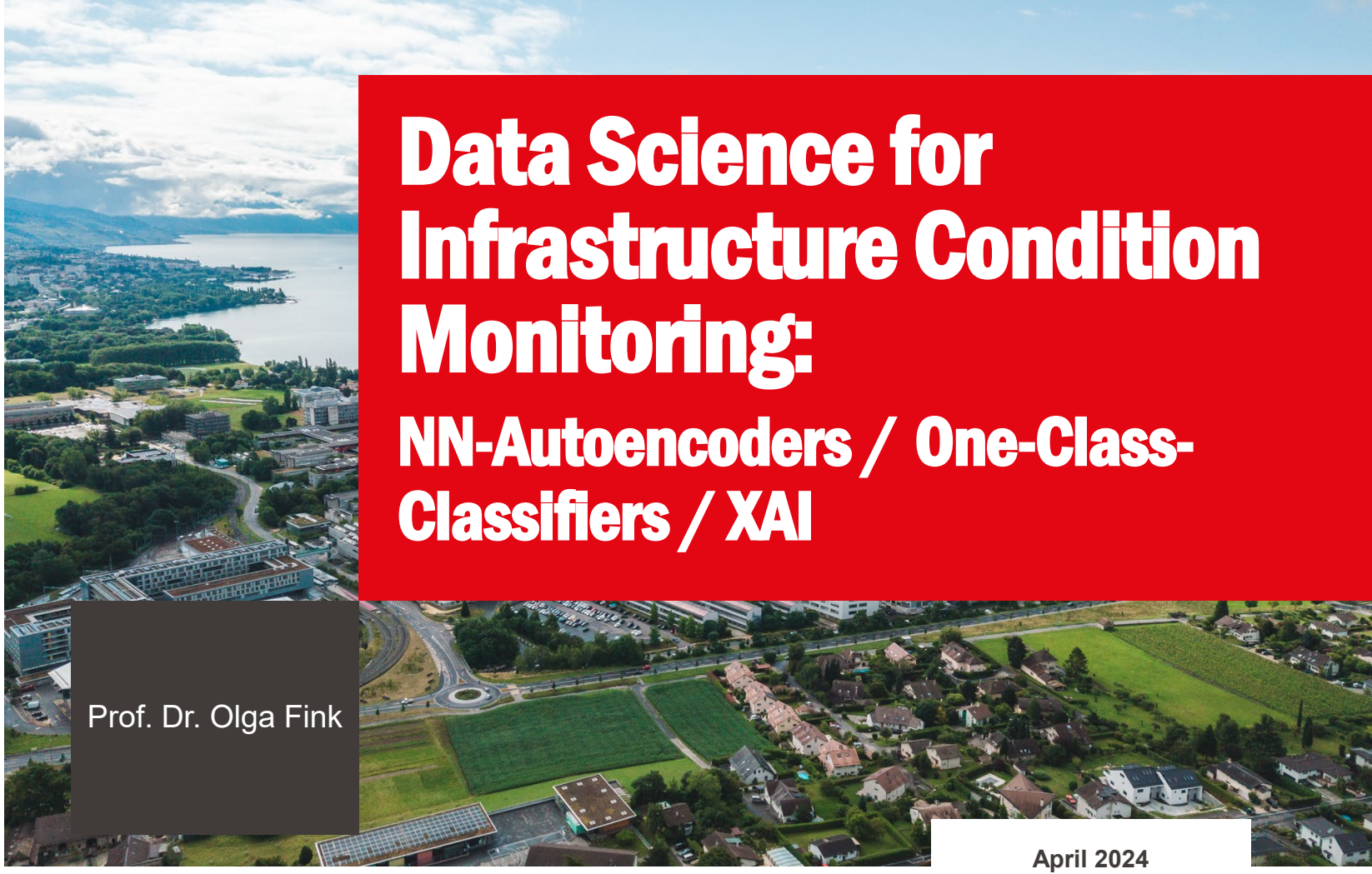


Data Science for Infrastructure Condition Monitoring:

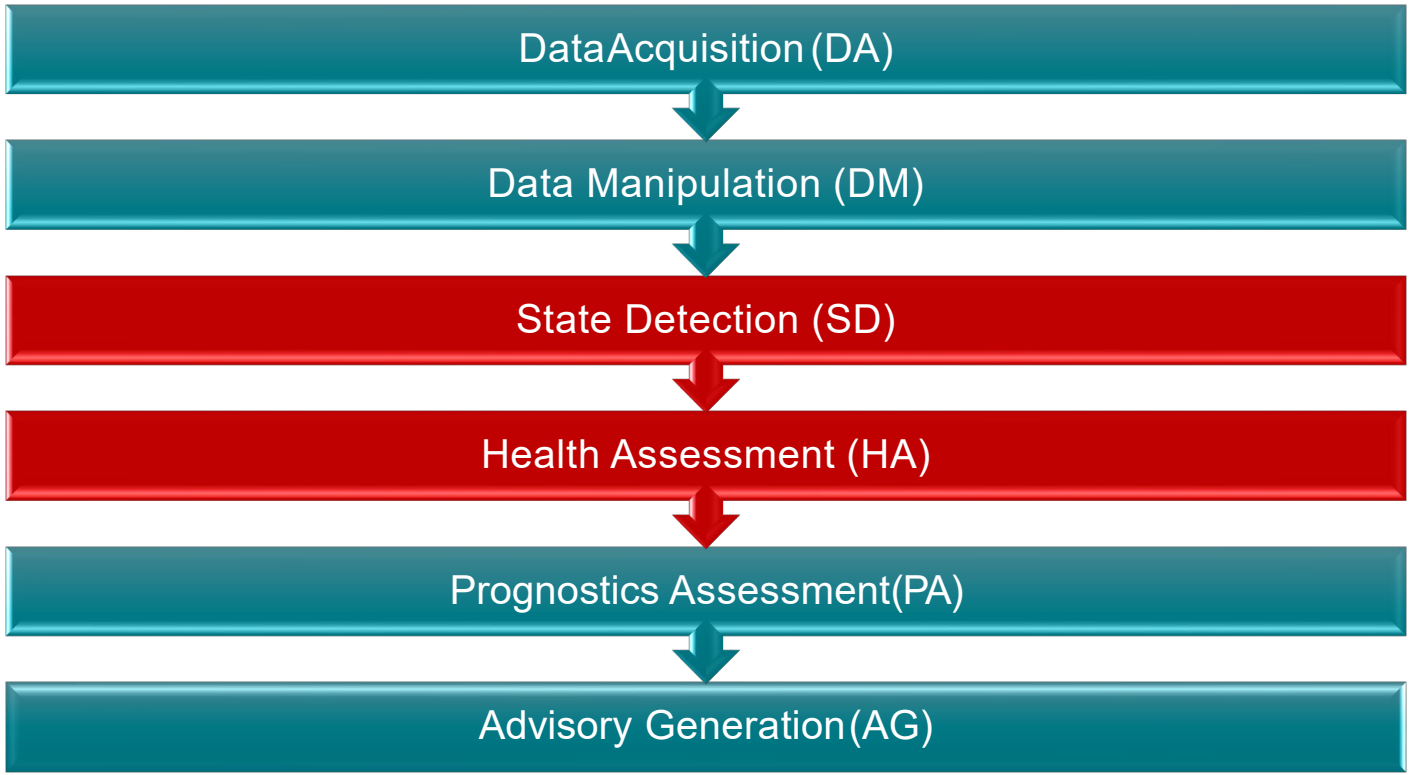
NN-Autoencoders / One-Class-Classifiers / XAI

