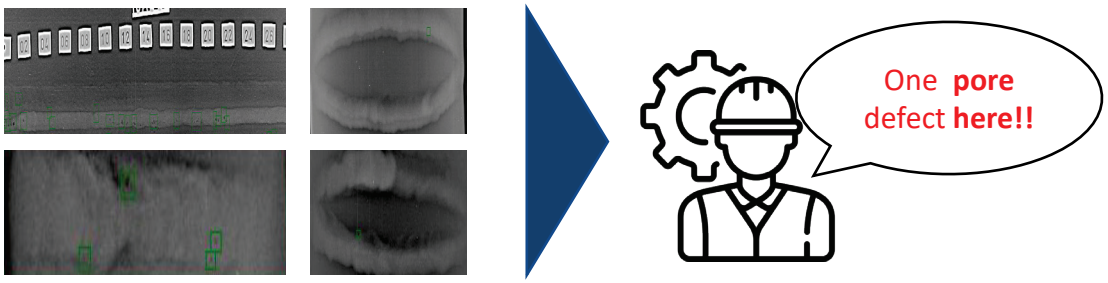
Application

CNNs for weld monitoring in industrial plants

Weld Monitoring for Defect Detection

- Weld is the most **critical** part in all large metal structures (e.g. reactor pressure vessel and pipelines)

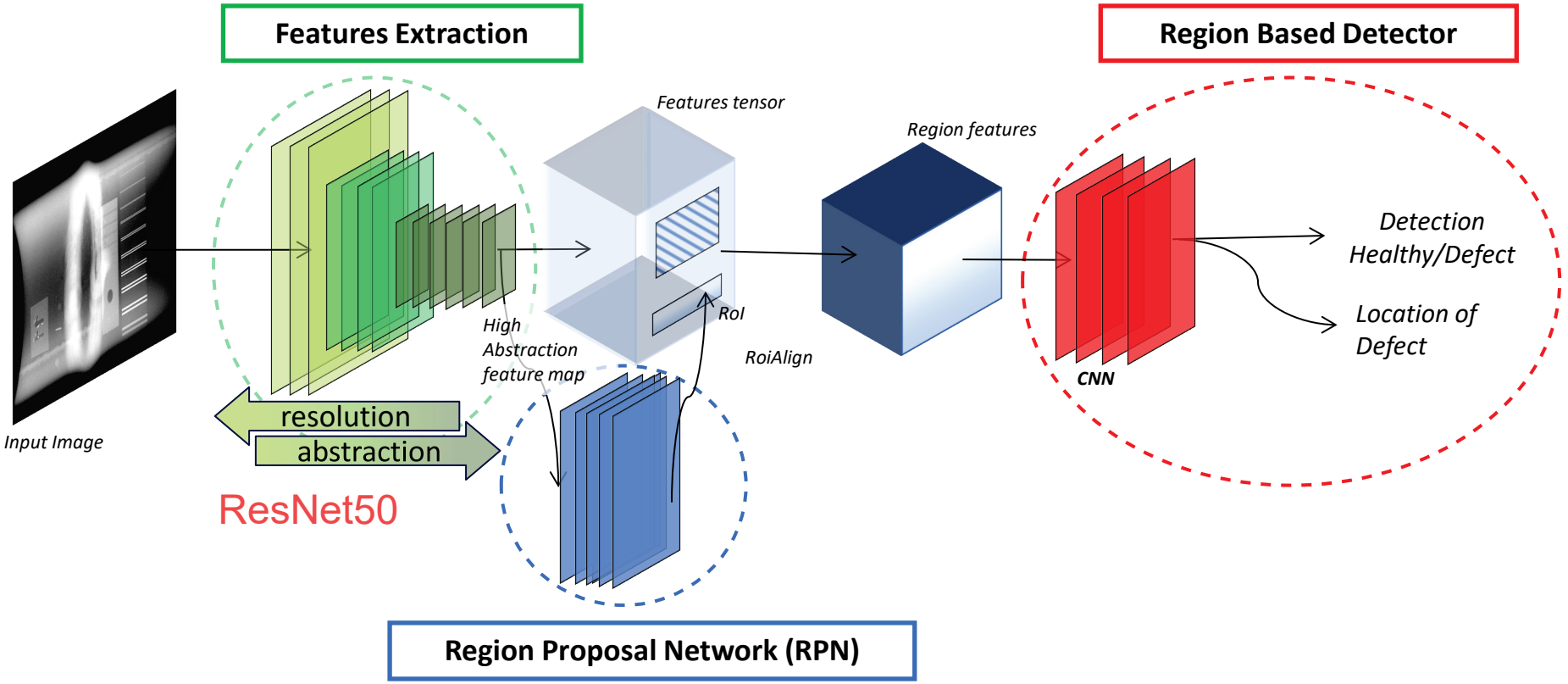
- **Visual Inspection** through Radiographic Testing (RT) systematically produce X-ray images during qualification checks



