Anomaly Detection and Domain Adaptation

Giacomo Boracchi, Francesco Trovò

June 3rd, 2020

Politecnico di Milano, DEIB

giacomo.boracchi@polimi.it

Outline

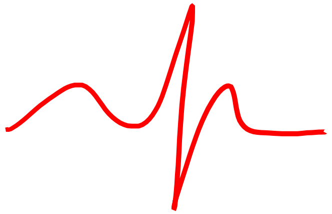
- Anomaly Detection Recap
- Anomaly Detection Methods
- Out of the «Random Variable World»
  - Reconstruction based-methods
  - Feature-based methods
- Domain Adaptation

Bonus slides

RV word

$x \sim \phi_0$

x



Statistical Approaches to Anomaly Detection

..a different monitoring problem

The Change/Anomaly Detection Problems

Change-detection problem:

*Given the previously estimated model, the arrival of new data invites the question: **"Is yesterday's model capable of explaining today's data?"***

Detecting process changes is important to understand the monitored phenomenon

Anomaly-detection problem:

*Locate those **samples that do not conform the normal ones** or a **model explaining normal ones***

Anomalies in data translate to significant information

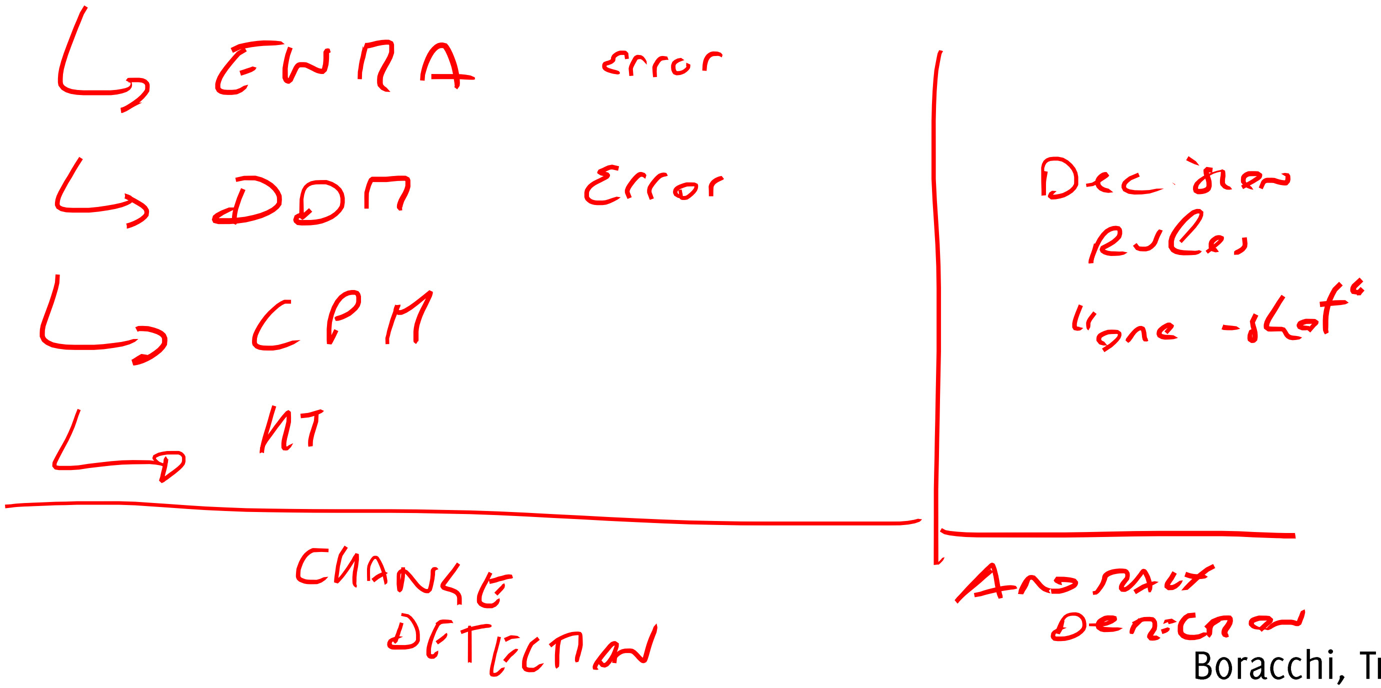
V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

C. J. Chu, M. Stinchcombe, H. White "*Monitoring Structural Change*" Econometrica Vol. 64, No. 5 (Sep., 1996), pp. 1045-1065.

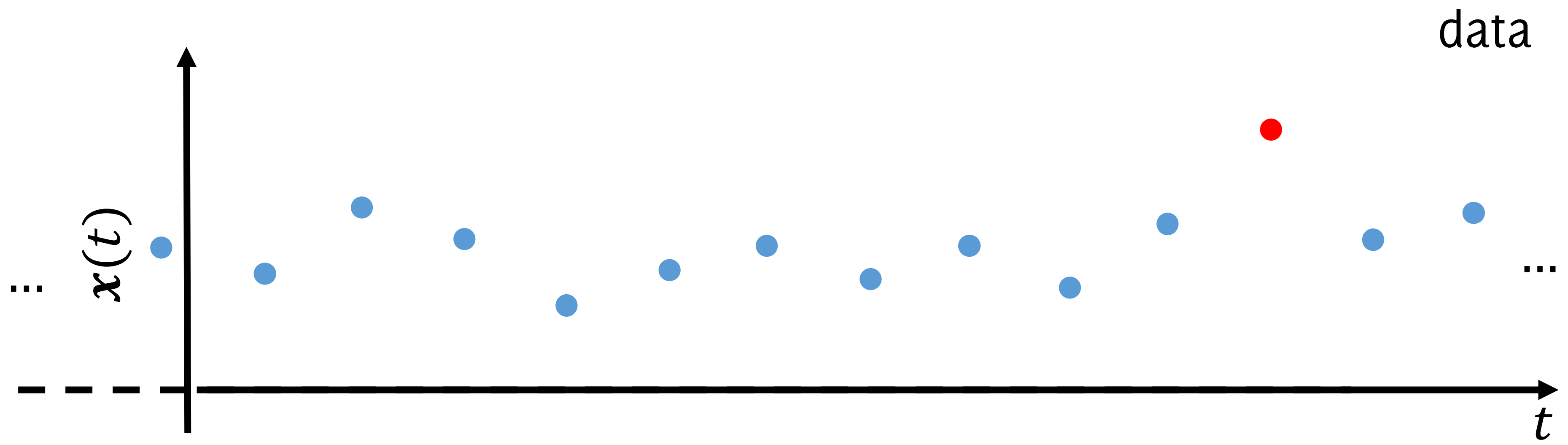
The Typical Anomaly Detection Solutions

Most algorithms are composed of:

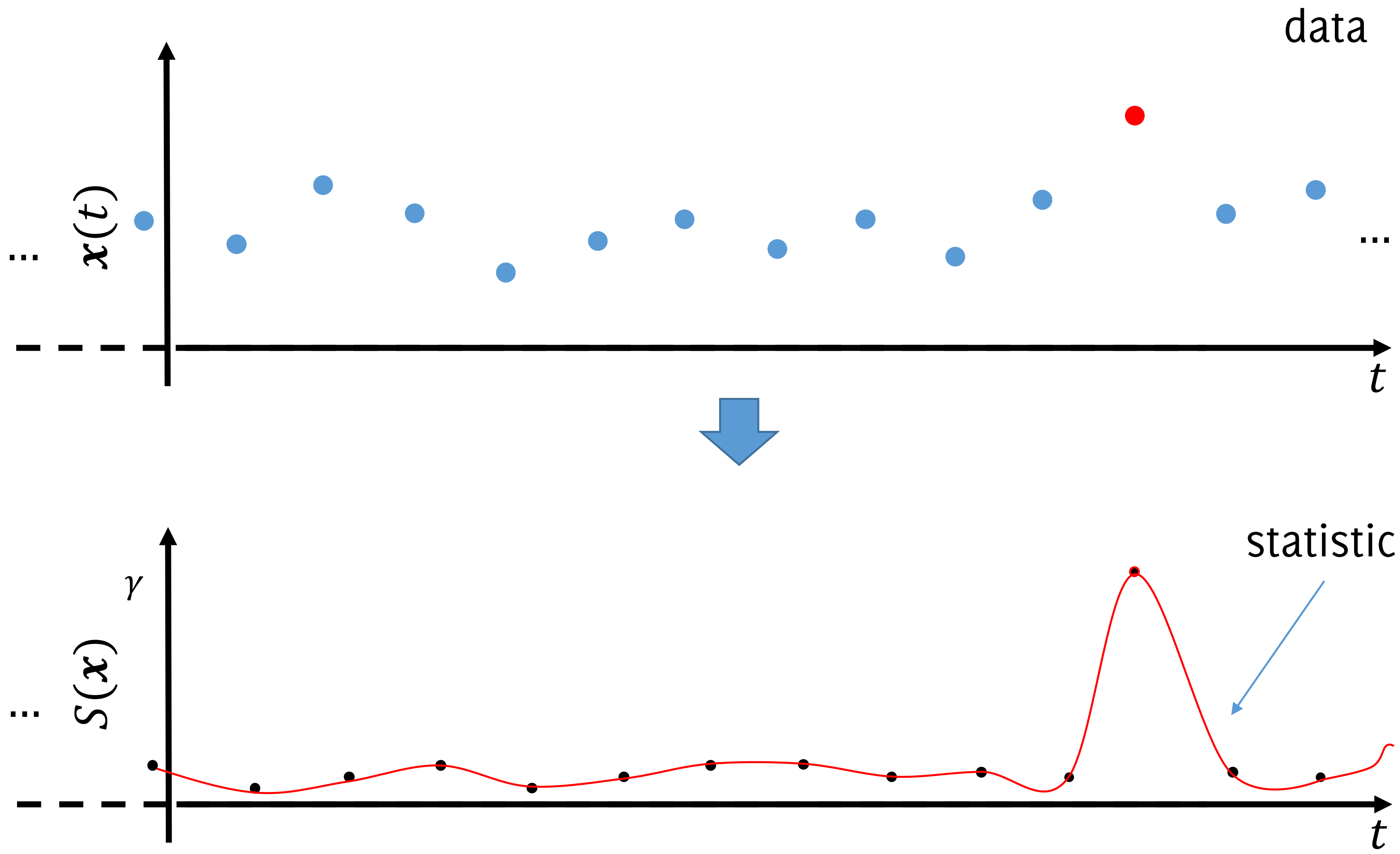
- A **statistic** that has a known response to normal data (e.g., the average, the sample variance, the log-likelihood, the confidence of a classifier, an "anomaly score"…)
- A **decision rule** to analyze the statistic (e.g., an adaptive threshold, a confidence region)



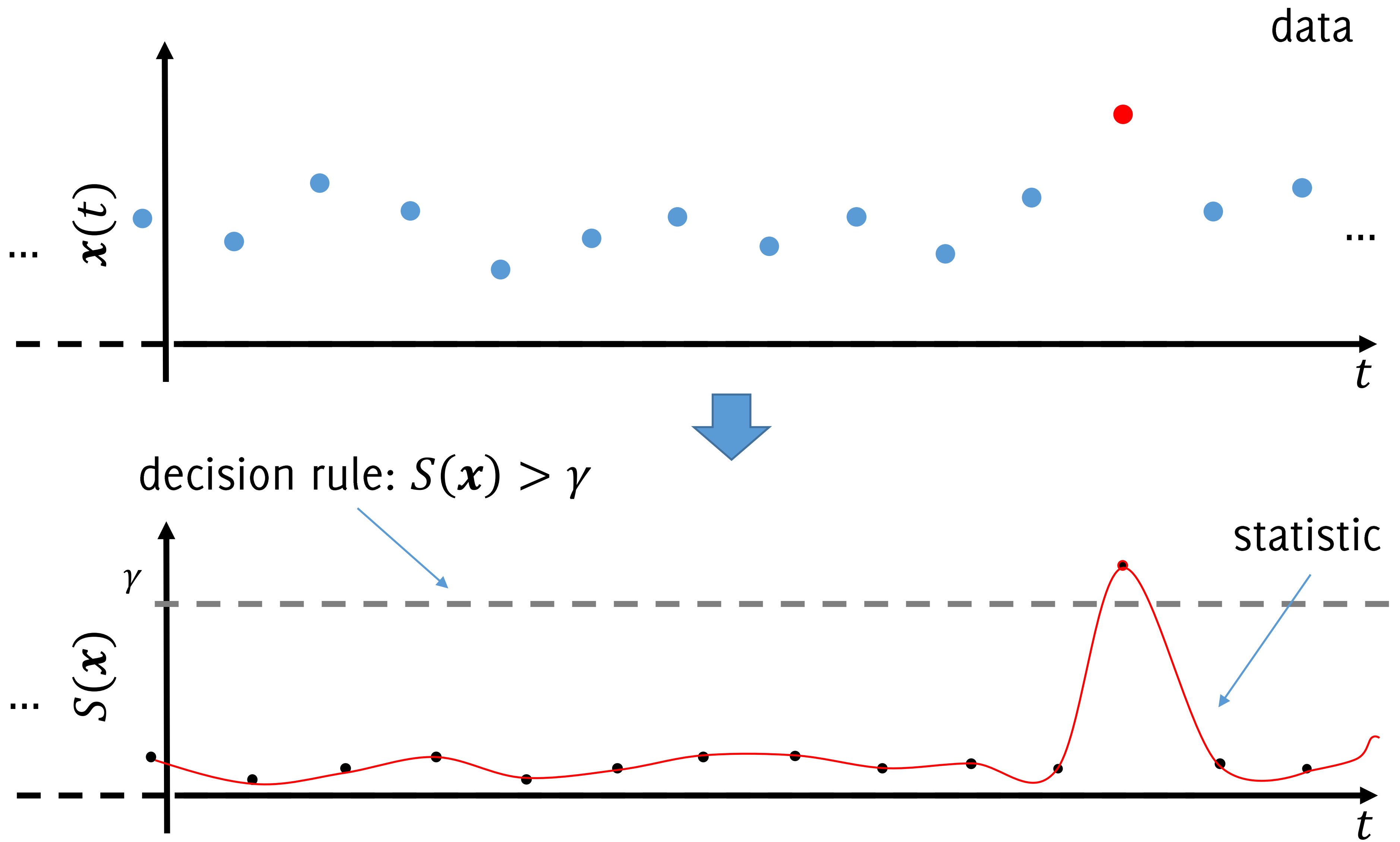
The Typical Anomaly Detection Solutions



The Typical Anomaly Detection Solutions



The Typical Anomaly Detection Solutions



Performance Measures

Assessing performance of anomaly detection algorithms

Anomaly-detection Performance

Anomaly detection performance:

- True positive rate: $TPR = \frac{\#\{\text{anomalies detected}\}}{\#\{\text{anomalies}\}}$ Recall
- False positive rate: $FPR = \frac{\#\{\text{normal samples detected}\}}{\#\{\text{normal samples}\}}$

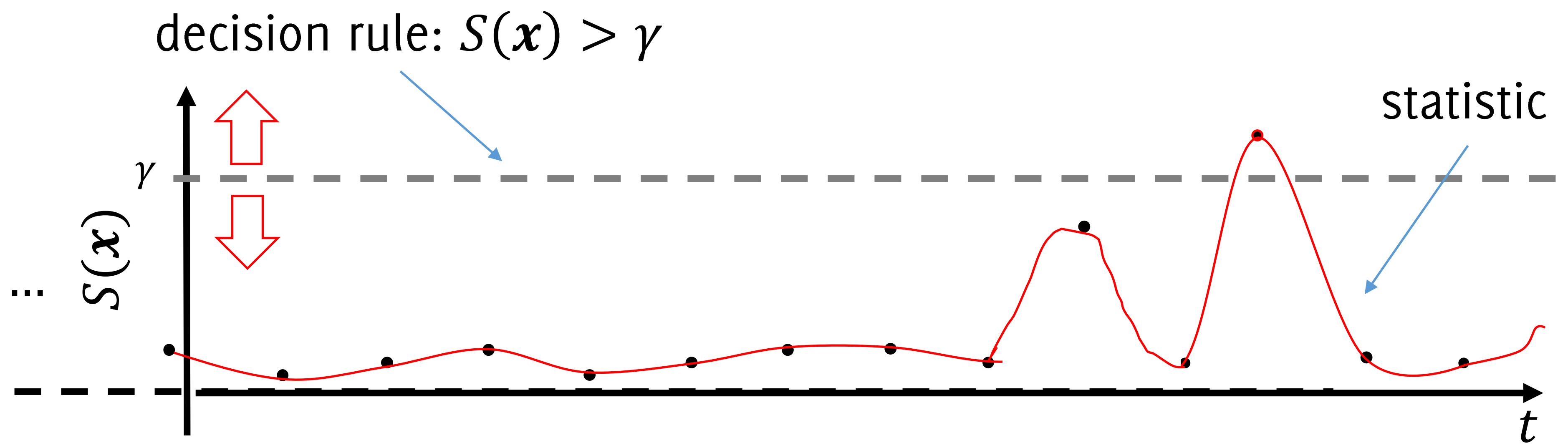
You have probably also heard of

- False negative rate (or miss-rate): $FNR = 1 - TPR$
- True negative rate (or specificity): $TNR = 1 - FPR$
- Precision on anomalies: $\frac{\#\{\text{anomalies detected}\}}{\#\{\text{detections}\}}$
- Recall on anomalies (or sensitivity, hit-rate): TPR

TPR and FPR Trade-off

There is always a **trade-off between TPR and FPR** (and similarly for derived quantities), which is ruled by algorithm parameters

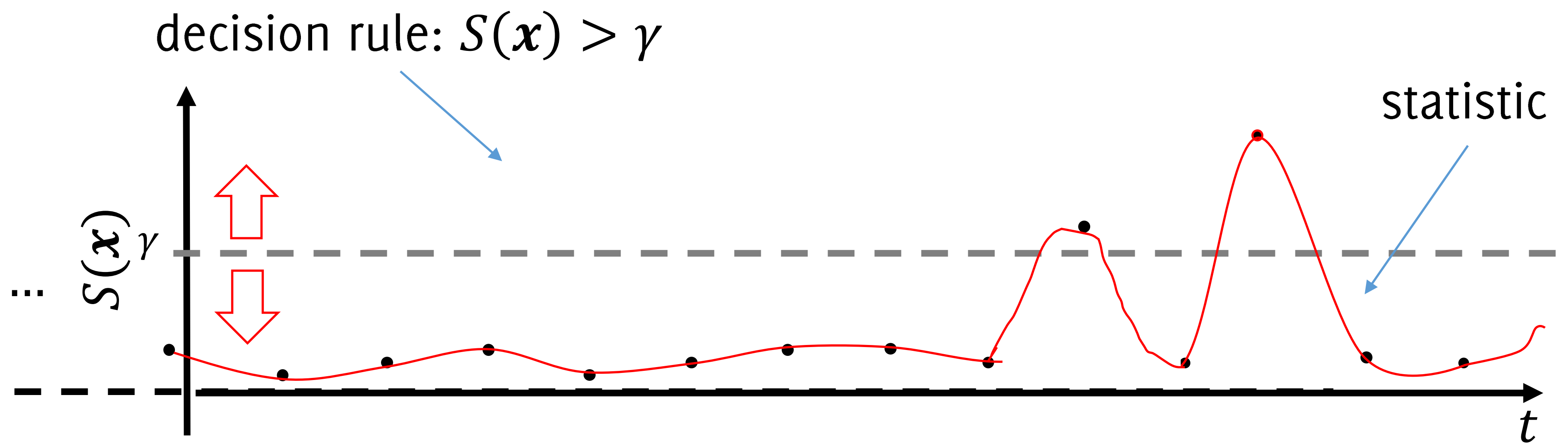
By changing γ performance changes (e.g. true positive increases but also false positives do)



TPR and FPR Trade-off

There is always a **trade-off between $\boldsymbol{TPR}$ and $\boldsymbol{FPR}$** (and similarly for derived quantities), which is ruled by algorithm parameters

By changing γ performance changes (e.g. true positive increases but also false positives do)



Anomaly-detection Performance

There is always a **trade-off between $\boldsymbol{TPR}$ and $\boldsymbol{FPR}$** (and similarly for derived quantities), which is ruled by algorithm parameters

Thus, to correctly assess performance it is necessary to consider at least **two indicators** (e.g., TPR, FPR)

Indicators combining both TPR and FPR:

Accuracy = $\frac{\#\{\text{anomalies detected}\} + \#\{\text{normal samples not detected}\}}{\#\{\text{samples}\}}$

F1 score = $\frac{2\#\{\text{anomalies detected}\}}{\#\{\text{detections}\} + \#\{\text{anomalies}\}}$

These equal 1 in case of “ideal detector” which detects all the anomalies and has no false positives

Anomaly-detection Performance

Comparing different methods might be tricky since we have to make sure that both have been configured in their best conditions

Testing a large number of parameters lead to the **ROC** (receiver operating characteristic) **curve**

The ideal detector would achieve:

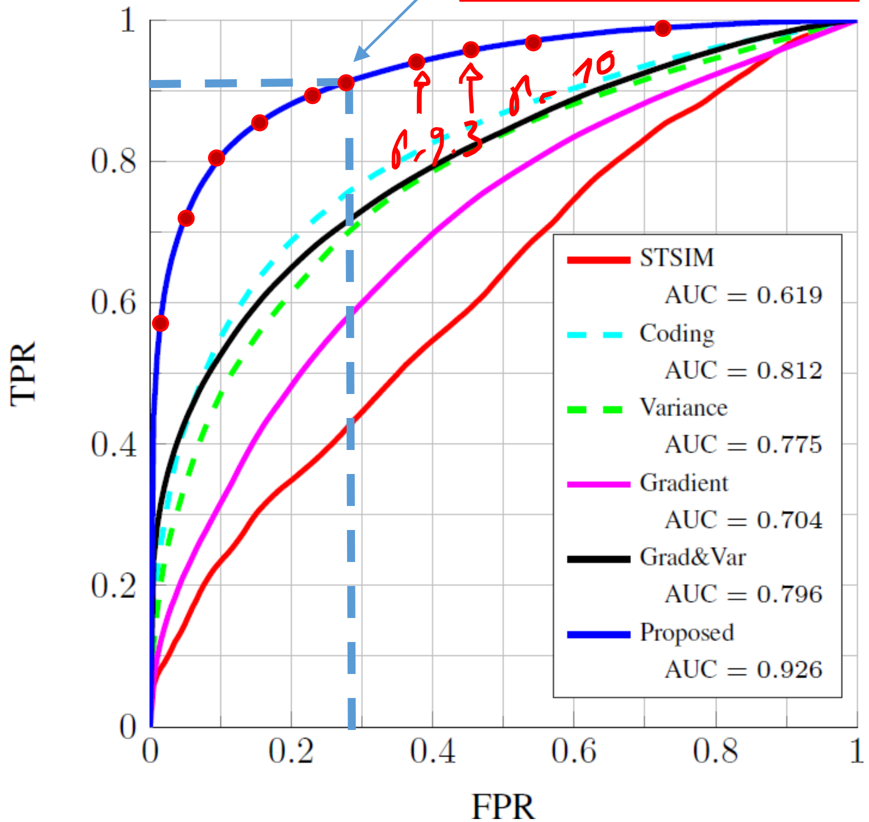
- $FPR = 0\%$,
- $TPR = 100\%$

Thus, the closer to (0,1) the better

The largest the **Area Under the Curve** (AUC), the better

The optimal parameter is the one yielding the point closest to (0,1)

(FPR, TPR) for a specific parameter



Anomaly detection approaches

…when ϕ_0 and ϕ_1 are unknown

Anomaly detection when $\boldsymbol{\phi_0}$ and $\boldsymbol{\phi_1}$ are unknown

Most often, **only a training set** TR is **provided:**

There are three scenarios:

- **Supervised:** Both normal and anomalous training data are provided in TR.
- **Semi-Supervised:** Only normal training data are provided, i.e. no anomalies in TR.
- **Unsupervised:** TR is provided without label.

$x \sim \phi_0$ $\quad TR = \{x \sim \phi_0\}$ $\quad x_i \sim \phi_1 \nexists \phi_0$

V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

Anomaly detection when $\boldsymbol{\phi_0}$ and $\boldsymbol{\phi_1}$ are unknown

Most often, **only a training set** TR is **provided**:

There are three scenarios:

- **Supervised:** Both normal and anomalous training data are provided in TR.
- **Semi-Supervised:** Only normal training data are provided, i.e. no anomalies in TR.
- **Unsupervised:** TR is provided without label.

V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

Supervised anomaly detection – disclaimer

Most papers and reviews agree that **supervised methods have not to be considered part of anomaly detection**, because:

- Anomalies in general lacks of a statistical coherence
- Not (enough) training samples are provided for anomalies

$TR: \{x \sim p_0,\ x \sim p_1\}$

However,

$TR = \{(x, y),\ x \in \mathbb{R}^d,\ y \in \{0,1\}\}$ $\quad y \in \{0,1\}$

- **Some supervised problems are often referred to as «detection»**, in case of severe class imbalance (e.g. fraud detection)
- Supervised models can be transferred in unsupervised methods, in particular for deep learning

T. Ehret, A. Davy, JM Morel, M. Delbracio *"Image Anomalies: A Review and Synthesis of Detection Methods"*, Journal of Mathematical Imaging and Vision, 1-34

Supervised anomaly detection - solutions

In **supervised methods** training data are annotated and divided in normal $(+)$ and anomalies $(-)$:

$$TR = \{(\boldsymbol{x}(t), y(t)), \qquad t < t_0, \boldsymbol{x} \in \mathbb{R}^d, y \in \{+, -\}\}$$

Solution:

- Train a two-class classifier to distinguish normal vs anomalous data.

During training:

- Learn a classifier $\mathcal{K}$ from TR.

During testing:

- Compute the classifier output $\mathcal{K}(\boldsymbol{x})$, or
- Set a threshold on the posterior $p_{\mathcal{K}}(-|\boldsymbol{x})$, or
- Select the k −most likely anomalies

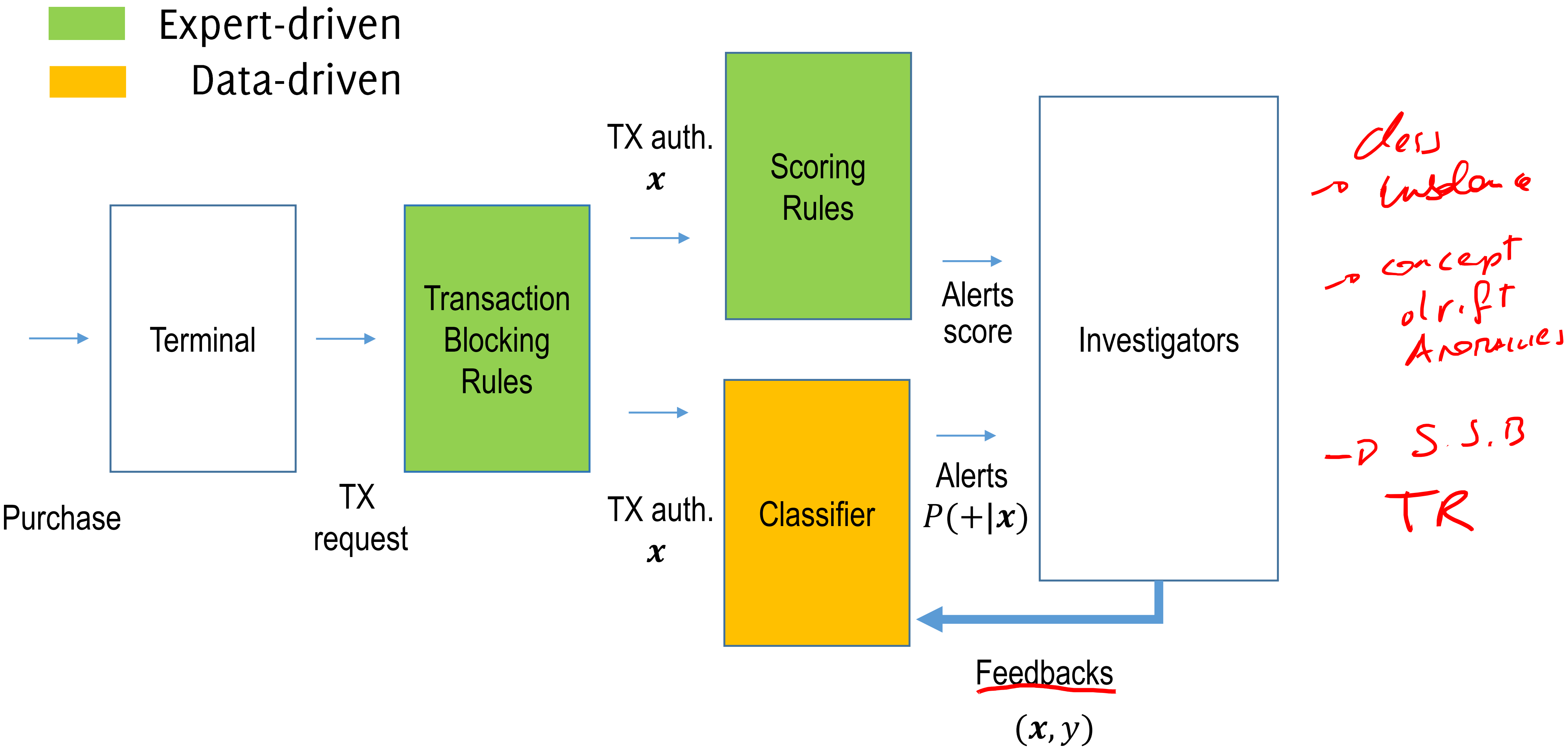
Supervised anomaly detection – challenges

These **classification problems are challenging** because these anomaly-detection settings typically imply:

- **Class Imbalance:** Normal data far outnumber anomalies
- **Concept Drift:** Anomalies might **evolve** over time, thus the few annotated anomalies might not be representative of anomalies occurring during operations
- **Selection Bias:** Training samples are typically selected through a **closed-loop and biased procedure**. Often only detected anomalies are annotated, and the vast majority of the stream remain unsupervised. This biases the selection of training samples.

Dal Pozzolo A., Boracchi G., Caelen O., Alippi C. and Bontempi G., ***"Credit Card Fraud Detection: a Realistic Modeling and a Novel Learning Strategy"*****, IEEE TNNL 2017, 14 pages**

Fraud Detection



Dal Pozzolo A., Boracchi G., Caelen O., Alippi C. and Bontempi G., *"Credit Card Fraud Detection: a Realistic Modeling and a Novel Learning Strategy"*, IEEE TNNL 2017, 14 pages

Supervised anomaly detection – An Example

This is **what typically happens in fraud detection**.

Class Imbalance:

- Frauds are typically less than 1% of genuine transactions

Concept Drift:

- Fraudster always implement new strategies

Sampling Selection Bias:

- Only alerted / reported transactions are controlled and annotated
- Old transactions that have not been disputed are considered genuine transactions

Dal Pozzolo A., Boracchi G., Caelen O., Alippi C. and Bontempi G., *"Credit Card Fraud Detection: a Realistic Modeling and a Novel Learning Strategy"*, IEEE TNNL 2017, 14 pages

Anomaly detection when $\boldsymbol{\phi_0}$ and $\boldsymbol{\phi_1}$ are unknown

Most often, **only a training set** TR is **provided**:

There are three scenarios:

- **Supervised:** Both normal and anomalous training data are provided in TR.
- **Semi-Supervised:** Only normal training data are provided, i.e. no anomalies in TR.
- **Unsupervised:** TR is provided without label.

V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

Practical Operating conditions

In semi-supervised methods the TR is composed of normal data

$$TR = \{x(t), t < t_0, x \sim \phi_0\}$$

There are many reasons to opt for a semi-supervised / unsupervised approach

- **Normal data are easy to gather.** A training set of normal signals denoted as TR is provided
- **Anomalous** / changed data are **difficult to collect**
- Training examples in TR might not be **representative of all the possible anomalies** / changes that can occur
- **In some cases** TR is not labeled, but it is reasonable to assume that **normal data are the vast majority**

V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.
Pimentel, M. A., Clifton, D. A., Clifton, L., Tarassenko, L. *"A review of novelty detection"* Signal Processing, 99, 215-249 (2014)

Semi-supervised Anomaly-Detection Methods

In semi-supervised methods the TR is composed of normal data

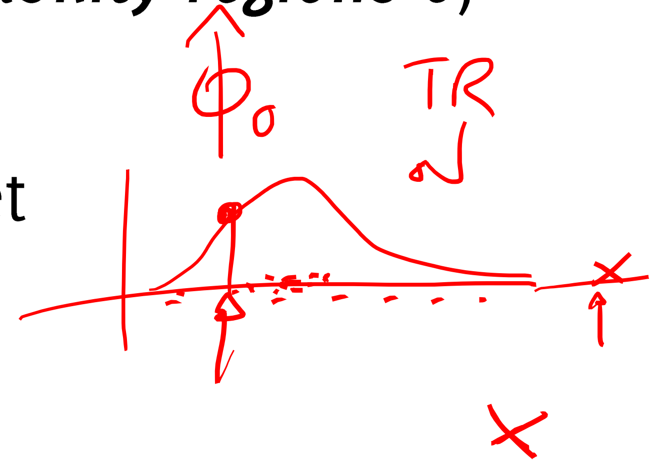
$$TR = \{x(t), t < t_0, x \sim \phi_0\}$$

Moreover… all in all… it is sometimes **safer** to **detect any data departing from** the **normal conditions**

Semi-supervised anomaly-detection methods are also referred to as **novelty-detection methods**

Density-based methods

Density-Based Methods: ***Normal*** *data occur in* ***high probability regions*** *of a stochastic model, while* ***anomalies*** *occur in the* ***low probability regions*** *of the model*



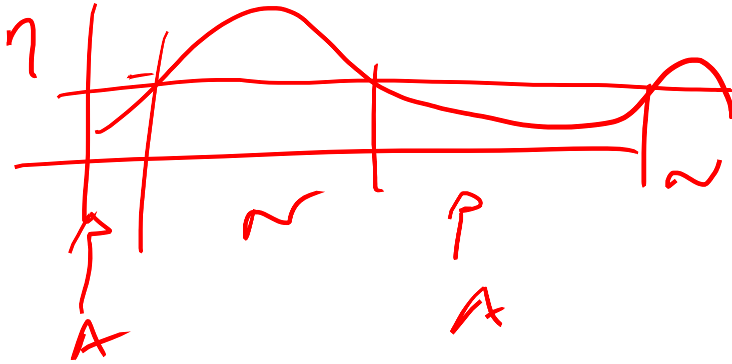
During training: $\hat{\phi}_0$ can be **estimated** from the training set

$$TR = \{x(t), t < t_0, x \sim \phi_0\}$$

- parametric models (e.g., Gaussian mixture models)
- nonparametric models (e.g. KDE, histograms)

During testing:

- Anomalies are detected as data yielding $\hat{\phi}_0(\boldsymbol{x}) < \eta$



V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

Density-based methods

Advantages:

- $\hat{\phi}_0(\boldsymbol{x})$ indicates how safe a detection is (like a p-value)
- If the density estimation process is robust to outliers, it is possible to tolerate few anomalous samples in TR
- **Histograms are simple to compute** in relatively small dimensions

Challenges:

- **It is challenging to fit models for high-dimensional data**
- Histograms traditionally suffer of **curse of dimensionality** when d increases
- Often the **1D histograms** of the marginals are monitored, **ignoring the correlations** among components

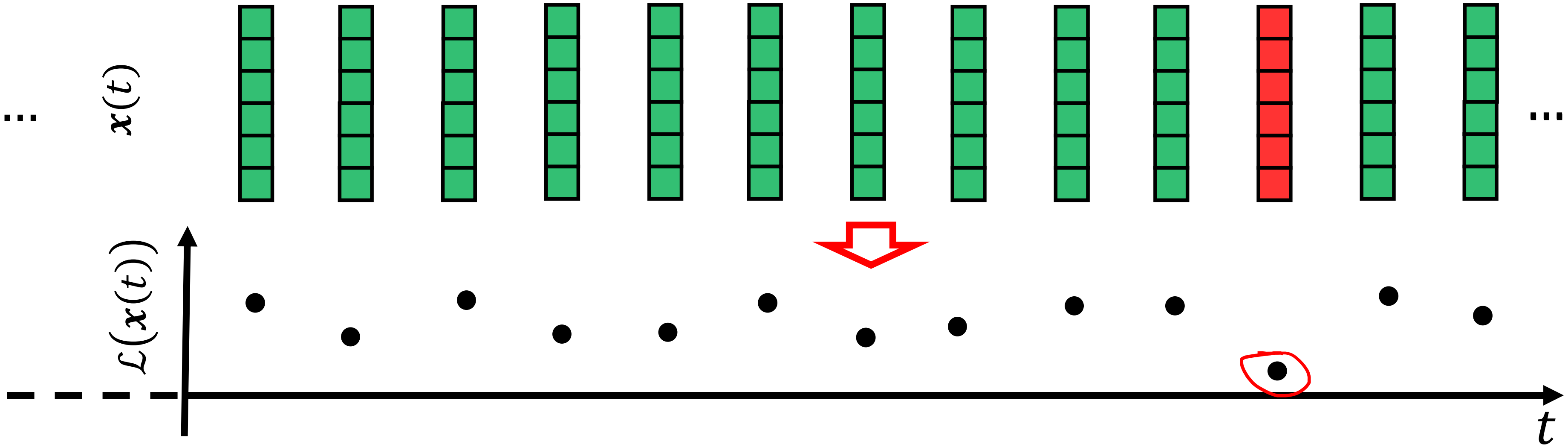
V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

Density-based methods: Monitoring the Log-likelihood

Monitoring the log-likelihood of data w.r.t $\hat{\phi}_0$ allow to address anomaly-detection problem in multivariate data

1. During training, estimate $\hat{\phi}_0$ from TR
2. During testing, compute
$$\mathcal{L}\big(\boldsymbol{x}(t)\big) = \log(\hat{\phi}_0(\boldsymbol{x}(t)))$$
3. Monitor $\{\mathcal{L}\big(\boldsymbol{x}(t)\big),\ t = 1, \dots\}$

$\mathcal{L}(x(t)) < \gamma$



Density-based methods: Monitoring the Log-likelihood

Monitoring the log-likelihood of data w.r.t $\hat{\phi}_0$ allow to address anomaly-detection problem in multivariate data

1. During training, estimate $\hat{\phi}_0$ from TR
2. During testing, compute
$$\mathcal{L}\big(\boldsymbol{x}(t)\big) = \log(\hat{\phi}_0(\boldsymbol{x}(t)))$$
3. Monitor $\{\mathcal{L}\big(\boldsymbol{x}(t)\big),\ t = 1, \ldots\}$

This is quite a popular approach in either anomaly and change detection algorithms

L. I. Kuncheva, “Change detection in streaming multivariate data using likelihood detectors," IEEE TKDE 2013.

X. Song, M. Wu, C. Jermaine, and S. Ranka, “Statistical change detection for multidimensional data” KDD, 2007.

J. H. Sullivan and W. H. Woodall, “Change-point detection of mean vector or covariance matrix shifts using multivariate individual observations," IIE transactions, vol. 32, no. 6, 2000.

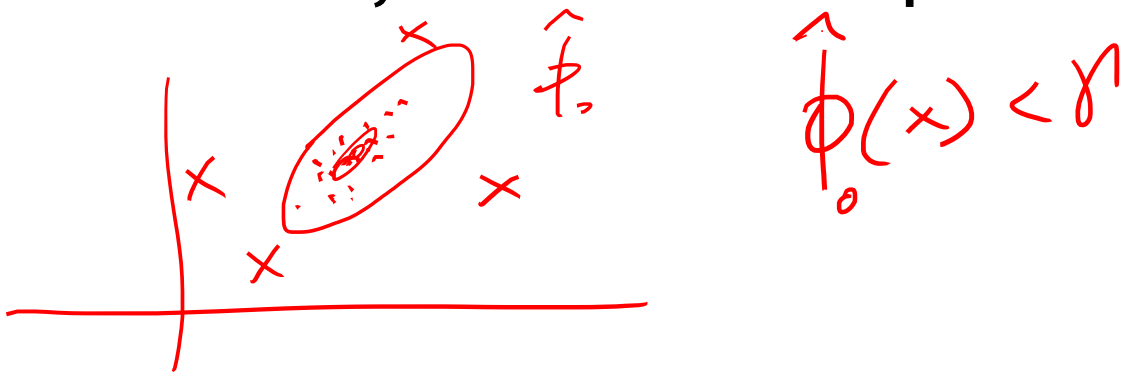
C. Alippi, G. Boracchi, D. Carrera, M. Roveri, "Change Detection in Multivariate Datastreams: Likelihood and Detectability Loss" IJCAI 2016,

Domain-based methods

Domain-based methods: *Estimate a boundary around normal* *data, rather than the density of normal data.*

A **drawback of density-estimation methods** is that they are meant to be accurate in high-density regions, while anomalies live in low-density ones.