Prof. Dr. Olga Fink

École polytechnique fédérale de Lausanne

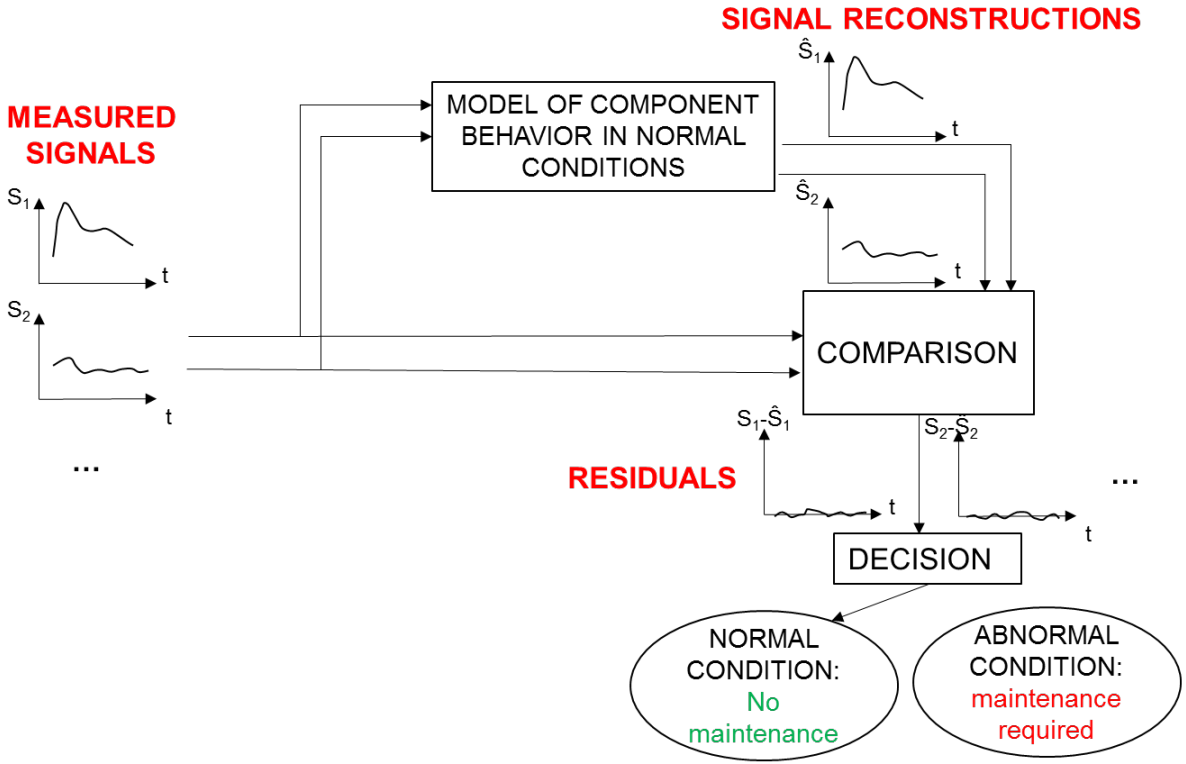
April 2024

PHM Process


DataAcquisition (DA)
Data Manipulation (DM)
State Detection (SD)
Health Assessment (HA)
Prognostics Assessment(PA)
Advisory Generation (AG)

NN Autoencoders

Signal reconstruction

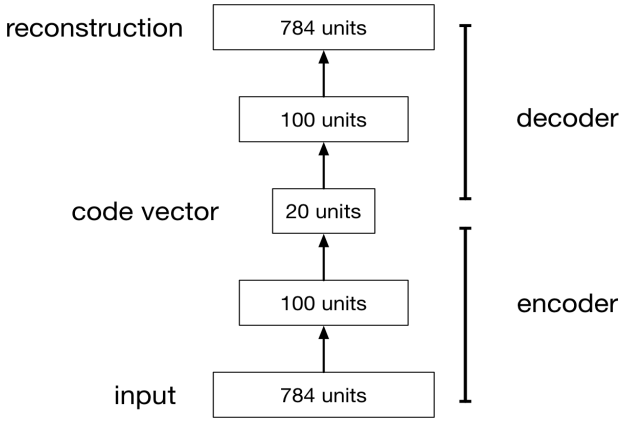


NN-Autoencoders

- A type of artificial neural network used to learn efficient representations of data
- Typically for the purpose of dimensionality reduction or feature learning
- Designed to learn a compressed representation of input data
- By passing the data through a network that first encodes the input into a lower-dimensional space
- Then reconstructs the output from this representation.

NN Autoencoders

- A NN autoencoder is a feed-forward neural net whose job it is to take an input x and predict $\hat{x}$.
- To make this non-trivial, we need to add a bottleneck layer whose dimension is much smaller than the input.
- Example:



Encoder / Decoder

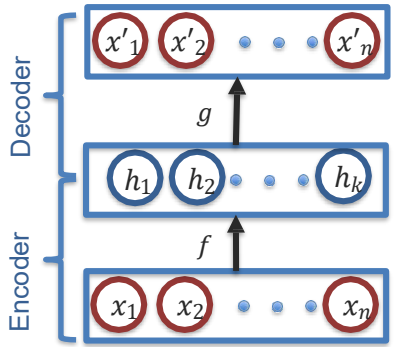
- **Encoder**: This part of the network compresses the input into a latent-space representation. It encodes the high-dimensional data into a lower-dimensional code. This process involves a series of layers that gradually downsample the input data.
- **Decoder**: After encoding, the network attempts to reconstruct the input data from the compressed code, resulting in the output that mirrors the original data. The decoder essentially reverses the process of the encoder, up-sampling from the latent space back to the original input dimension.

AE learning

- The learning process is unsupervised
- Does not require labeled input-output pairs
- It uses a loss function that measures the difference between the input and its reconstruction (often using a mean squared error (MSE))
- By minimizing this reconstruction error during training, the autoencoder learns to preserve as much of the relevant information in the input data as possible.

Autoencoders

- Network is trained to output the input (learn identify function).
- Two parts encoder/decoder
  - $x' = g(f(x))$
  - g - decoder
  - f - encoder



Trivial solution unless:
- Constrain number of units in Layer 2 (learn compressed representation), or
- Constrain Layer 2 to be **sparse**

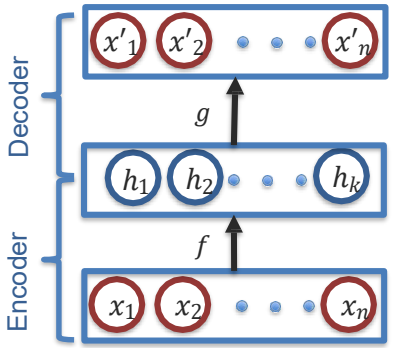
Source: J.C. Kao, UCLA

Basic principles of an autoencoder

If the input is $x \in \mathbb{R}^n$ an autoencoder will produce a $h \in \mathbb{R}^d$ where $d < n$, which is designed to contain most of the important features of $\mathbf{x}$ to reconstruct it.

Autoencoder performs the following steps:

- **Encoder**: Perform a dimensionality reduction step on the data, $\mathbf{x} \in \mathbb{R}^n$ to obtain features $\mathbf{h} \in \mathbb{R}^d$.
- **Decoder**: Map the features $\mathbf{h} \in \mathbb{R}^d$ to closely reproduce the input, $\hat{\mathbf{x}} \in \mathbb{R}^n$.



Thus, the autoencoder implements the following problem:

Let $\mathbf{x} \in \mathbb{R}^n$, $f(\cdot) : \mathbb{R}^n \to \mathbb{R}^d$ and $g(\cdot) : \mathbb{R}^d \to \mathbb{R}^n$. Let

$$\hat{\mathbf{x}} = g(f(\mathbf{x}))$$

Define a loss function, $\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}})$, and minimize $\mathcal{L}$ with respect to the parameters of $f(\cdot)$ and $g(\cdot)$.

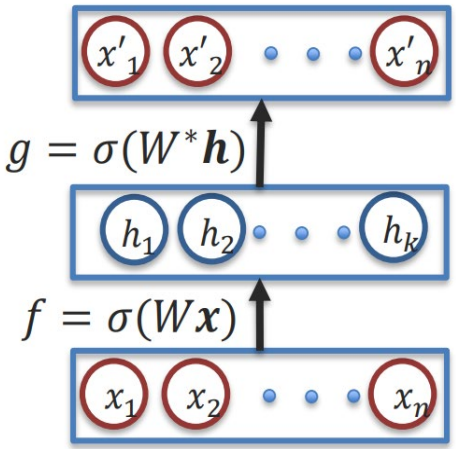
There are different loss functions that you could consider, but a common one is the squared loss:

$$\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}}) = \|\mathbf{x} - \hat{\mathbf{x}}\|^2$$

Source: J.C. Kao, UCLA

Nonlinear dimensionality reduction

- In a more general form, f() and g() could be deep neural networks, learning potentially more nonlinear and expressive features h.



Sparse autoencoders

A sparse autoencoder is one that is regularized to not only minimize the loss, but to also incorporate sparse features. If $\mathbf{h} = f(\mathbf{x})$ and $\hat{\mathbf{x}} = \mathbf{g}(\mathbf{h})$, then the sparse encoder has the following loss:

$$\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}}) + \lambda \sum_i |h_i|$$

where h_i is the ith element of $\mathbf{h}$. This is intuitive, as we know L1-regularization introduces sparsity.

Lasso Regression (Least Absolute Shrinkage and Selection Operator)

Source: J.C. Kao, UCLA

Denoising autoencoders

Say you wanted to obtain an autoencoder that was robust to noise. One could generate noise, ε, and add it to the input $\mathbf{x}$, so that $\tilde{\mathbf{x}} = \mathbf{x} + \varepsilon$. Then, the loss function of the autoencoder would have loss:

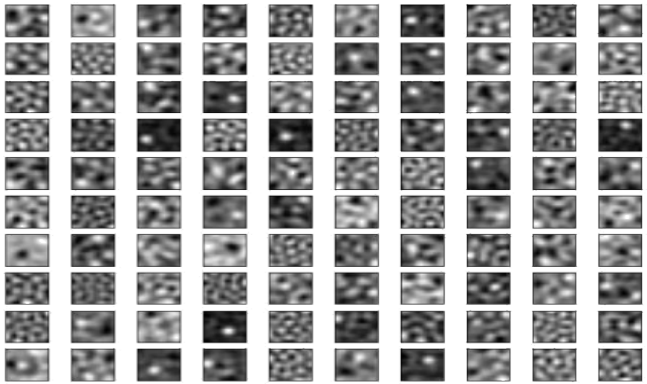
$$\mathcal{L}(\mathbf{x}, g(f(\tilde{\mathbf{x}})))$$

and it would learn to denoise $\tilde{\mathbf{x}}$ to reproduce $\mathbf{x}$. This can cause your autoencoder to be robust to certain types of noise.