- Slow & Time consuming
- Not accurate
- Subjective
- Expensive

Detection AND Localization

CNN Architecture: Mask-R-CNN



Dataset

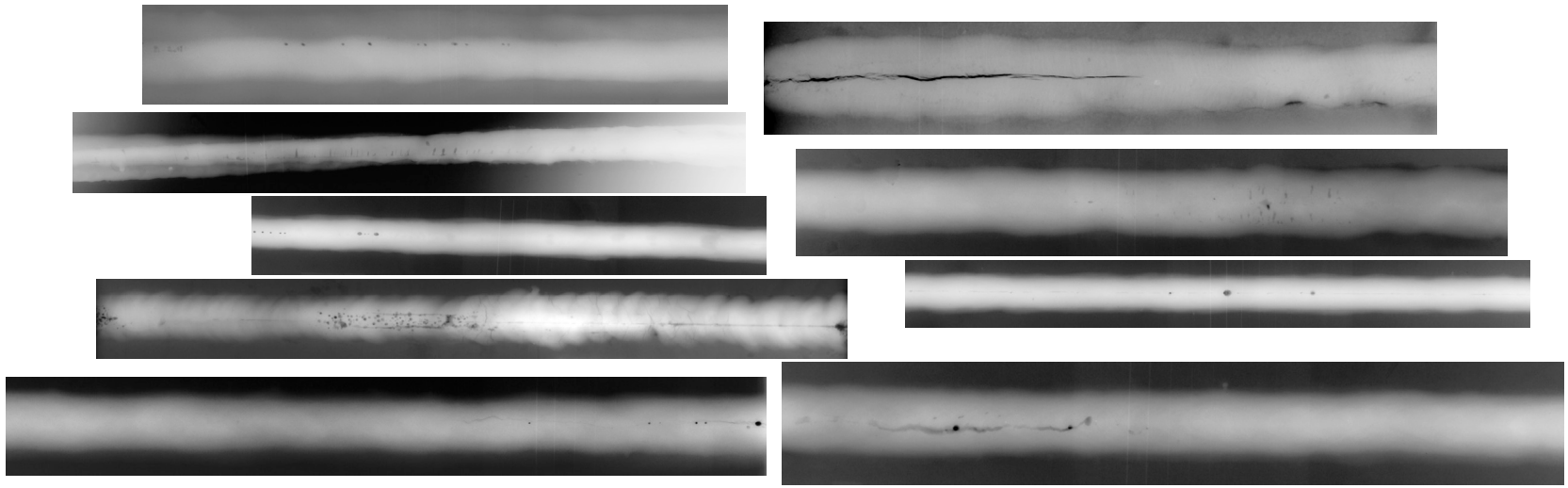
❑ Benchmark: GDXray *Domingo Mery et al. (2015) Journal of NDT*

10 large X-ray images with dimension ≃ 400 x 9000 pixel

High resolution in terms of density of pixels per inch (dpi)