One-Class SVM are domain-based methods defined by **the normal samples at the periphery of the distribution.**



Schölkopf, B., Williamson, R. C., Smola, A. J., Shawe-Taylor, J., Platt, J. C. "*Support Vector Method for Novelty Detection*". In NIPS 1999 (Vol. 12, pp. 582-588).

Tax, D. M., Duin, R. P. "*Support vector domain description*". Pattern recognition letters, 20(11), 1191-1199 (1999)

Pimentel, M. A., Clifton, D. A., Clifton, L., Tarassenko, L. "*A review of novelty detection*" Signal Processing, 99, 215-249 (2014)

One-class svm (Schölkopf et al. 1999)

Idea: define boundaries by estimating a **binary function** f that **captures regions of the input space where density is higher**.

As in support vector methods, f is **defined in the feature space** F and **decision boundaries are defined by a few support vectors** (i.e., a few normal data).

Let $\psi(\boldsymbol{x})$ the feature associated to $\boldsymbol{x}$, f is defined as

$$f(\boldsymbol{x}) = \text{sign}(< w, \psi(\boldsymbol{x}) > -\rho)$$

Where the hyperplane parameters w, ρ are optimized to yield a **function that is positive on most training samples.** Thus in the feature space, normal points can be separated from the origin.

A linear separation in the feature space corresponds to a **variety of nonlinear boundaries in the space of** $\boldsymbol{x}$.

Schölkopf, B., Williamson, R. C., Smola, A. J., Shawe-Taylor, J., Platt, J. C. "*Support Vector Method for Novelty Detection*". In NIPS 1999 (Vol. 12, pp. 582-588).

One-class svm (Tax and Duin 1999)

Boundaries of normal region can be also defined by an **hypersphere that, in the feature space, encloses most of the normal data.**

Similar detection formulas hold, **measuring the distance in the feature space from the sphere center for each** $\psi(\boldsymbol{x})$ for $\boldsymbol{x} \in TR$.

The function is always defined by a few support vectors.

Remarks: In both one-class approaches, the amount of samples that falls within the margin (outliers) is controlled by regularization parameters.

This parameter regulates the number of outliers in the training set and the detector sensitivity.

Tax, D. M., Duin, R. P. "*Support vector domain description*". Pattern recognition letters, 20(11), 1191-1199 (1999)

Anomaly detection when $\boldsymbol{\phi}_0$ and $\boldsymbol{\phi}_1$ are unknown

Most often, **only a training set** TR is **provided**:

There are three scenarios:

- **Supervised:** Both normal and anomalous training data are provided in TR.
- **Semi-Supervised:** Only normal training data are provided, i.e. no anomalies in TR.
- **Unsupervised:** TR is provided without label.

V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

Unsupervised anomaly-detection

The training set TR might contain **both normal and anomalous data.** However, **no labels** are provided

$$TR = \{x(t), t < t_0\}$$

Underlying assumption: *Anomalies are rare* *w.r.t. normal data TR*

One in principle could use:

- Density/Domain based methods that are **robust to outliers** can be applied in an unsupervised scenario
- Unsupervised methods can be improved whenever labels are available

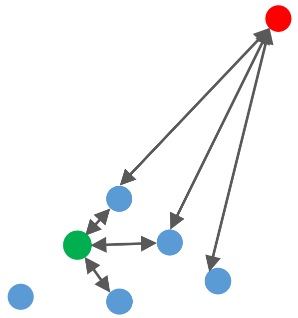
Distance-based methods

Distance-based methods: ***normal*** *data fall in* ***dense neighborhoods****, while* ***anomalies*** *are* ***far from their closest neighbors****.*

A critical aspect is the **choice of the similarity measure** to use.

Anomalies are detected by **monitoring**:

- **distance** between each data and its $\boldsymbol{k}$ **–nearest neighbor**



V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

Zhao, M., Saligrama, V. *"Anomaly detection with score functions based on nearest neighbor graphs"*. NIPS 2009

A. Zimek, E. Schubert, H. Kriegel. *"A survey on unsupervised outlier detection in high-dimensional numerical data"* SADM 2012.

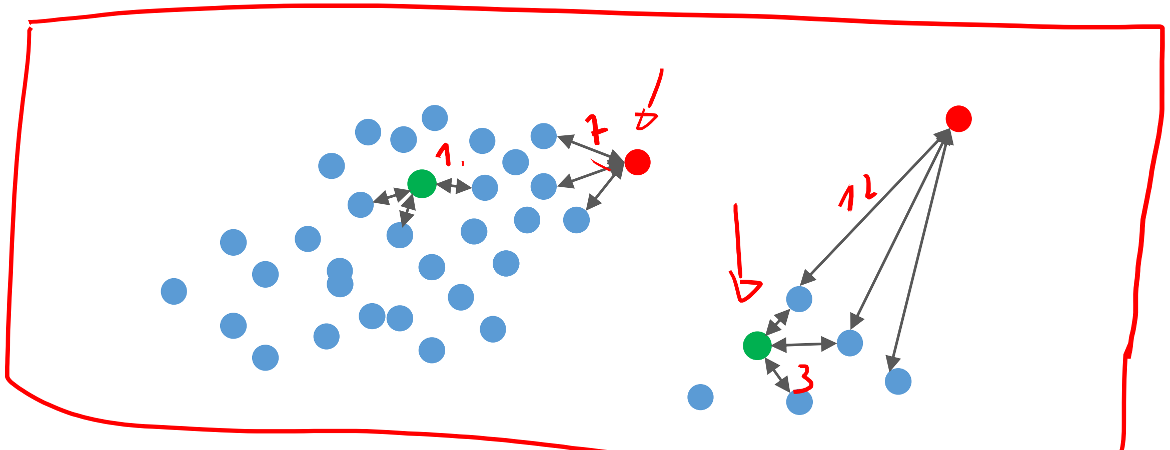
Distance-based methods

Distance-based methods: ***normal*** *data fall in* ***dense neighborhoods****, while* ***anomalies*** *are* ***far from their closest neighbors****.*

A critical aspect is the **choice of the similarity measure** to use.

Anomalies are detected by **monitoring:**

- **distance** between each data and its $\boldsymbol{k}$ **−nearest neighbor**
- the **above distance** considered **relatively to neighbors**



V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

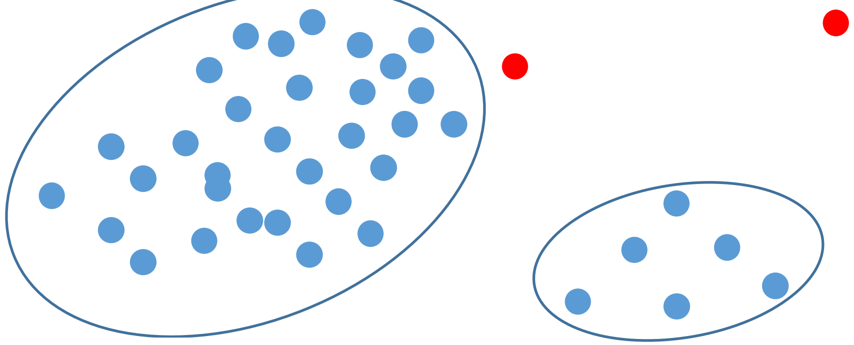
Distance-based methods

Distance-based methods: ***normal*** *data fall in* ***dense neighborhoods****, while* ***anomalies*** *are* ***far from their closest neighbors***.

A critical aspect is the **choice of the similarity measure** to use.

Anomalies are detected by **monitoring**:

- **distance** between each data and its $\boldsymbol{k}$ **−nearest neighbor**
- the **above distance** considered **relatively to neighbors** LOF
- whether they do not belong to **clusters**, or are at the cluster periphery, or belong to small and sparse clusters

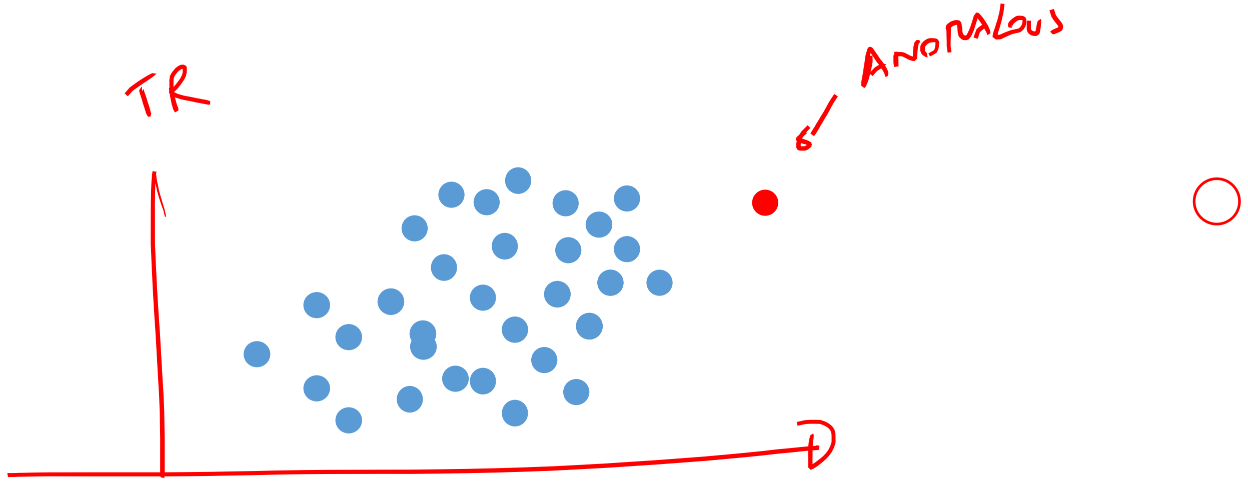


V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

IFOR: Isolation Forest

Builds upon the rationale that **«anomalies are easier to separate from the rest of normal data»**

This idea is implemented very efficiently through **a forest of binary trees** that are constructed via an iterative procedure

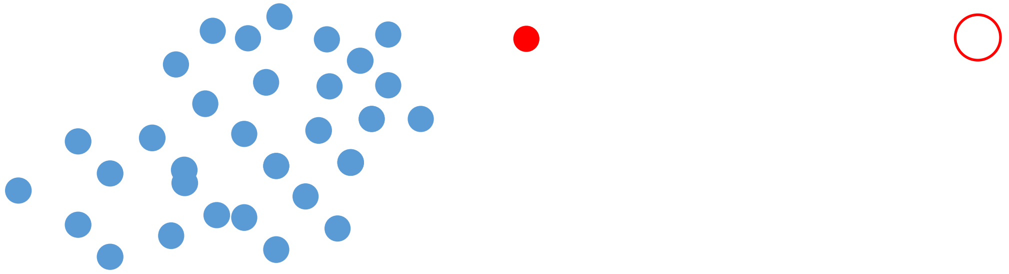


Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: Isolation Forest

Builds upon the rationale that **«anomalies are easier to separate from the rest of normal data»**

This idea is implemented very efficiently through **a forest of binary trees** that are constructed via an iterative procedure



Randomly choose

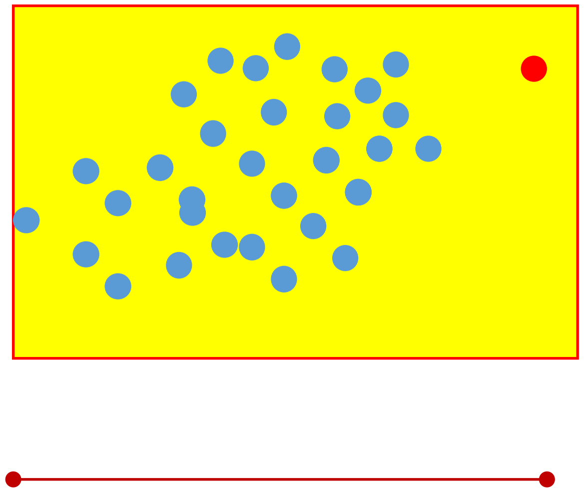
1. a component x_i

Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: Isolation Forest

Builds upon the rationale that «anomalies are easier to separate from the rest of normal data»

This idea is implemented very efficiently through a forest of binary trees that are constructed via an iterative procedure



Randomly choose

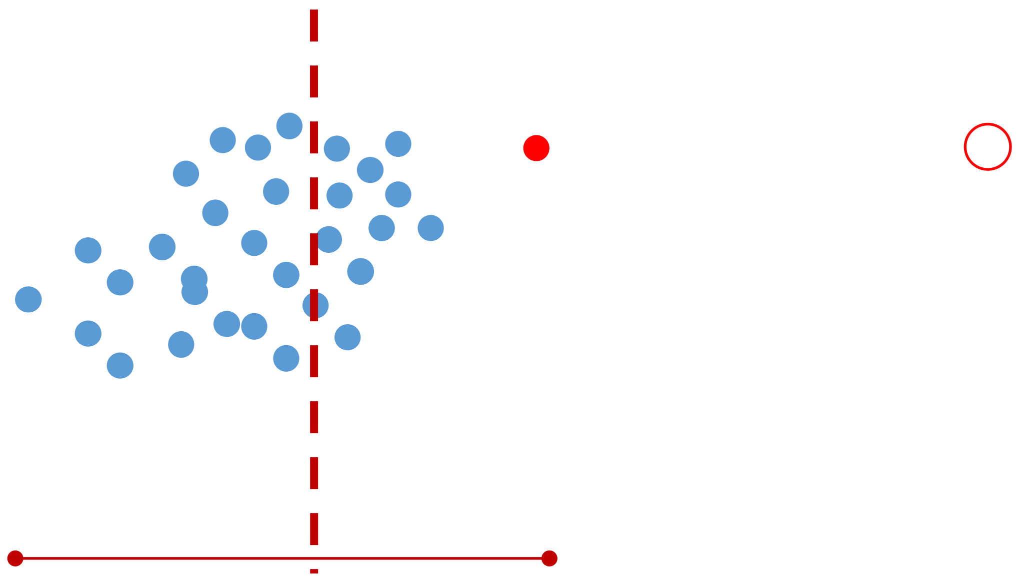
1. a component x_i

Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: Isolation Forest

Builds upon the rationale that «anomalies are easier to separate from the rest of normal data»

This idea is implemented very efficiently through a forest of binary trees that are constructed via an iterative procedure



Randomly choose

1. a component x_i
2. a value in the range of projections of TR over the i-th component

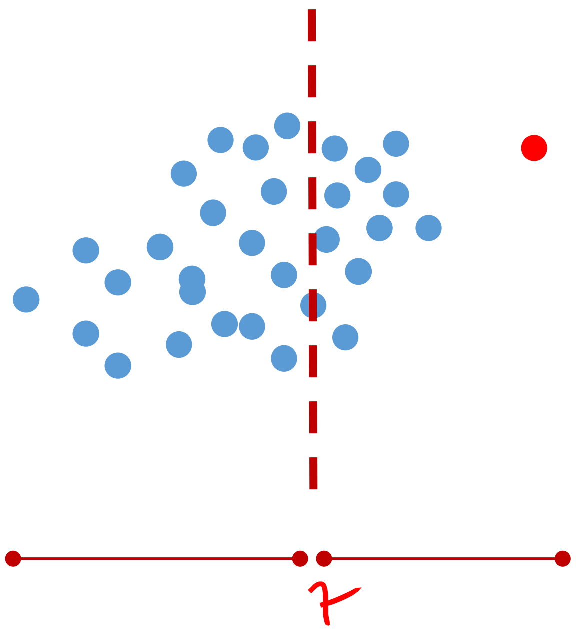
This yields a splitting

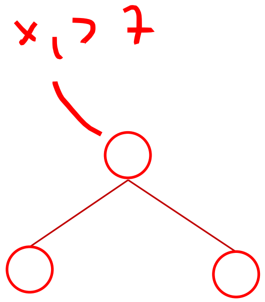
Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: Isolation Forest

Builds upon the rationale that «anomalies are easier to separate from the rest of normal data»

This idea is implemented very efficiently through a forest of binary trees that are constructed via an iterative procedure





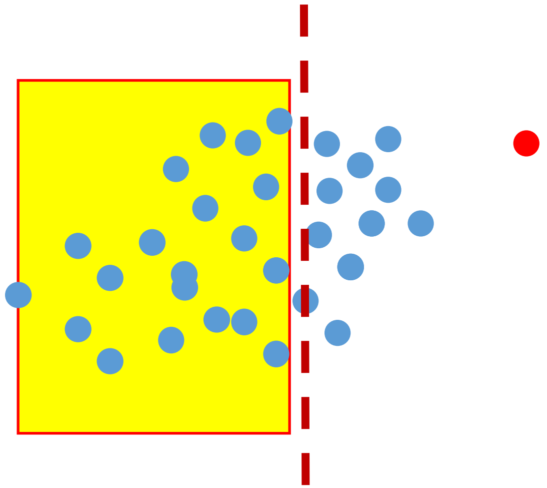
This yields a splitting criteria

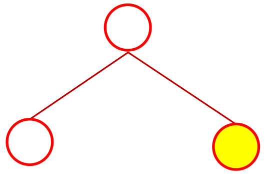
Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: Isolation Forest

Builds upon the rationale that «anomalies are easier to separate from the rest of normal data»

This idea is implemented very efficiently through a forest of binary trees that are constructed via an iterative procedure





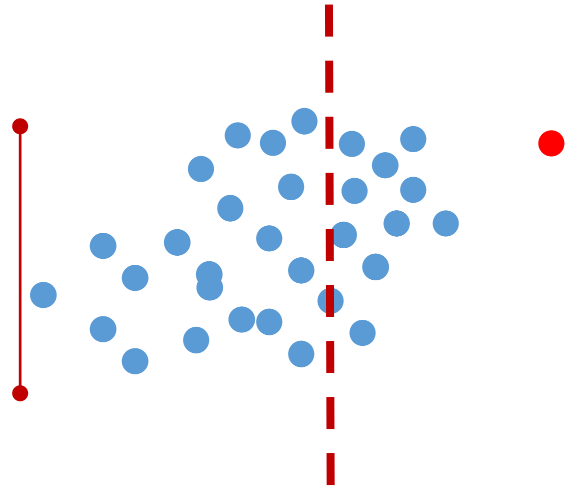
Repeat the procedure on each node:

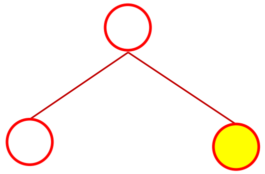
Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: Isolation Forest

Builds upon the rationale that «anomalies are easier to separate from the rest of normal data»

This idea is implemented very efficiently through a forest of binary trees that are constructed via an iterative procedure





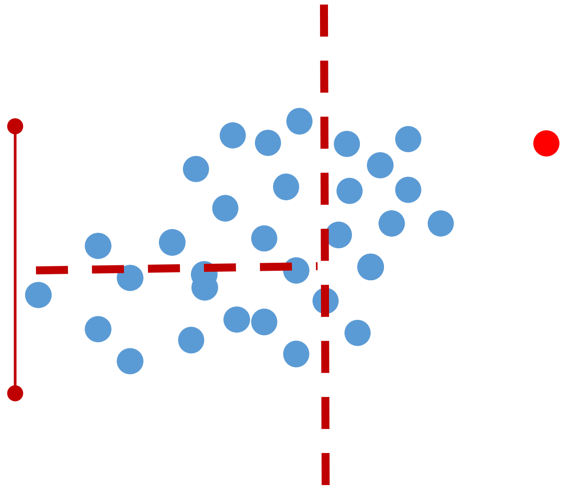
Repeat the procedure on each node: Randomly select a component

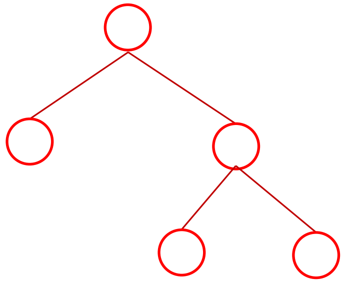
Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: Isolation Forest

Builds upon the rationale that «anomalies are easier to separate from the rest of normal data»

This idea is implemented very efficiently through a forest of binary trees that are constructed via an iterative procedure





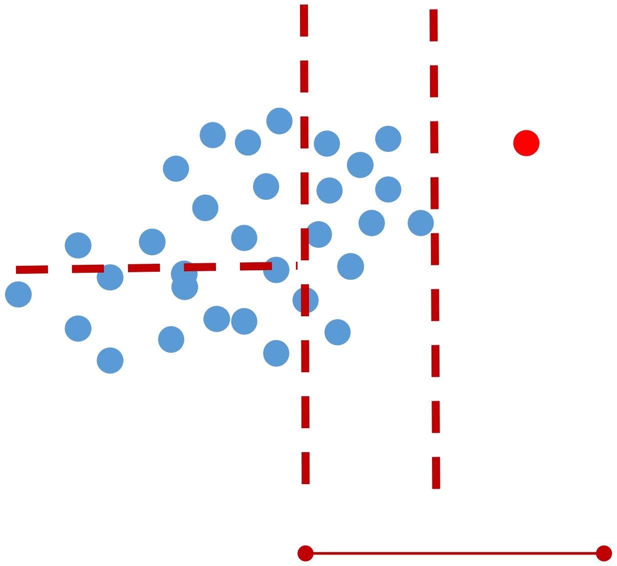
Repeat the procedure on each node: Randomly select a component and a cut point

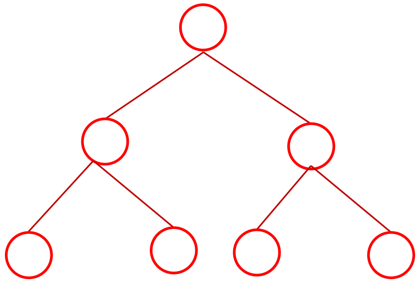
Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: Isolation Forest

Builds upon the rationale that «anomalies are easier to separate from the rest of normal data»

This idea is implemented very efficiently through a forest of binary trees that are constructed via an iterative procedure





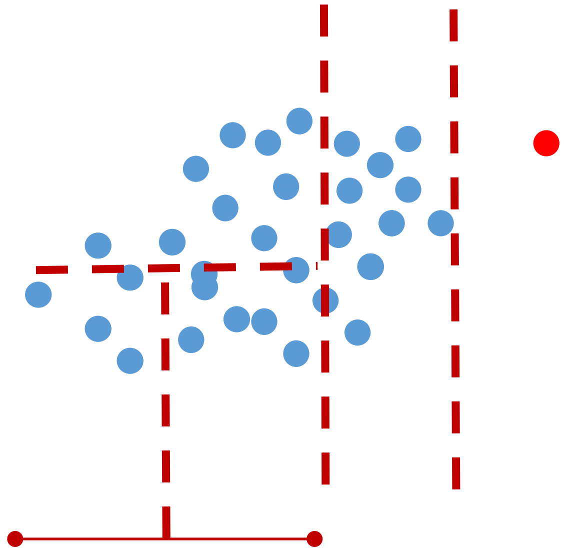
Randomly choose a component and a value within the range and define a splitting criteria

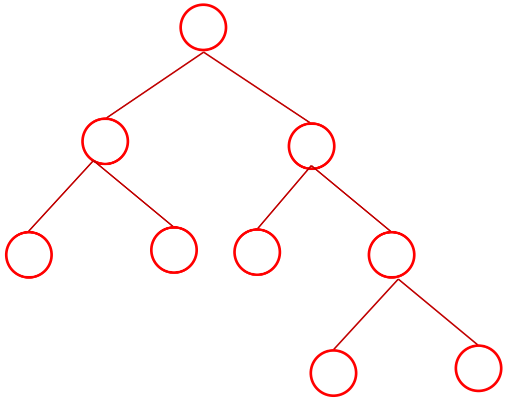
Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: Isolation Forest

Builds upon the rationale that «anomalies are easier to separate from the rest of normal data»

This idea is implemented very efficiently through a forest of binary trees that are constructed via an iterative procedure





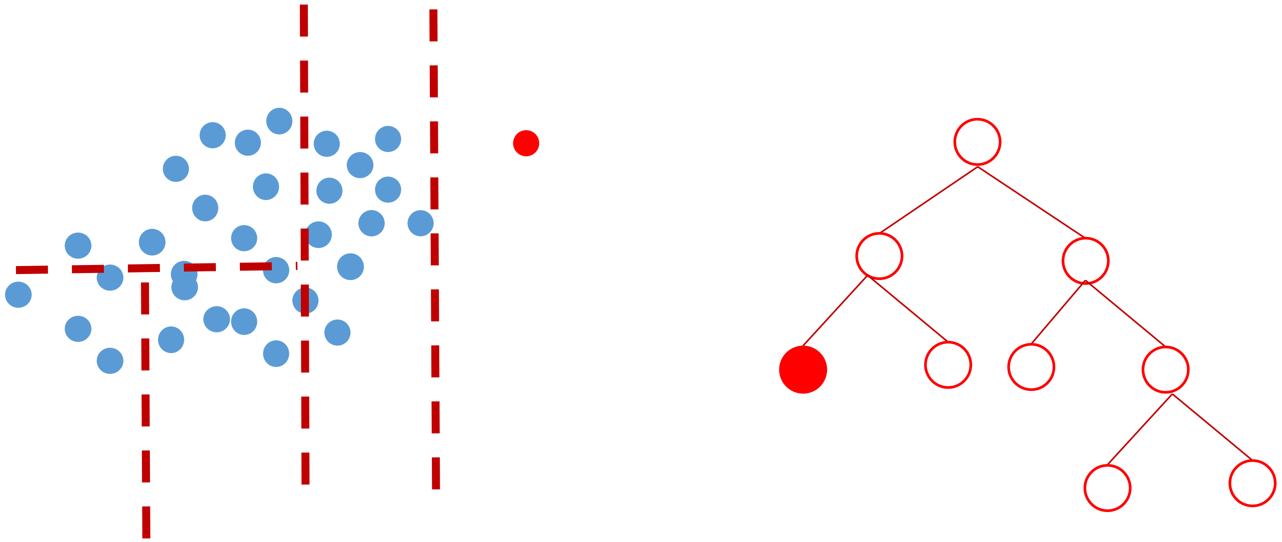
Repeat the procedure on the nodes:
Randomly select a component and a cut point

Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: Isolation Forest

Builds upon the rationale that «anomalies are easier to separate from the rest of normal data»

This idea is implemented very efficiently through a forest of binary trees that are constructed via an iterative procedure



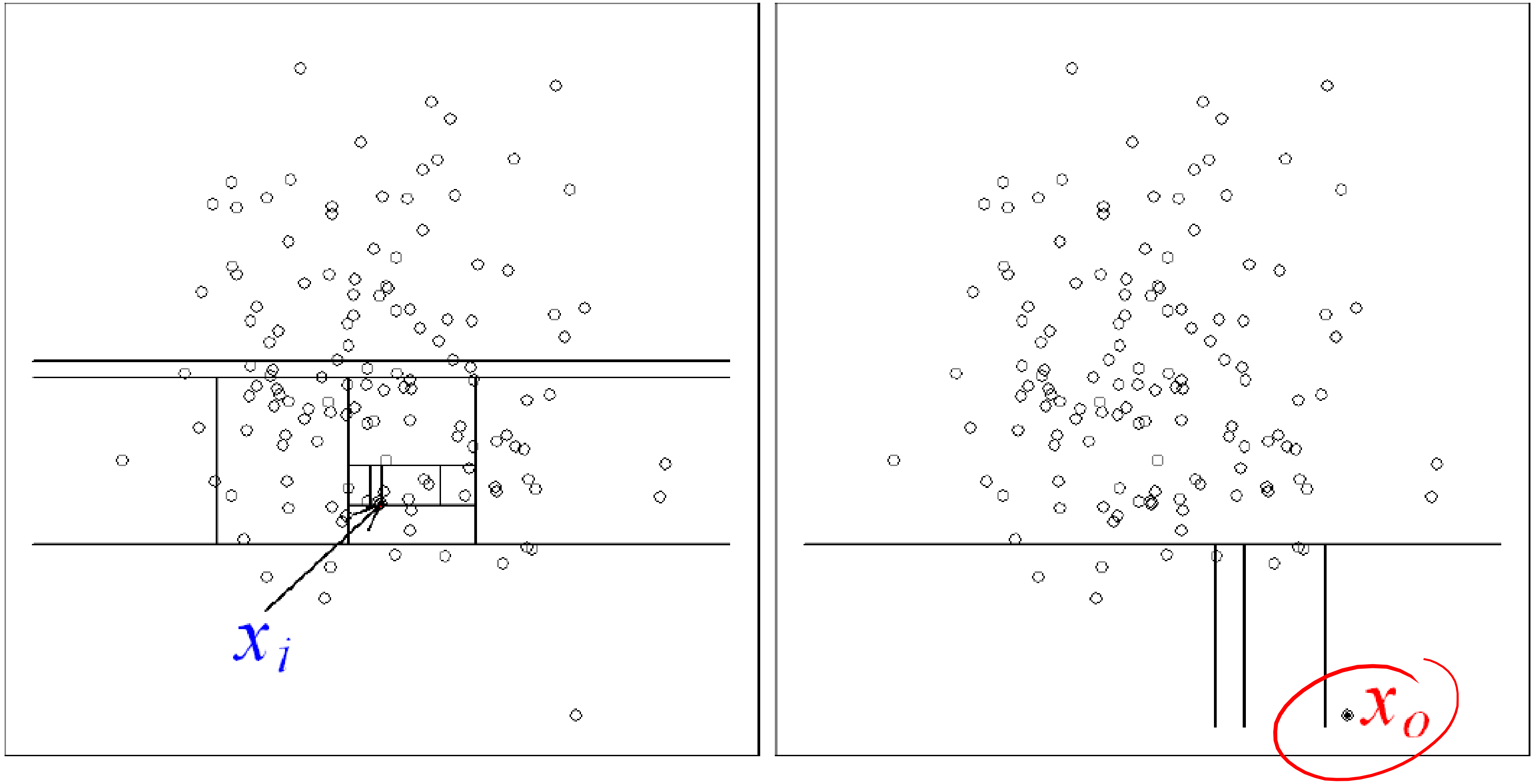
Anomalies lies in leaves close to the root.

Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: Isolation Forest

An anomalous point (x_0) can be easily isolated

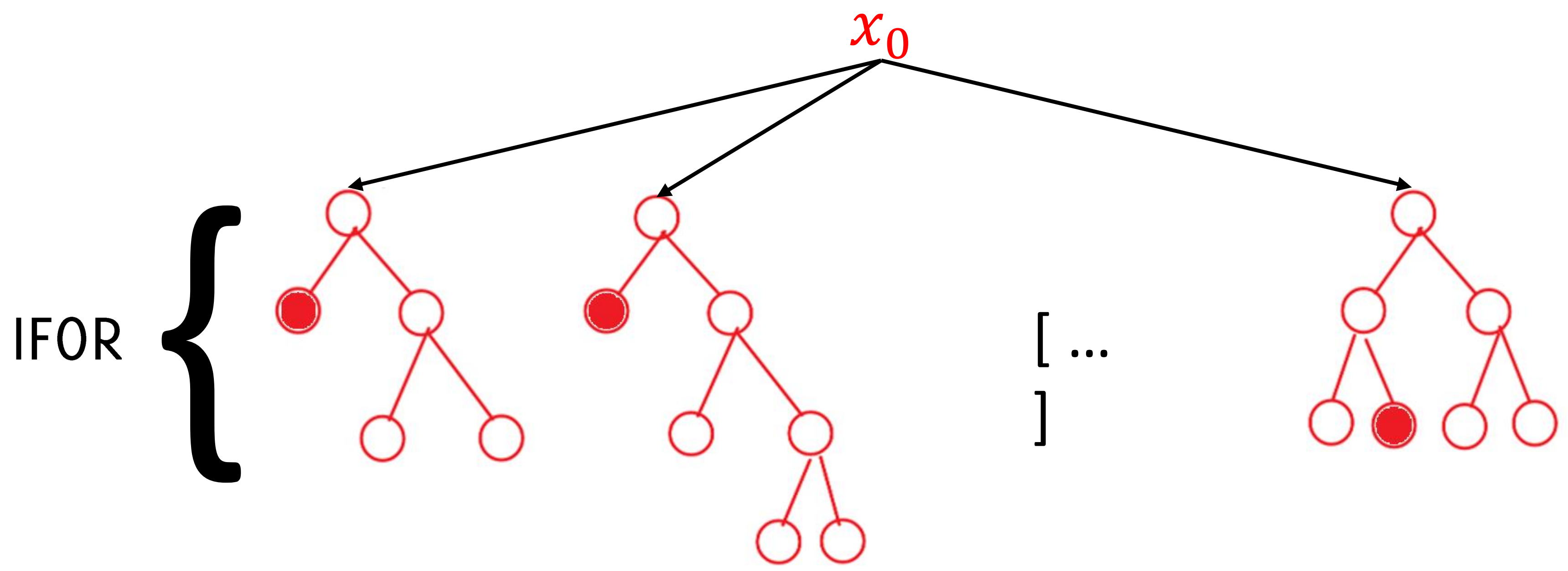
Genuine points (x_i) are instead difficult to isolate.



Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: Isolation Forest

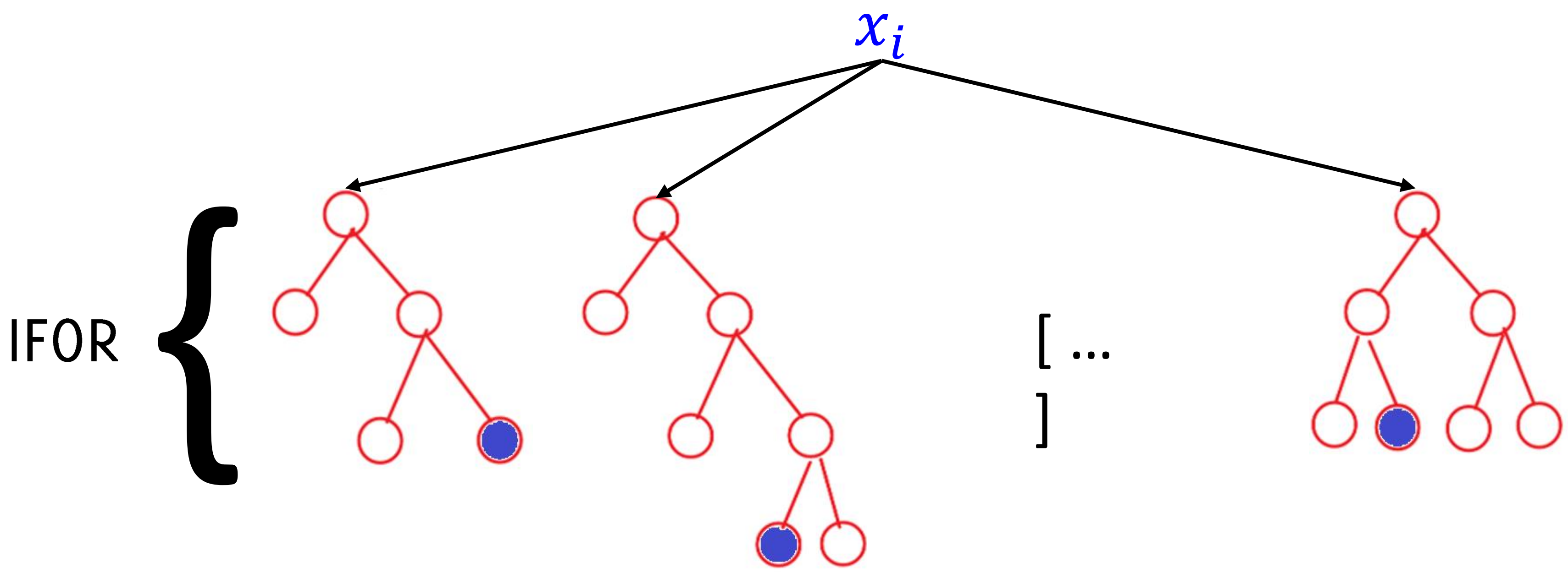
Anomalies



Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: Isolation Forest

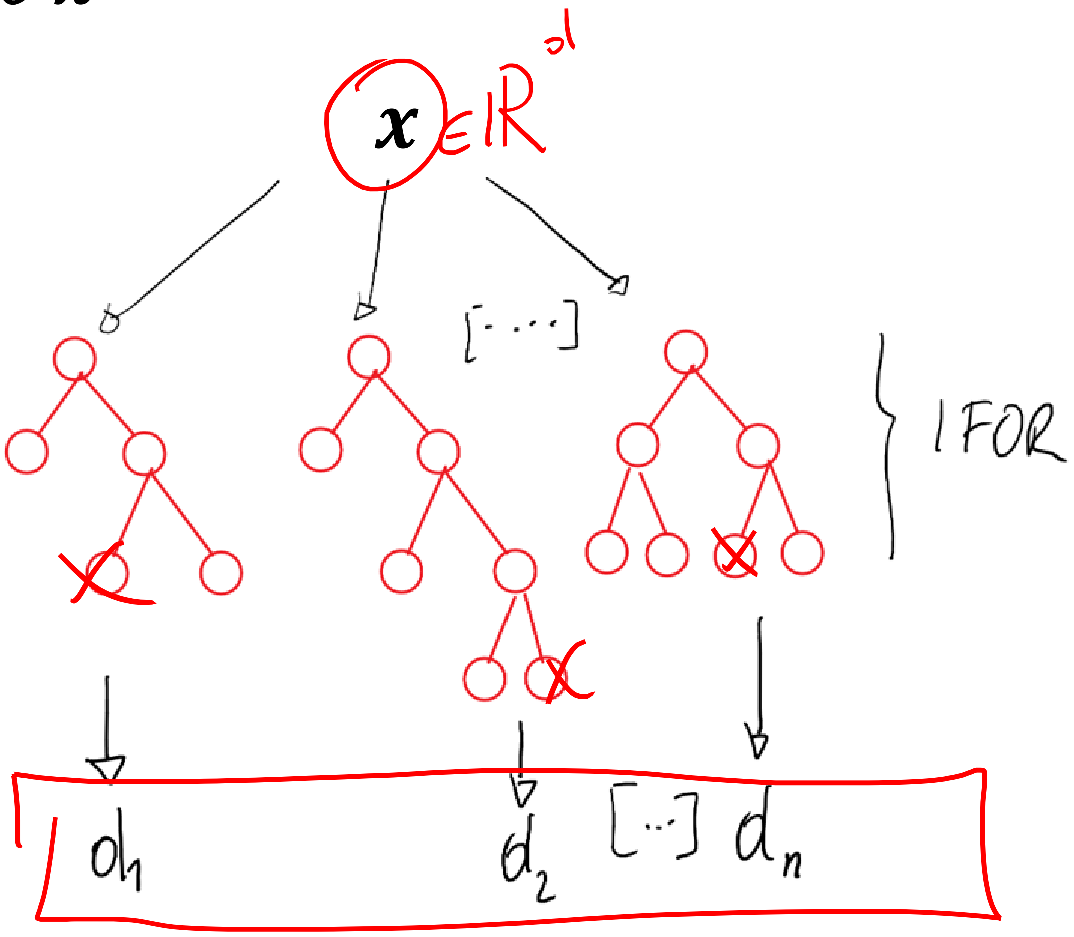
Normal data



Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: testing

Compute $E(h(\boldsymbol{x}))$, the **average path length among all the trees in the forest**, of a test sample $\boldsymbol{x}$



Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

IFOR: testing

A test sample is identified as **anomalous** when:

$$\mathcal{A}(\boldsymbol{x}) = 2^{-\frac{E(h(\boldsymbol{x}))}{c(n)}} > \gamma$$

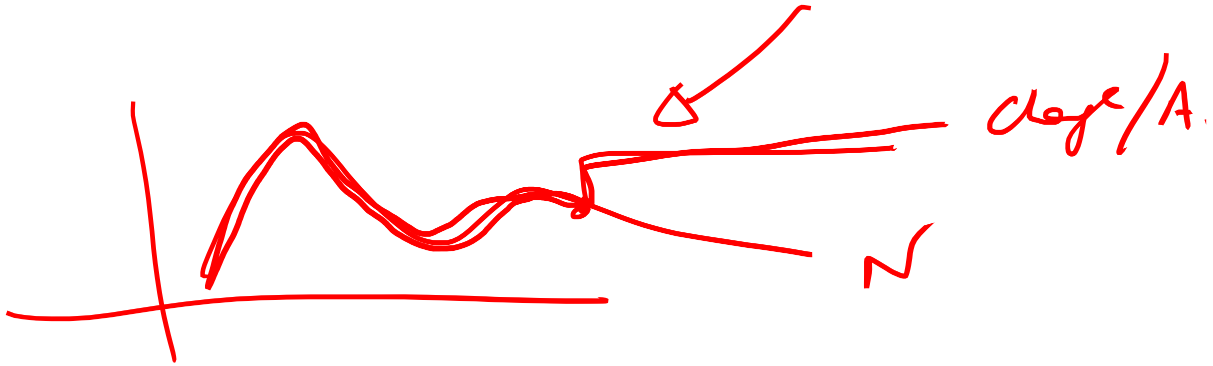
- n: number of samples in TR
- $c(n)$: average path length of unsuccessful search in a binary tree