Source: J.C. Kao, UCLA

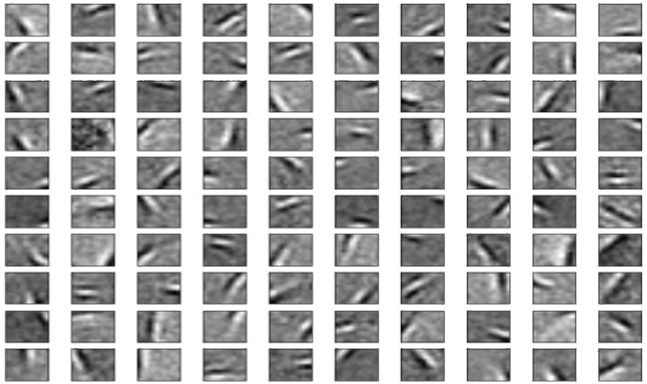
Sparse Vs Denoising

- Filter weights, 12x12 patches



Sparse AE

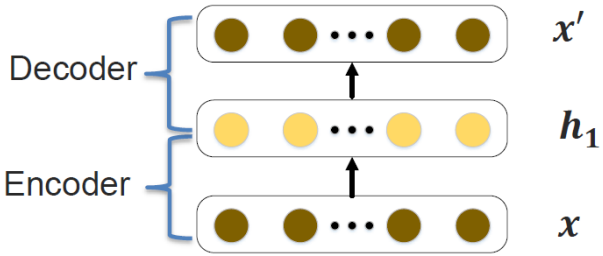
Actually meaningless

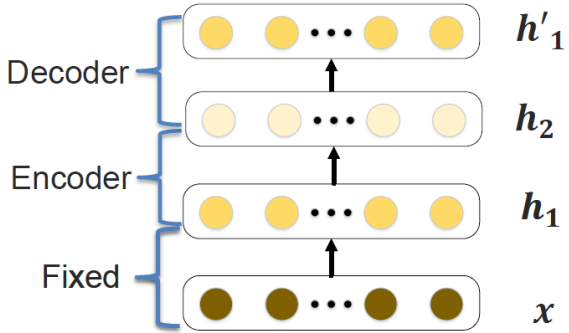


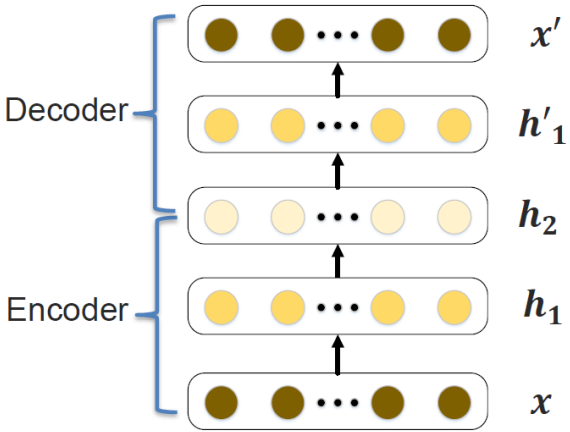
Denoising AE

[Vincent et al. 2010]

Stacked autoencoders



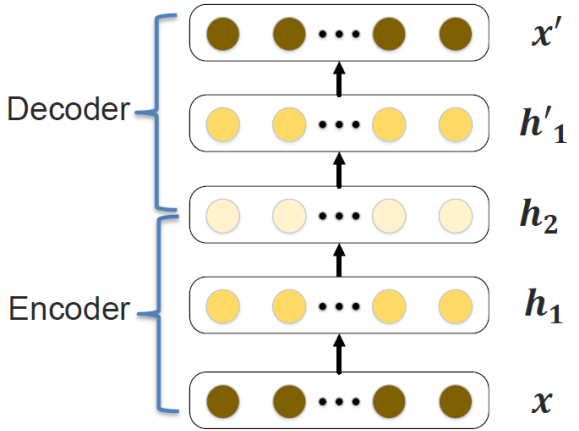




Source: L.P. Morency

Stacked denoising autoencoders

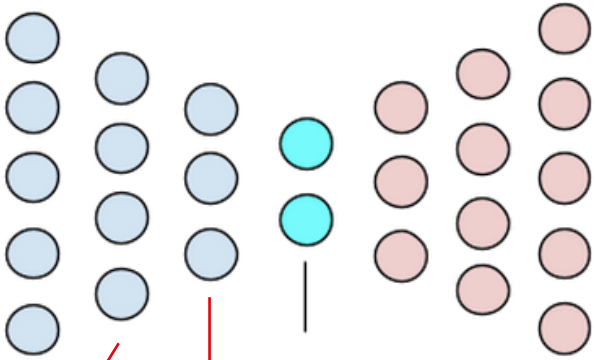
- Can extend this to a denoising model
- Add noise when training each of the layers
- Often with increasing amount of noise per layer
- 0.1 for first, 0.2 for second, 0.3 for third



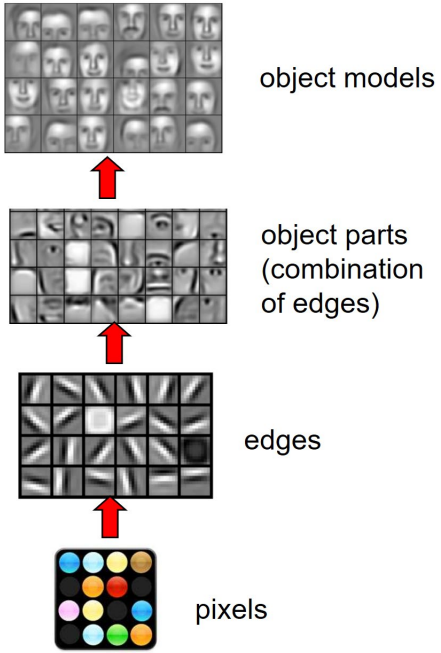
Source: L.P. Morency

Learning features with deep learning algorithms

Input

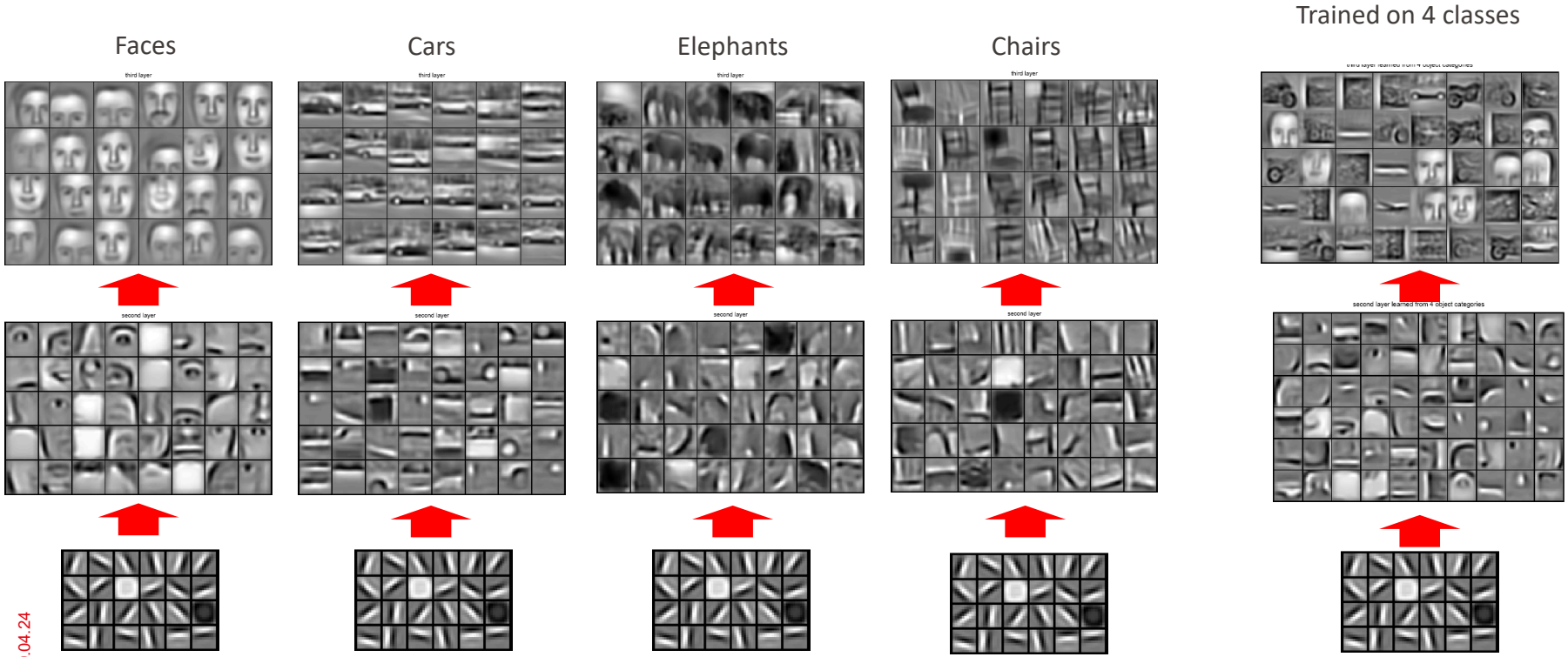


Output



Source: A. NG

Examples of learned object parts from object categories



Source: A. NG

One-Class Classifiers

General Concepts (1/2)

- Designed to identify whether a given instance belongs to a specific class or not
- Particularly useful in situations where you have a lot of data for one class, often referred to as the "target" or "positive" class, and very little to no data for the "outlier" or "negative" class
- The classic use case for a one-class classifier is anomaly detection, where the goal is to detect rare events.