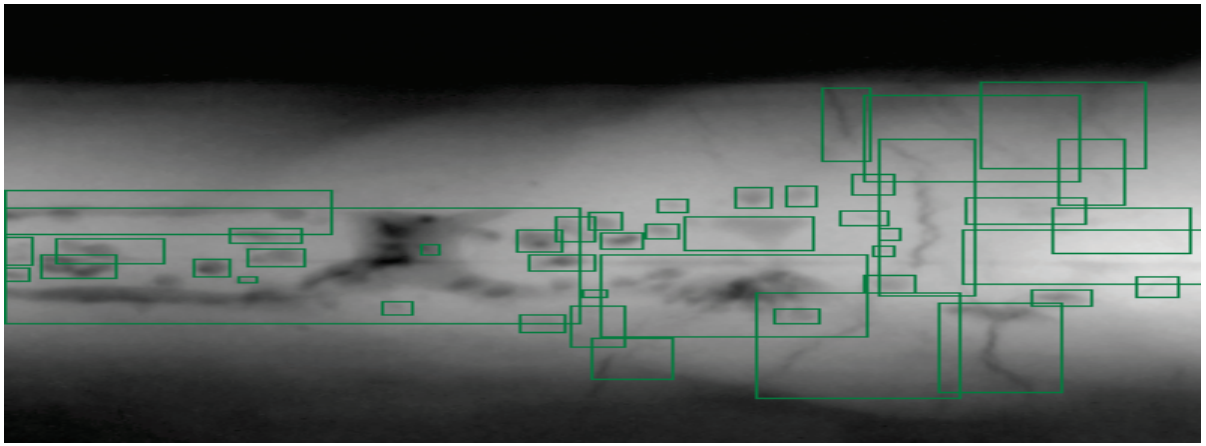
582 bounding-boxes

Data Set example

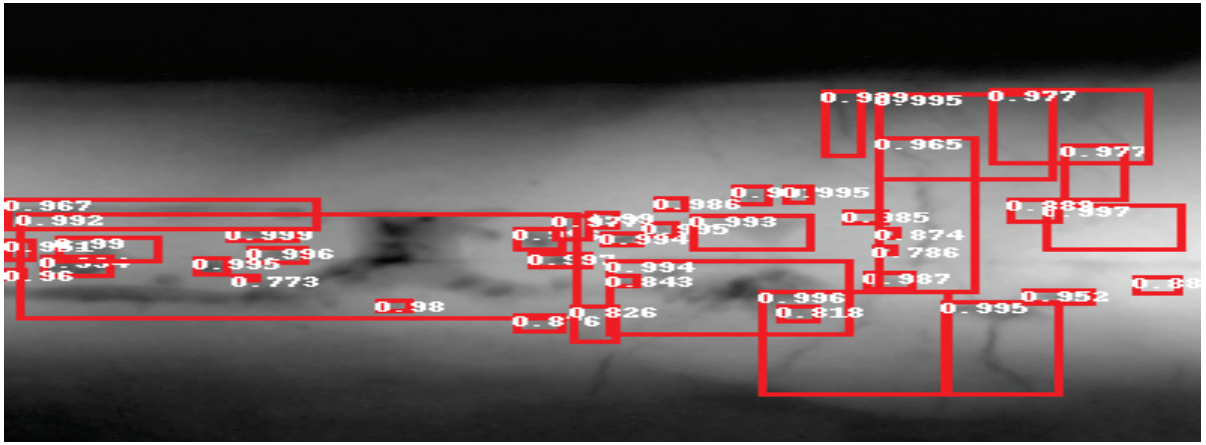


Detection

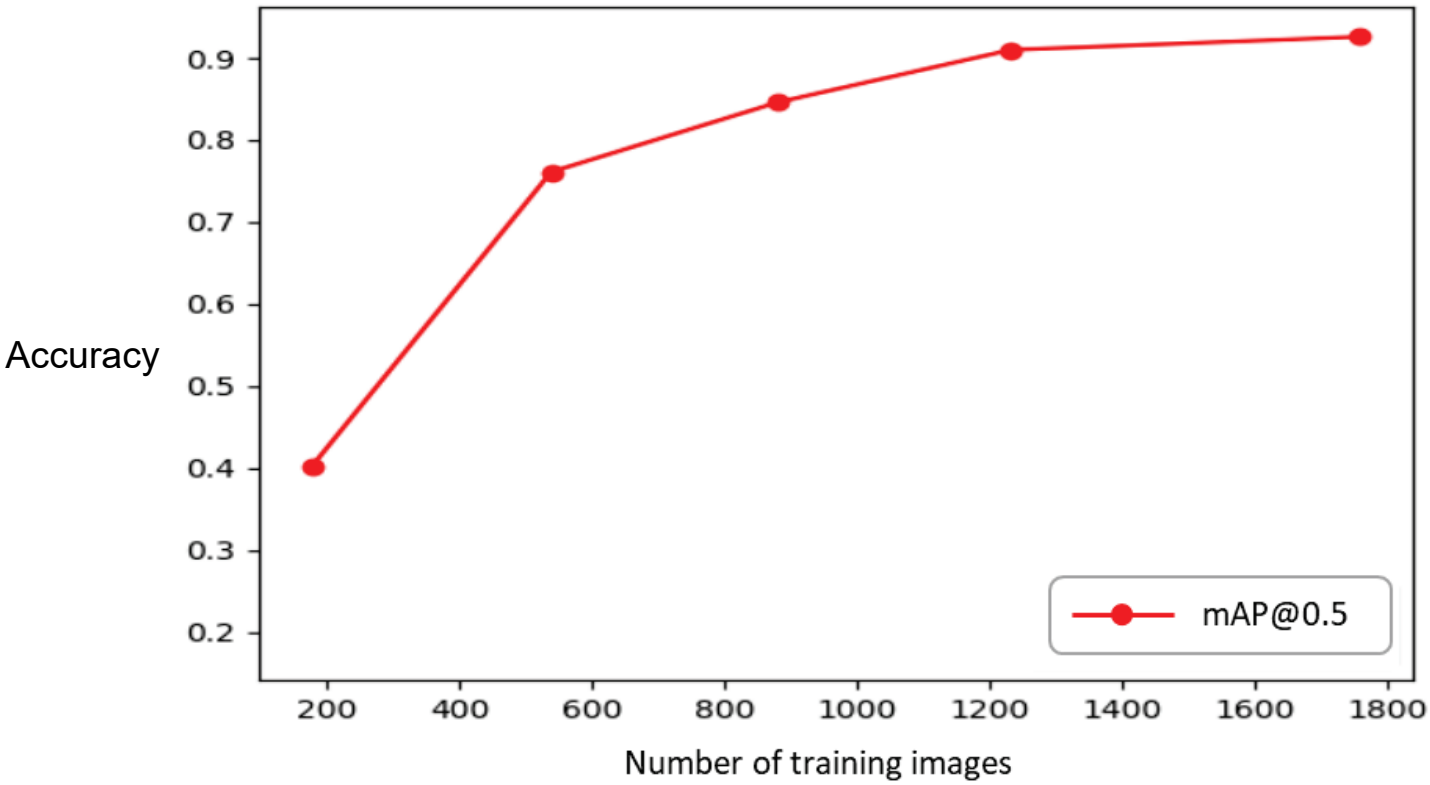
Ground Truth



Model Detection



Accuracy

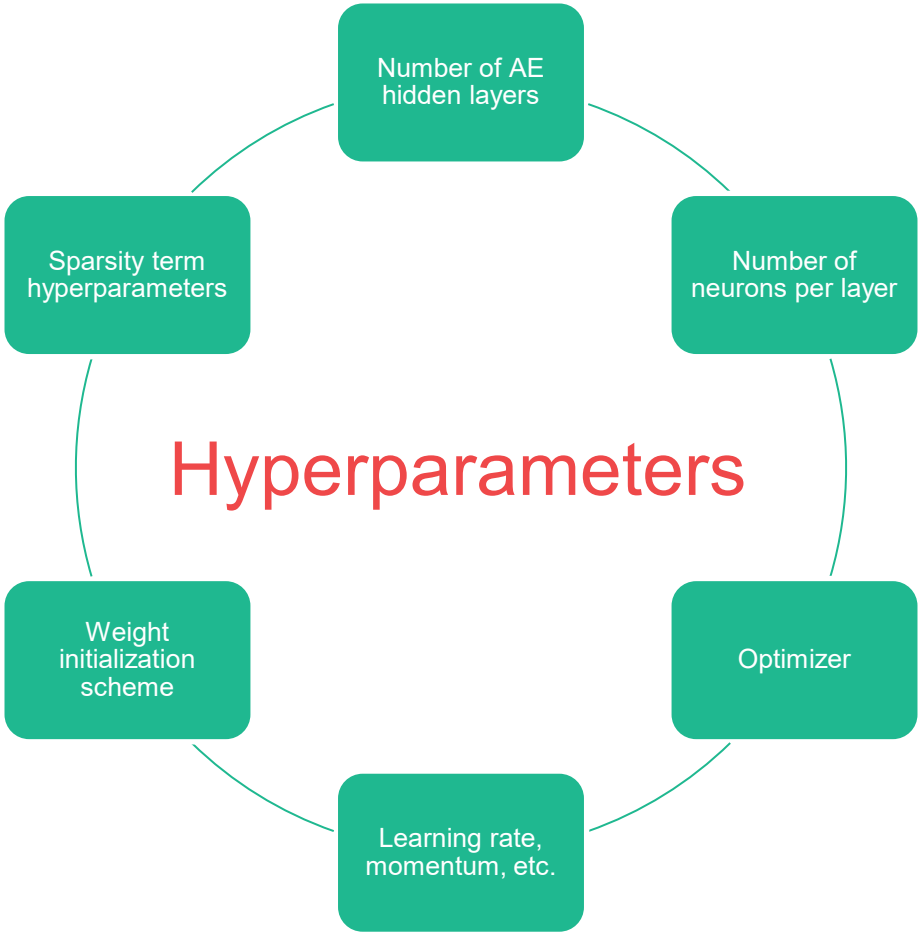


A relatively **large** number of images is needed for **satisfactory results**

Thank you!

Case study: Setting the hyperparameters

The extracted features and their monotonicity is highly dependent on the hyperparameter setting.



Case study: Setting the hyperparameters

Setting the hyperparameters → An optimization problem

Objective:

Maximizing the monotonicity of the most monotonic extracted feature

Solutions:

- Trial and error
  - Tedious and time-consuming
- Exhaustive search
  - Not feasible

- Genetic Algorithm

- **Coevolutionary Algorithm (For details, read [1])**

[1] Ali Eftekhari Milani, Federico Antonello, Piero Baraldi, Enrico Zio, "A Coevolutionary Optimization Approach with Deep Sparse Autoencoder for the Extraction of Equipment Degradation Indicators", ESREL2020-PSAM15 Conference, 2020