Fei Tony Liu, Kai Ming Ting and Zhi-Hua Zhou, *Isolation Forest, ICDM 2008*

Out of the «Random Variable World»

Anomaly Detection Methods for Signals and Images

$x \sim \phi_0$

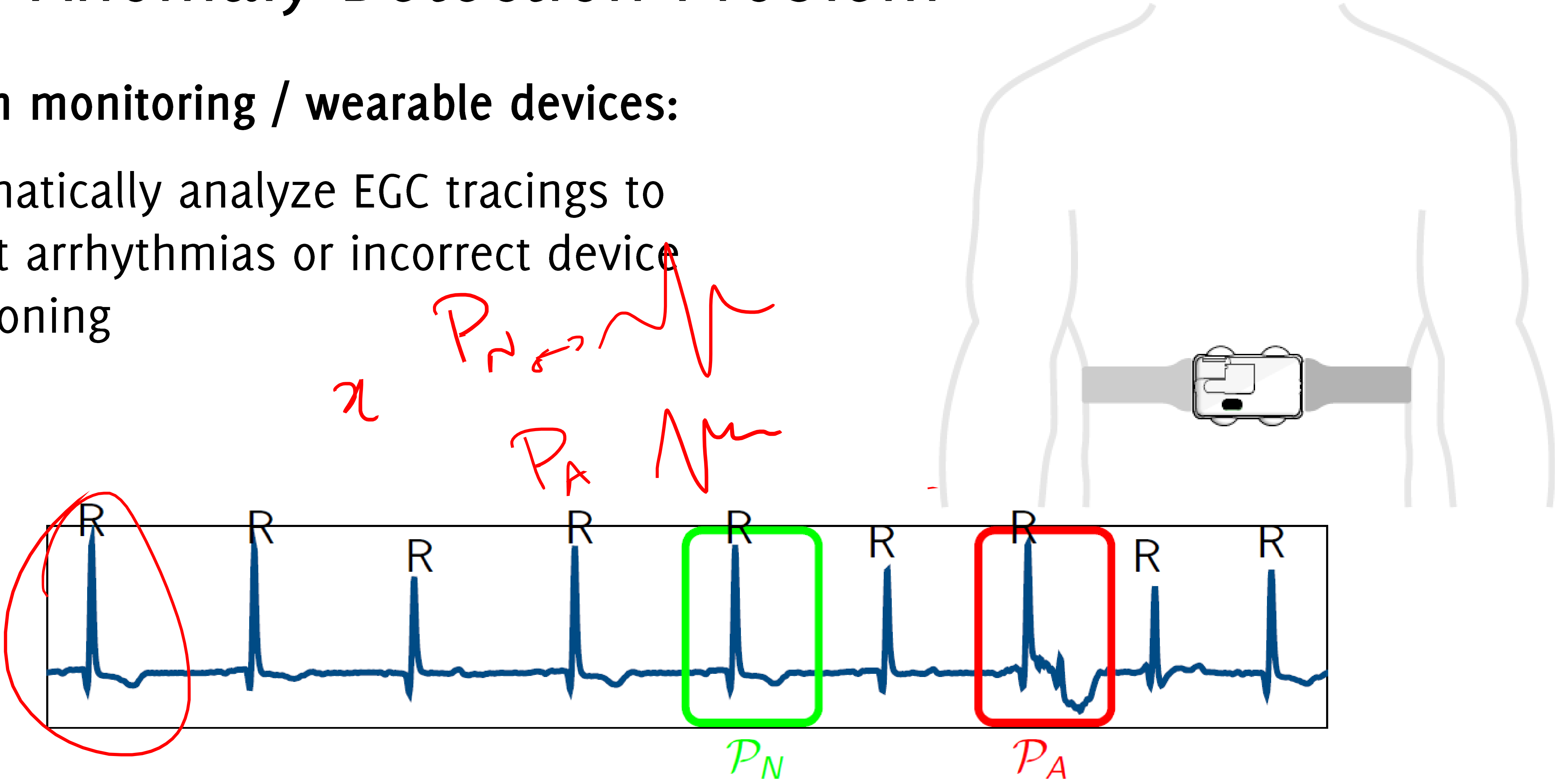


Boracchi, Trovò

… An Anomaly-Detection Problem

Health monitoring / wearable devices:

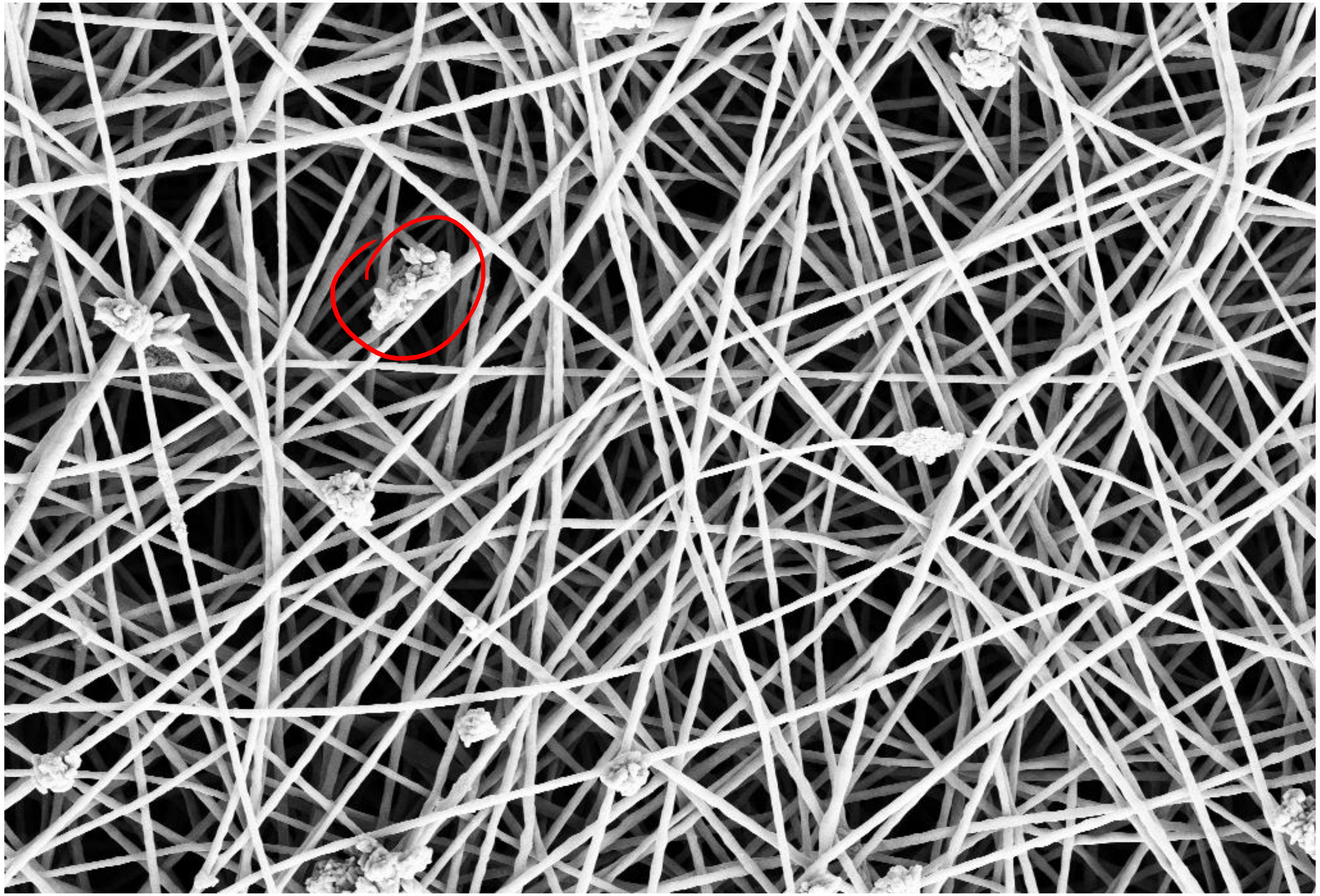
Automatically analyze EGC tracings to detect arrhythmias or incorrect device positioning



D. Carrera, B. Rossi, D. Zambon, P. Fragneto, and G. Boracchi "*ECG Monitoring in Wearable Devices by Sparse Models*" in Proceedings of ECML-PKDD 2016, 16 pages

… An Anomaly-Detection Problem

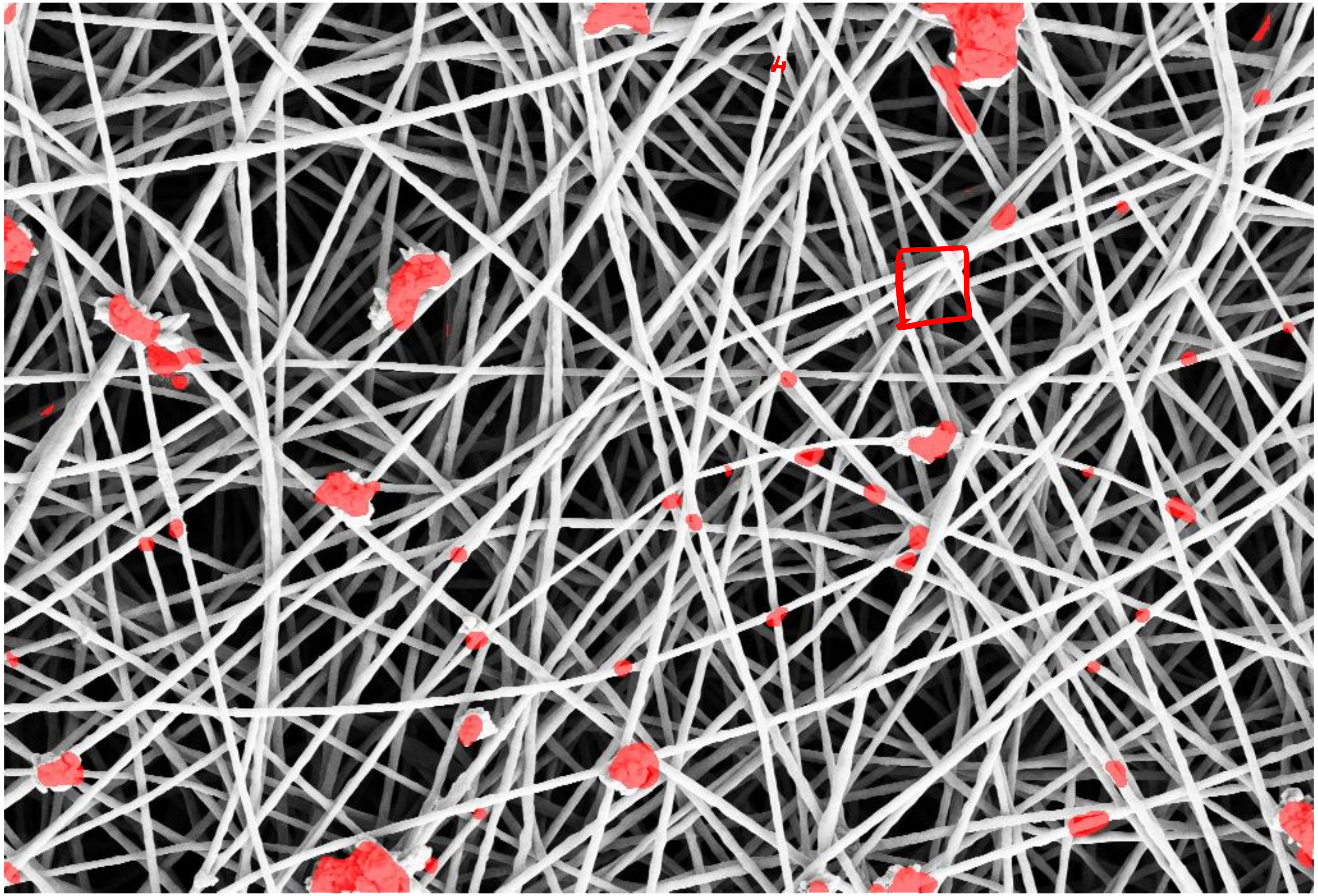
Quality Inspection Systems: monitoring the nanofiber production



Carrera D., Manganini F., Boracchi G., Lanzarone E. *"Defect Detection in SEM Images of Nanofibrous Materials"*, IEEE Transactions on Industrial Informatics 2017, 11 pages, doi:10.1109/TII.2016.2641472

… An Anomaly-Detection Problem

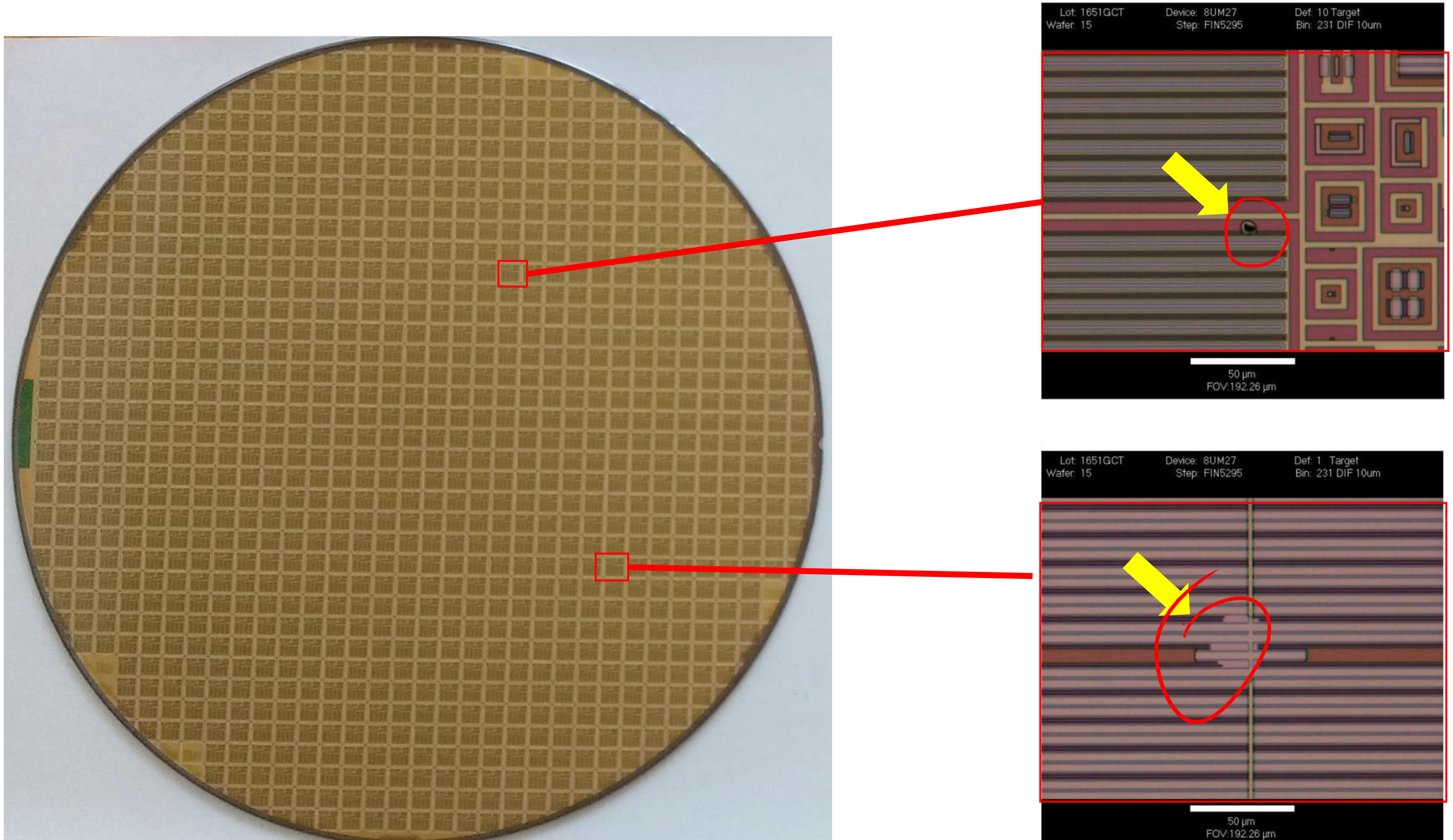
Quality Inspection Systems: monitoring the nanofiber production



Carrera D., Manganini F., Boracchi G., Lanzarone E. *"Defect Detection in SEM Images of Nanofibrous Materials"*, IEEE Transactions on Industrial Informatics 2017, 11 pages, doi:10.1109/TII.2016.2641472

… An Anomaly-Detection Problem

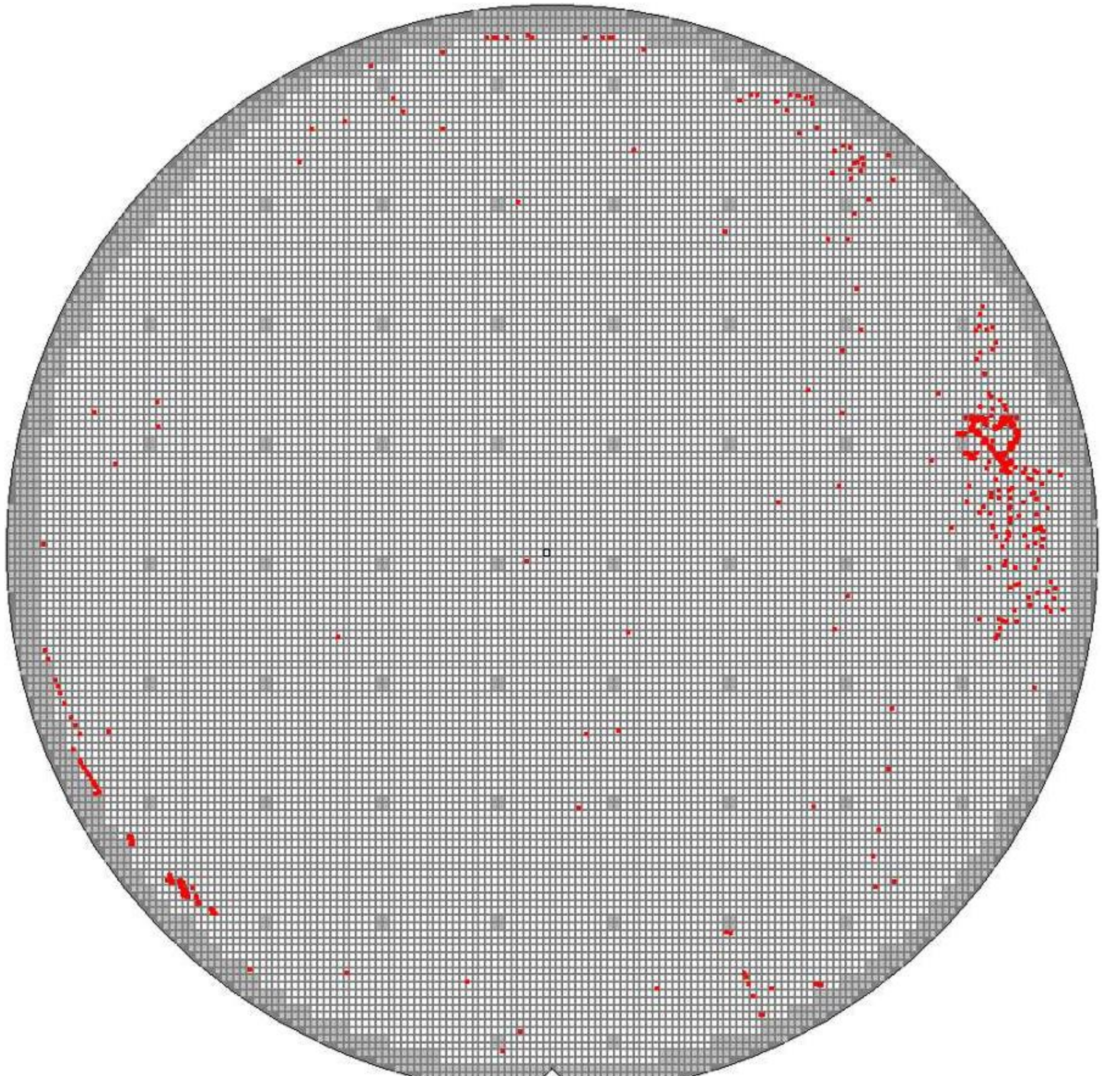
Detection of anomalies in chip production



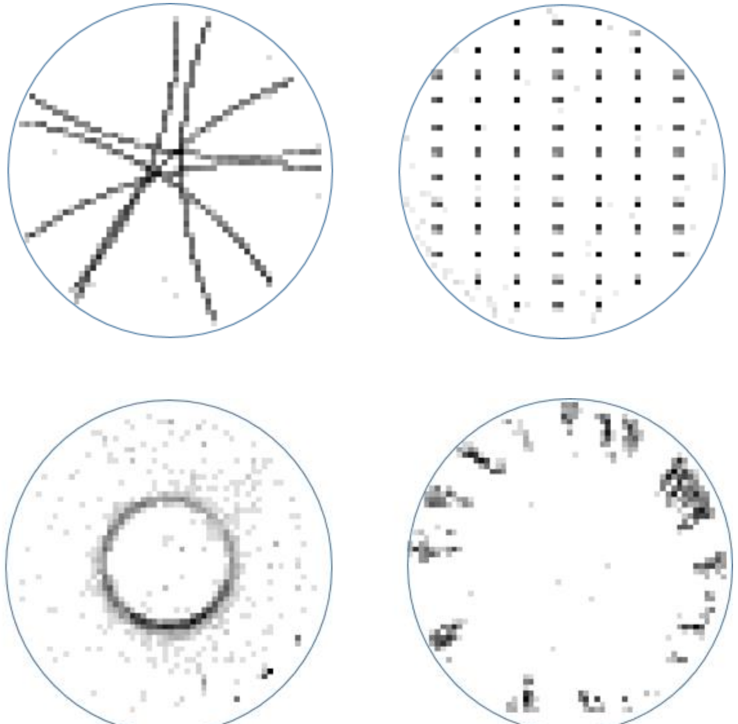
Boracchi, Trovò

… An Anomaly-Detection Problem

Detect **anomalous patterns** in the layout of defective chips, i.,e in the **wafer defect map**.



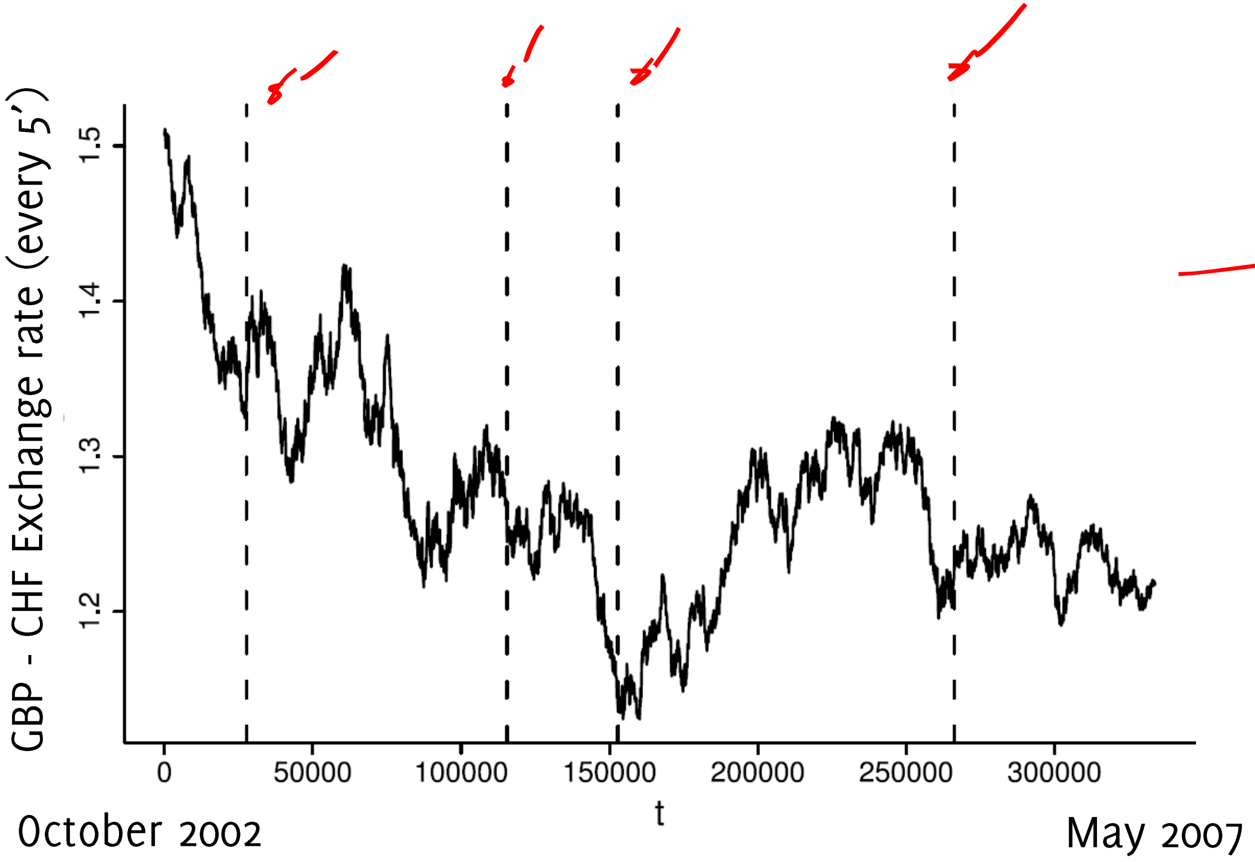
These might indicate faults, problems or malfunctioning in the chip production.

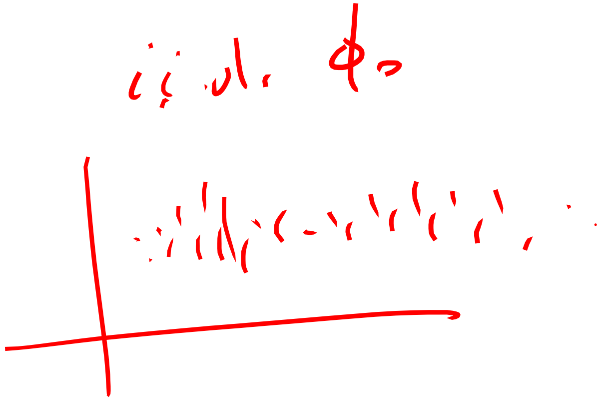


Boracchi, Trovò

… A Change-Detection Problem

Time-series (including financial ones) are typically subject to changes, as the **data-generating process evolves** over time.





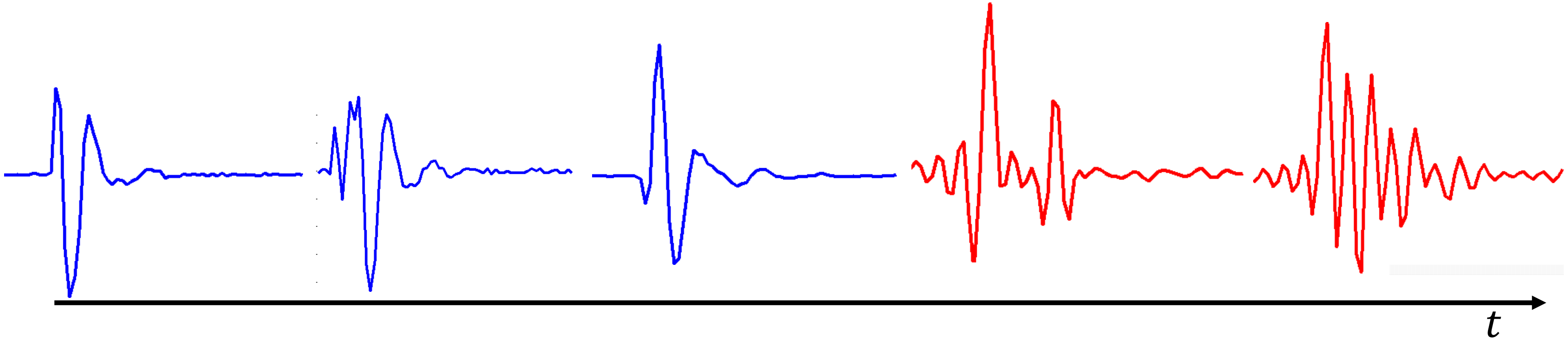
CPM

G.J. Ross, D.K. Tasoulis, N.M. Adams *"Nonparametric monitoring of data streams for changes in location and scale"* Technometrics 53 (4), 379-389, 2011

… A Change-Detection Problem

Environmental Monitoring

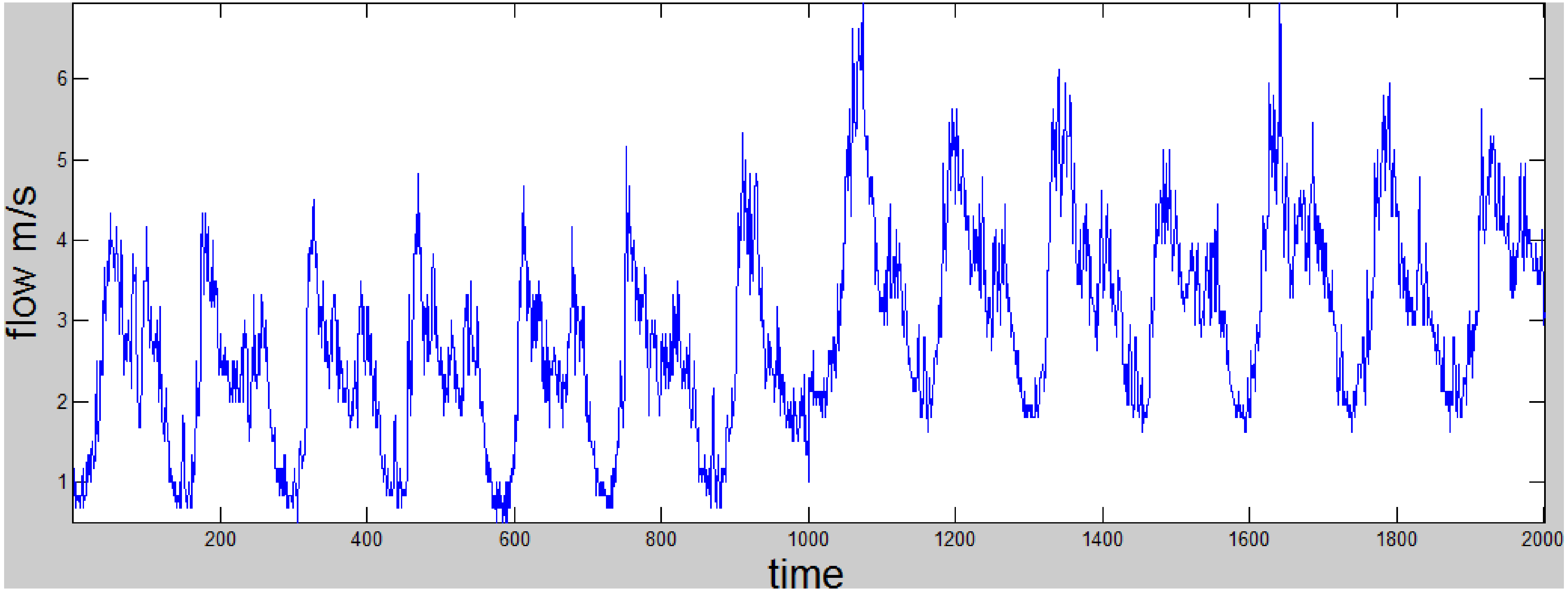
A sensor network monitoring rock faces: detecting changes in the waveforms that are recorded by MEMS sensors in network units.



C. Alippi, G. Boracchi, B. Wohlberg "*Change Detection in Streams of Signals with Sparse Representations*" in Proceedings of IEEE ICASSP 2014

… A Change-Detection Problem

Leak detection in Water Distribution Networks

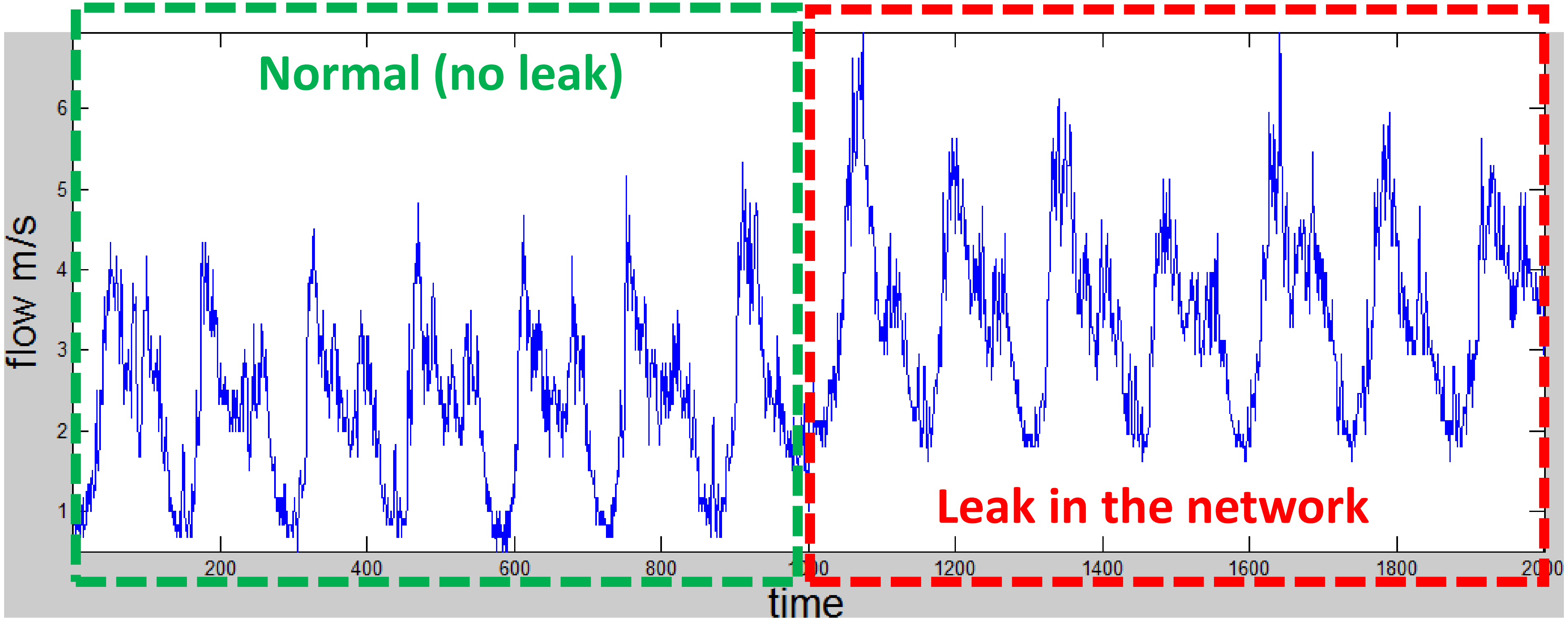


G. Boracchi and M. Roveri *"Exploiting Self-Similarity for Change Detection"* , IJCNN 2014, pp 3339 - 3346

… A Change-Detection Problem

Leak detection in Water Distribution Networks

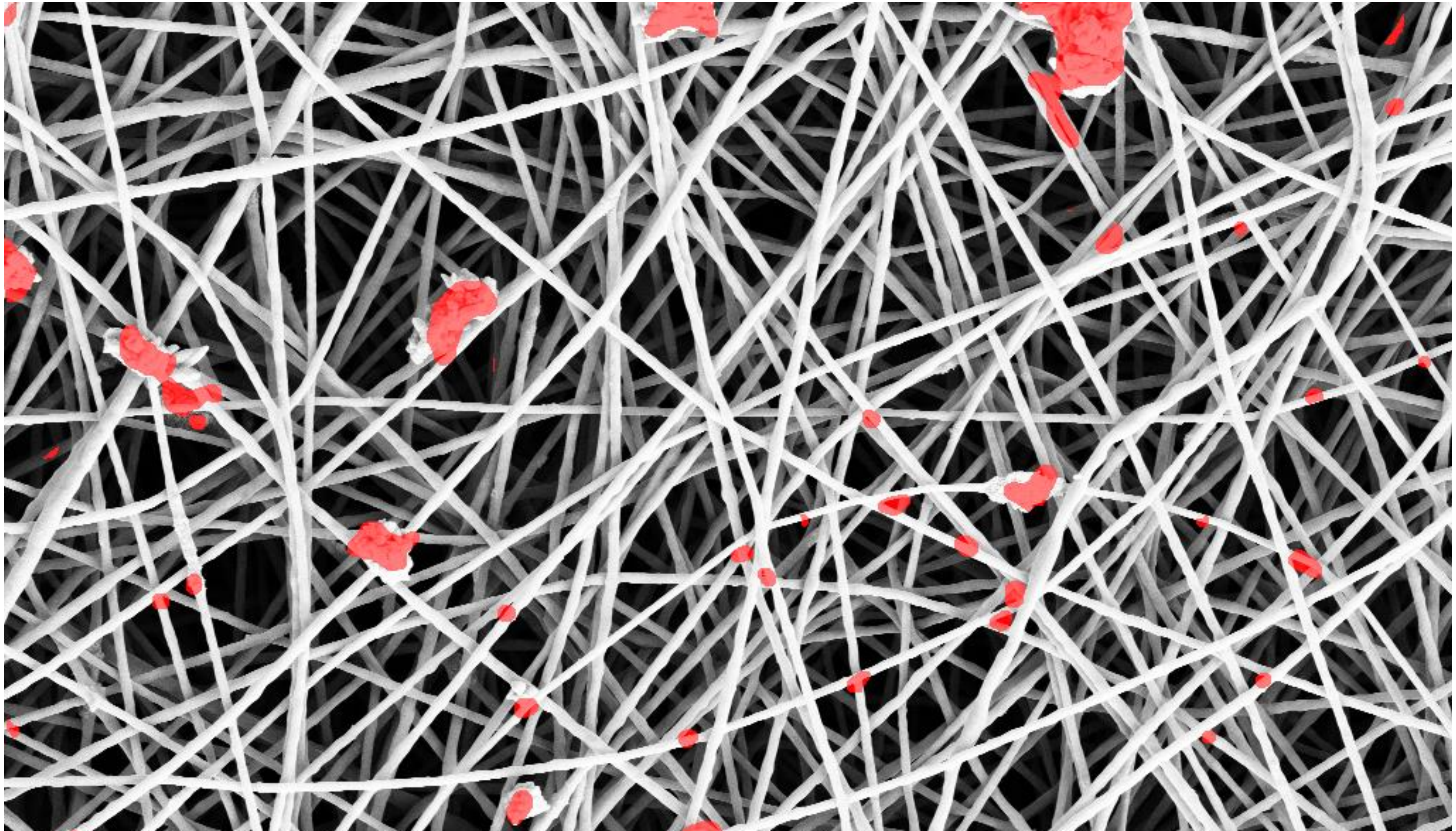
Similar problems arise in other critical infrastructure monitoring scenarios



G. Boracchi and M. Roveri *"Exploiting Self-Similarity for Change Detection"* , IJCNN 2014, pp 3339 - 3346

OUR Running example

Goal: Automatically measure area covered by defects



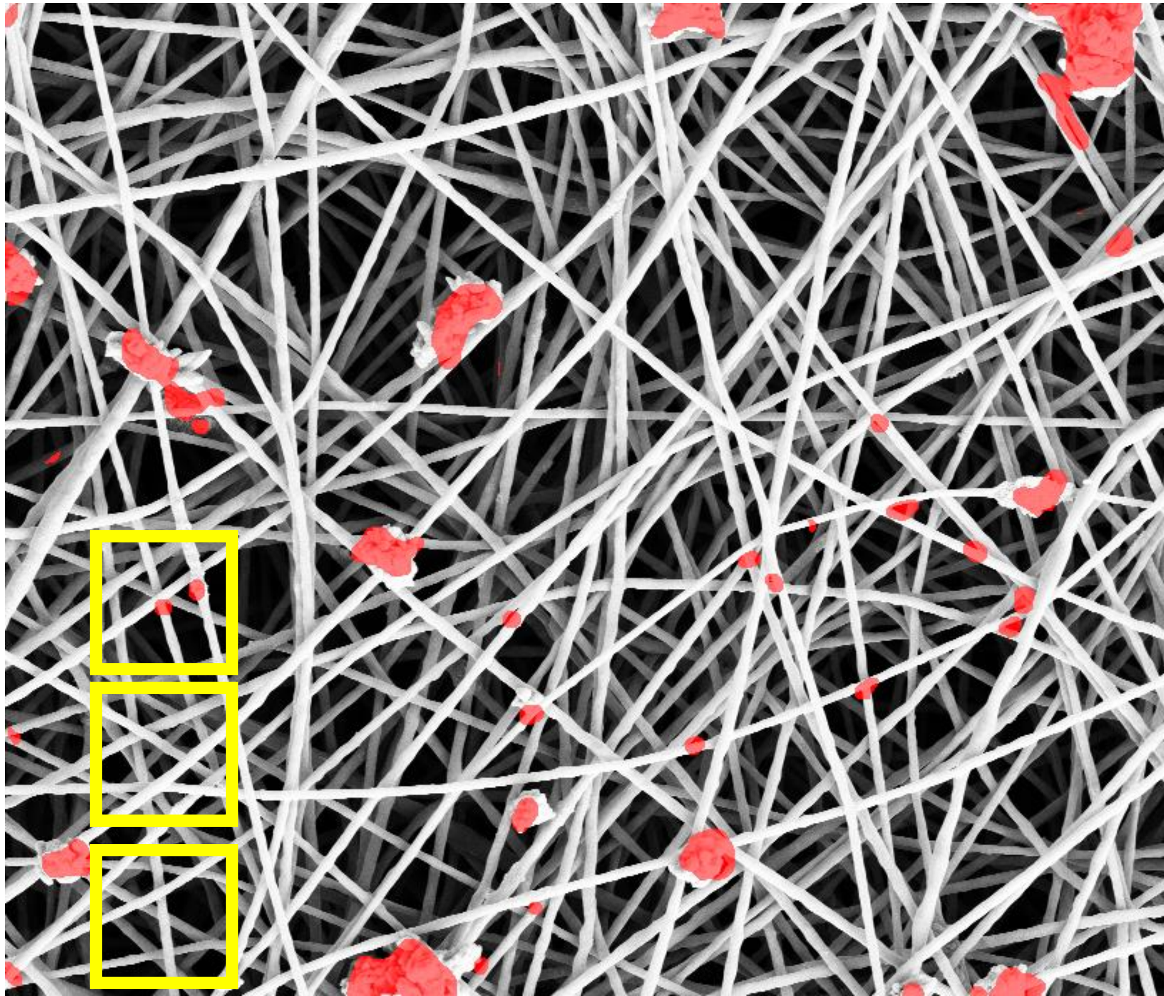
Boracchi, Trovò

Anomaly Detection in Images

The goal not determining whether the whole image is normal or anomalous, but **locate/segment possible anomalies**

Therefore, it is convenient to

1. **Analyze the image patch-wise**
2. Isolate regions containing patches that are detected as as anomalies



Can we pursue approaches meant for random variables on image patches?