General Concepts (2/2)

- **Target Class Representation**: The one-class classifier learns only from the target class during the training phase. The model tries to capture the distribution, patterns, or boundaries of the target class data.
- **Anomaly Detection**: During inference, the classifier predicts whether new instances resemble the learned target class. Instances that deviate significantly from the target class pattern are identified as anomalies or outliers.

Examples

- **Autoencoders**: In the context of one-class classification, autoencoders can be trained on the target class data and used to reconstruct new data points. Large reconstruction errors can indicate outliers or anomalies.
- **One-Class SVM**: One-class Support Vector Machines (SVM) are a popular approach. They work by finding the largest margin between the target class data points and the origin in a transformed feature space, effectively separating the target data from the origin.
- **Isolation Forest**: This algorithm isolates anomalies instead of profiling normal data points. It uses random trees to partition the data, and anomalies are expected to be isolated closer to the root of the tree, with fewer splits needed than for normal points.

One-Class SVM

One-class SVM

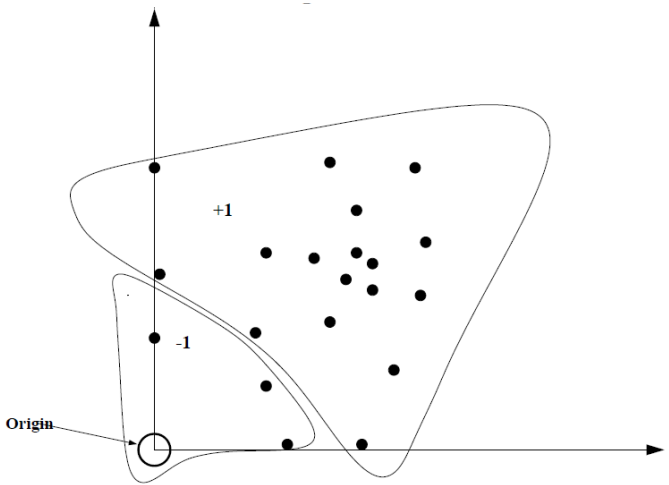
- Suppose that a dataset has a probability distribution *P* in the feature space.
- Find a “simple” subset *S* of the feature space such that the probability that a test point from *P* lies outside *S* is bounded by some a priori specified value $v \in (0, 1)$
- The solution for this problem is obtained by estimating a function *f* which is positive on *S* and negative on the complement $\bar{S}$.

$$f(x) = \begin{cases} +1 & \text{if } x \in S \\ -1 & \text{if } x \in \overline{S} \end{cases}$$

Source: Manevitz, 2001

One-class SVM

- The algorithm can be summarized as mapping the data into a feature space *H* using an appropriate kernel function, and then trying to separate the mapped vectors from the origin with maximum margin



Source: Manevitz, 2001

One-class SVM

$$\min_{w,\,\xi_i,\,\rho} \frac{1}{2}\|w\|^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - \rho$$

subject to:

$$(w \cdot \phi(x_i)) \geq \rho - \xi_i \qquad \text{for all } i = 1, \ldots, n$$

$$\xi_i \geq 0 \qquad \text{for all } i = 1, \ldots, n$$

Source: Manevitz, 2001

ν-parameter

- In this formula it is the parameter ν that characterizes the solution;
- It sets an upper bound on the fraction of outliers (training examples regarded out-of-class)
- It is a lower bound on the number of training examples used as Support Vector

Source: Manevitz, 2001

Decision function

If *w* and *ρ* solve this problem, then the decision function

$$f(x) = \operatorname{sgn}((w \cdot \phi(x_i)) - \rho) = \operatorname{sgn}(\sum_{i=1}^{n} \alpha_i K(x, x_i) - \rho)$$

will be positive for most examples x_i contained in the training set.

Source: Manevitz, 2001

Support Vector Data Description (SVDD)

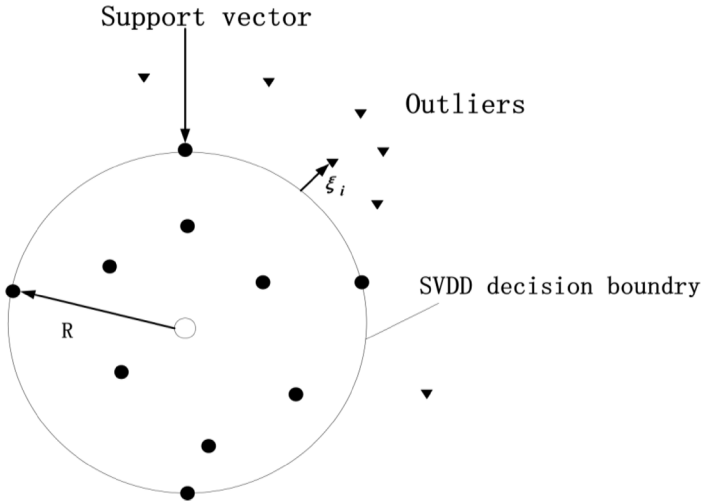
- Obtains a spherical boundary, in feature space, around the data.
- The volume of this hypersphere is minimized→ minimizes the effect of incorporating outliers in the solution

$$\min_{R,\,\mathbf{a}} R^2 + C\sum_{i=1}^{n}\xi_i$$

subject to:

$$\|x_i - \mathbf{a}\|^2 \leq R^2 + \xi_i \qquad \text{for all } i = 1,\ldots,n$$

$$\xi_i \geq 0 \qquad \text{for all } i = 1,\ldots,n$$



Source: Tax & Duin

Support Vector Data Description (SVDD)

- Obtains a spherical boundary, in feature space, around the data.
- The volume of this hypersphere is minimized→ minimizes the effect of incorporating outliers in the solution

$$\min_{R,\,\mathbf{a}} R^2 + C\sum_{i=1}^{n}\xi_i$$

subject to:

$$\|x_i - \mathbf{a}\|^2 \leq R^2 + \xi_i \qquad \text{for all } i = 1,\ldots,n$$