Density-based approach on image patches

A density-based approach to AD would be:

Training

i. Split the normal image in patches $\boldsymbol{s}$

ii. Fit a statistical model $\hat{\phi}_0 = \mathcal{N}(\mu, \Sigma)$ describing normal patches.

Testing

i. Split the test image in patches

ii. Compute $\hat{\phi}_0(\boldsymbol{s})$ the likelihood of each test patch $\boldsymbol{s}$

iii. Detect anomalies by thresholding the likelihood

Du, B., Zhang, L.: *Random-selection-based anomaly detector for hyperspectral imagery.* IEEE Transactions on Geoscience and Remote sensing

Density-based approach on image patches

A density-based approach to AD would be:

Training

i. Split the normal image in patches $\boldsymbol{s}$

ii. Fit a statistical model $\hat{\phi}_0 = \mathcal{N}(\mu, \Sigma)$ describing normal patches.

This model is rarely accurate on natural images.
Small patches (e.g. 2×2 or 5×5) are typically preferred

Du, B., Zhang, L.: *Random-selection-based anomaly detector for hyperspectral imagery*. IEEE Transactions on Geoscience and Remote sensing
X Xie, M Mirmehdi *"Texture exemplars for defect detection on random textures"* – ICPR 2005

Density-based approach on image patches

A density-based approach to AD would be:

Training

i. Split the normal image in patches $\boldsymbol{s}$

ii. Fit a statistical model $\hat{\phi}_0 = \mathcal{N}(\mu, \Sigma)$ describing normal patches.

In some cases (textures) a Gaussian Mixture was used as a more general model

Du, B., Zhang, L.: *Random-selection-based anomaly detector for hyperspectral imagery.* IEEE Transactions on Geoscience and Remote sensing

X Xie, M Mirmehdi *"Texture exemplars for defect detection on random textures"* – ICPR 2005

Density-based approach on image patches

A density-based approach to AD would be:

Training

i. Split the normal image in patches $\boldsymbol{s}$

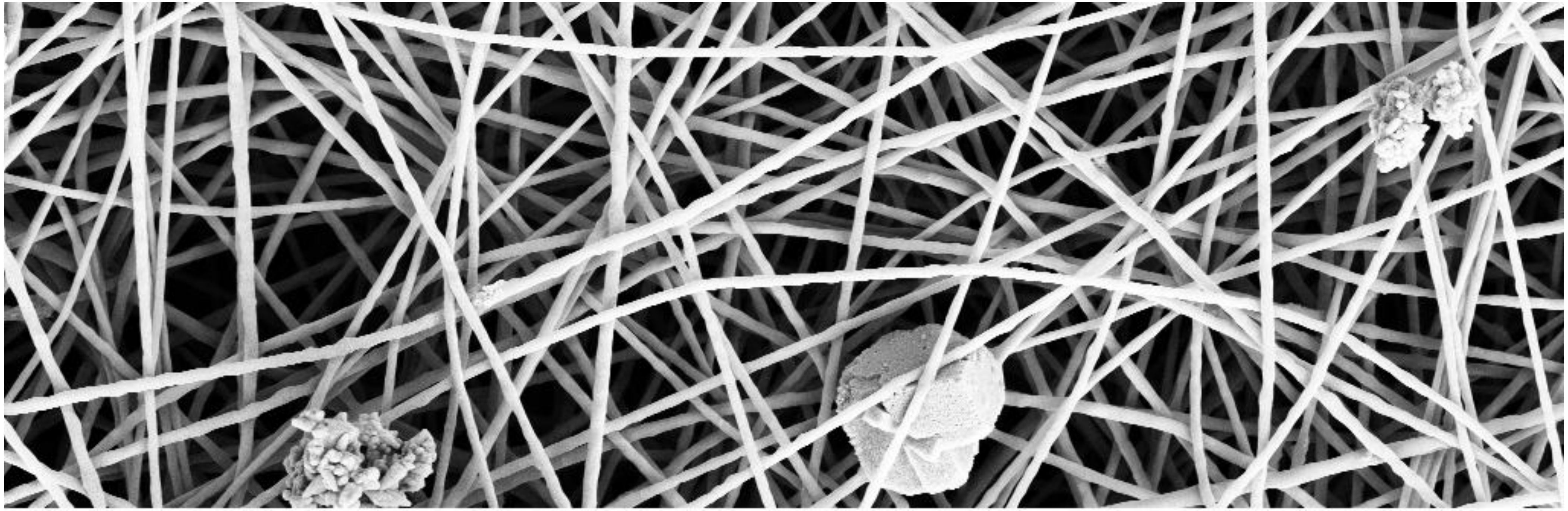
ii. Fit a statistical model $\hat{\phi}_0 = \mathcal{N}(\mu, \Sigma)$ describing normal patches.

Random selection procedures can be employed to minimize the risk of including outliers

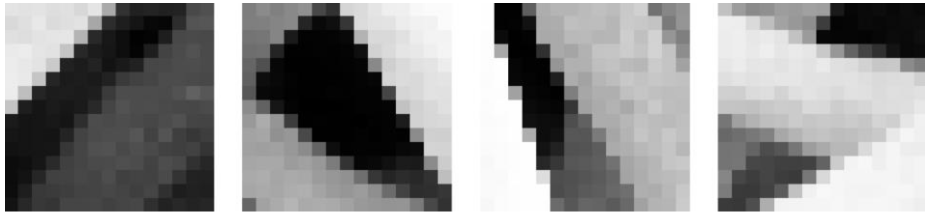
Du, B., Zhang, L.: *Random-selection-based anomaly detector for hyperspectral imagery*. IEEE Transactions on Geoscience and Remote sensing

The limitations of the Random variable model

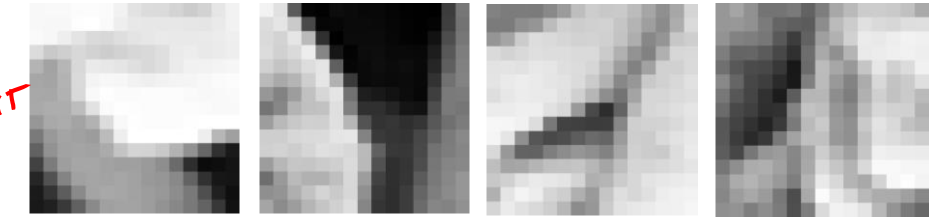
In many anomaly-detection problems in imaging, **normal regions exhibit peculiar structures and spatial correlation**



Normal regions

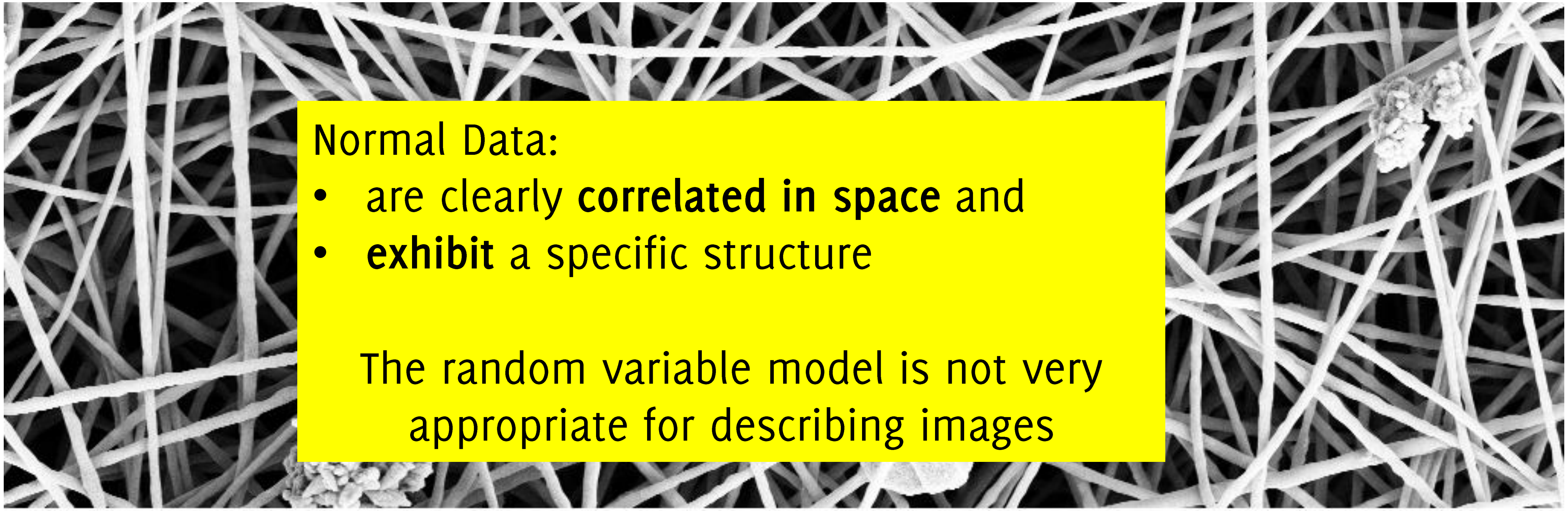


Anomalous regions



The limitations of the Random variable model

In many anomaly-detection problems in imaging, **normal regions exhibit peculiar structures and spatial correlation**

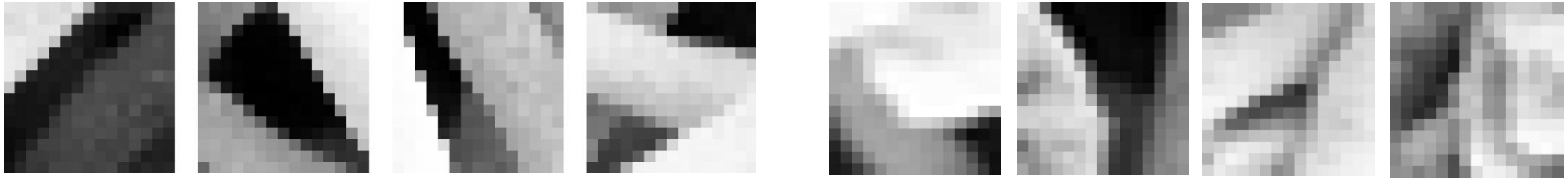


Normal Data:
- are clearly **correlated in space** and
- **exhibit** a specific structure

The random variable model is not very appropriate for describing images

Normal regions

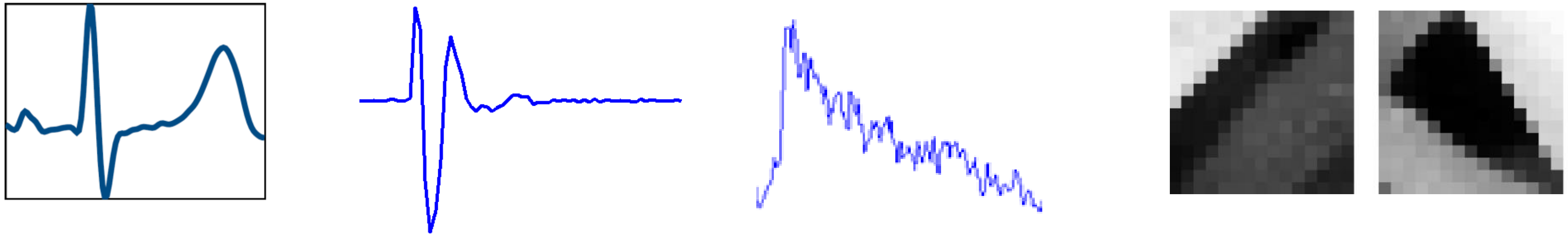
Anomalous regions



Boracchi, Trovò

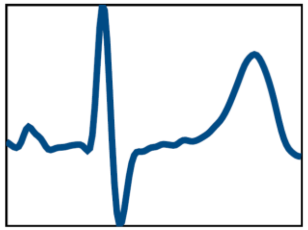
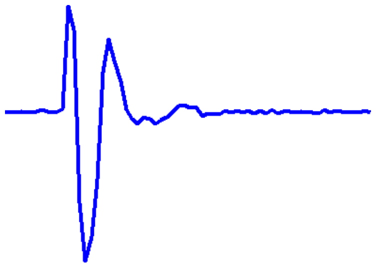
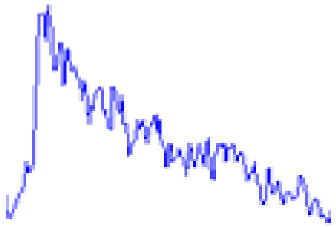
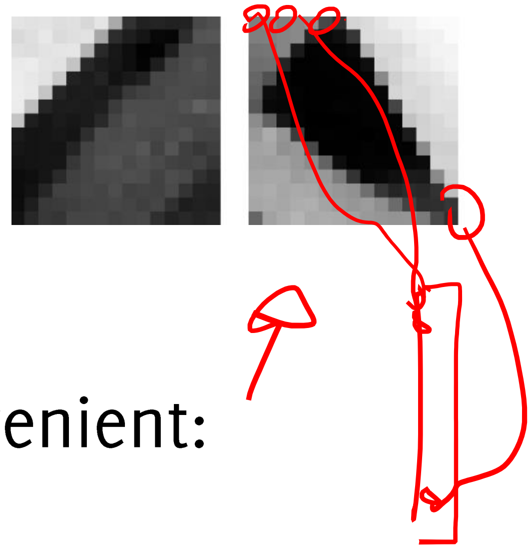
Real World Detection Problems

Random variable model **does not successfully apply to signals or images** (not even small portions)



Real World Detection Problems

Random variable model **does not successfully apply to signals or images** (not even small portions)

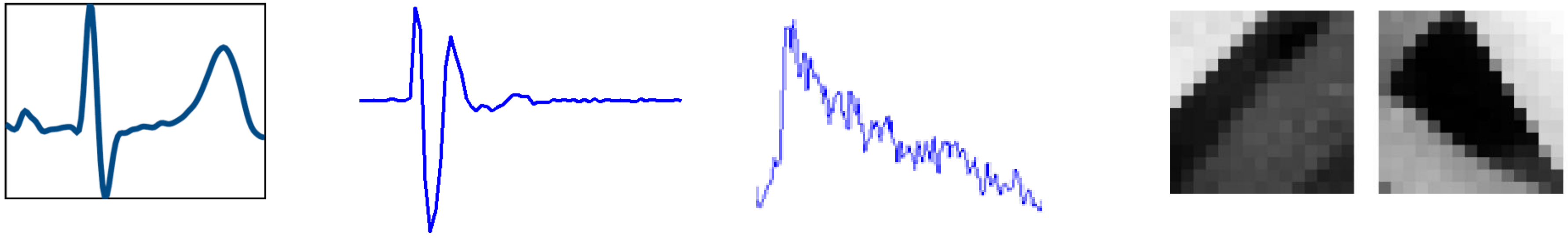





Stacking each signal $\boldsymbol{s} \in \mathbb{R}^d$ in a vector $\boldsymbol{x}$ is not convenient:

- **Data dimension** d can become **huge**
- **Correlation among components is difficult to model**

Real World Detection Problems

Random variable model **does not successfully apply to signals or images** (not even small portions)



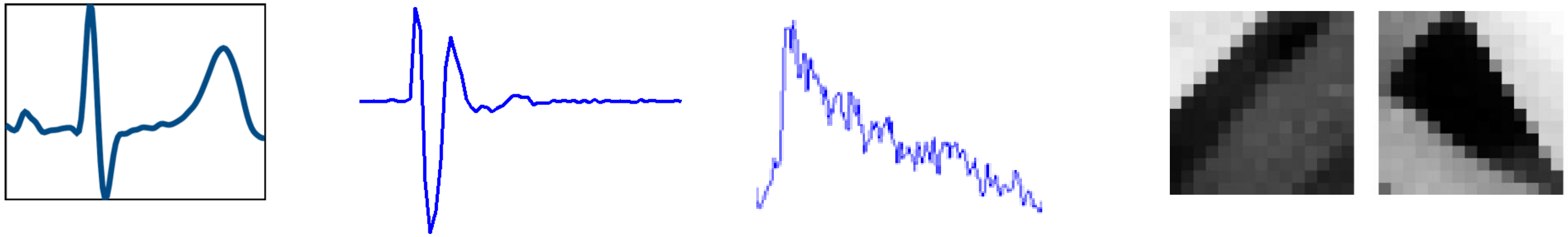
Stacking each signal $\boldsymbol{s} \in \mathbb{R}^d$ in a vector $\boldsymbol{x}$ is not convenient:

- **Data dimension** d can become **huge**
- **Correlation among components is difficult to model**

It is not easy to **estimate a density model** or threat these as **realizations of a random variable**

Real World Detection Problems

Random variable model **does not successfully apply to signals or images** (not even small portions)



Stacking each signal $\boldsymbol{s} \in \mathbb{R}^d$ in a vector $\boldsymbol{x}$ is not convenient:

- **Data dimension** d can become **huge**
- **Correlation among components is difficult to model**

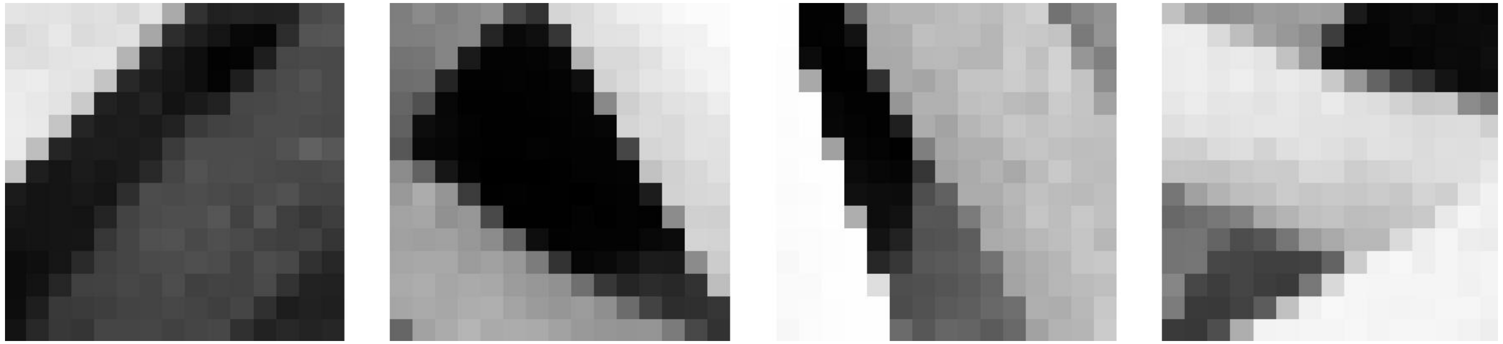
It is not easy to **estimate a density model** or threat these as **realizations of a random variable**

Moreover, when **normal data** exhibit a peculiar **structure**, we are interested in **detecting changes/anomalies affecting that structure**

Real World Detection Problems

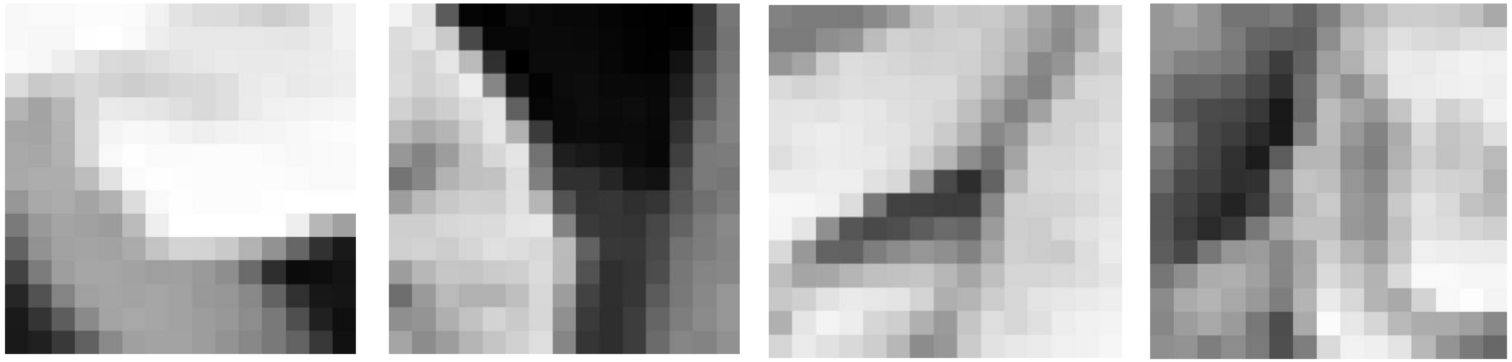
Normal patches -> background

- Exhibit a specific structure (geometry) or intensities



Anomalous patches:

- Are rare elements that do not confrom with the background



Anomaly Detection Out of the “Random Variable” World

Model-based approaches for images

The Typical Approach

Most of the considered methods

1. **Estimate** a **model** describing **normal data** (background model)
2. Provide, for each test sample, an **anomaly score,** or measure of rareness, w.r.t. the learned model
3. Apply a **decision rule** to detect anomalies (typically thresholding)
4. **[optional]** Perform **post-processing** operations to enforce smooth detections and avoid isolated pixels that are not consistent with neighbourhoods

Remark: Statistical-based approaches seen before use as background model the statistical distribution $\hat{\phi}_0$ and a statistic as anomaly score

The Typical Approach

Most of the considered methods

1. **Estimate** a **model** describing **normal data** (background model)
2. Provide, for each test sample, an **anomaly score,** or measure of rareness, w.r.t. the learned model
3. Apply a **decision rule** to detect anomalies (typically thresholding)
4. **[optional]** Perform **post-processing** operations to enforce smooth detections and av sistent with neighborhoods

Remark: Statistical-ba as background model the statistical nomaly score

The background model is used to **bring an image patch into the "random variable world"** (regression, encoding, feature extraction…)

The Typical Approach

Most of the considered methods

1. **Estimate** a **model** describing **normal data** (background model)
2. Provide, for each test sample, an **anomaly score,** or measure of rareness, w.r.t. the learned model
3. Apply a **decision rule** to detect anomalies (typically thresholding)
4. **[optional]** Perform **post-processing** operations to enforce smooth detections and ... tent with neighborhoods ...

Remark: Statistical- ... background model the statistic ... maly score

Once “applied“ the background model, one can use **most of anomaly detection methods for the “random variable world”**.
This might require **fitting an additional model**

The Typical Approach

Different options to learn the background model

- **semi-supervised approach,** background model is learned exclusively normal data
- **unsupervised approach**, background model is fit to both normal and anomalous but it is robust to outliers

Semi-supervised AD methods out of the RVW

Out of the "Random Variable" world

- Detrending-based methods
- Reconstruction-based methods
  - Subspace methods
- Feature-based monitoring
  - Expert-driven Features
  - Data-driven Features

Semi-supervised AD methods out of the RVW

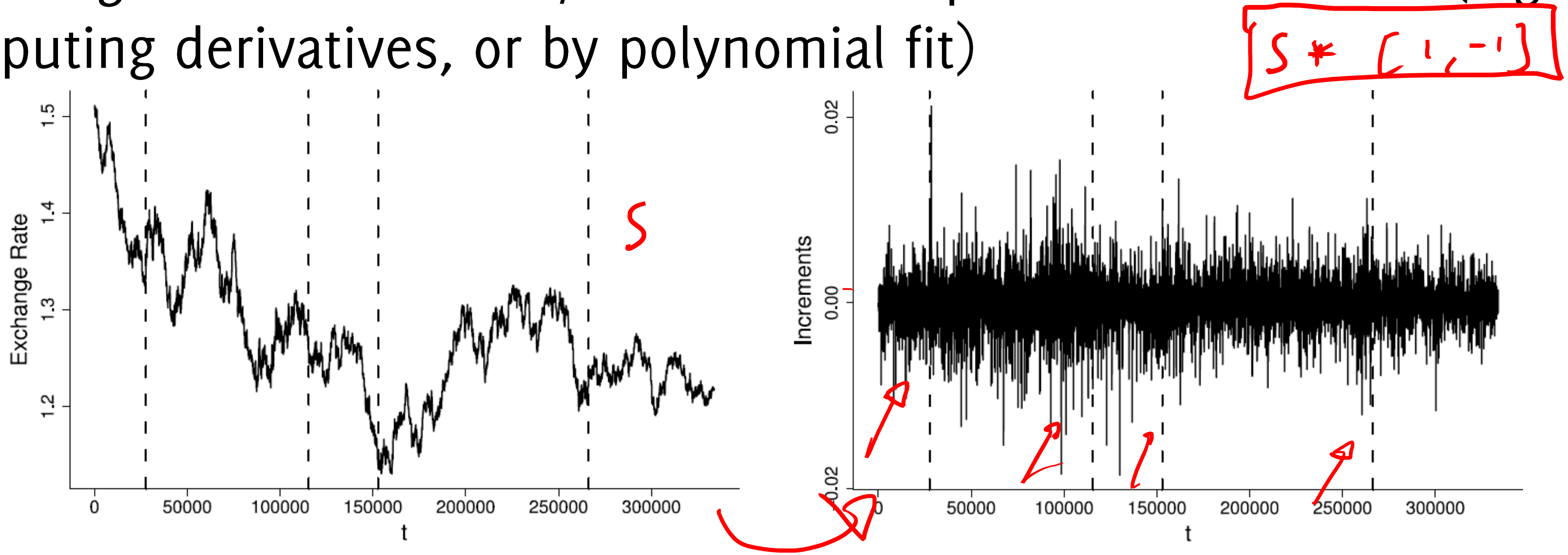
Out of the "Random Variable" world

- Detrending-based methods
- Reconstruction-based methods
  - Subspace methods
- Feature-based monitoring
  - Expert-driven Features
  - Data-driven Features

… Out of the random variable world

We can "get rid of the structure" by **detrending/filtering:**

- removing the deterministic/correlated components of the data (e.g. by computing derivatives, or by polynomial fit)



(a) Original sequence (b) First differences

But this might not always apply, since we get rid of "all the structures", thus also those from anomalous signals.

G.J. Ross, D.K. Tasoulis, N.M. Adams *"Nonparametric monitoring of data streams for changes in location and scale"* Technometrics 53 (4), 379-389, 2011

Semi-supervised AD methods out of the RVW

Out of the "Random Variable" world

- Detrending-based methods
- Reconstruction-based methods
  - Subspace methods
- Feature-based monitoring
  - Expert-driven Features
  - Data-driven Features

Reconstruction-based Methods

*Fit a statistical model to the observation to **describe dependence**, **apply anomaly detection** on the independent **residuals**.*

Detection is performed by using a model $\mathcal{M}$ which **represents normal data**:

- **During training:** learn the model $\mathcal{M}$ from training set TR
- **During testing:**
  - Reconstruct each test signal $\boldsymbol{s}$ through $\mathcal{M}$.
  - Assess the **residuals** between $\boldsymbol{s}$ and its reconstruction

The rationale is that $\mathcal{M}$ can **reconstruct only normal data**, thus anomalies are expected to yield large reconstruction errors.

Bengio, Y., Courville, A., Vincent, P. *"Representation learning: A review and new perspectives"*. IEEE TPAMI 2013

Reconstruction-based Methods

Popular models are:

- autoregressive models for time series (ARMA, ARIMA…)
- neural networks, in particular auto-encoders, for higher dimensional data
- projection on subspaces / manifolds
- dictionaries yielding sparse-representations

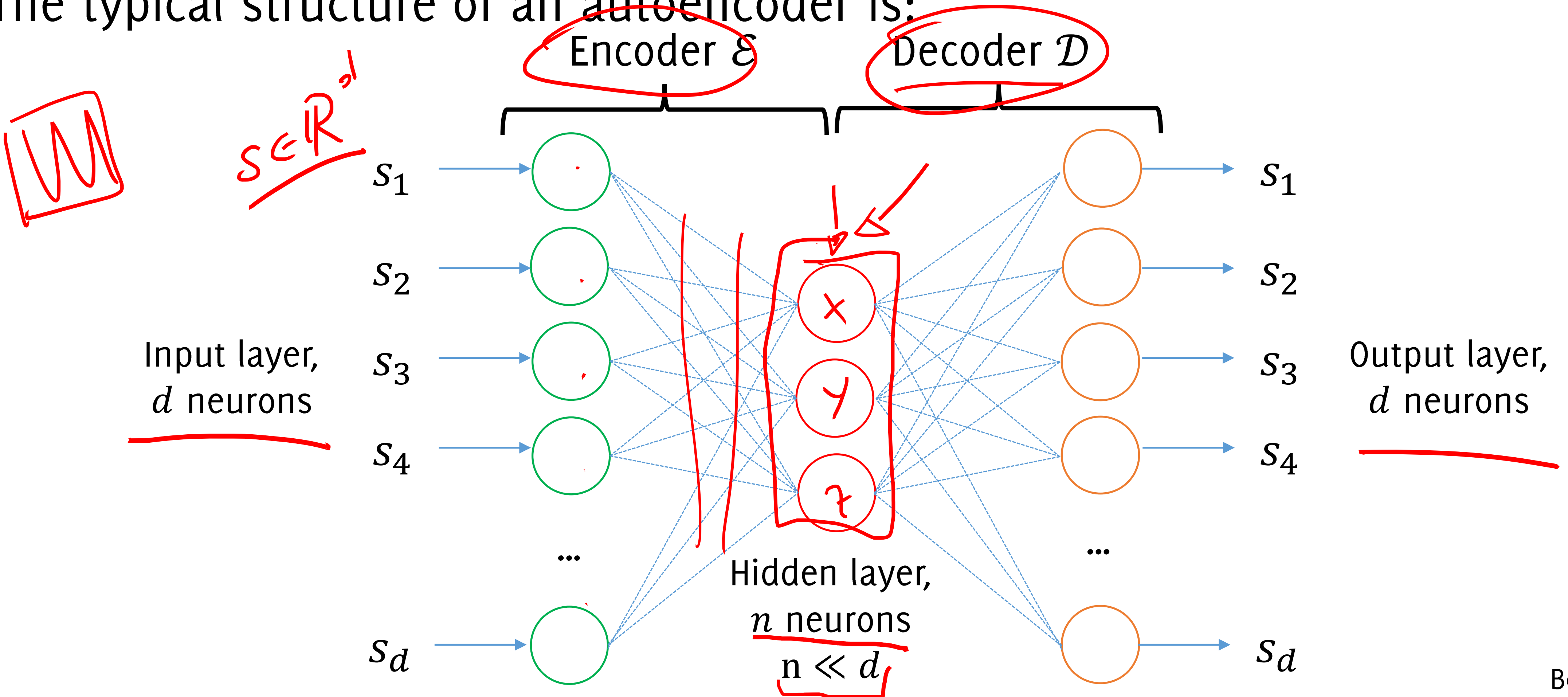
× DL models

The two latter can be also interpreted as subspace methods

Reconstruction-based Methods

Autoencoders are non-parametric models (neural networks) trained to reconstruct data in a training set.

The typical structure of an autoencoder is:



Reconstruction-based Methods

Autoencoders are non-parametric models (neural networks) trained to reconstruct data in a training set. The typical loss function is:

$$\sum_{s \in S} \left\| \boldsymbol{s} - \mathcal{D}\left(\mathcal{E}(\boldsymbol{s})\right) \right\|_2$$

and training of $\mathcal{D}\left(\mathcal{E}(\cdot)\right)$ is performed through standard backpropagation algorithms (e.g. SGD)

Remarks

- AE typically does not provide exact reconstruction since $n \ll d$.
- Additional regularization terms might be included in the loss function

Bengio, Y., Courville, A., Vincent, P. *"Representation learning: A review and new perspectives"*. IEEE TPAMI 2013

Mishne, G., Shaham, U., Cloninger, A., & Cohen, I. *Diffusion nets*. Applied and Computational Harmonic Analysis (2017).

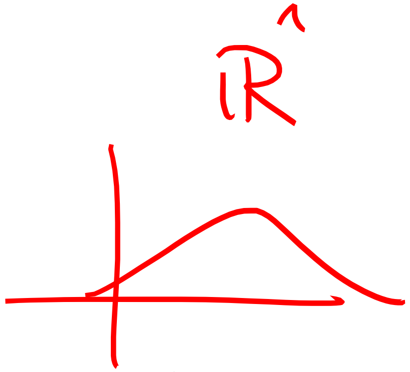
Monitoring the Reconstruction Error

Detection by reconstruction error monitoring (AE notation)

Training (Monitoring the Reconstruction Error):

1. Train the model $\mathcal{D}(\mathcal{E}(\cdot))$ from the training set TR
2. Learn the distribution of reconstruction errors

$$\text{err}(\boldsymbol{s}) = \left\|\boldsymbol{s} - \mathcal{D}\big(\mathcal{E}(\boldsymbol{s})\big)\right\|_2, \qquad \boldsymbol{s} \in V$$



over a validation set $V \neq TR$ and define a threshold γ (bootstrap)

Testing (Monitoring the Reconstruction Error):

1. Perform encoding and compute the reconstruction error

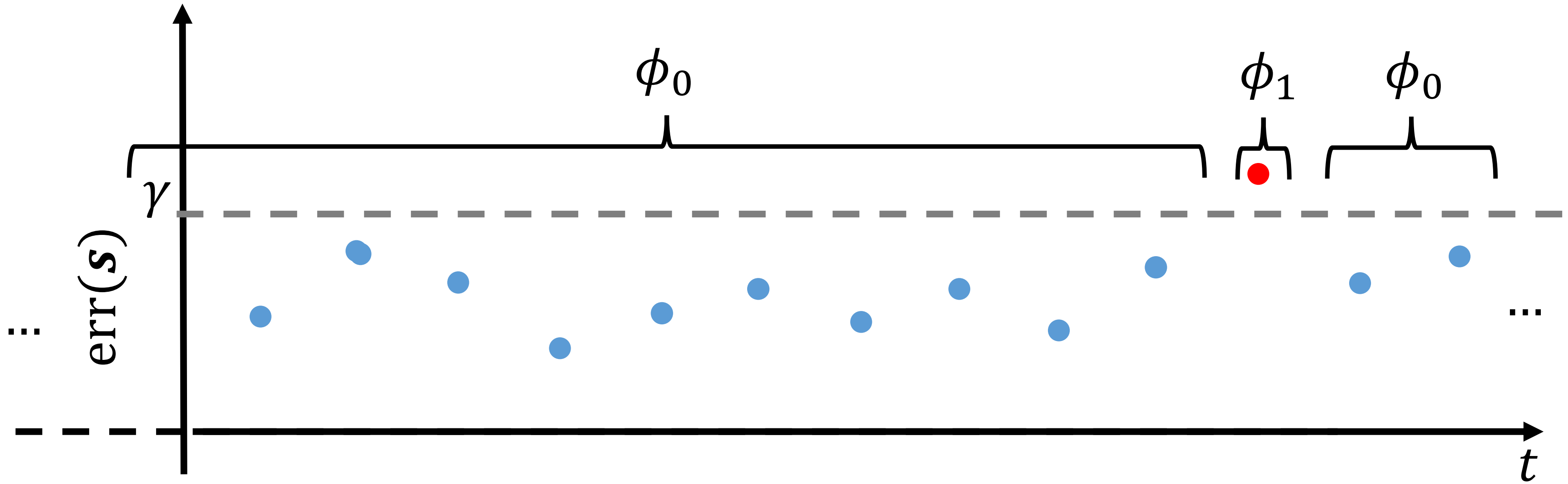
$$\text{err}(\boldsymbol{s}) = \left\|\boldsymbol{s} - \mathcal{D}\big(\mathcal{E}(\boldsymbol{s})\big)\right\|_2$$

2. Consider $\boldsymbol{s}$ anomalous when $\text{err}(\boldsymbol{s}) > \gamma$

Monitoring the Reconstruction Error

Normal data are expected to yield values of $\text{err}(\boldsymbol{s})$ that **are low**, while anomalies do not. This property holds **when the model $\mathcal{M}$ was specifically learned to describe normal data**

Outliers can be detected by a threshold on $\text{err}(\boldsymbol{s})$



Semi-supervised AD methods out of the RVW

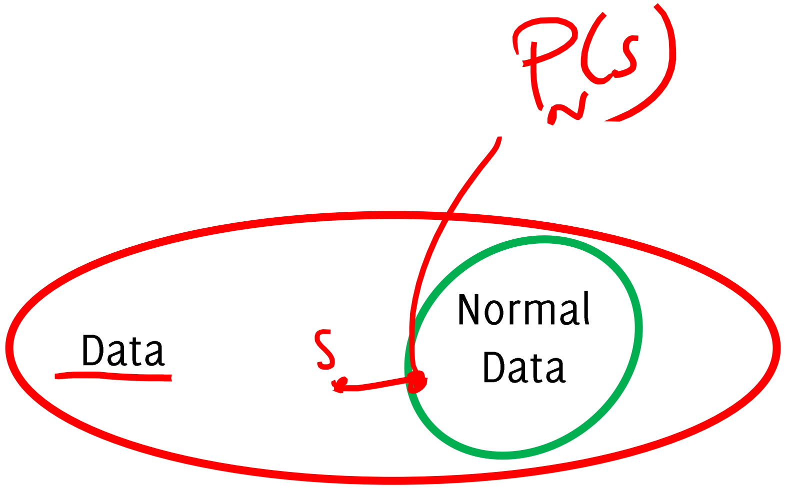
Out of the "Random Variable" world

- Detrending-based methods
- Reconstruction-based methods
  - Subspace methods
- Feature-based monitoring
  - Expert-driven Features
  - Data-driven Features

Subspace methods

The underlying assumption is that

- **normal data live in a subspace** that can be identified by TR
- **anomalies can be detected by projecting test data** in such subspace and by monitoring the reconstruction error (distance with the projection)

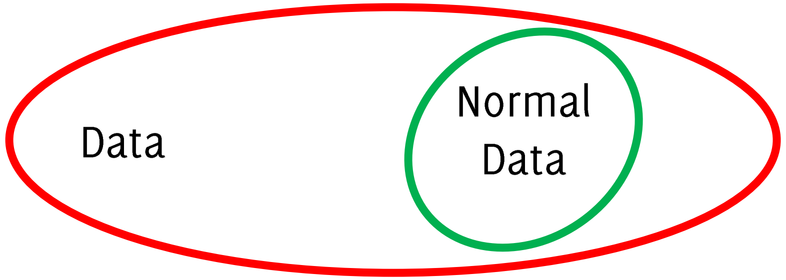


V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

Subspace Methods

A few example of **models used for describing normal patches**:

- **Fourier transform**: normal data can be expressed by a few specific frequencies
- **PCA**: normal data live **in the linear subspace of the first components.**
- Robust PCA: defined on the ℓ^1 distance to be insensitive to outliers in normal data.
- Kernel PCA: normal patches live in a non-linear manifold.
- **Random projections**



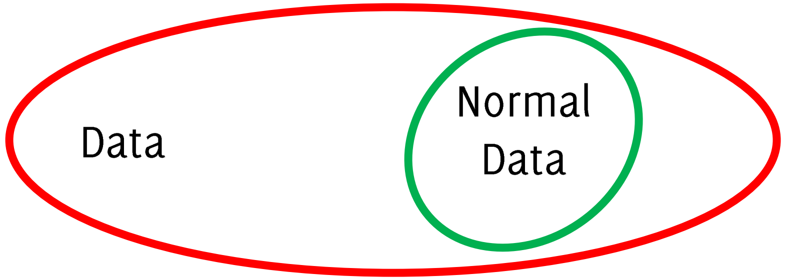
V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

Subspace Methods

A few example of **models used for describing normal patches**:

- **Fourier transform**: normal data can be expressed by a few specific frequencies
- **PCA**: normal data live **in the linear subspace of the first components.**
- Robust PCA: defined on the ℓ^1 distance to be insensitive to outliers in normal data.
- Kernel PCA: normal patches live in a non-linear manifold.
- **Random projections**

Data Driven



V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

Subspace Methods: Statistics

Compute the **the projection over a subspace characterizing normal data,**

$$\boldsymbol{x} = P^T \boldsymbol{s}, \qquad P \in \mathbb{R}^{m \times d}, \qquad m \ll d$$

which is the projection over the first m principal components and a way to reduce data-dimensionality. When $m = d$ you get perfect reconstruction on any normal and anomalous data!

- Monitor the **reconstruction error**:

$$\text{err}(\boldsymbol{s}) = \|\boldsymbol{s} - PP^T\boldsymbol{s}\|_2$$

  which is the distance between $\boldsymbol{s}$ and its projection $PP^T\boldsymbol{s}$ over the subspace of normal patches

- Alternatively, monitor the **least-principal component only**, which like an anomaly score should be low in normal data.

- Alternatively, **monitor projections** coefficient $\boldsymbol{x} = P^T\boldsymbol{s}$ via multivariate statistical model/test

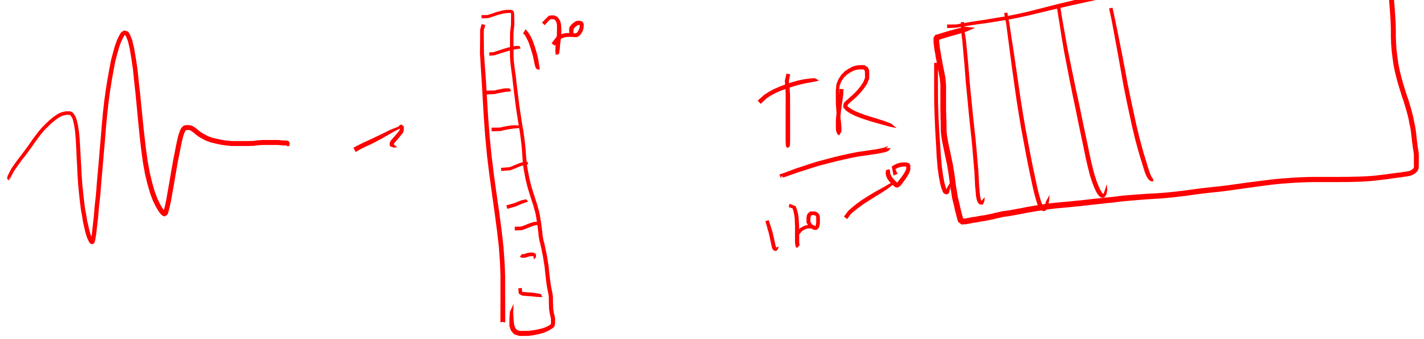
Fifth Matlab Assignment

Fifth Matlab Assignment

Goal: You have to implement a model based on PCA for detecting anomalous heart-beats.

Data: You will be provided with both normal and anomalous ECG signals, already split and aligned in portions corresponding to an heartbeat.

Annotation are also provided but these can be used for performance assessment only



TR → PCA projection

$[T, Score, L] = PCA(TR)$

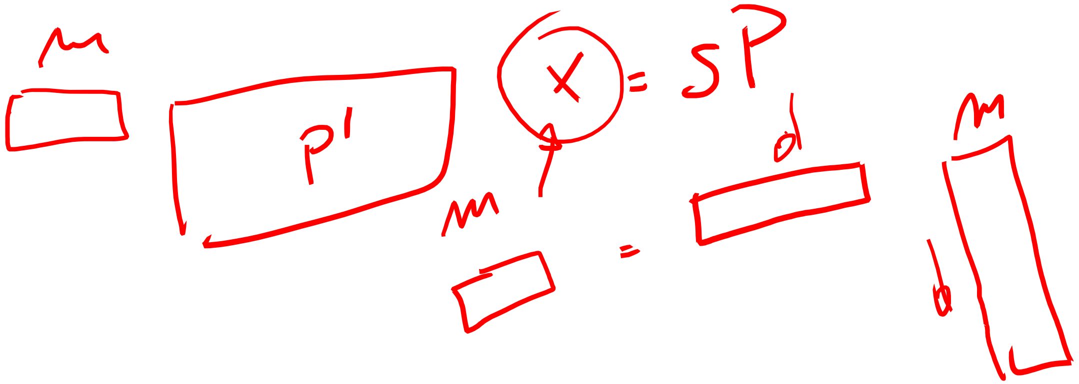
$L(1 - m)$

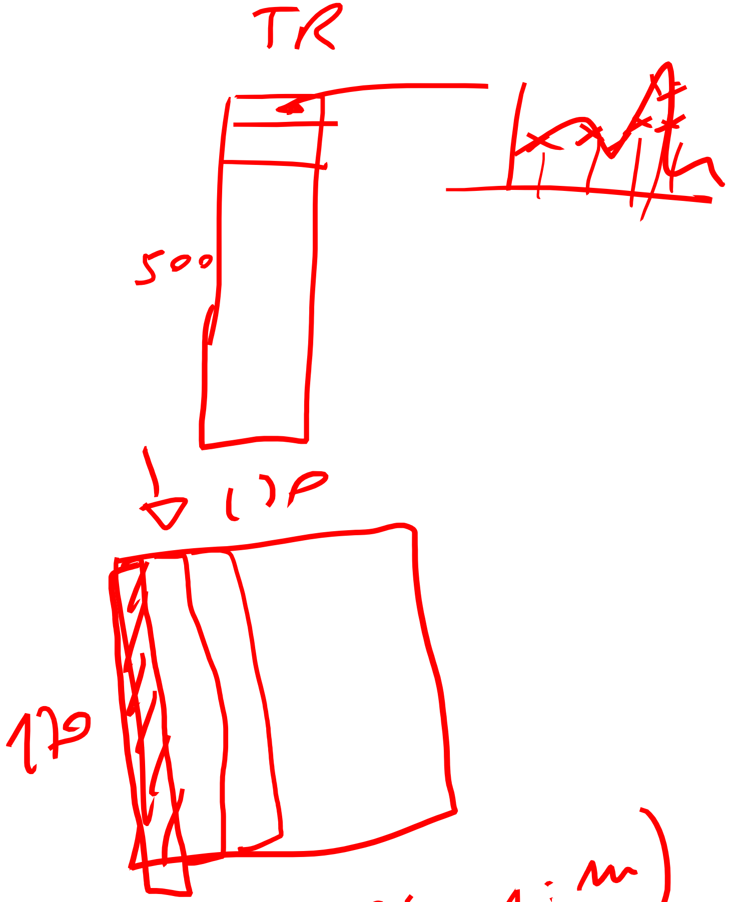
$S \in \mathbb{R}^{170}$

t

xP'

sT





$P = T$

sPP'

$P = T(:, 1:m)$

$E(s)$

The Data

Vectors arranged in matrices corresponding to normal/anomalous HB

Watch out:

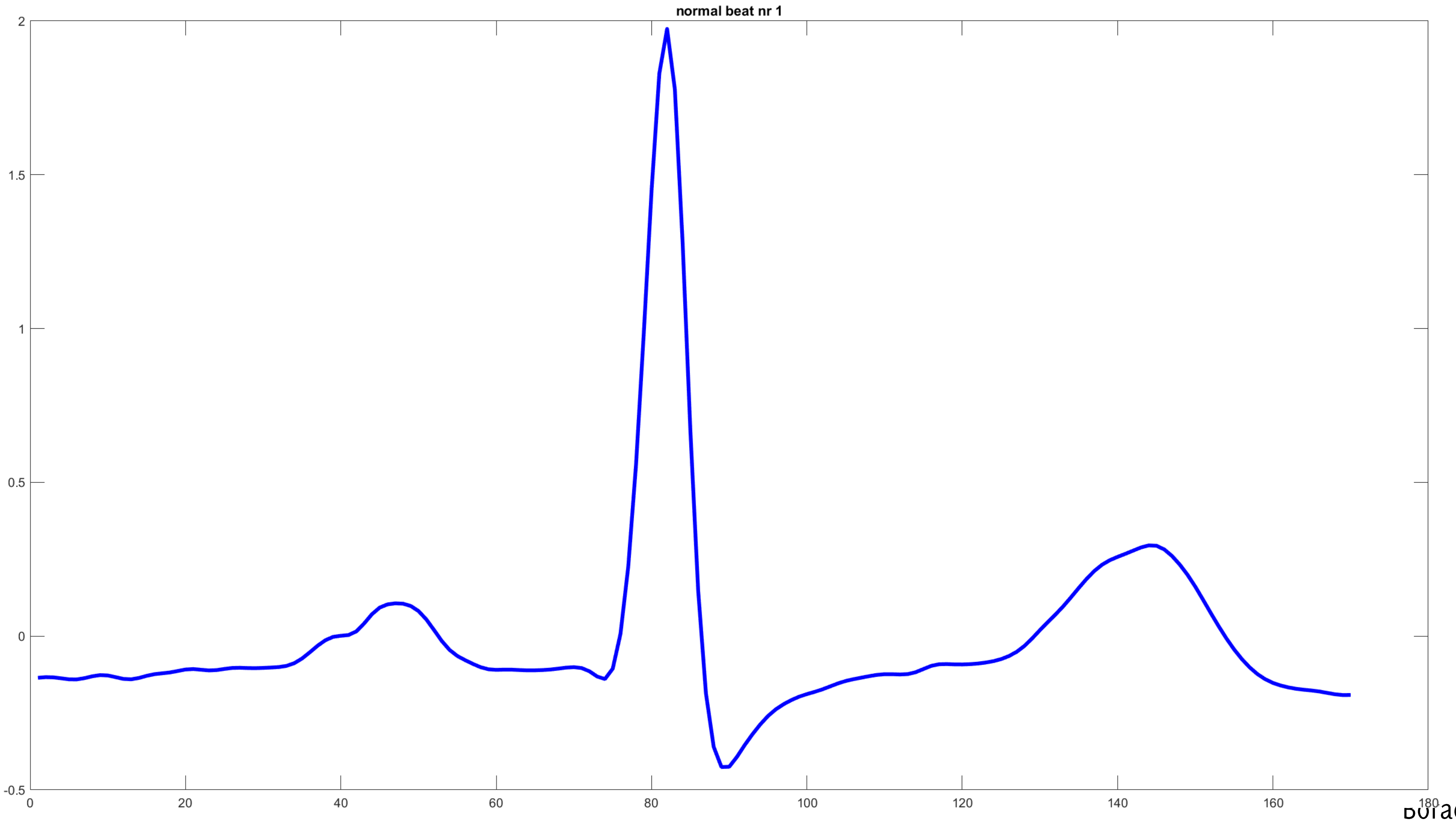
- Each signal is a **row vector, as this is how PCA operates (we used column vectors instead so far)**
- It is convenient to **center each sample before applying transformation.** This can be done by subtracting the mean of each sample.

Training and Test set has been already separated in two matrices.

Annotated labels are provided on the test set.

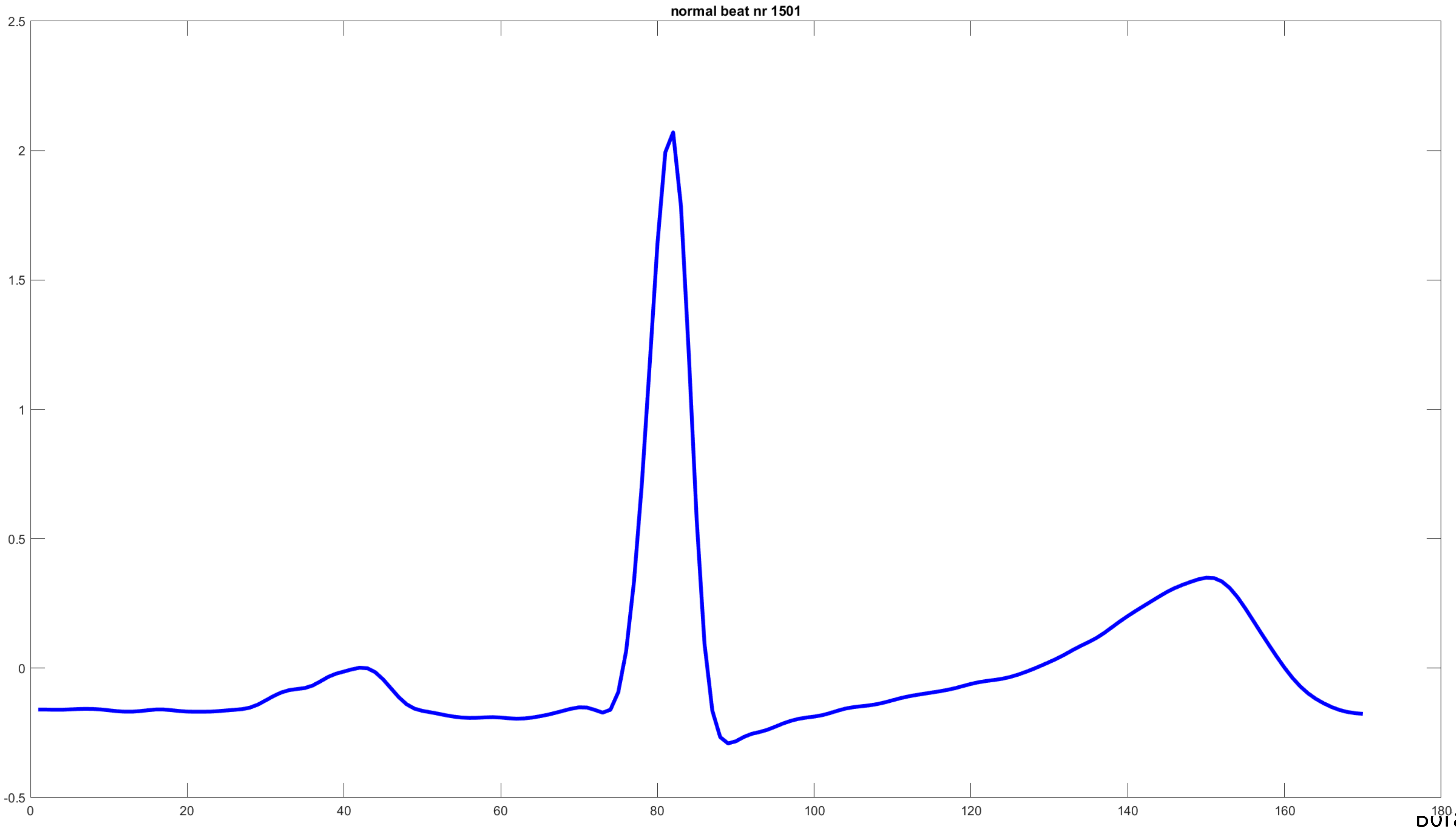
Training set is made of normal data only (semi-supervised settings)

Normal HB

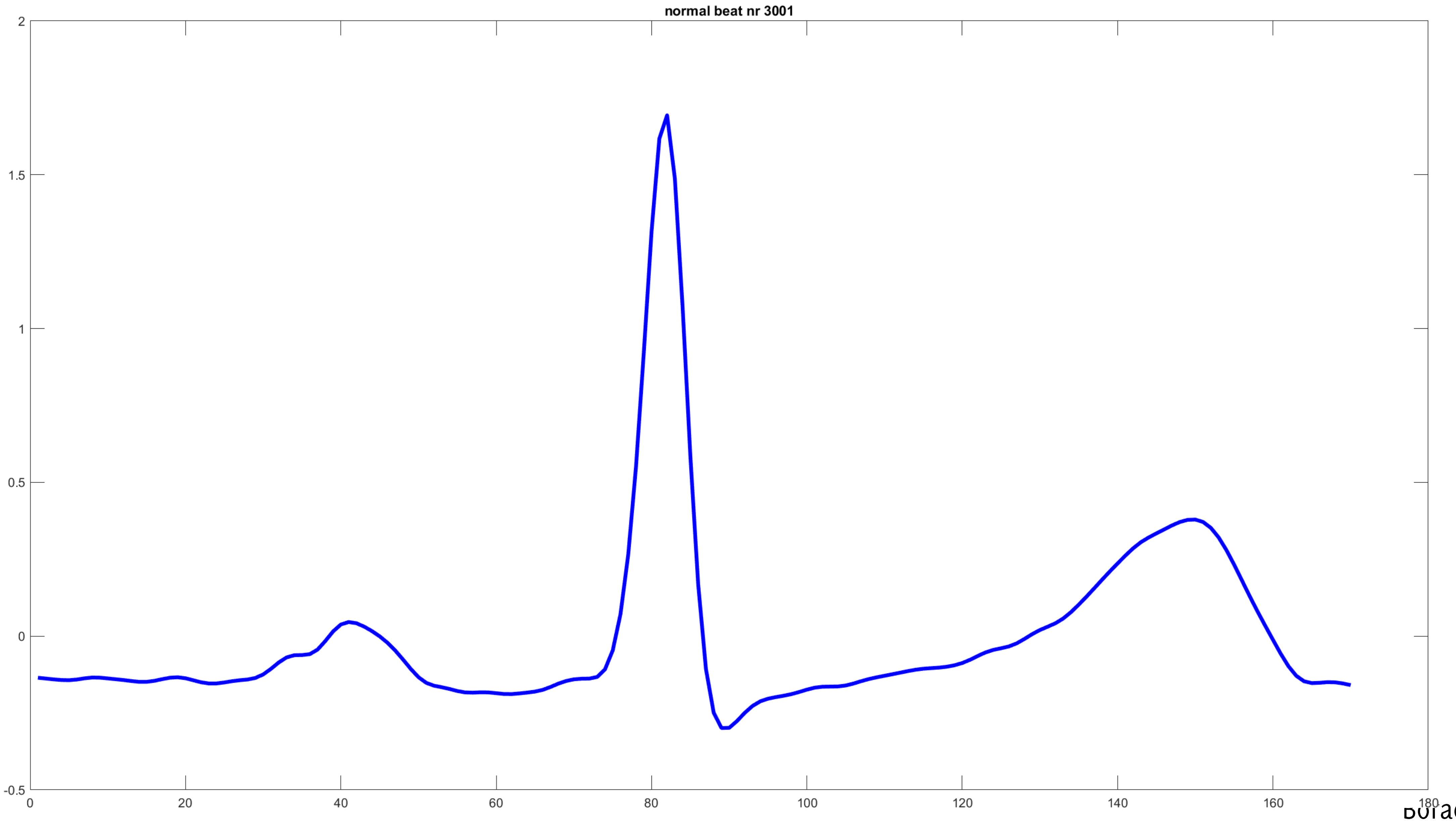


Boracchi, Trovò

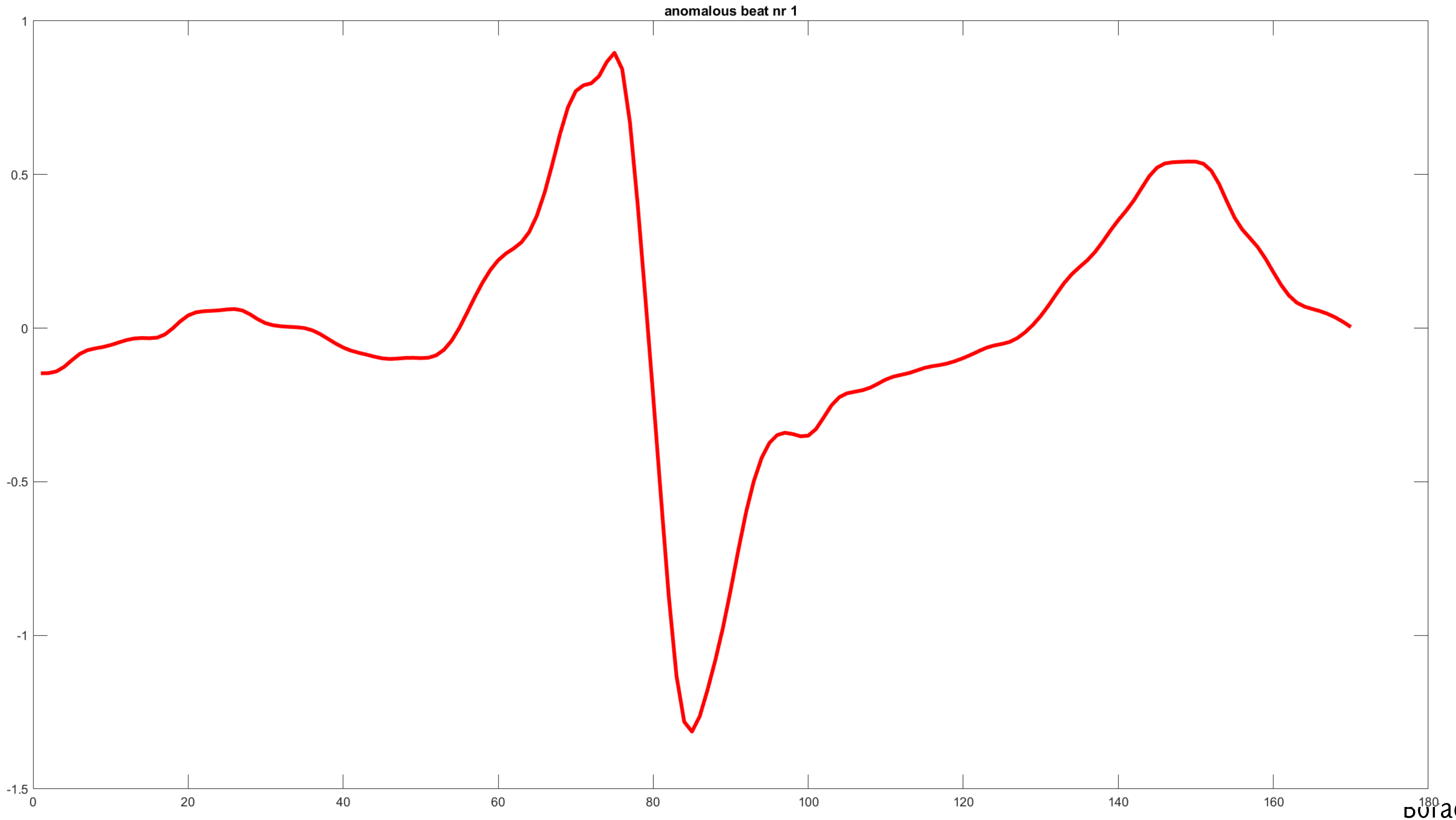
Normal HB


normal beat nr 1501
2.5
2
1.5
1
0.5
0
-0.5
0
20
40
60
80
100
120
140
160
180

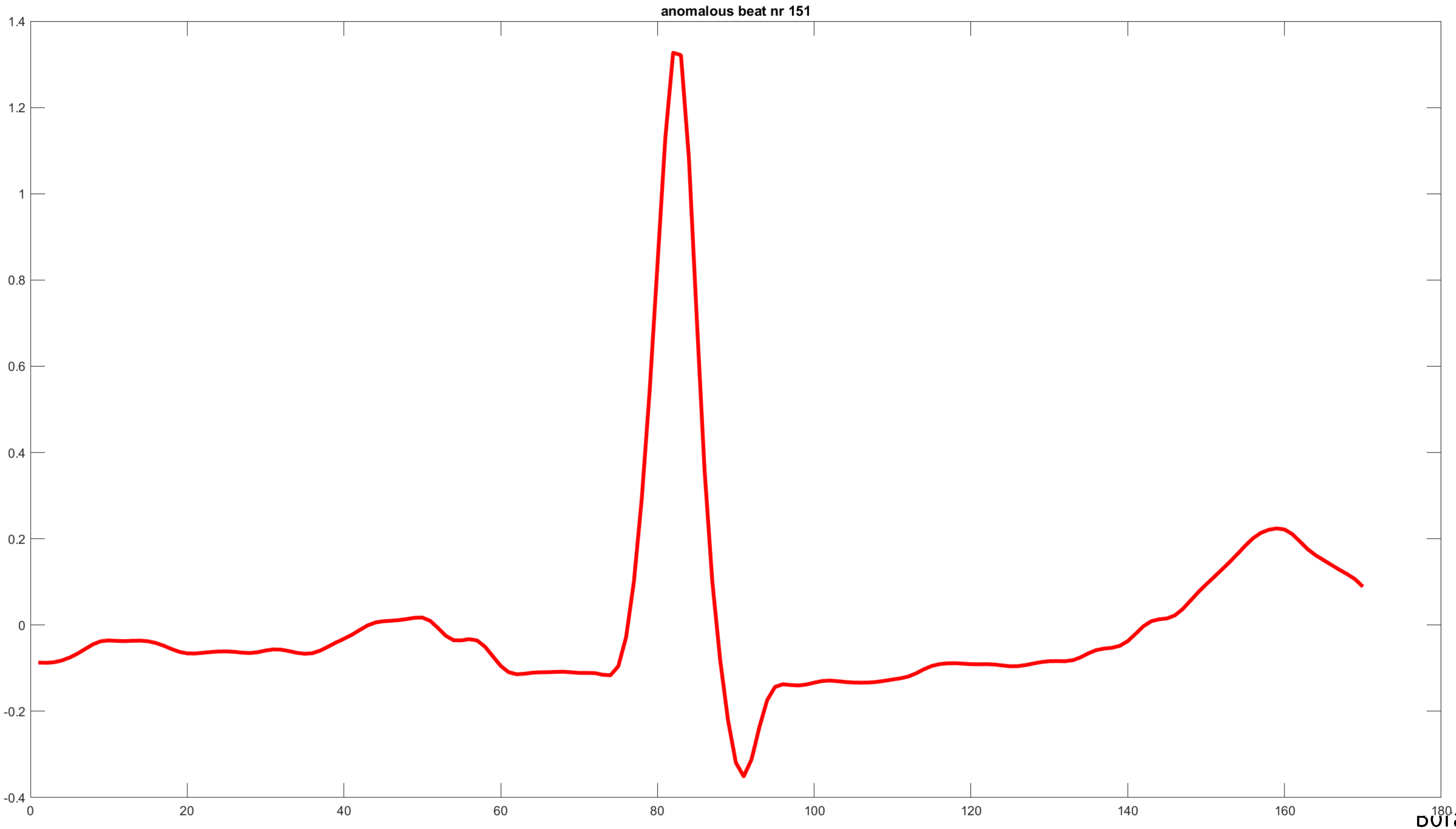
Normal HB



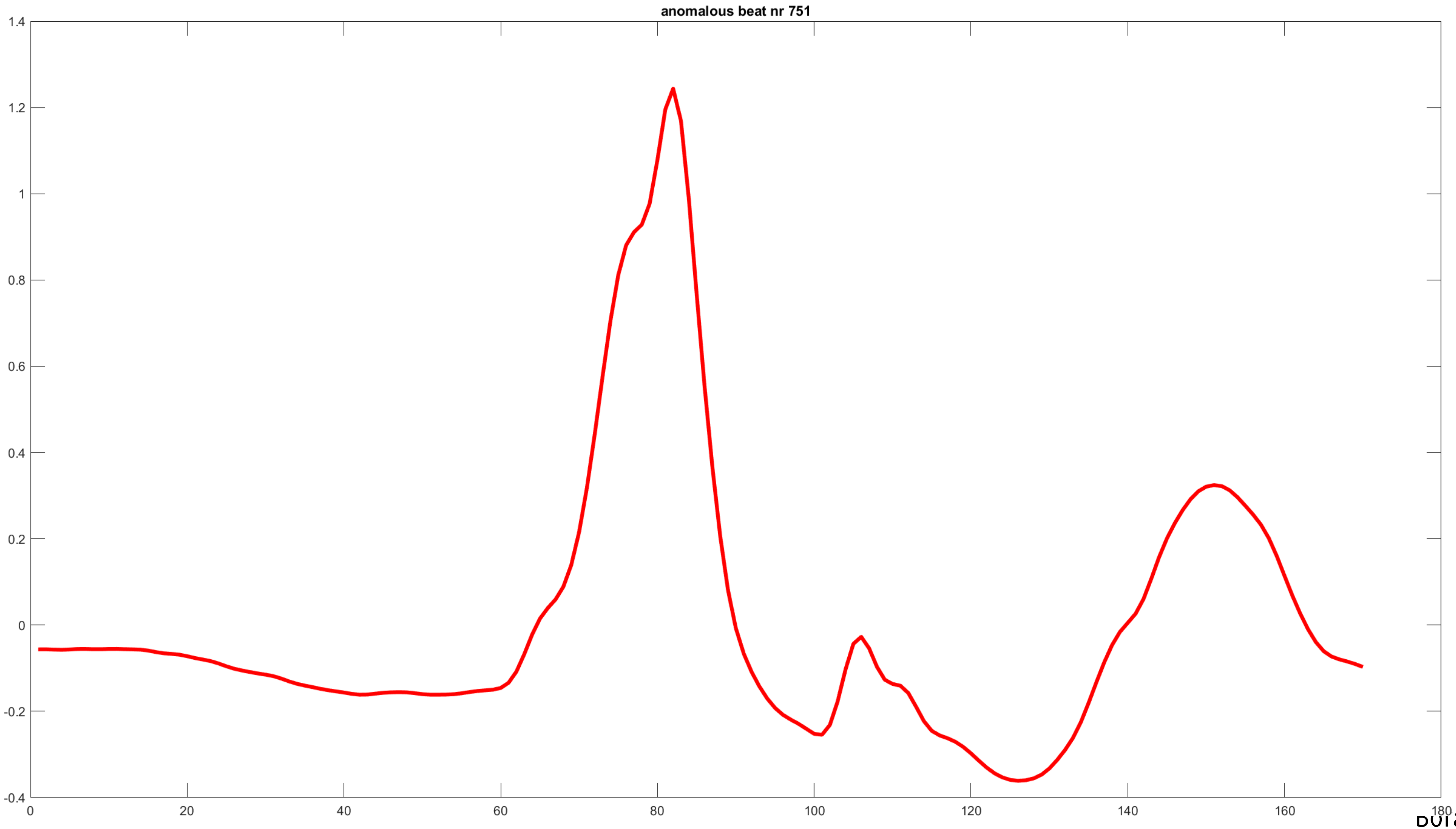
Anomalous HB



Anomalous HB



Anomalous HB

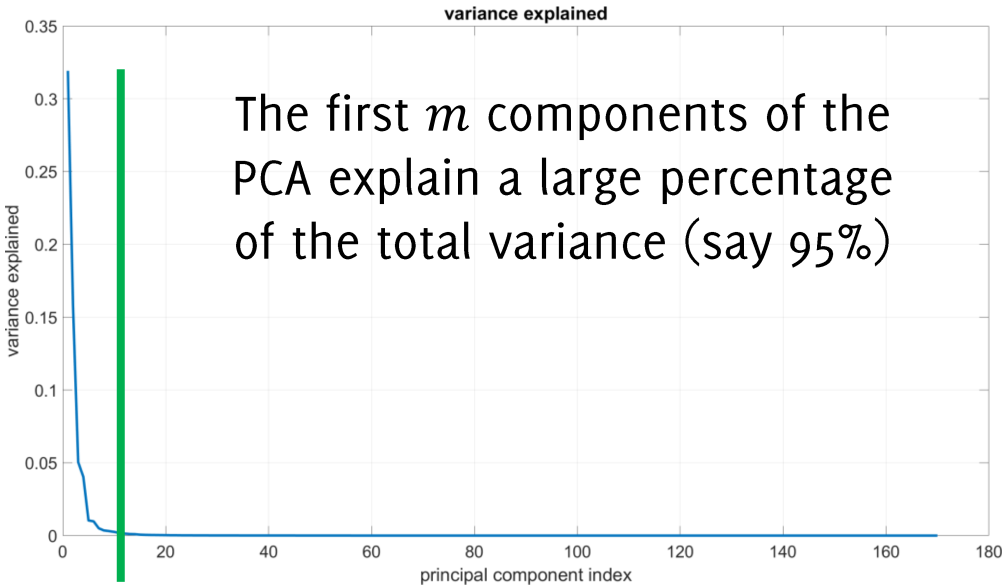


The Model $\mathcal{M}$ for normal data

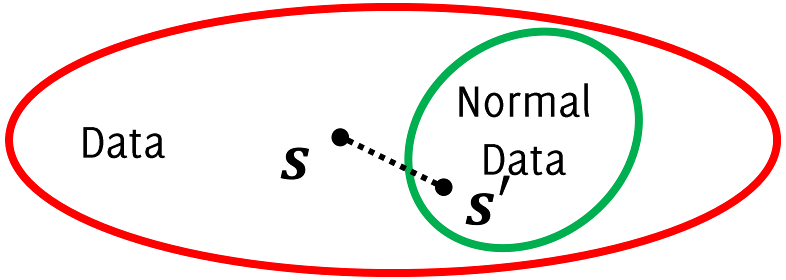
We assume normal data lives in a m −dimensional space that can be learned from the TS

We define this projection by **truncating the PCA transformation** computed over the training set TS

$m << d$



This is the subspace of normal HB that is spanned by the first m PCs



The Projection

Once the first m components of the PCA have been identified, it is possible to compute the coefficients of the projection over the PCA subspace $\forall \boldsymbol{s} \in \mathbb{R}^{1,d}$

$$\boldsymbol{s} \to \boldsymbol{x} = \boldsymbol{s}\, P$$

Being $P \in \mathbb{R}^{d,m}, m \ll d$, $\boldsymbol{x} \in \mathbb{R}^{m}$ are the first m principal components (i.e. the m columns of the PCA transformation matrix)

Remember **now signals are arrange row-wise in vectors**

The Reconstruction

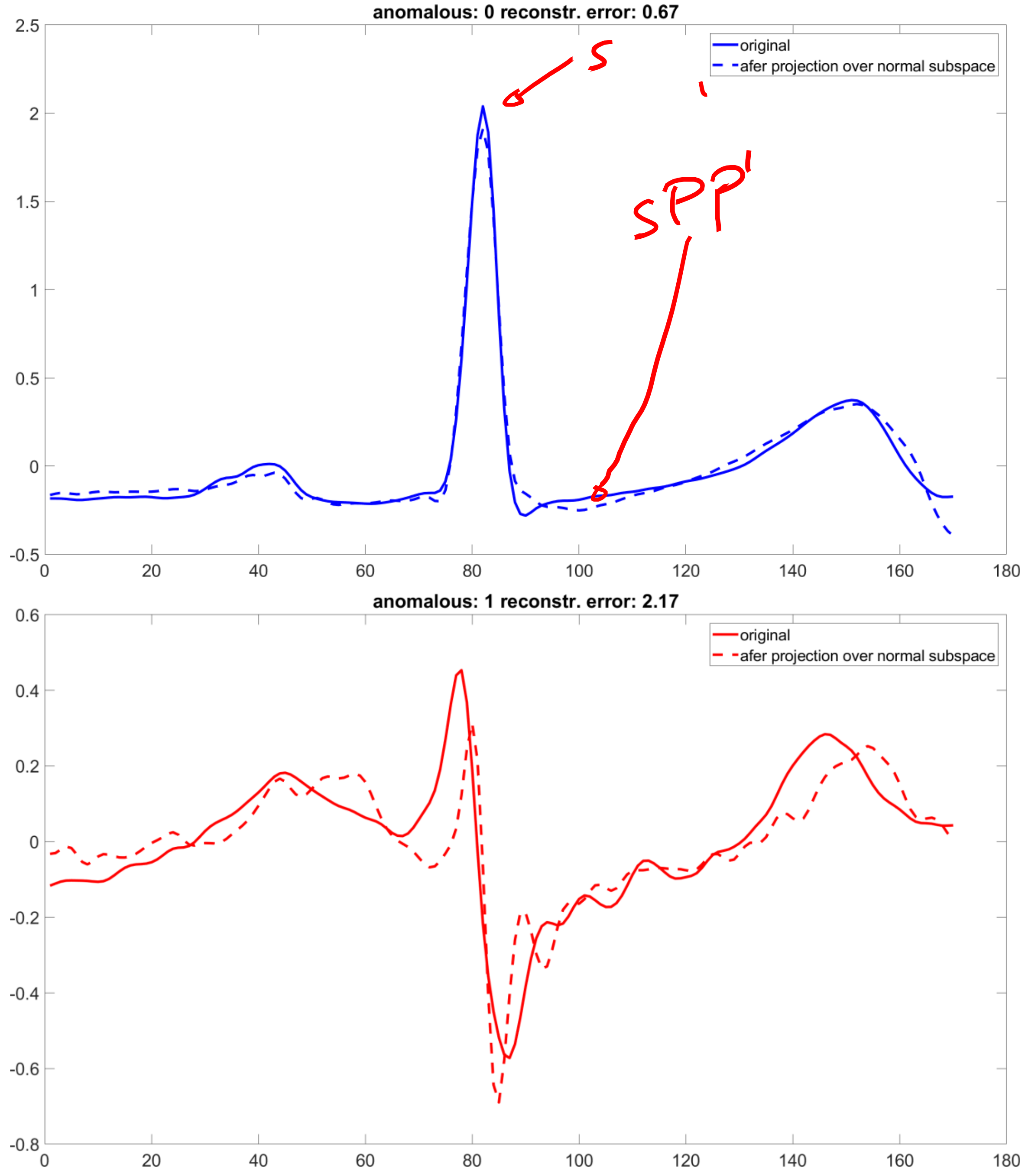
The reconstructed signal from the projection is:

$$\boldsymbol{x}P^T \in \mathbb{R}^d$$

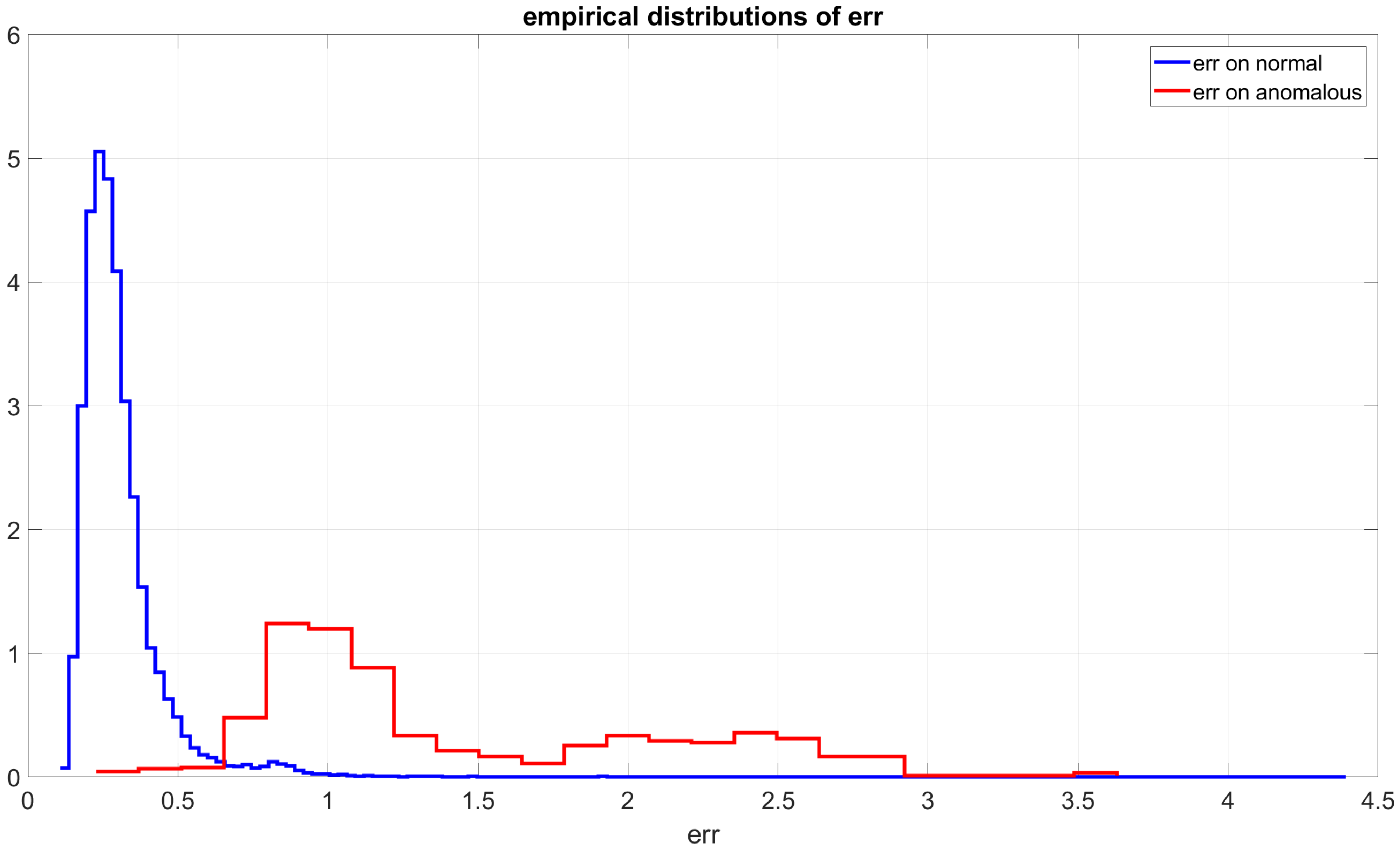
Which can be compared with the original signal

$$\text{err}(\boldsymbol{s}) = ||\boldsymbol{s} - \boldsymbol{s}PP^T||_2$$

That can be used as an anomaly score

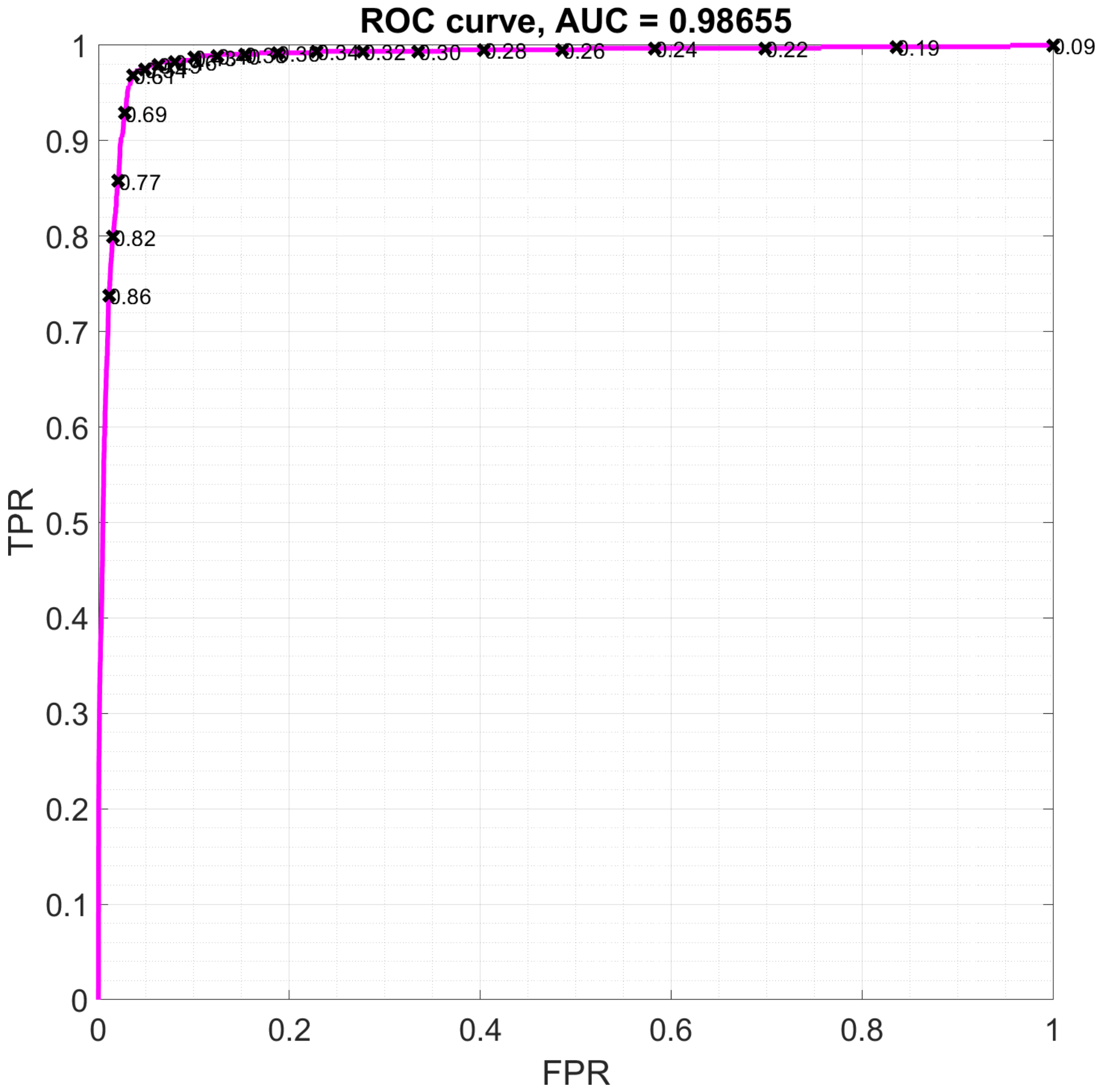


Different Distributions of err($\boldsymbol{s}$)



Boracchi, Trovò

Good Detection by Thresholding



Boracchi, Trovò

Anomaly Detection Based on Sparse Representations

Subspace Methods: Sparse Representations

Basic assumption: normal data live in **a union of low-dimensional subspaces** of the input space

The model learned from S is a matrix: the **dictionary** D.