$$\xi_i \geq 0 \qquad \text{for all } i = 1,\ldots,n$$

Source: Tax & Duin

Support Vector Data Description (SVDD) according to Tax and Duin

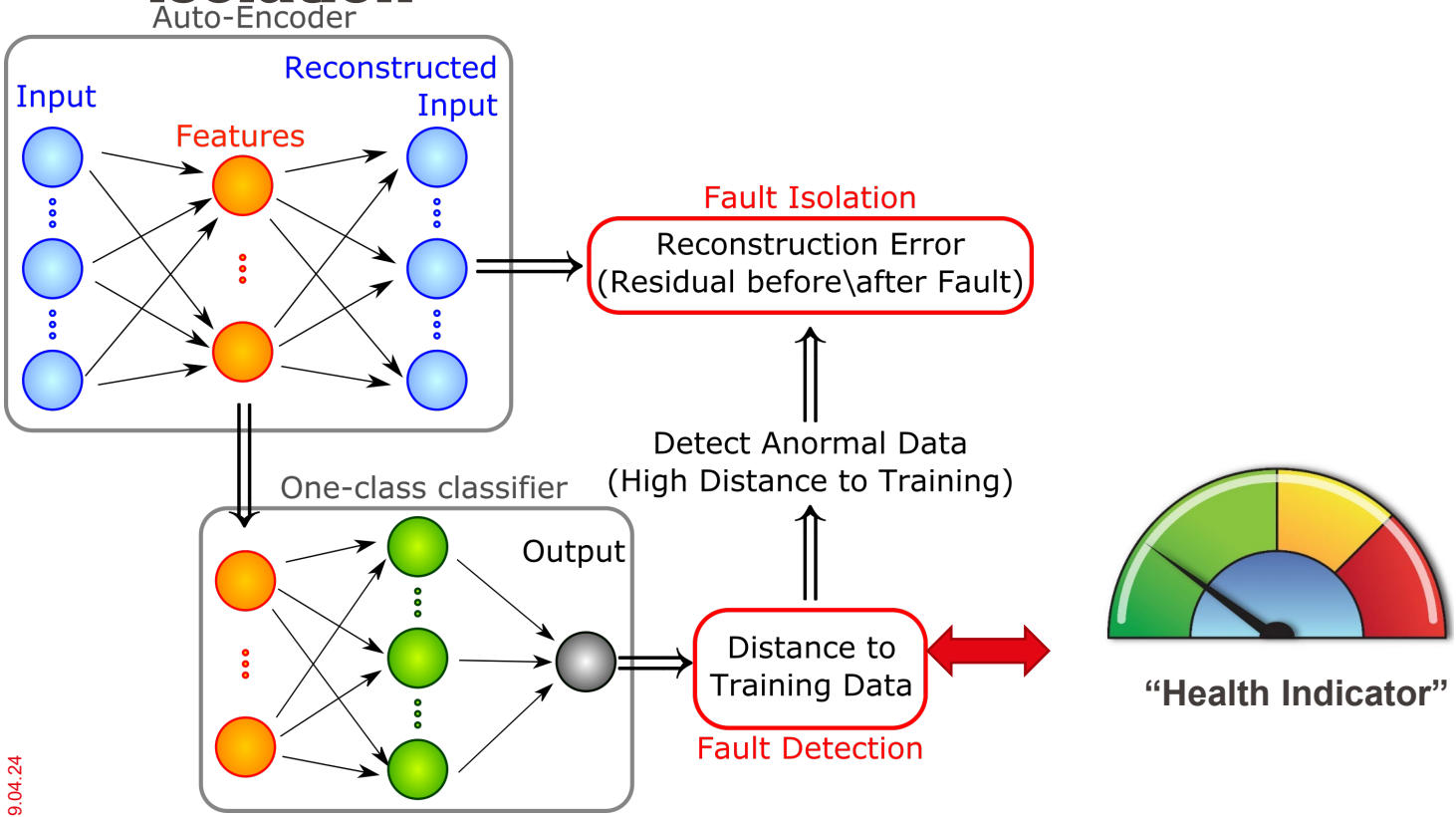
- After solving this by introduction Lagrange multipliers α_i, a new data point z can be tested to be in or out of class.
- It is considered in-class when the distance to the center is smaller than or equal to the radius, by using the Gaussian kernel as a distance function over two data points:

$$\|z - \mathbf{x}\|^2 = \sum_{i=1}^{n} \alpha_i \exp\left(\frac{-\|z - x_i\|^2}{\sigma^2}\right) \geq -R^2/2 + C_R$$

Source: Tax & Duin

Combining AE + One-Class-Classifiers

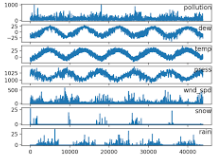
Analysing the reconstruction residuals for fault isolation

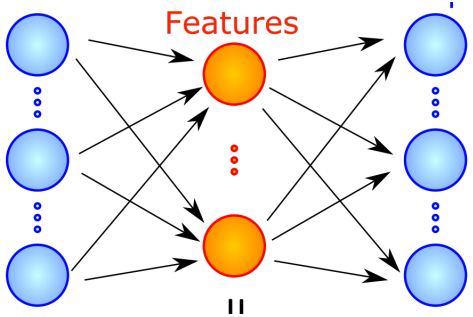


Michau, G., T. Palmé, and O. Fink (2017): Deep Feature Learning Network for Fault Detection and Isolation, Annual conference of the PHM society, October 2017

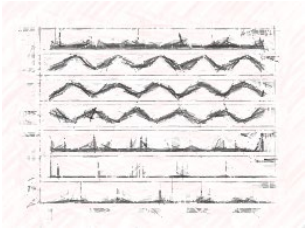
Decoder for Failure Isolation

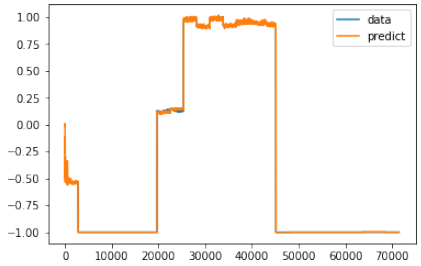
Input (x)



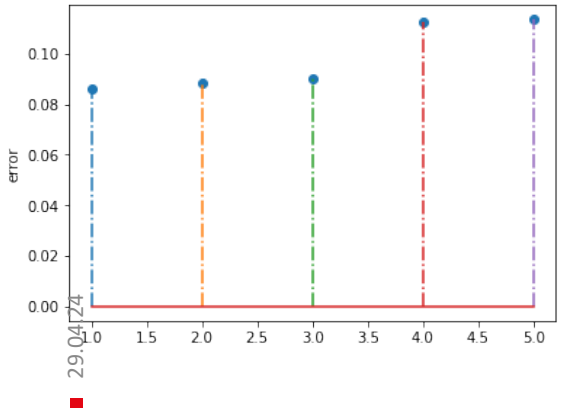


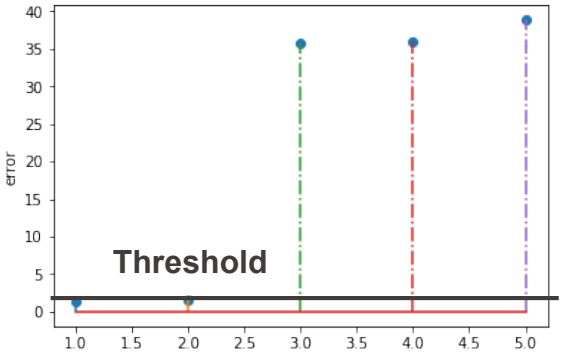
Reconstructed Input ($\bar{x}$)





Residual = $x - \bar{x}$





1. **Training**

   Residual = $\mathbf{abs}(x_i - \bar{x}_i) \sim 0$

2. **Validation**

   Residual threshold defined with healthy data

3. **Test**

   If $\mathbf{abs}(x_i - \bar{x}_i) \gg 0$ then signal i faulty

29.04.24

Generator Health Monitoring

320 monitoring sensors:

- Partial discharge
- Rotor shaft voltage
- Rotor flux
- Stator end winding vibration
- Stator Water Temperature

275 days of recorded operation,

60 000 observations

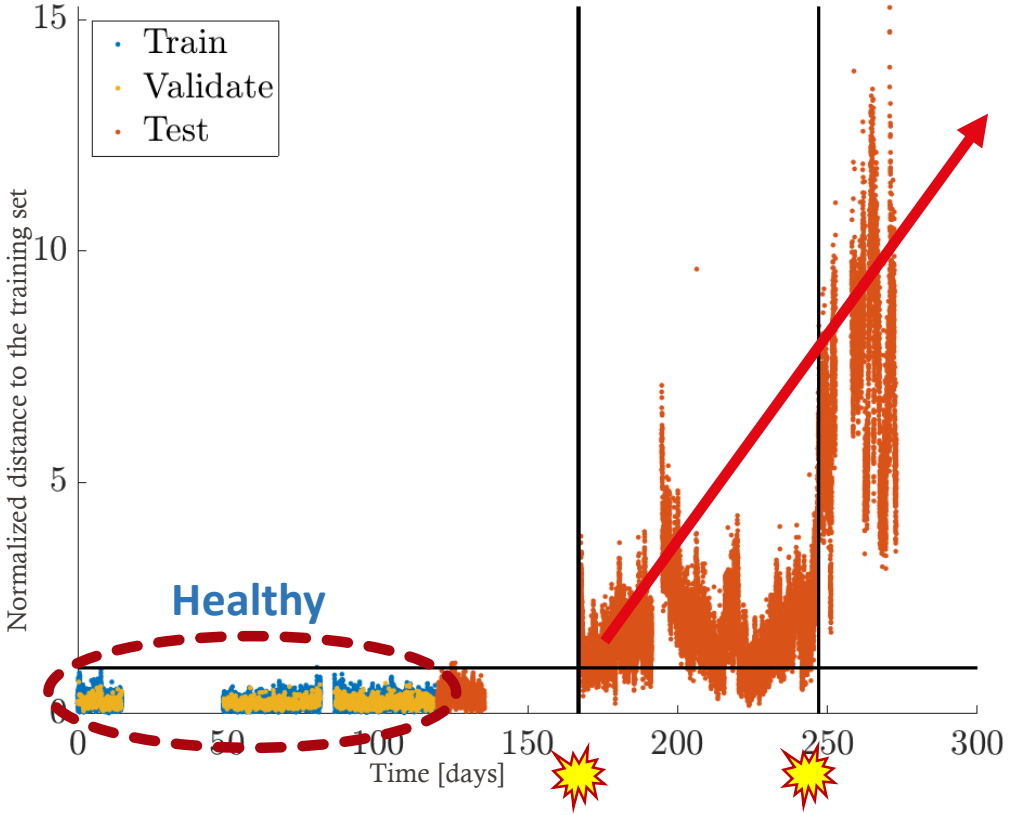
1 fault

Can only use Healthy data for training!