Each signal is decomposed as **a sum of a few dictionary atoms** (representation is constrained to be **sparse**).

Atoms represent the many **building blocks** that can be used to reconstruct normal signals.

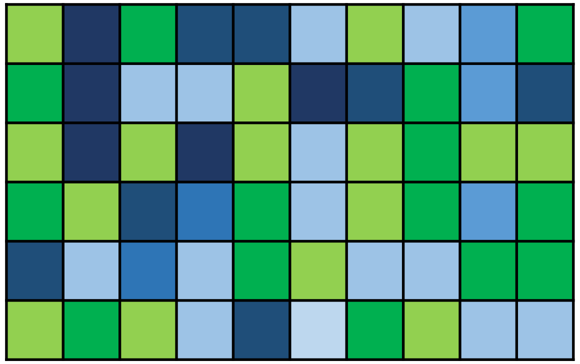
There are typically **more atoms** than the signal dimension.

Effective as long as the learned **dictionary** D is **very specific for normal data**

M. Elad *"Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing"*, Springer, 2010

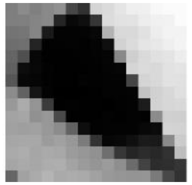
Dictionaries Yielding Sparse Representations

Dictionaries are just matrices! $D \in \mathbb{R}^{d \times m}$

$D =$ 

Dictionaries Yielding Sparse Representations

Dictionaries are just matrices! $D \in \mathbb{R}^{d \times m}$

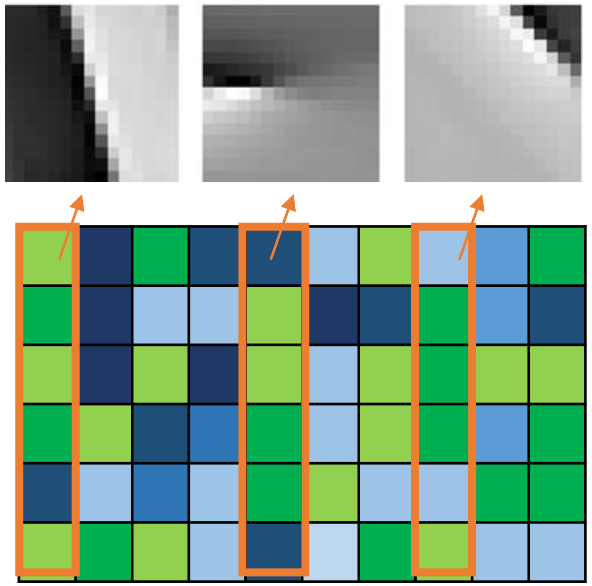


$\boldsymbol{s}$

Each column is an atom:

- lives in the input space
- it is one of the learned building blocks to reconstruct the input signal

$D =$



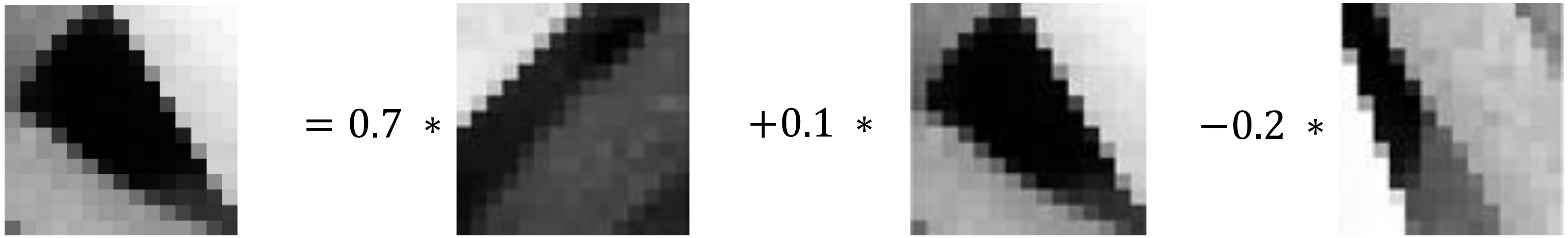
Sparse Representations

Let $\boldsymbol{s} \in \mathbb{R}^n$ be the input signal, a sparse representation is

$$\boldsymbol{s} = \sum_{i=1}^{M} \alpha_i \, \boldsymbol{d_i}$$

a linear combination of **few dictionary atoms** $\{\boldsymbol{d_i}\}$, i.e., most of coefficients are such that $\alpha_i = 0$

An illustrative example in case of our patches

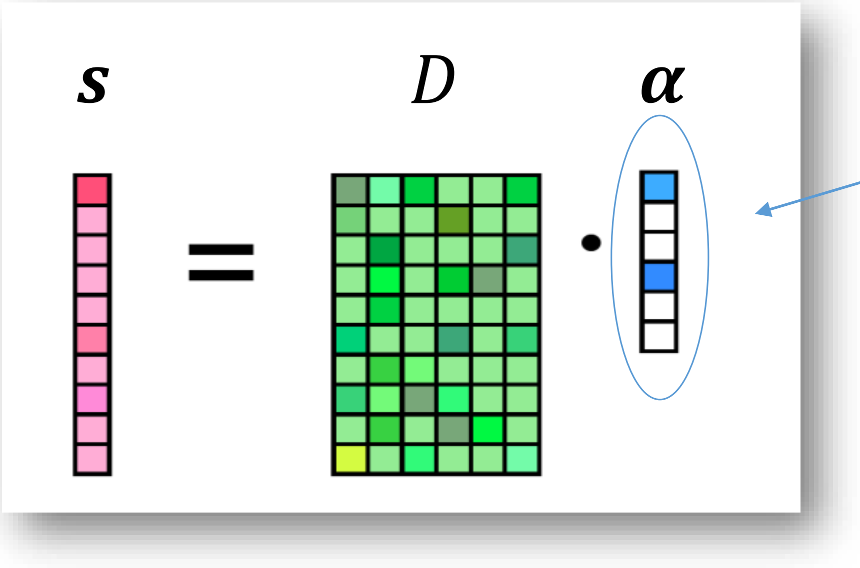


Sparse Representations… Matrix Expression

Let $\boldsymbol{s} \in \mathbb{R}^n$ be the input signal, a sparse representation is

$$\boldsymbol{s} = \sum_{i=1}^{M} \alpha_i \, \boldsymbol{d_i} = D\boldsymbol{\alpha}$$

a linear combination of **few dictionary atoms** $\{\boldsymbol{d_i}\}$ and $||\boldsymbol{\alpha}||_0 < L$, i.e. only a few coefficients are nonzero, i.e. $\boldsymbol{\alpha}$ is sparse.



This vector
$\boldsymbol{\alpha} = [\alpha_1, \quad \dots, \quad \alpha_M]$
is **sparse**

Sparse Coding…

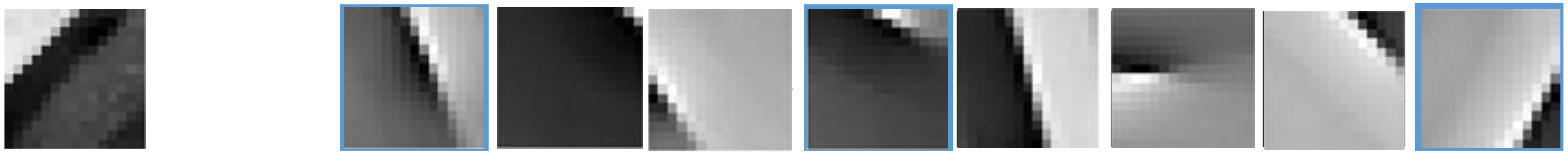
Sprase Coding: computing the sparse representation for an input signal $\boldsymbol{s}$ w.r.t. D

$$\boldsymbol{s} \in \mathbb{R}^d \rightarrow \boldsymbol{\alpha} \in \mathbb{R}^n$$

It is solved as the following optimization problem, (e.g. via the Orthogonal Matching Pursuit, OMP)

$$\boldsymbol{\alpha} = \underset{\boldsymbol{a} \in \mathbb{R}^n}{\operatorname{argmin}} \ ||D\boldsymbol{a} - s||_2 \ \text{ s.t. } \ ||\boldsymbol{a}||_0 < L$$

$\boldsymbol{s}$



$\boldsymbol{\alpha} =$ 0.7 0 0 0.1 0 0 0 -0.2

In the previous illustration $\boldsymbol{\alpha} = [0.7, 0, 0, 0.1, 0, 0, 0, -0.2]$

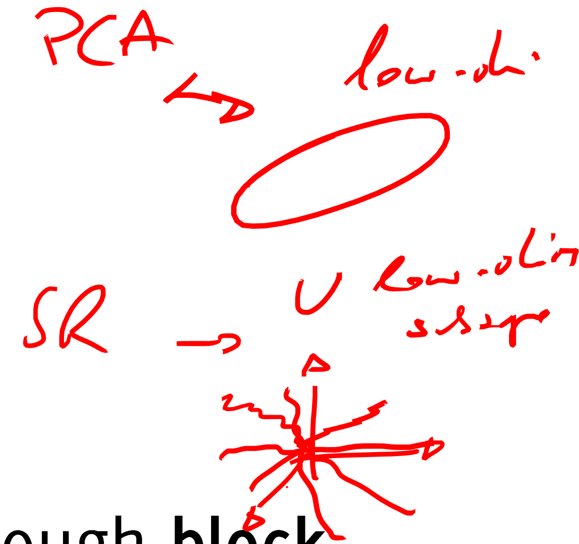
Pati, Y.; Rezaiifar, R.; Krishnaprasad, P. *Orthogonal Matching Pursuit: recursive function approximation with application to wavelet decomposition*. Asilomar Conf. on Signals, Systems and Comput. 1993

… and Dictionary Learning

Dictionary Learning: estimate D from a training set of M

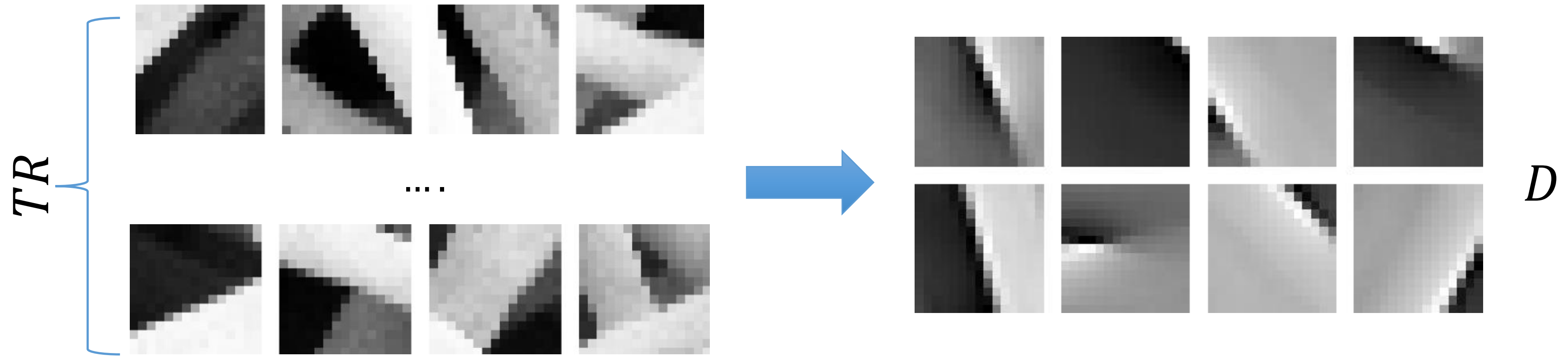
$\in \mathbb{R}^{d \times n}$

$$S = \{\boldsymbol{s_1}, \dots \boldsymbol{s_M}\} \Rightarrow D \in \mathbb{R}^{d \times n}$$



It is solved as the following optimization problem typically through **block-coordinates descent** (e.g. KSVD algorithm)

$$[D, X] = \underset{A \in \mathbb{R}^{d \times n},\, Y \in \mathbb{R}^{n \times M}}{\operatorname{argmin}} \|AY - S\|_2 \ \text{s.t.} \ \|\boldsymbol{y_i}\|_0 < L, \quad \forall \boldsymbol{y_i}$$



Aharon, M.; Elad, M. Bruckstein, A. *K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation* IEEE TSP, 2006

Sparse representation monitoring: statistics

Anomalies can be directly **detected during the sparse coding** stage, by changing the functional being optimized.

A set of test signals is modeled as:

$$S = DX + E + V$$

where X is sparse, V is a noise term, and E is a matrix having most columns set to zero. Columns $\boldsymbol{e}_i \neq \mathbf{0}$ indicate anomalies, as they do not admit a sparse representation w.r.t. D

A. Adler, M. Elad, Y. Hel-Or, and E. Rivlin, *"Sparse coding with anomaly detection"* Journal of Signal Processing Systems, vol. 79, no. 2, pp. 179–188, 2015.

Sparse representation monitoring: statistics

Anomalies can be detected by solving (through ADMM) the following sparse coding problem

$$\underset{X,E}{\operatorname{argmin}}\left(\frac{1}{2}\,\|S - DX - E\|_F^2 + \lambda\|X\|_1 + \mu\|E\|_{2,1}\right)$$

Data-fidelity for normal data — $\|S - DX - E\|_F^2$

Sparsity — $\|X\|_1$

Group sparsity regularization, only a few columns can be nonzero — $\|E\|_{2,1}$

.. and identifying as anomalies the signals corresponding to columns of E that are nonzero.

A. Adler, M. Elad, Y. Hel-Or, and E. Rivlin, "*Sparse coding with anomaly detection*" Journal of Signal Processing Systems, vol. 79, no. 2, pp. 179–188, 2015.

If you want to know more:

«Learning Sparse Representations for Image and Signal Modeling»

PhD course 2021

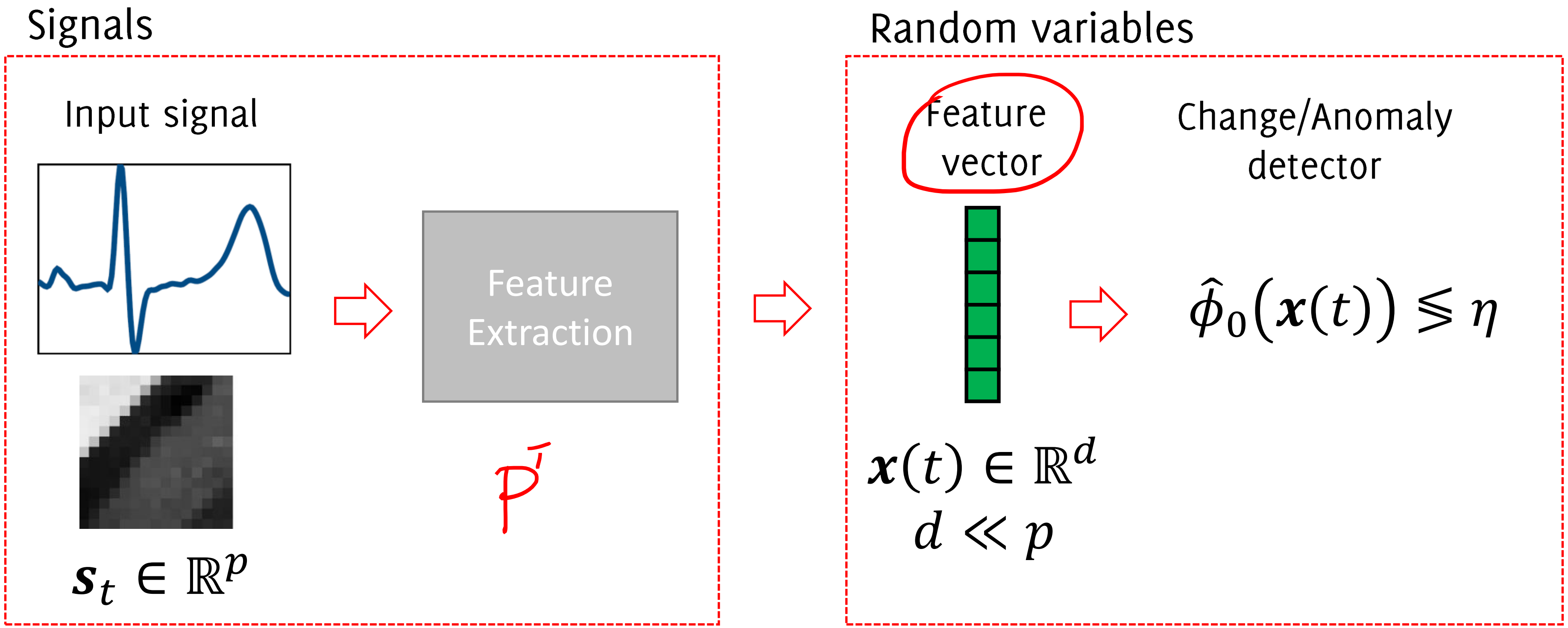
Semi-supervised AD methods out of the RVW

Out of the "Random Variable" world

- Detrending-based methods
- Reconstruction-based methods
  - Subspace methods
- Feature-based monitoring
  - Expert-driven Features
  - Data-driven Features

Monitoring Features

Feature extraction: meaningful indicators to be monitored which have a known / controlled response w.r.t. normal data



Feature Extraction: signal processing, a priori information, learning methods

The customary framework for change / anomaly detection

V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

Feature Extraction

The peculiar structures of normal images and signals suggest that **normal data live in a manifold** having **lower dimension** than the input domain

Data dimensionality can be reduced by extracting features

Good features should:

- Yield a **stable response** w.r.t. **normal data**
- Yield **unusual response** on **anomalies** / when data change

Reconstruction error and representation coefficients can be considered features.

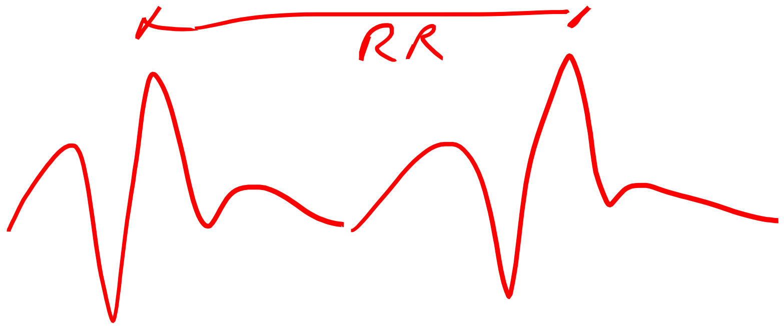
Features can be monitored in either one-shot/sequential monitoring schemes.

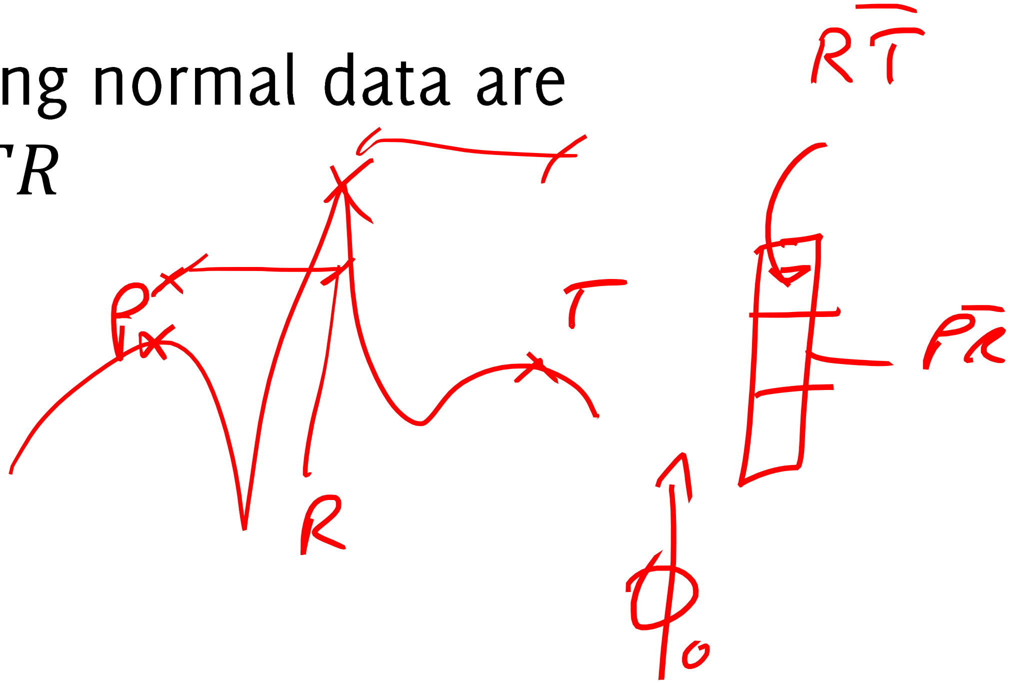
V. Chandola, A. Banerjee, V. Kumar. "*Anomaly detection: A survey*". ACM Comput. Surv. 41, 3, Article 15 (2009), 58 pages.

Feature Extraction approaches

There are two major approaches for extracting features:

- **Expert-driven (hand-crafted) features**: computational expressions that are **manually designed by experts** to distinguish between normal and anomalous data
- **Data-driven features**: **features** characterizing normal data are automatically **learned from training data** TR





Semi-supervised AD methods out of the RVW

Out of the "Random Variable" world

- Detrending-based methods
- Reconstruction-based methods
  - Subspace methods
- Feature-based monitoring
  - Expert-driven Features
  - Data-driven Features

Examples of Expert-Driven Features

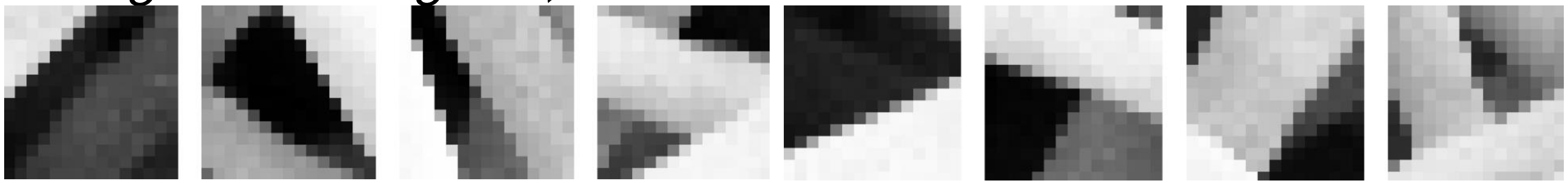
Expert-driven features: each patch of an image s

$$\mathbf{s}_c = \{s(c+u), u \in \mathcal{U}\}$$

Example of features are:

- the average,
- the variance,
- the total variation (the energy of gradients)

These can hopefully **distinguish normal** and **anomalous** patches (since image in anomalous region is expected to be flat or without edges characterizing normal regions)



Carrera D., Manganini F., Boracchi G., Lanzarone E. *"Defect Detection in SEM Images of Nanofibrous Materials"*, IEEE Transactions on Industrial Informatics 2017

Semi-supervised AD methods out of the RVW

Out of the "Random Variable" world

- Detrending-based methods
- Reconstruction-based methods
  - Subspace methods
- Feature-based monitoring
  - Expert-driven Features
  - Data-driven Features

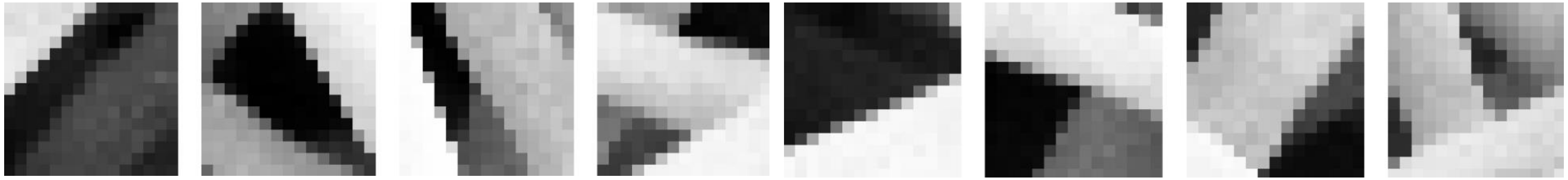
Examples of Data-Driven Features

Analyze each patch of an image s

$$\mathbf{s}_c = \{s(c+u), u \in \mathcal{U}\}$$

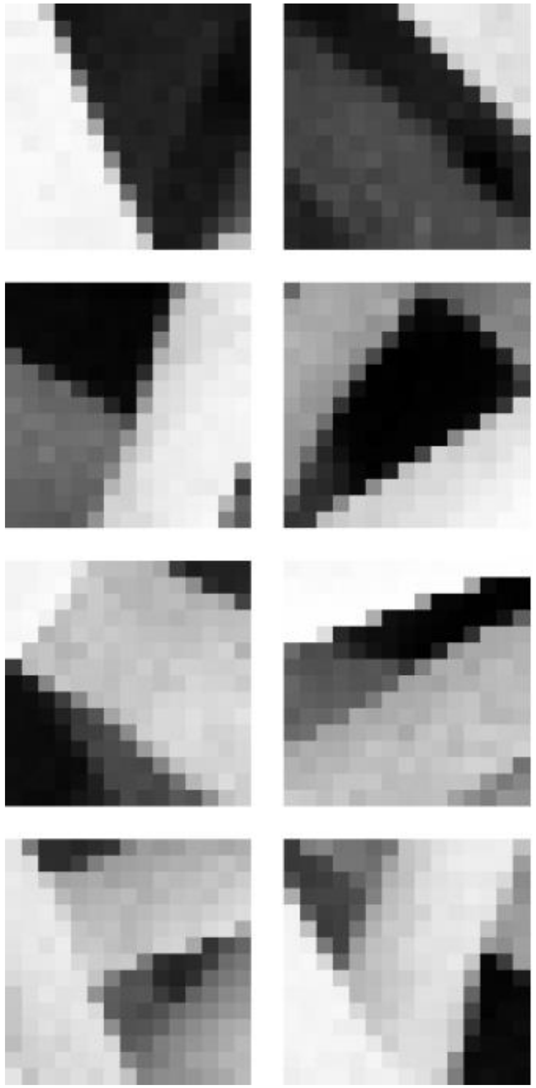
and determine whether it is normal or anomalous.

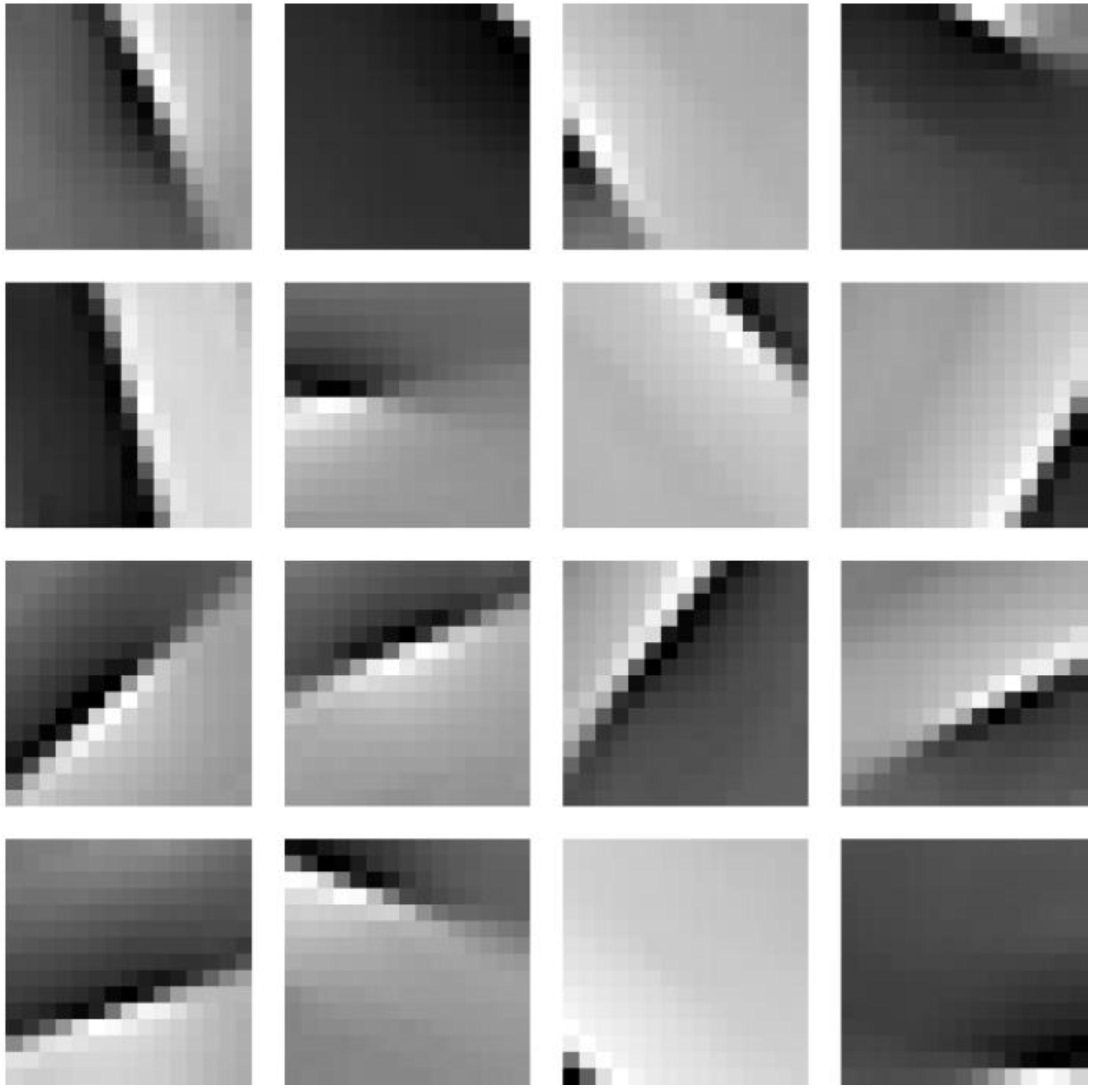
Data driven features: expressions to **quantitatively assess whether test patches conform** or not **with the model**, learned from normal data.



Carrera D., Manganini F., Boracchi G., Lanzarone E. *"Defect Detection in SEM Images of Nanofibrous Materials"*, IEEE Transactions on Industrial Informatics 2017, 11 pages, doi:10.1109/TII.2016.2641472

A Learned Dictionary from normal patches





Data-Driven Features

To assess the conformance of $\boldsymbol{s}_c$ with D we solve the following

Sparse coding:

$$\boldsymbol{\alpha} = \underset{\widetilde{\boldsymbol{\alpha}} \in \mathbb{R}^n}{\operatorname{argmin}} \|D\widetilde{\boldsymbol{\alpha}} - \mathbf{s}\|_2^2 + \lambda \|\widetilde{\boldsymbol{\alpha}}\|_1, \qquad \lambda > 0$$

which is the BPDN formulation and we solve using ADMM.

The penalized ℓ^1 formulation has more degrees of freedom in the reconstruction, **the conformance of $\boldsymbol{s}$ with $\boldsymbol{D}$ have to be assessed monitoring both terms of the functional**

Boyd S., Parikh N., Chu E., Peleato B., Eckstein J., ***"Distributed optimization and statistical learning via the alternating direction method of multipliers"*** **2011**

Data-driven features

Features then include both the **reconstruction error**

$$\text{err}(\boldsymbol{s}) = \|D\boldsymbol{\alpha} - \mathbf{s}\|_2^2$$

and **the sparsity** of the representation

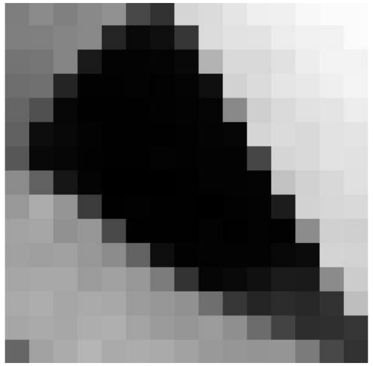
$$\|\boldsymbol{\alpha}\|_1$$

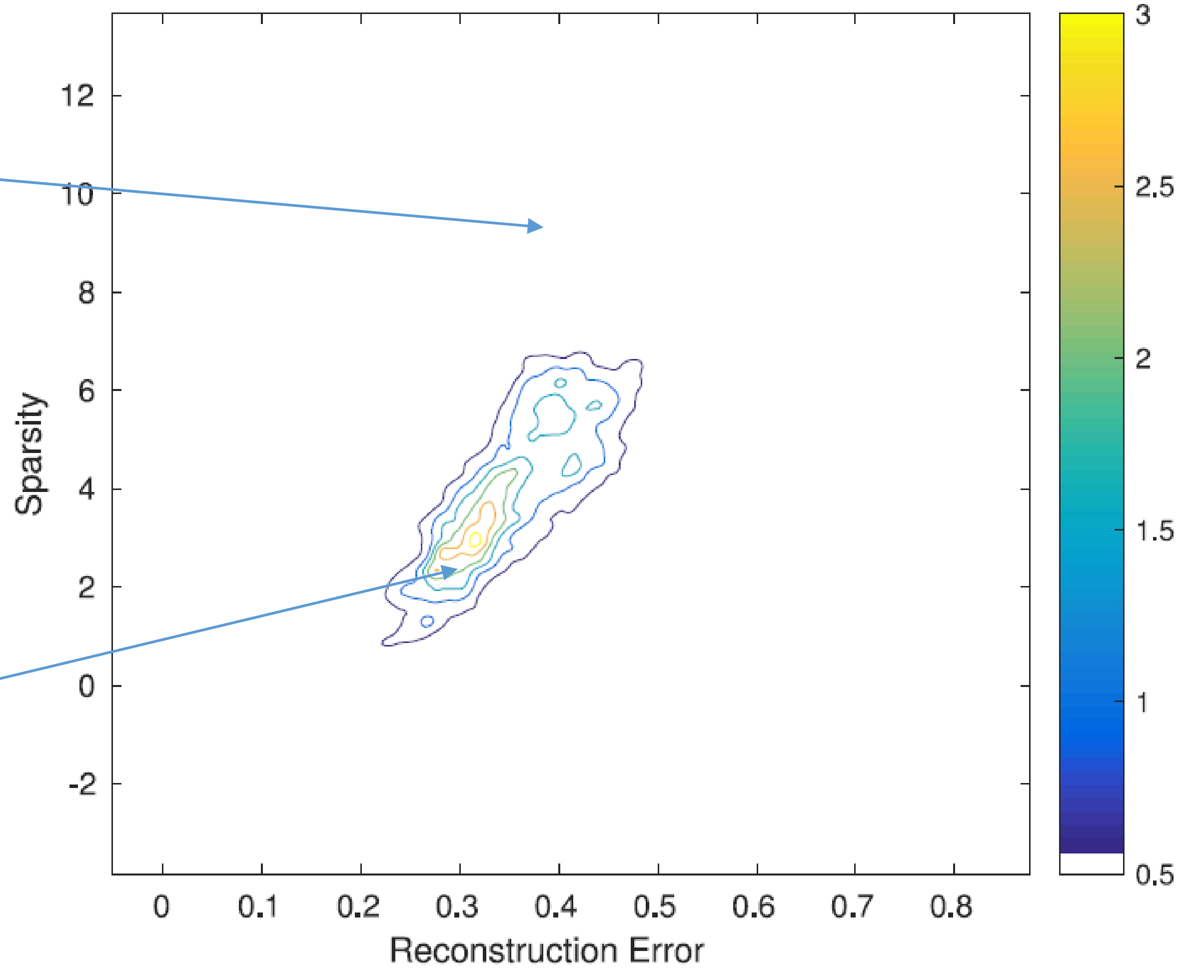
Thus obtaining **a data-driven feature vector** $\boldsymbol{x} = \begin{bmatrix} \|D\boldsymbol{\alpha} - \mathbf{s}\|_2^2 \\ \|\boldsymbol{\alpha}\|_1 \end{bmatrix}$

Density-based monitoring on Data-driven features

Anomalies

Normal patches:





Data-driven features

Training:

- **Learn** from $TR\backslash V$ the dictionary D
- **Learn** from V, the distribution $\hat{\phi}_0$ of normal features vectors $\boldsymbol{x}$.