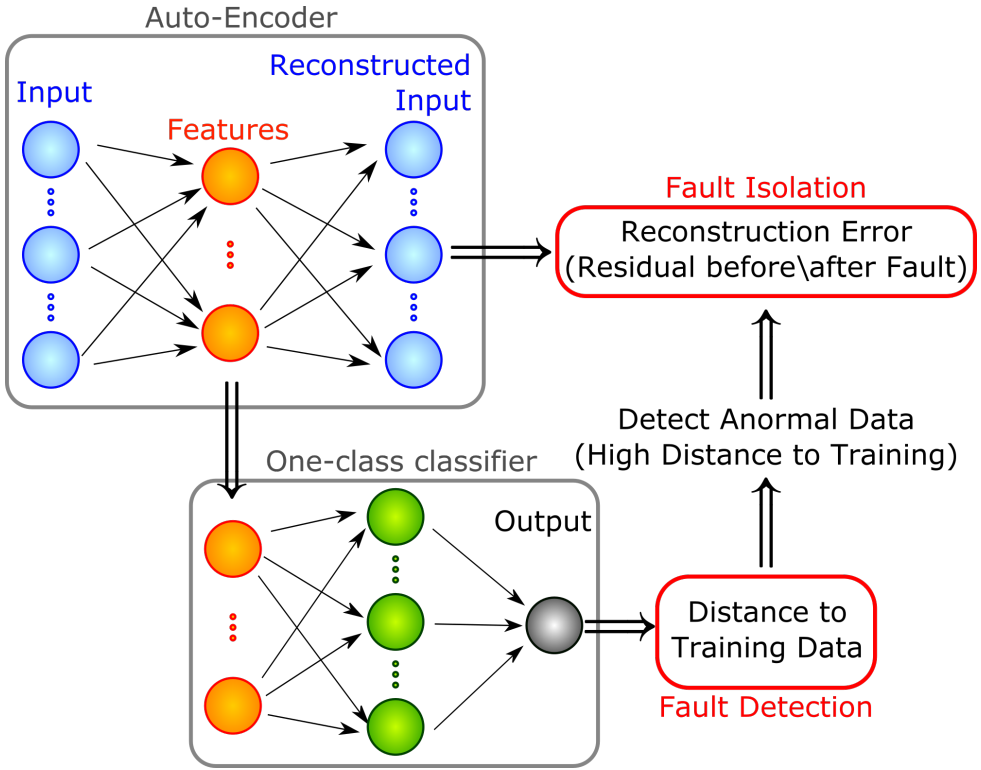
Michau, G., T. Palmé, and O. Fink (2017): Deep Feature Learning Network for Fault Detection and Isolation, Annual conference of the PHM society, October 2017

Generator Health Monitoring

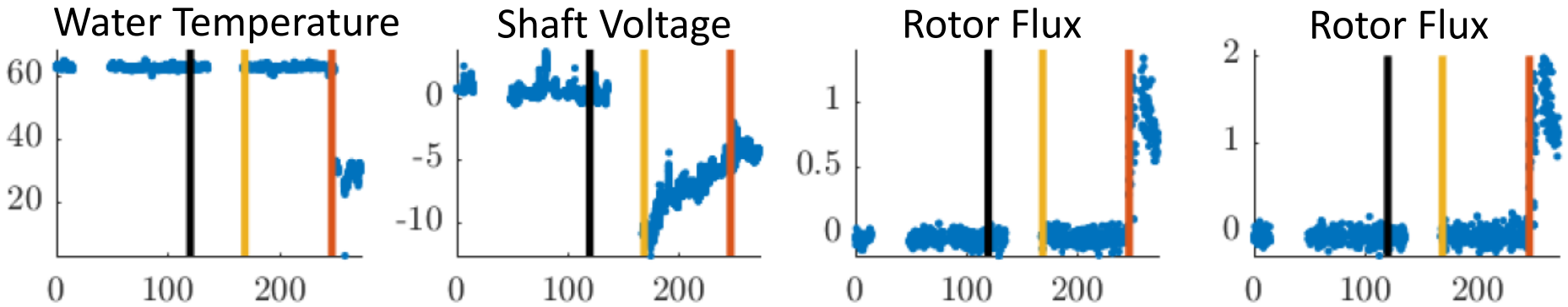
Abnormal behavior 100 days before!



Combined Architecture



Integrated fault diagnostics: Generator case study



Integrated!

At no additional cost!

Michau, G., T. Palmé, and O. Fink (2017): Deep Feature Learning Network for Fault Detection and Isolation, Annual conference of the PHM society, October 2017

Isolation Forest

Isolation Forest – overview

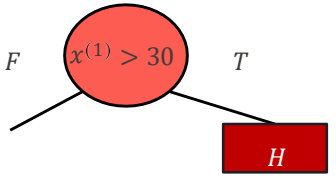
- **Aim:** provide a ranking that reflects the degree of “anomaly” for each data point
  - Sort data points according to their path lengths or anomaly scores
  - Outliers are the points with the biggest anomaly scores
- **Isolation Tree (iTree)**: binary tree where each node in the tree has exactly zero or two daughter nodes
- **Isolation Forest (iForest) algorithm**: Unsupervised Machine Learning algorithm inspired by random forests
  - Unsupervised: observations in the dataset are unlabeled
  - No need to profile normal instances and to calculate point-based distances
  - Builds an ensemble of random trees based on a mechanism called “isolation”, an iterative (random) partitioning process to separate outliers from normal points
  - Uses the observation that **outliers are more likely to be isolated with fewer steps**, compared to normal points

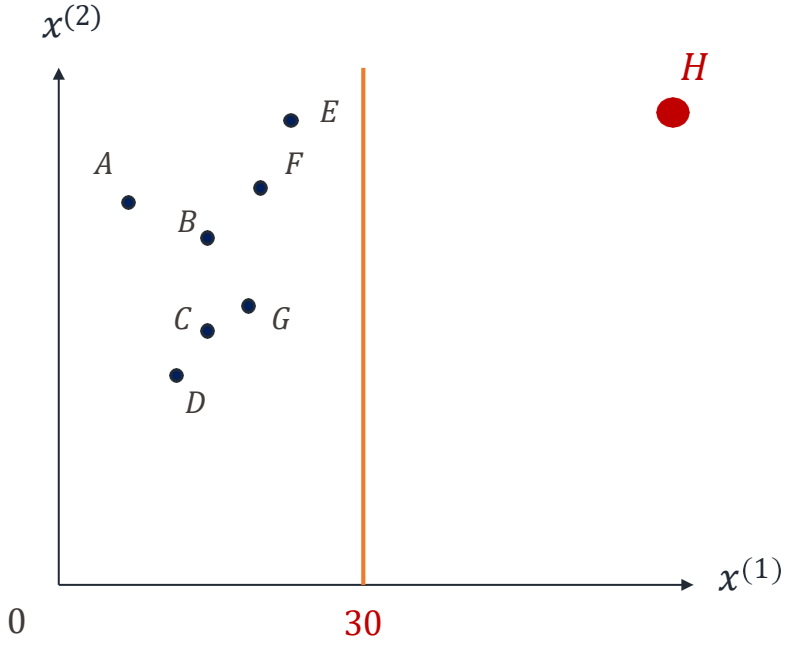
Liu, F. T., Ting, K. M., and Zhou, Z.-H. (2008). Isolation forest. In 2008 Eighth IEEE International Conference on Data Mining, pages 413-422. IEEE.

Source: Alexandre Boumezoued

Isolation Forest – example of iTree

- Example of an Isolation Tree, two-dimensional case ($d = 2$)
- Point H (outlier) is isolated with only 1 step
- More steps are needed to isolate the other points

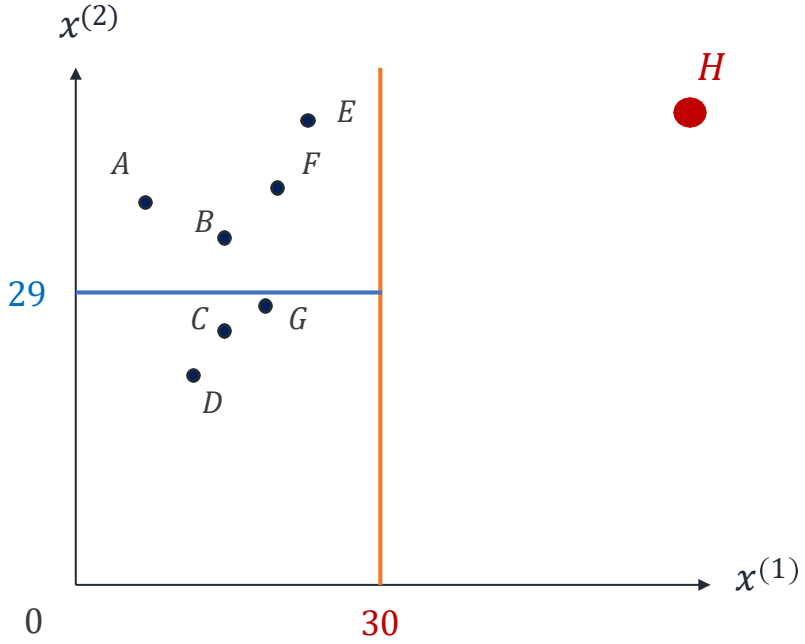


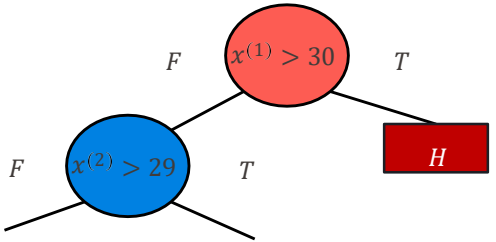


Source: Alexandre Boumezoued

Isolation Forest – example of iTree

- Example of an Isolation Tree, two-dimensional case ($d = 2$)
- Point H (outlier) is isolated with only 1 step
- More steps are needed to isolate the other points