Testing:

- Compute feature vectors $\boldsymbol{x}$ via sparse coding
- Detect anomalies when $\hat{\phi}_0(\boldsymbol{x}) < \eta$

Carrera D., Manganini F., Boracchi G., Lanzarone E. *"Defect Detection in SEM Images of Nanofibrous Materials"*, IEEE Transactions on Industrial Informatics 2017, 11 pages, doi:10.1109/TII.2016.2641472

Data-driven features

Training:

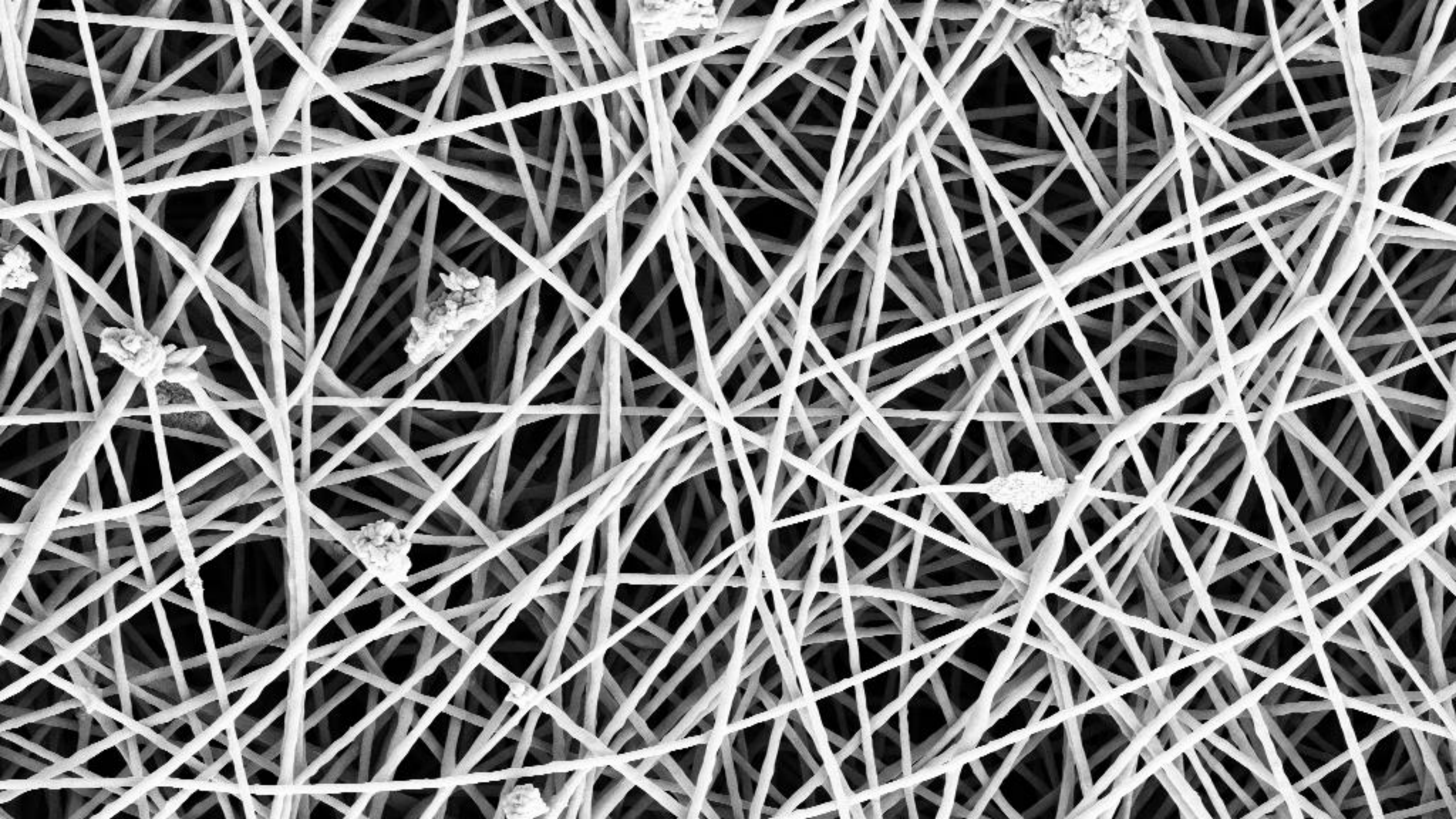
- **Learn** from $TR \backslash V$ the dictionary D
- **Learn** from V, the distribution $\hat{\phi}_0$ of normal features vectors $\boldsymbol{x}$.

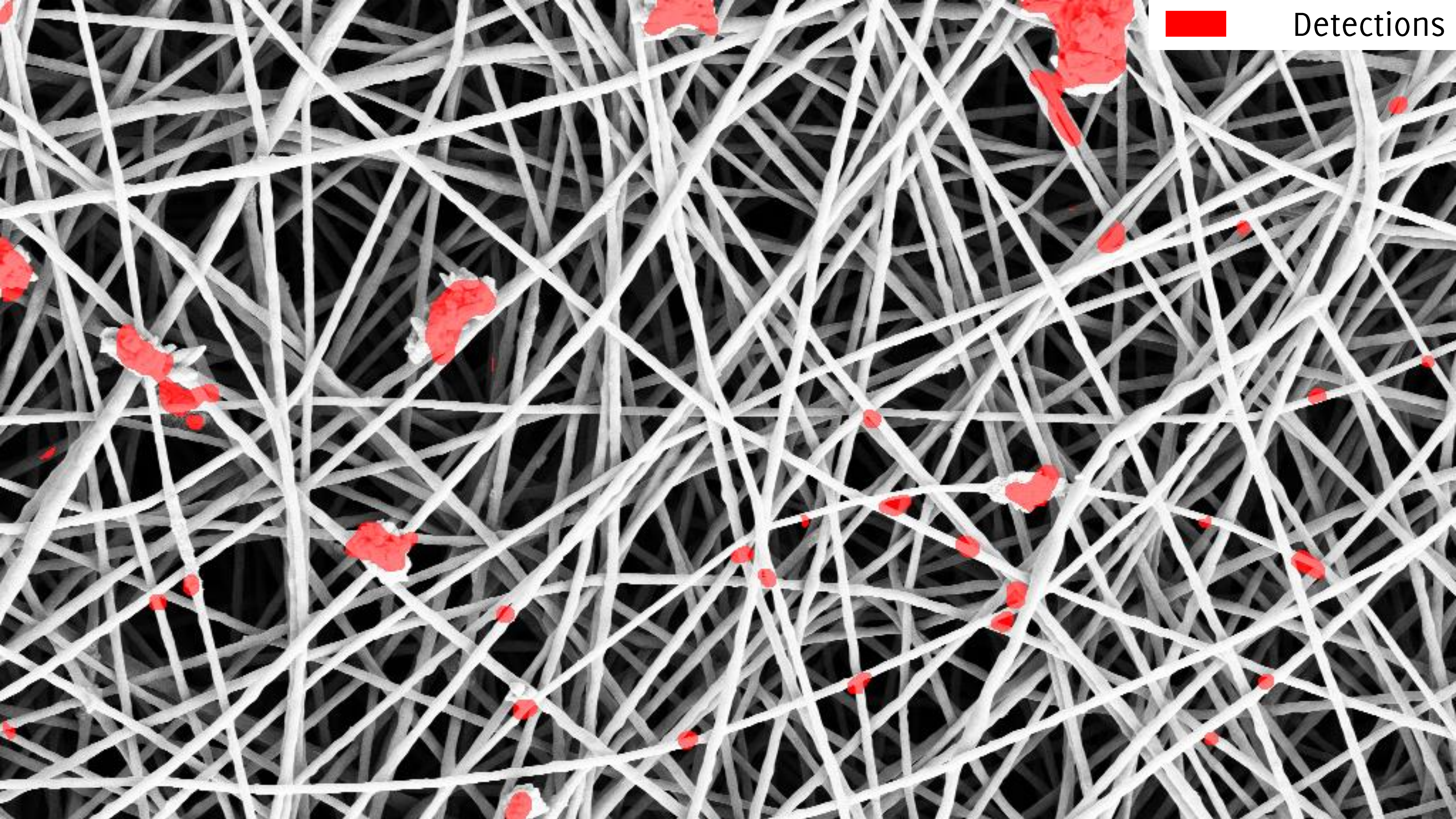
Testing:

- Compute feature vectors $\boldsymbol{x}$ via sparse coding
- Detect anomalies when $\hat{\phi}_0(\boldsymbol{x}) < \eta$

This solution is rather flexible and can be adapted when operating conditions changes (e.g. different zooming level)

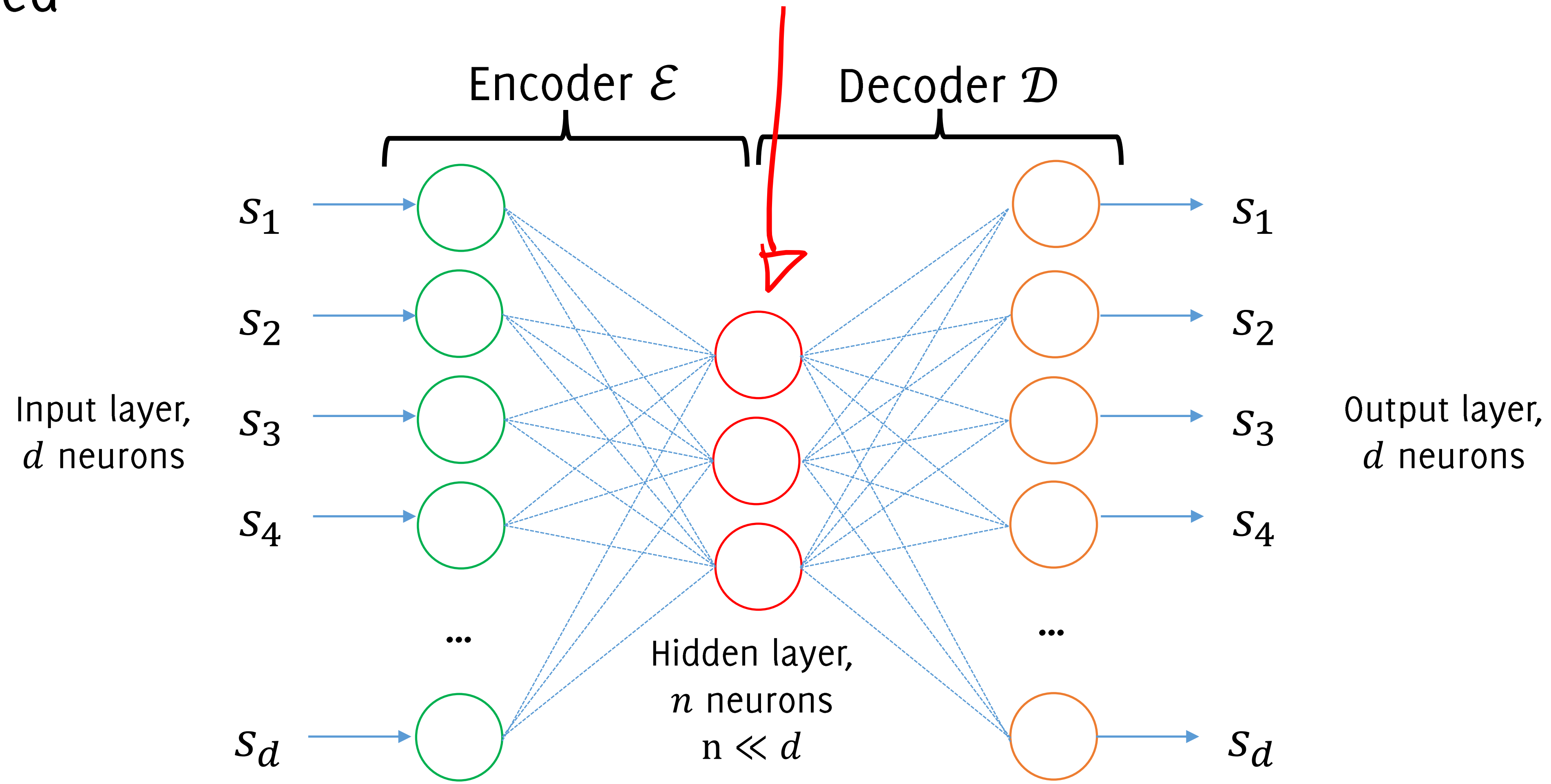
Carrera D., Boracchi G., Foi A., Wohlberg B. *"Scale-invariant Anomaly Detection With multiscale Group-sparse Models"* ICIP 2016




Detections

Feature-based Methods

Autoencoders can be also used in feature-based monitoring schemes, where the hidden representation of the input is the feature being monitored



Monitoring Feature Distribution

Detection by **feature monitoring** (AE notation)

Training (Monitoring Feature Distribution):

- Learn the autoencoder $\mathcal{D}(\mathcal{E}(\cdot))$ from the training set S
- Fit a density model $\hat{\phi}_0$ to the encoded features

$$\{\mathcal{E}(\boldsymbol{s}), \boldsymbol{s} \in V\}$$

  over a validation set $V \neq S$

- Define a suitable threshold γ for $\hat{\phi}_0(\boldsymbol{s})$

Testing (Monitoring Feature Distribution):

- Encode each incoming signal $\boldsymbol{s}$ through $\mathcal{E}$
- Detect anomalies if $\hat{\phi}_0(\mathcal{E}(\boldsymbol{s})) < \gamma$

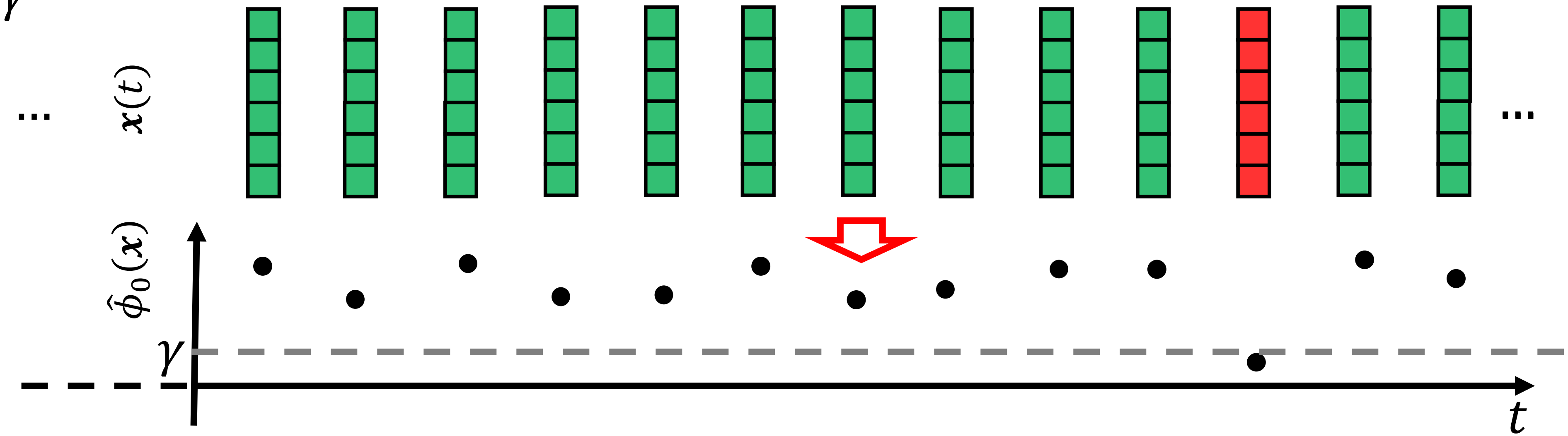
Monitoring Feature Distribution

Normal data are expected to yield $\mathcal{E}(\boldsymbol{s})$ that are **i.i.d. vectors (or features)** and that **follow an** unknown **distribution** ϕ_0.

Anomalous data do not, as they follow $\phi_1 \neq \phi_0$.

We are **back to our statistical framework** and we can

- learn $\hat{\phi}_0$ from a set features extracted from normal data
- detect anomalous data by computing $\boldsymbol{x} = \mathcal{E}(\boldsymbol{s})$ and then check whether $\hat{\phi}_0(\boldsymbol{x}) < \gamma$



Deep Learning Features

CNNs as data-driven feature extractor

The super-human performance achieved by CNNs in many fields indicates these are very **powerful feature extractors**.

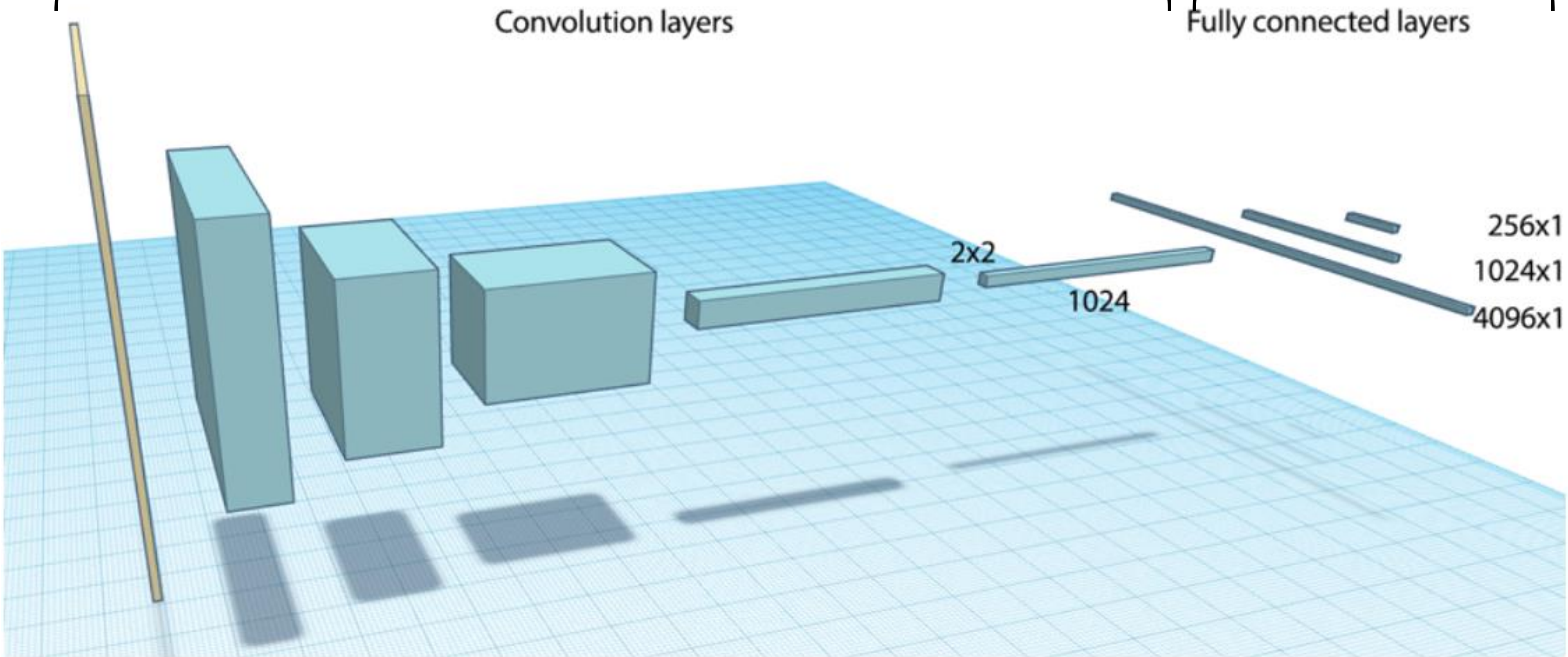
Not surprisingly, these have been also used for **anomaly detection**, giving rise to multiple approaches:

- Transfer learning
- Autoencoders
- Self-supervised learning
- Domain-based
- Generative-based

CNNs as data-driven feature extractor

Extract high-level features from pixel data

Classify



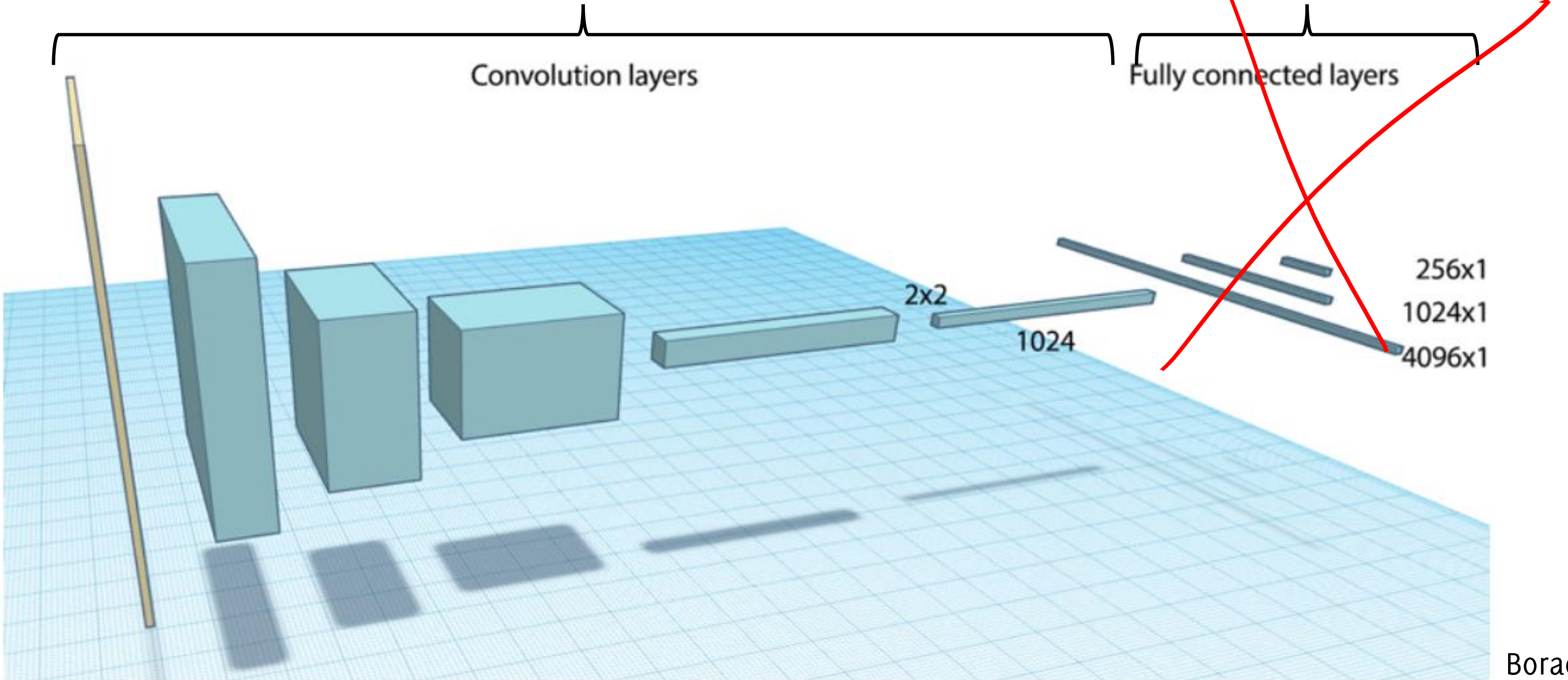
Boracchi, Trovò

CNNs as data-driven feature extractor

The feature vector extracted from the last layer can be modeled as a random vector

Extract high-level features from pixel data

Classify



Boracchi, Trovò

CNNs as data-driven feature extractor

- Transfer learning
- Autoencoders
- Self-supervised learning
- Domain-based
- Generative-based

Transfer Learning Supervised CNNs for AD

Idea:

- Use a pretrained network CNN (e.g. AlexNet), that was trained for a different task and on a different dataset
- Throw away the last layer(s)
- Use the CNN to build a new dataset TR' from TR:

$$TR' = \{\psi(\mathbf{s}_i),\ \mathbf{s}_i \in TR\}$$

- Train your favorite anomaly detector on TR'

CNN (s_i)

Transfer Learning Supervised CNNs for AD

- Features extracted from a CNN, i.e., $\psi(\boldsymbol{s})$ is typically very large for deep networks (e.g. ResNET). **Reduce data-dimensionality** by PCA defined on a set of normal features
- **Anomalies** can be **detected by measuring distance w.r.t. normal features**, possibly using clustering to speed up performance.
- Thresholds can be computed by the three-sigma rule or bootstrap.

Napoletano P., Piccoli F., Schettini R., *"Anomaly Detection in Nanofibrous Materials by CNN-Based Self-Similarity"*, Sensors 2018

Transfer Learning Supervised CNNs for AD

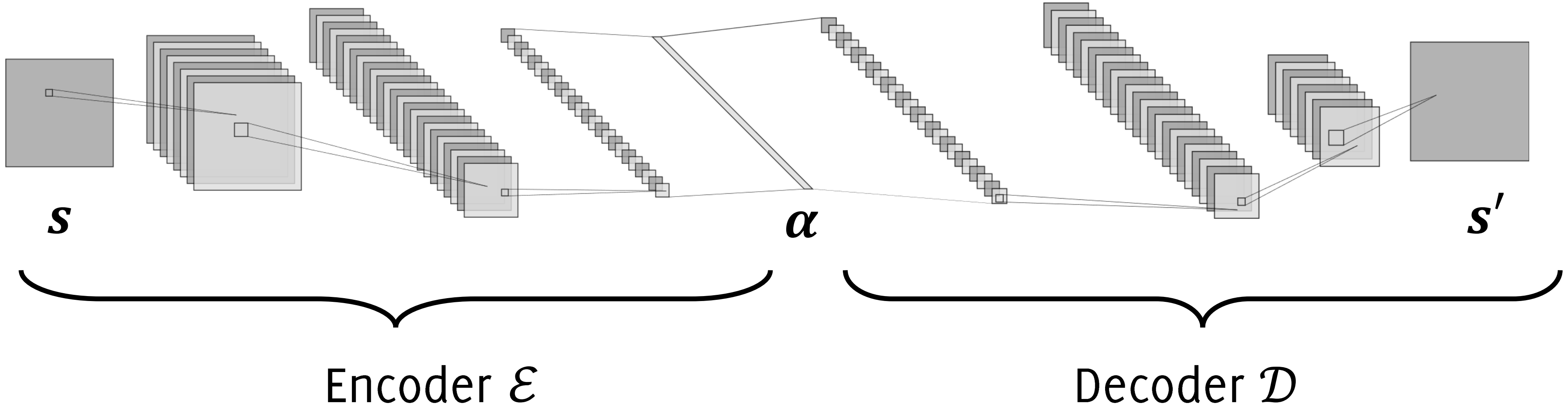
Pros: pretrained networks are very powerful models, since they usually trained on datasets with million of images

Cons: the network is **not trained on normal** data. Meaningful structures in normal images might not be successfully captured by network trained on images from a different domain (e.g. medical vs natural images)

CNNs as data-driven feature extractor

- Transfer learning
- Autoencoders
- Self-supervised learning
- Domain-based
- Generative-based

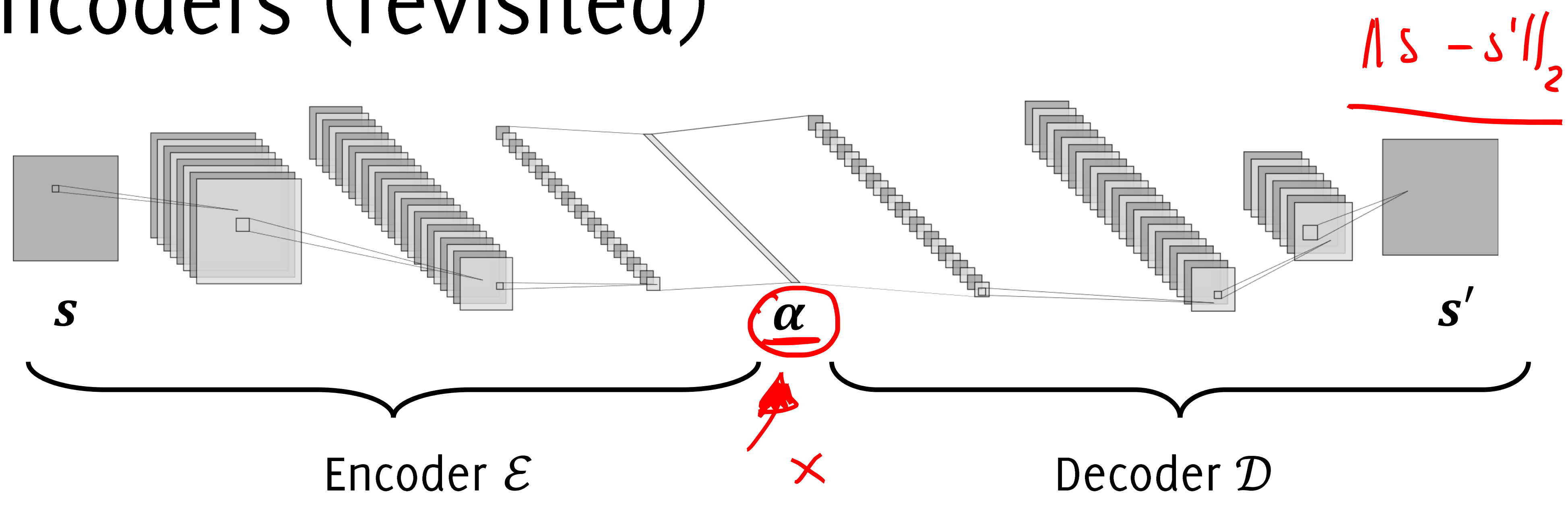
Autoencoders (revisited)



Autoencoders can be trained directly on normal data by minimizing the reconstruction loss:

$$\sum_{s \in TR} \left\| \boldsymbol{s} - \mathcal{D}\big(\mathcal{E}(\boldsymbol{s})\big) \right\|_2$$

Autoencoders (revisited)



We can fit a **density model** (e.g. Gaussian Mixture) on $\boldsymbol{\alpha} = \mathcal{E}(\boldsymbol{s})$:

$$\boldsymbol{\alpha} \sim \sum_i \pi_i \varphi_{\boldsymbol{\mu}_i, \Sigma_i},$$

Where $\varphi_{\boldsymbol{\mu}_i, \Sigma_i}$ is the pdf of $\mathcal{N}(\boldsymbol{\mu}_i, \Sigma_i)$

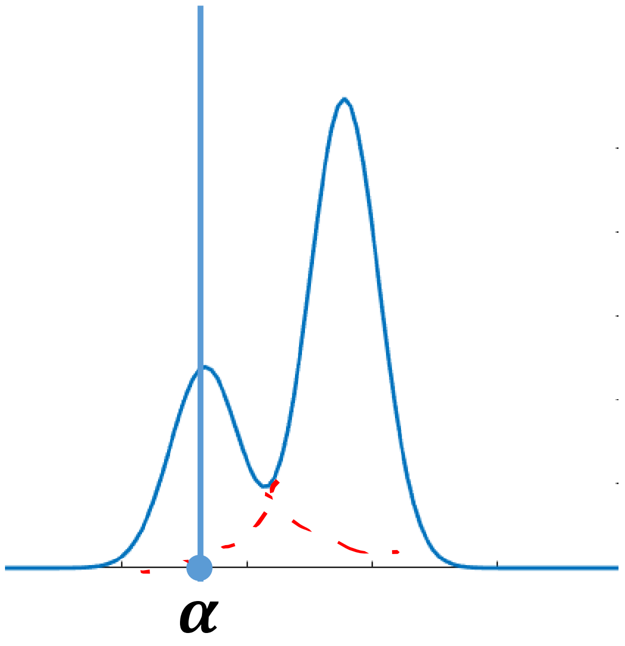
Em-algorithm for Gaussian Mixtures

pros

Gaussian Mixture parameters $\{\pi_i, \boldsymbol{\mu}_i, \Sigma_i\}$ are typically estimated from a training set $\{\boldsymbol{\alpha}_n\}_n$via EM-algorithm, which iterates the **E-step** and **M-step**

- **E-step:** compute the membership weights $\gamma_{n,i}$ for each training sample $\boldsymbol{\alpha}_n$

$$\gamma_{n,i} = \frac{\pi_i \varphi_{\boldsymbol{\mu}_i, \Sigma_i}(\boldsymbol{\alpha}_n)}{\sum_k \pi_k \varphi_{\boldsymbol{\mu}_k, \Sigma_k}(\boldsymbol{\alpha}_n)}$$



$\gamma_1 \sim 1$
$\gamma_2 \sim 0$

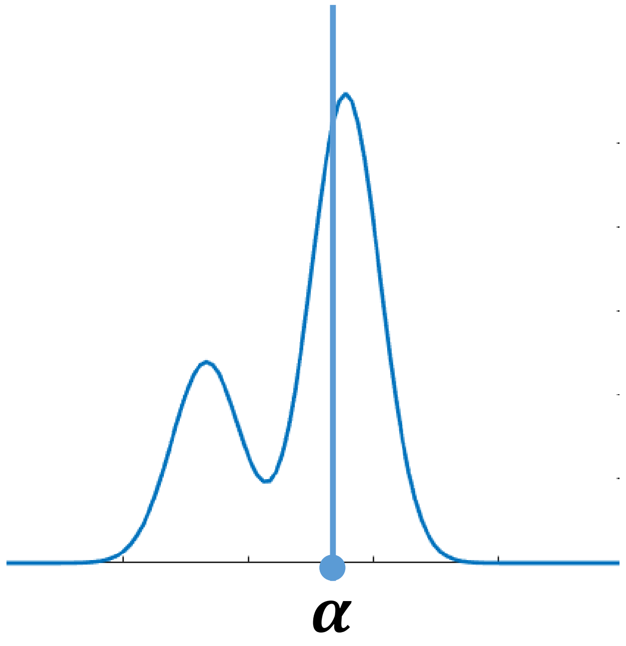
Slide Credits Diego Carrera

Em-algorithm for Gaussian Mixtures

Gaussian Mixture parameters $\{\pi_i, \boldsymbol{\mu}_i, \Sigma_i\}$ are typically estimated from a training set $\{\boldsymbol{\alpha}_n\}_n$ via EM-algorithm, which iterates the **E-step** and **M-step**

- **E-step:** compute the membership weights $\gamma_{n,i}$ for each training sample $\boldsymbol{\alpha}_n$

$$\gamma_{n,i} = \frac{\pi_i \varphi_{\boldsymbol{\mu}_i,\Sigma_i}(\boldsymbol{\alpha}_n)}{\sum_k \pi_k \varphi_{\boldsymbol{\mu}_k,\Sigma_k}(\boldsymbol{\alpha}_n)}$$



$\gamma_1 \sim 0$

$\gamma_2 \sim 1$

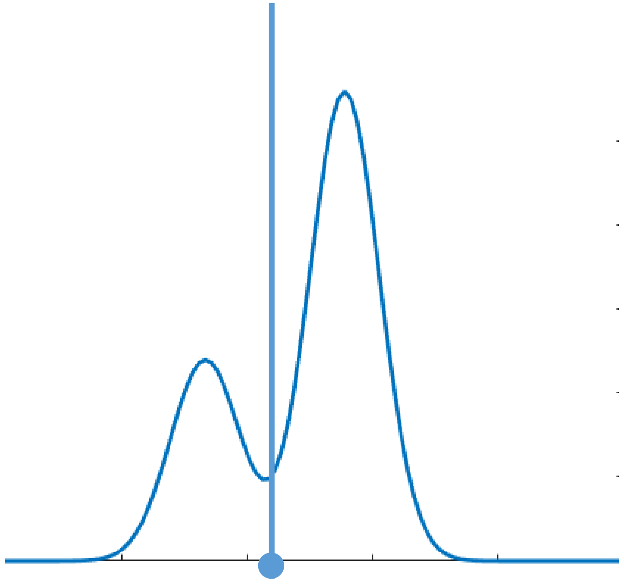
Slide Credits Diego Carrera

Em-algorithm for Gaussian Mixtures

Gaussian Mixture parameters $\{\pi_i, \boldsymbol{\mu}_i, \Sigma_i\}$ are typically estimated from a training set $\{\boldsymbol{\alpha}_n\}_n$via EM-algorithm, which iterates the **E-step** and **M-step**

- **E-step:** compute the membership weights $\gamma_{n,i}$ for each training sample $\boldsymbol{\alpha}_n$

$$\gamma_{n,i} = \frac{\pi_i \varphi_{\boldsymbol{\mu}_i,\Sigma_i}(\boldsymbol{\alpha}_n)}{\sum_k \pi_k \varphi_{\boldsymbol{\mu}_k,\Sigma_k}(\boldsymbol{\alpha}_n)}$$



$$\gamma_1 \sim \frac{1}{2}$$

$$\gamma_2 \sim \frac{1}{2}$$

Slide Credits Diego Carrera

Em-algorithm for Gaussian Mixtures

Gaussian Mixture parameters $\{\pi_i, \boldsymbol{\mu}_i, \Sigma_i\}$ are typically estimated from a training set $\{\boldsymbol{\alpha}_n\}_n$via EM-algorithm, which iterates the **E-step** and **M-step**

- **E-step:** compute the membership weights $\gamma_{n,i}$ for each training sample $\boldsymbol{\alpha}_n$

$$\gamma_{n,i} = \frac{\pi_i \varphi_{\boldsymbol{\mu}_i,\Sigma_i}(\boldsymbol{\alpha}_n)}{\sum_k \pi_k \varphi_{\boldsymbol{\mu}_k,\Sigma_k}(\boldsymbol{\alpha}_n)}$$

- **M-step:** update the parameters of the Gaussian Mixture

$$\pi_i = \frac{1}{N}\sum_n \gamma_{n,i}$$

$$\mu_i = \frac{\sum_n \gamma_{n,i}\boldsymbol{\alpha}_n}{\Sigma_n \gamma_{n,i}}$$

$$\Sigma_i = \frac{\sum_n \gamma_{n,i}(\boldsymbol{\alpha}_n - \boldsymbol{\mu}_i)(\boldsymbol{\alpha}_n - \boldsymbol{\mu}_i)^T}{\Sigma_n \gamma_{n,i}}$$

Slide Credits Diego Carrera

Em-algorithm for Gaussian Mixtures

Gaussian Mixture parameters $\{\pi_i, \boldsymbol{\mu}_i, \Sigma_i\}$ are typically estimated from a training set $\{\boldsymbol{\alpha}_n\}_n$ via EM-algorithm, which iterates the **E-step** and **M-step**

- **E-step:** compute the membership weights $\gamma_{n,i}$ for each training sample $\boldsymbol{\alpha}_n$

$$\gamma_{n,i} = \frac{\pi_i \varphi_{\boldsymbol{\mu}_i,\Sigma_i}(\boldsymbol{\alpha}_n)}{\sum \ldots}$$

- **M-step:** up

The whole procedure can be initialized by a k-means round to identify the GM parameters

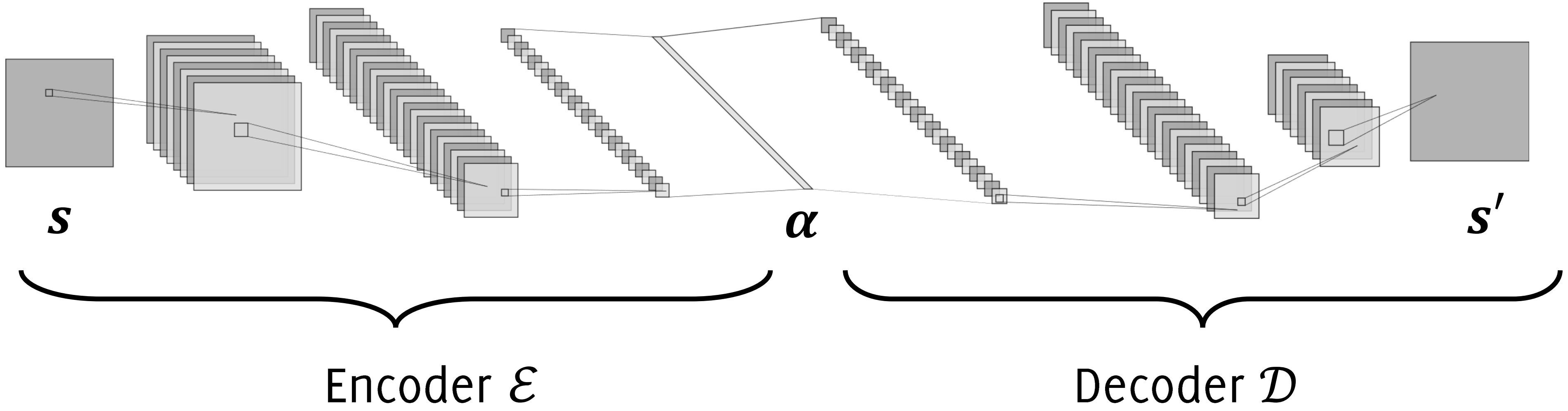
$$\pi_i = \frac{1}{N} \sum_n \gamma_{n,i}$$

$$\mu_i = \frac{\sum_n \gamma_{n,i} \boldsymbol{\alpha}_n}{\sum_n \gamma_{n,i}}$$

$$\Sigma_i = \frac{\sum_n \gamma_{n,i} (\boldsymbol{\alpha}_n - \boldsymbol{\mu}_i)(\boldsymbol{\alpha}_n - \boldsymbol{\mu}_i)^T}{\sum_n \gamma_{n,i}}$$

Slide Credits Diego Carrera

Autoencoders (revisited)

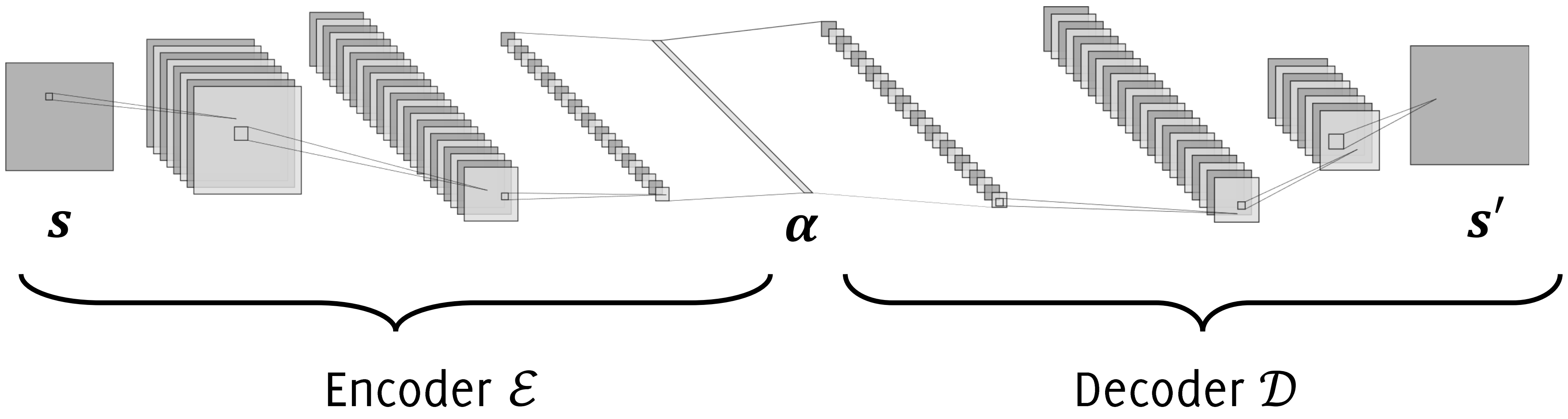


We can compute the likelihood of a test sample $\boldsymbol{s}$ as:

$$\mathcal{L}(\boldsymbol{s}) = \sum_i \pi_i \varphi_{\boldsymbol{\mu}_i, \Sigma_i}(\mathcal{E}(\boldsymbol{s})),$$

Slide Credits Diego Carrera

Autoencoders (revisited)



We can compute the likelihood of a test sample $\boldsymbol{s}$ as:

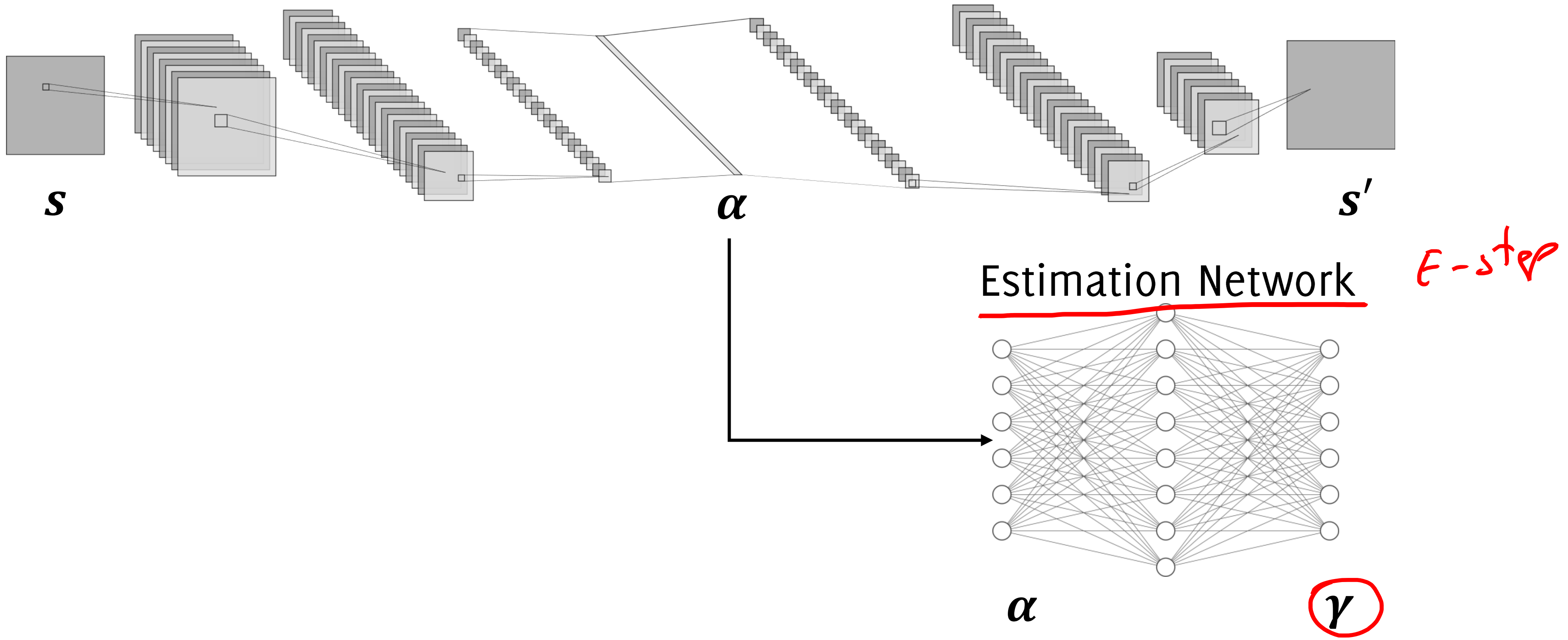
$$\mathcal{L}(\boldsymbol{s}) = \sum_i \pi_i \varphi_{\boldsymbol{\mu}_i, \Sigma_i}(\mathcal{E}(\boldsymbol{s})),$$

The autoencoder and the Gaussian Mixture are not jointly learned!

Slide Credits Diego Carrera

Joint learning of autoencoder and density model

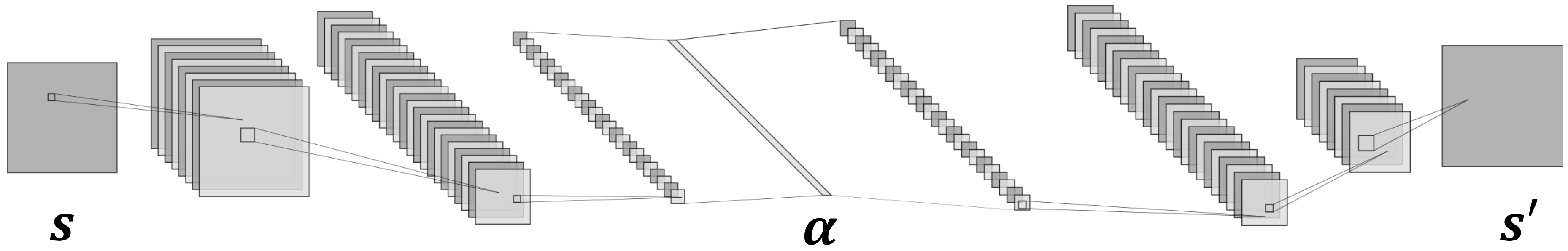
Idea: given a training set of N samples use a NN to predict the membership weights of each sample



Zong et al, *"Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection"*, ICLR 2018

Joint learning of autoencoder and density model

Idea: given a training set of N samples use a NN to predict the membership weights of each sample



Estimate the GM parameters as:

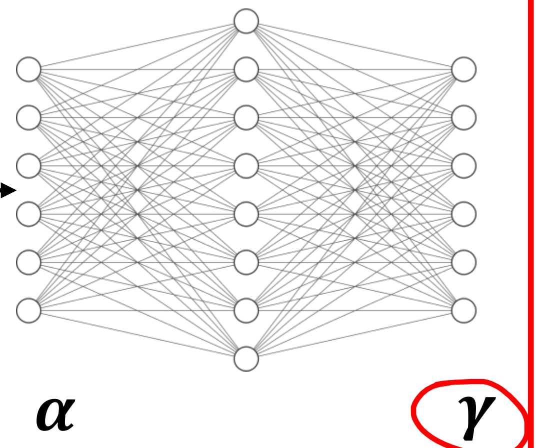
$$\pi_i = \frac{1}{N}\sum_n \gamma_{n,i}$$

$$\mu_i = \frac{\sum_n \gamma_{n,i}\boldsymbol{\alpha}_n}{\sum_n \gamma_{n,i}}$$

$$\Sigma_i = \frac{\sum_n \gamma_{n,i}(\boldsymbol{\alpha}_n - \boldsymbol{\mu}_i)(\boldsymbol{\alpha}_n - \boldsymbol{\mu}_i)^T}{\sum_n \gamma_{n,i}}$$

M-step

Estimation Network



E-step

Zong et al, *"Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection"*, ICLR 2018

The parameters of (π_i, μ_i, Σ_i), can be entirely expressed w.r.t. γ_i

The network training loss is a sum encompassing:

- Reconstruction error $||s - s'||_2$
- The negative log-likelihood of $\mathcal{E}(s)$ for all the training samples w.r.t the identified GM
- Reguarization term for the GM of the mixtures to avoid singular Σ_i

The last two terms are parametrized on γ_i, thus it is possible to backpropagate

$\boldsymbol{s}$ $\boldsymbol{\alpha}$ $\boldsymbol{s}'$

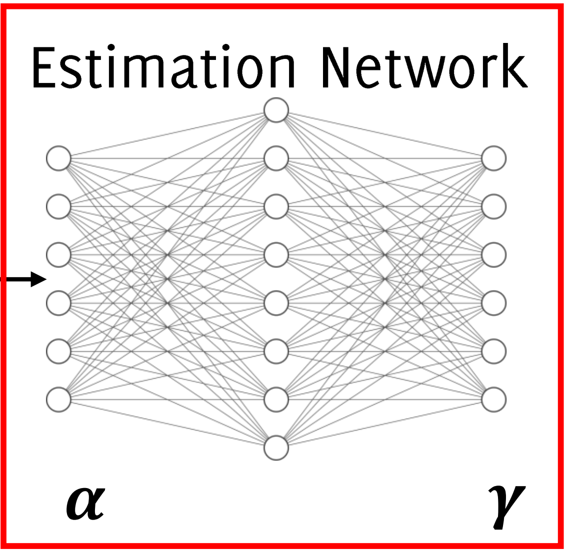
Estimate the GM parameters as:

$$\pi_i = \frac{1}{N}\sum_n \gamma_{n,i}$$

$$\mu_i = \frac{\sum_n \gamma_{n,i}\boldsymbol{\alpha}_n}{\sum_n \gamma_{n,i}}$$

$$\Sigma_i = \frac{\sum_n \gamma_{n,i}(\boldsymbol{\alpha}_n - \boldsymbol{\mu}_i)(\boldsymbol{\alpha}_n - \boldsymbol{\mu}_i)^T}{\sum_n \gamma_{n,i}}$$

M-step



E-step

Zong et al, *"Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection"*, ICLR 2018

CNN as data-driven feature extractor

- Transfer learning
- Autoencoders
- Self-supervised learning
- Domain-based
- Generative-based

These other approaches requires a few more notions of Deep Learning and are out of scope for this course…
If you are interested in knowing more, come to our A2NDL course!

Domain Adaptation

Example of Domain Adaptation Questions

These are rather **common questions** when using an classifier for images

- Can we train classifiers with Flickr photos, as they have already been collected and annotated, and hope the classifiers still work well on mobile camera images?
- Image classifiers optimized on benchmark dataset often exhibit significant degradation in recognition accuracy when evaluated on another one

Problem Formulation

Consider a classification problem from an input space to a label space

$$\mathcal{X} \rightarrow \mathcal{Y}$$

And denote by

- A **domain** $\mathcal{D} = \{\mathcal{X}, \phi_{\boldsymbol{x}}\}$ such that $\boldsymbol{x} \sim \phi_{\boldsymbol{x}}$
- A **task** $\mathcal{T} = \{\mathcal{Y}, \phi(\cdot\,|\boldsymbol{x})\}$ which consists in associating the label to an input $\boldsymbol{x}$

S. J. Pan and Q. Yang, "A survey on transfer learning", TKDE 2010.

Problem Formulation

Definition: Transfer Learning

Given a ***source domain*** $\mathcal{D}_S$ *and learning task* $\mathcal{T}_S$, *a* ***target domain*** $\mathcal{D}_T$ *and learning task* $\mathcal{T}_T$, *transfer learning aims to help* ***improve the learning of the target predictive function K in*** $\mathcal{D}_T$ ***using the knowledge in*** $\mathcal{D}_S$ ***and*** $\mathcal{T}_S$, *where* $\mathcal{D}_S \neq \mathcal{D}_T$ *or* $\mathcal{T}_S \neq \mathcal{T}_T$

S. J. Pan and Q. Yang, "A survey on transfer learning", TKDE 2010.

Problem Formulation

Remarks:

$\mathcal{D}_S \neq \mathcal{D}_T$ implies that either

- $\mathcal{X}_S \neq \mathcal{X}_T$ (e.g. different input size / features)
- $\phi_{\boldsymbol{x}}^S \neq \phi_{\boldsymbol{x}}^T$ (e.g. different style of an image)

$\mathcal{T}_S \neq \mathcal{T}_T$ implies that either

- $\mathcal{Y}_S \neq \mathcal{Y}_T$ (e.g. new labels in target)
- $\phi^S(y|\boldsymbol{x}) \neq \phi^T(y|\boldsymbol{x})$ (e.g. different class imbalance between S and T)

Obviously, $\mathcal{D}_S = \mathcal{D}_T$ and $\mathcal{T}_S = \mathcal{T}_T$ corresponds to traditional machine learning settings

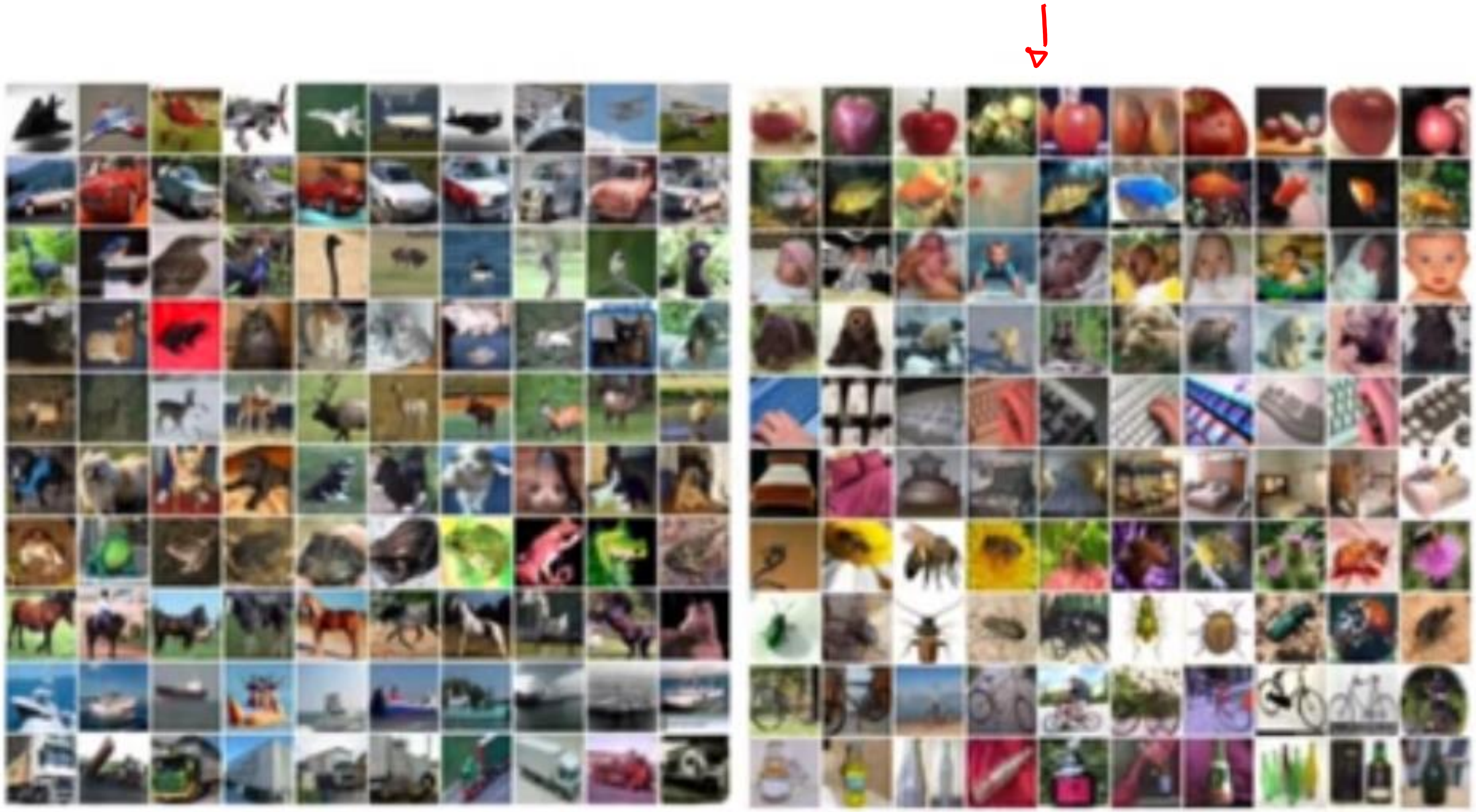
Example: $\mathcal{X}_S \neq \mathcal{X}_T$ same task, different domains

- Classification of documents in different languages
- Images of different sizes
- Colour vs grayscale images
- Daytime vs Nigh-time

Example: $\phi_x^S \neq \phi_x^T$

Venkateswara, H., Eusebio, J., Chakraborty, S., & Panchanathan, S. (2017). Deep hashing network for unsupervised domain adaptation. CVPR 2017

Example: $\mathcal{Y}_S \neq \mathcal{Y}_T$ different task, same domains



(b) CIFAR-10 (c) CIFAR-100

Example: $\phi^S(y|\boldsymbol{x}) \neq \phi^T(y|\boldsymbol{x})$

In the case of text classification, words might have a different meaning depending on the domain

Consider

$\mathcal{D}_S$: articles from newspapers

$\mathcal{D}_T$: articles from a computer magazine

The word «monitor» in the two context is very likely to have different meanings, leading to different classes of documents in each domain

Standard Settings

Consider a classification problem where we are provided with

$$TR = \{(\boldsymbol{x}_0, y_0), \dots, (\boldsymbol{x}_n, y_n) \in \mathcal{X}_S \times \mathcal{Y}_S\}$$

That are from the source domain $\mathcal{D}_S$ and representative of $\mathcal{T}_S$.

You are asked to prepare a classifier K that is able to operate in the target domain $\mathcal{D}_T$ to address the task $\mathcal{T}_T$

Usually there are other constraints over the target domain (task), such that little or even no supervised information is provided.

S. J. Pan and Q. Yang, "A survey on transfer learning", TKDE 2010.

Supervised Case

The fully supervised case:

- we have access to

$$TR_S = \{(\boldsymbol{x}_0, y_0), \dots, (\boldsymbol{x}_n, y_n) \in \mathcal{X}_S \times \mathcal{Y}_S\}$$

a large, annotated corpus of data from the **source domain $\mathcal{D}_S$ and representative of $\mathcal{T}_S$.**

- we spend a little money to annotate a small corpus in the target domain $\mathcal{D}_T$ **and representative of $\mathcal{T}_T$**

$$TR_T = \{(\boldsymbol{x}_0, y_0), \dots, (\boldsymbol{x}_m, y_m) \in \mathcal{X}_T \times \mathcal{Y}_T\}$$

Thus $m \ll n$.

Goal: learn a classifier K that is very accurate in $\mathcal{D}_T$ to solve the task $\mathcal{T}_T$

Unsupervised Case

The fully unsupervised case:

- we have access to

$$TR_S = \{(\boldsymbol{x}_0, y_0), \dots, (\boldsymbol{x}_n, y_n) \in \mathcal{X}_S \times \mathcal{Y}_S\}$$

a large, annotated corpus of data from the **source domain $\boldsymbol{\mathcal{D}_S}$ and representative of $\boldsymbol{\mathcal{T}_S}$.**

- we have no annotations over $\boldsymbol{\mathcal{D}_T}$

$$X_T = \{\boldsymbol{x_0}, \dots, \boldsymbol{x_m} \in \mathcal{X}_T\}$$

And m might be even larger than n.

Goal: learn a classifier K that is very accurate in $\boldsymbol{\mathcal{D}_T}$ to solve the task $\boldsymbol{\mathcal{T}_T}$

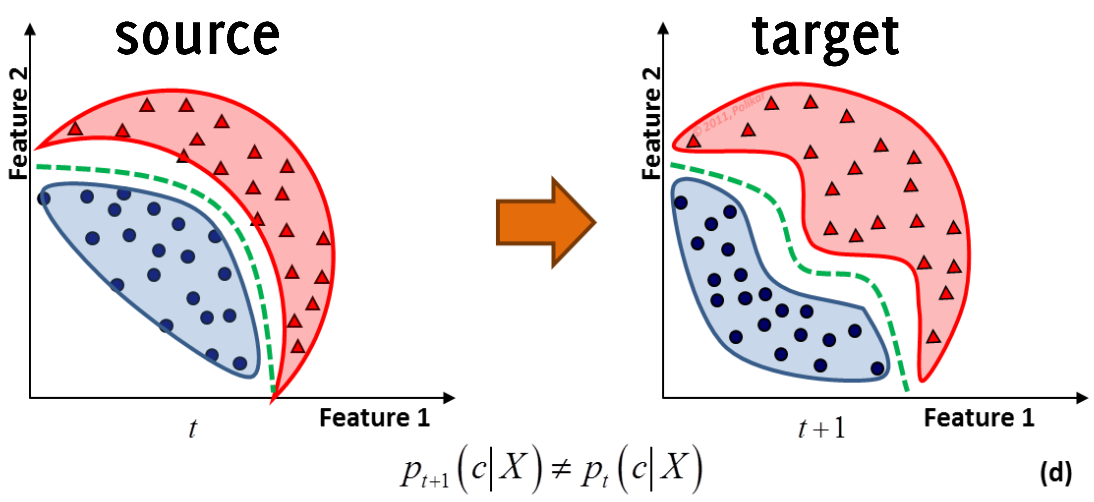
Does this remind you something?

Connection with Concept Drift

It’s the **same problem to be addressed in Datastreams affected by Concept Drift, but without**

- Temporal dimension
- Need to detect changes

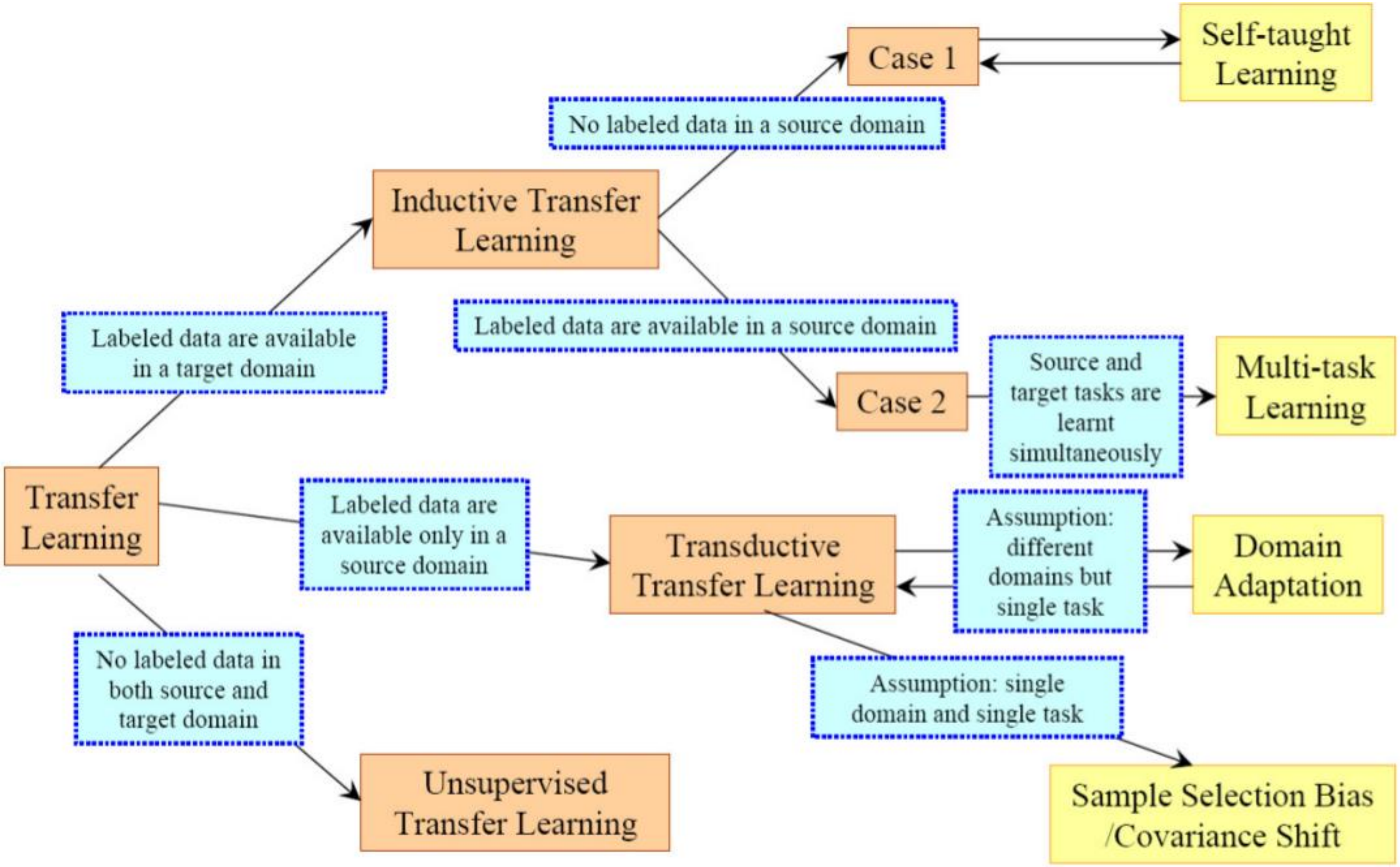
You are given a training set that is (stationary) from the source domain and you have to operate in a different (stationary) target domain



Boracchi, Trovò

Major Approaches to Domain Adaptation

Transfer Learning Taxhonomy



S. J. Pan and Q. Yang, "A survey on transfer learning", TKDE 2010.

Reweighting

Correct a sample bias by reweighting source labeled data:

source instances close to target instances are more important

Motivations:

- Domains share the same support (i.e. bag of words)
- Distribution shift is caused by sampling bias/shift between marginals

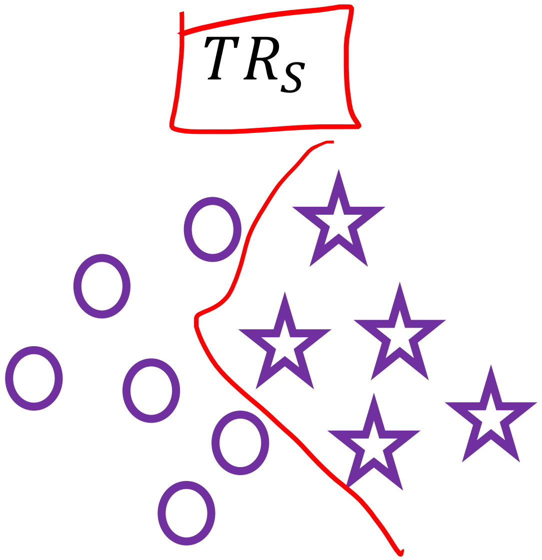
Idea:

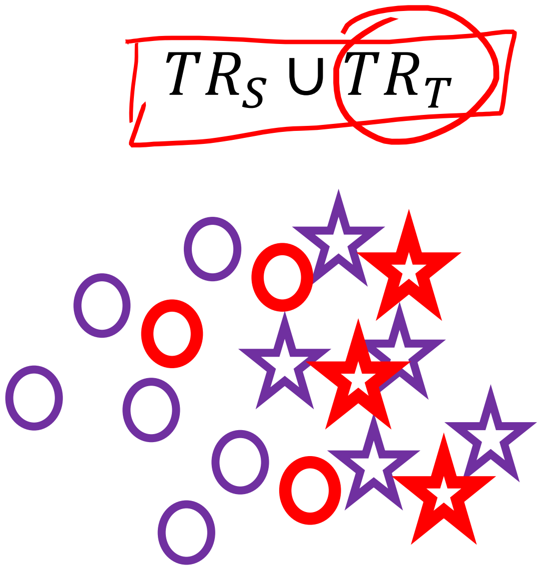
- Reweight or select instances to reduce the discrepancy between source and target domains

Supervised Reweighting

Reweight or select instances to reduce the discrepancy between source and target domains

Supervised case: (where $\mathcal{Y}_S = \mathcal{Y}_T$), it is possible to train a classifier over $TR_S \cup TR_T$ by weighting more the loss over instances from TR_T (or by resampling TR_T)





Unsupervised Reweighting over TR_S

Estimate $\hat{\phi}_{S,x}$ and $\hat{\phi}_{T,x}$ from X_S and X_T and compute

$$w_i = \frac{\hat{\phi}_{T,\boldsymbol{x}}(x_i)}{\hat{\phi}_{S,\boldsymbol{x}}(x_i)}$$

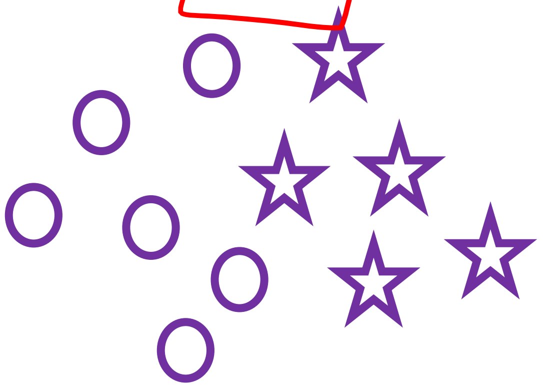
$x_i \in X_S$

Train a classifier to minimize the weighted loss

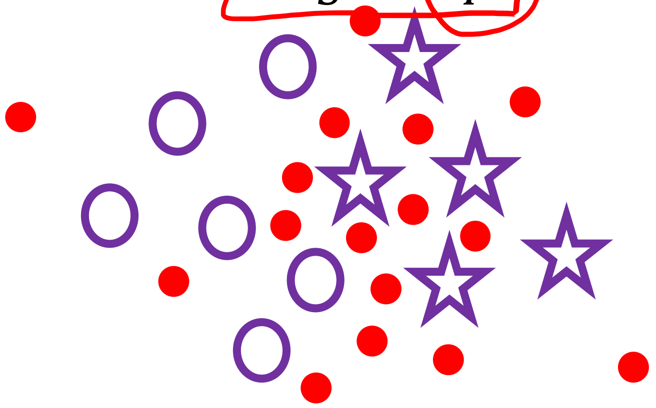
$$\mathcal{L}(TR_S) = \sum_{x_i \in TR_S} w_i \, I[K(\boldsymbol{x_i}) \neq y_i]$$

TR_S

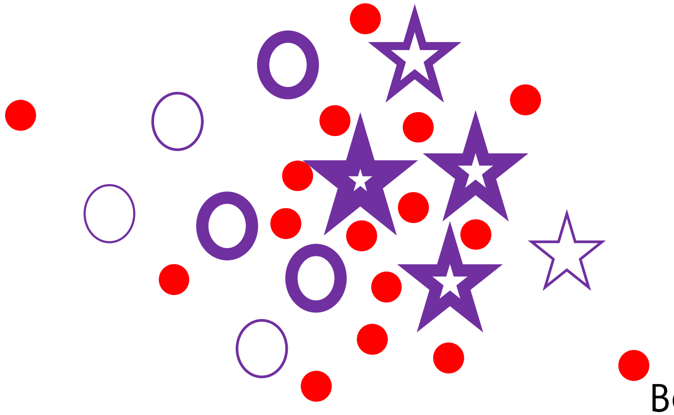
TR_S



$TR_S \cup X_T$



$TR_S \cup X_T$ and reweighting



Boracchi, Trovò

Feature-based Methods

Find a common space where source and target are close (projection, new features, etc)

Change the feature representation $\mathcal{X}$ to better represent shared characteristics between the two domains

- some features are domain-specific,
- others are generalizable
- or there exist mappings from the original space

Idea:

- Make source and target domain explicitly similar
- Learn a new feature space by embedding or projection

Supervised, Feature-based Methods

Train a classifier K_S over TR_S

Use the output of K_S as an additional feature to train the classifier K_T over augmented features

$$[x_i;\ K_S(x_i)]$$

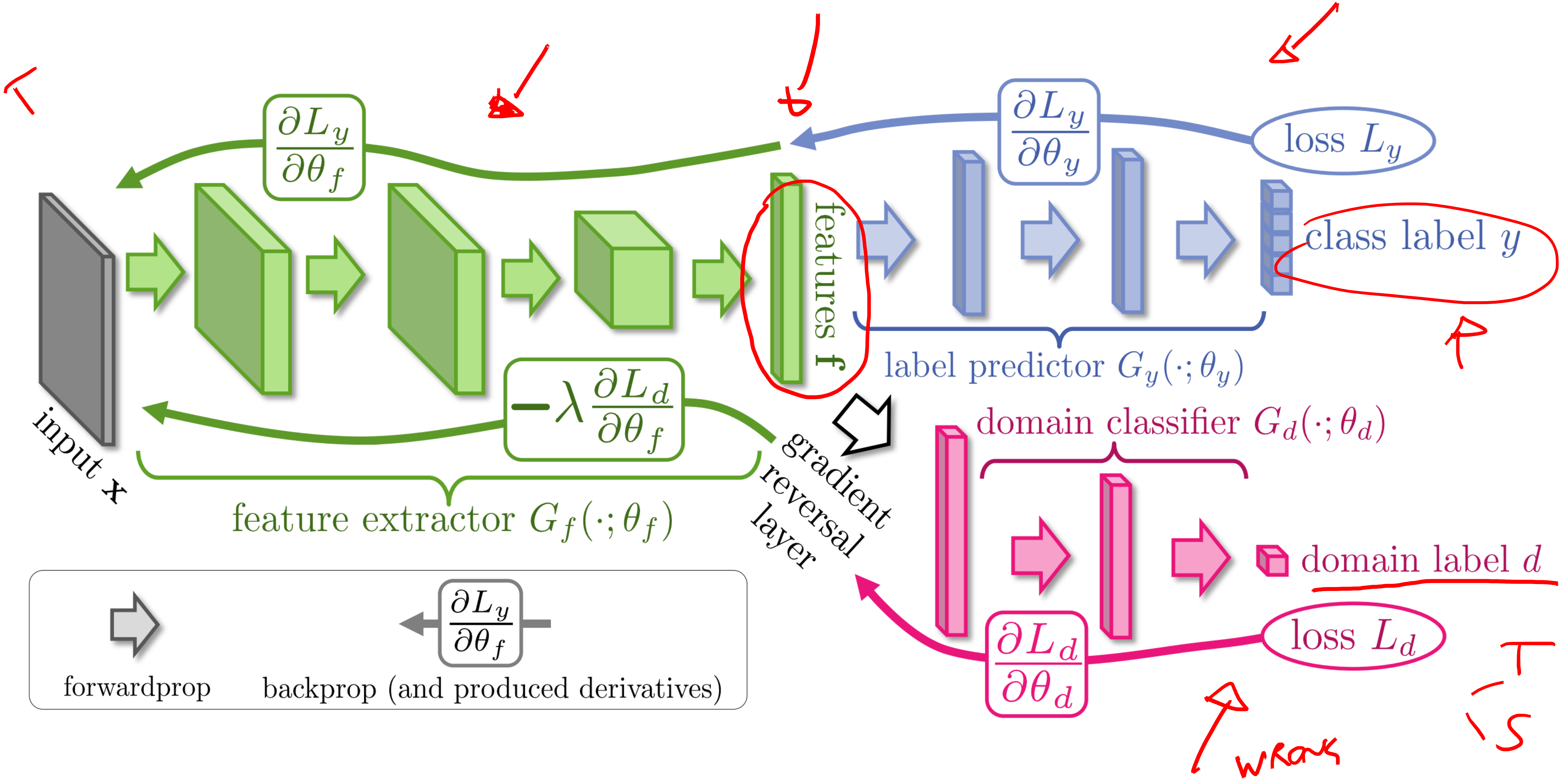
Or, train a joint classifier over an augmented training set TR_A where inputs are defined as:

$$\boldsymbol{x_i} \rightarrow [\boldsymbol{x_i}; \boldsymbol{x_i}; \boldsymbol{0}] \text{ when } \boldsymbol{x_i} \text{ from } TR_S$$
$$\boldsymbol{x_i} \rightarrow [\boldsymbol{x_i}; \boldsymbol{0}; \boldsymbol{x_i}] \text{ when } \boldsymbol{x_i} \text{ from } TR_T$$

During operations augment everything as in TR_T

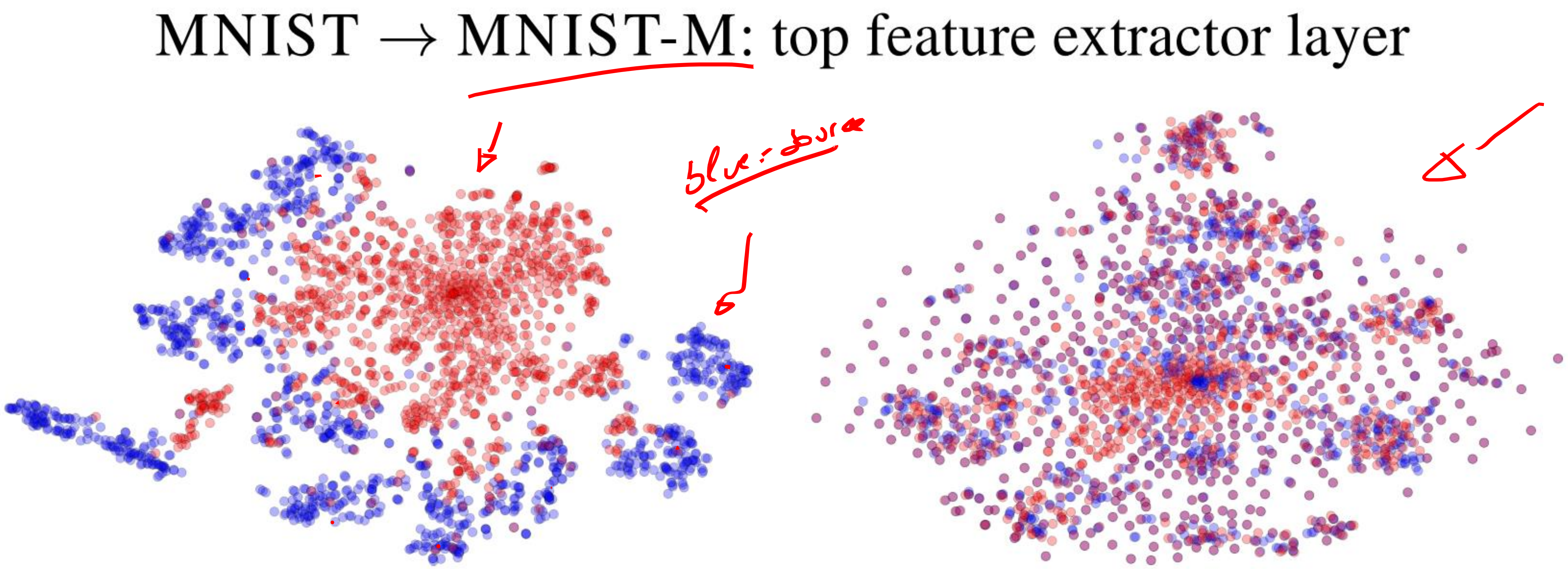
Daumé III, H. Frustratingly easy domain adaptation. ACL 2007

Unsupervised Domain Adaptation



Ganin, Y., & Lempitsky, V. Unsupervised domain adaptation by backpropagation. *ICML 2015*

Unsupervised Domain Adaptation



(a) Non-adapted (b) Adapted

Ganin, Y., & Lempitsky, V. Unsupervised domain adaptation by backpropagation. *ICML 2015*