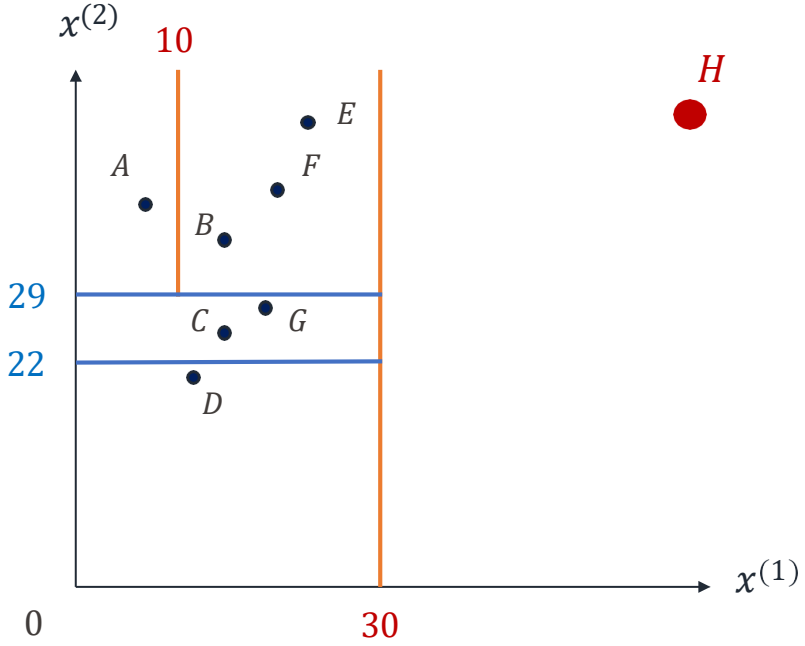


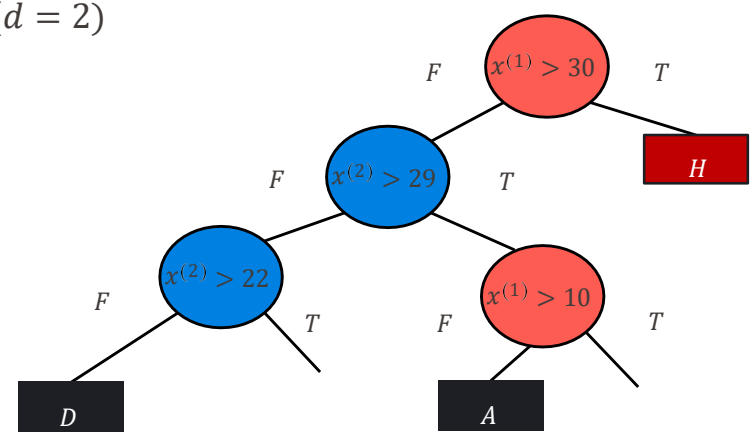


Source: Alexandre Boumezoued

Isolation Forest – example of iTree

- Example of an Isolation Tree, two-dimensional case ($d = 2$)
- Point H (outlier) is isolated with only 1 step
- More steps are needed to isolate the other points

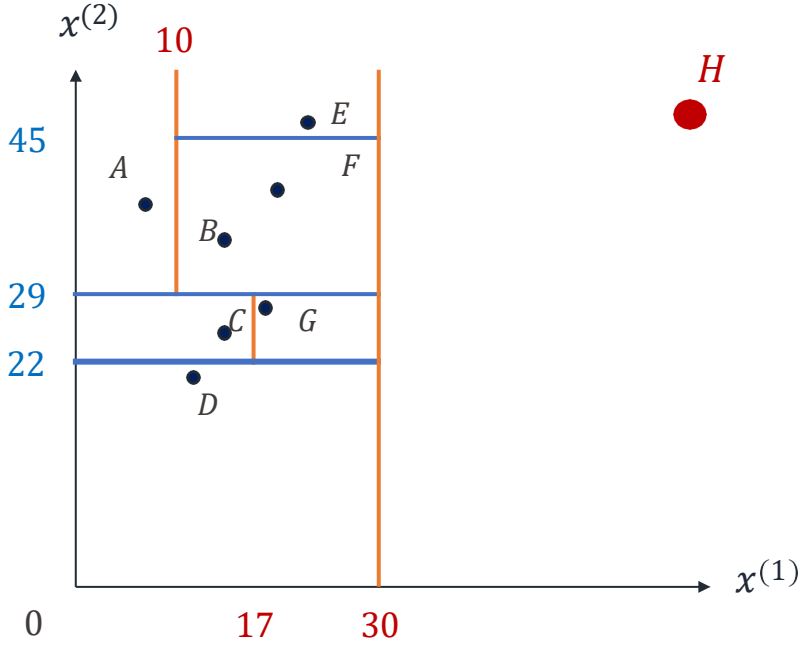


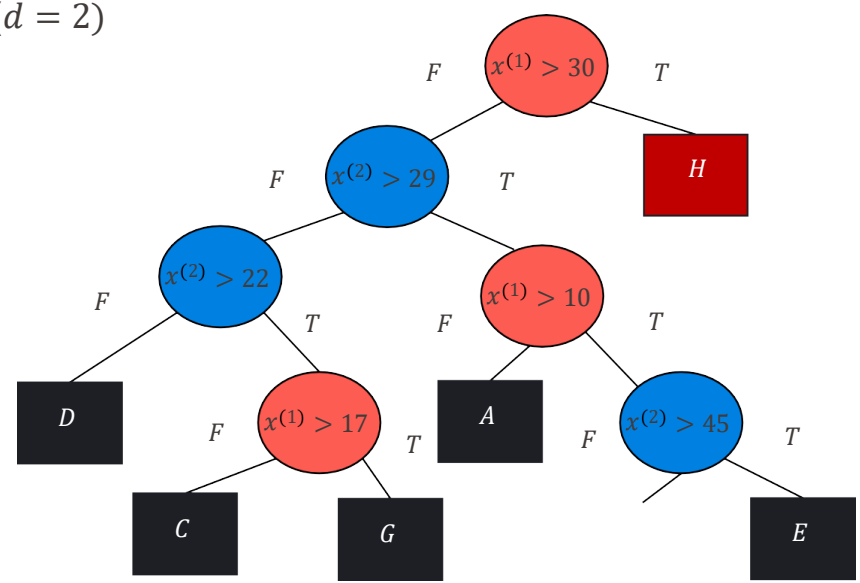


Source: Alexandre Boumezoued

Isolation Forest – example of iTree

- Example of an Isolation Tree, two-dimensional case ($d = 2$)
- Point H (outlier) is isolated with only 1 step
- More steps are needed to isolate the other points

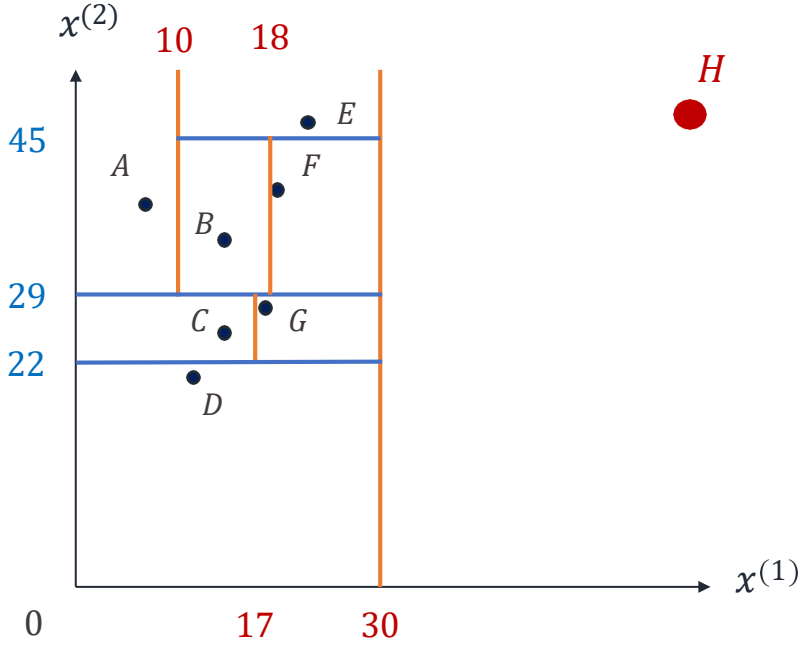


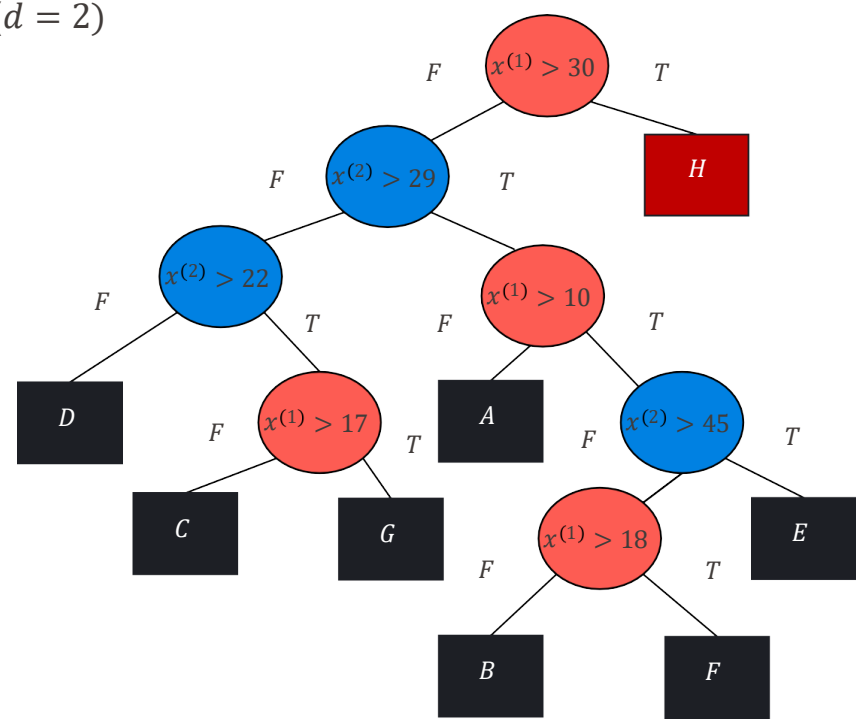


Source: Alexandre Boumezoued

Isolation Forest – example of iTree

- Example of an Isolation Tree, two-dimensional case ($d = 2$)
- Point H (outlier) is isolated with only 1 step
- More steps are needed to isolate the other points

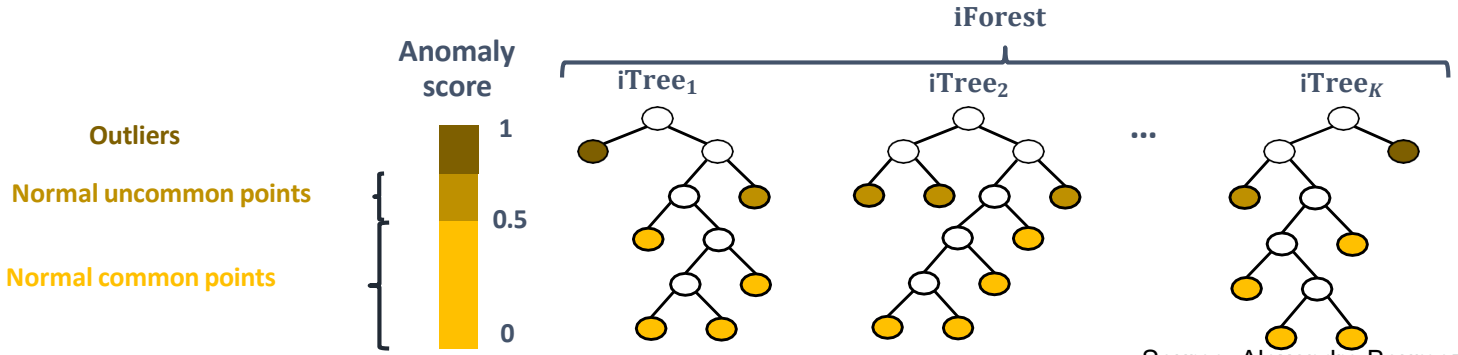




Source: Alexandre Boumezoued

Isolation Forest – from iTree to iForest

- **Ensemble method:** generates multiple iTrees → iForest
  - Path length of an observation obtained as the sum of:
    - total number of splits needed to isolate it
    - adjustment term to add if observation terminates at an external node. (Accounts for an unbuilt subtree beyond some tree height limit ℓ → saves computational time)
  - Compute the **average path lengths** $\overline{h(X_i)}$ for each observation X_i
- **Calculation of the anomaly scores**
  - Normalization by the average (universal) path length L in a binary tree
  - Anomaly score: $s(X_i) = 2^{-\frac{\overline{h(X_i)}}{L}} \in [0,1]$ ⇒ small average path length = high anomaly score



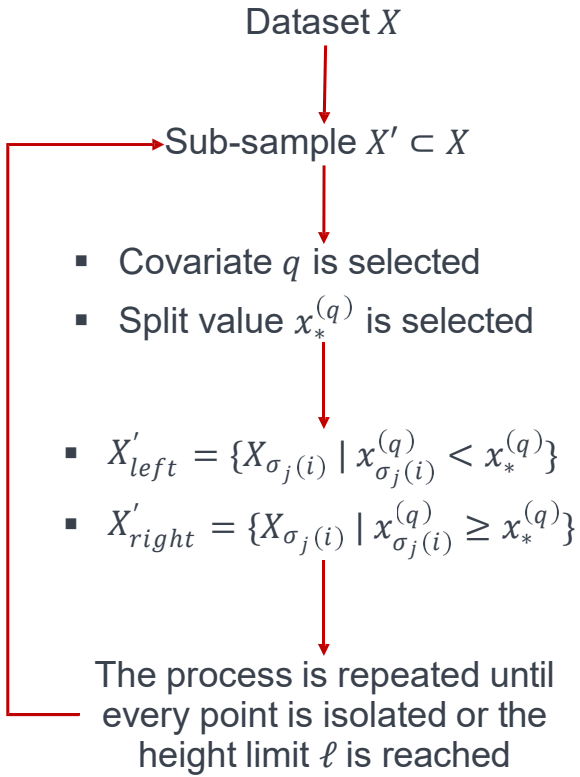
Source: Alexandre Boumezoued

Isolation Forest – details

- **Dataset**: $X = (X_1, \dots, X_n)$, with $X_i = \left(x_i^{(1)}, \dots, x_i^{(d)}\right) \in \mathbb{R}^d$ where:
  - n is the number of instances
  - d is the number of covariates

- **Sub-sampling**: An iTree j is obtained by selecting a random subset $X' \subset X$, where $|X'| = \psi < n$, $X' = \left(X_{\sigma_j(1)}, \dots, X_{\sigma_j(\psi)}\right)$, and $\sigma_j : [\![1, \psi]\!] \to [\![1, n]\!]$ is a (random) injective function.

- X' is then divided recursively by **randomly selecting a covariate** $q \in \{1, \dots, d\}$ and a **split value** $x_*^{(q)} \in [\min_{1 \le i \le \psi} x_{\sigma_j(i)}^{(q)}, \max_{1 \le i \le \psi} x_{\sigma_j(i)}^{(q)}]$ until:
  - Either the tree reaches the height limit ℓ, which is approximately the average tree height
  - Or $|X'| = 1$, i.e. there is only one unique point remaining



Source: Alexandre Boumezoued

Isolation Forest – about swamping and masking

- Swamping & masking are standard issues in outlier detection problems
  - **Swamping**: wrong identification of normal instances as outliers in the case where many variables are non informative on the “outlier” nature. (the split based on these variables is not appropriate)
  - **Masking**: when too many outliers coexist in the dataset, the splitting rules are not efficient to isolate data points since many iterations are needed
- Both problems are consequences of too many data for the purpose of outlier detection
- Solution of iForest algorithm: **Sub-sampling**
  - Controls data size, which helps to better isolate examples of outliers
  - Each iTree can be specialized, each sub-sample including a different set of outliers
- It has been shown that iForest's outlier detection ability is superior when sub-sampling is used

Source: Alexandre Boumezoued

Isolation Forest – pros and challenges

- **Pros**
  - Unsupervised method: does not require labels of outliers provided by expert judgements
  - No model needed (the aim is not to model normal instances)
  - Provides a hierarchy by assigning an anomaly score to each observation
  - Does not require examples of outliers in the training set
  - Requires relatively small samples from large datasets to derive an outlier detection function
  - The algorithm can be trained once and reused without computational cost
  - Achieves a linear time complexity with low memory requirement
    - by using sub-sampling
    - by avoiding building trees after reaching a height limit ℓ
  - Overcomes the problem of swamping/masking by using sub-sampling
- **Challenges**
  - Requires working on the data to provide appropriate format of the covariates
  - Tuning parameters need to be set
  - To avoid the black-box syndrome, it benefits from a pre-selection of covariates in line with the problem to be tackled

Source: Alexandre Boumezoued

Deep Semi-Supervised Anomaly Detection Deep SAD

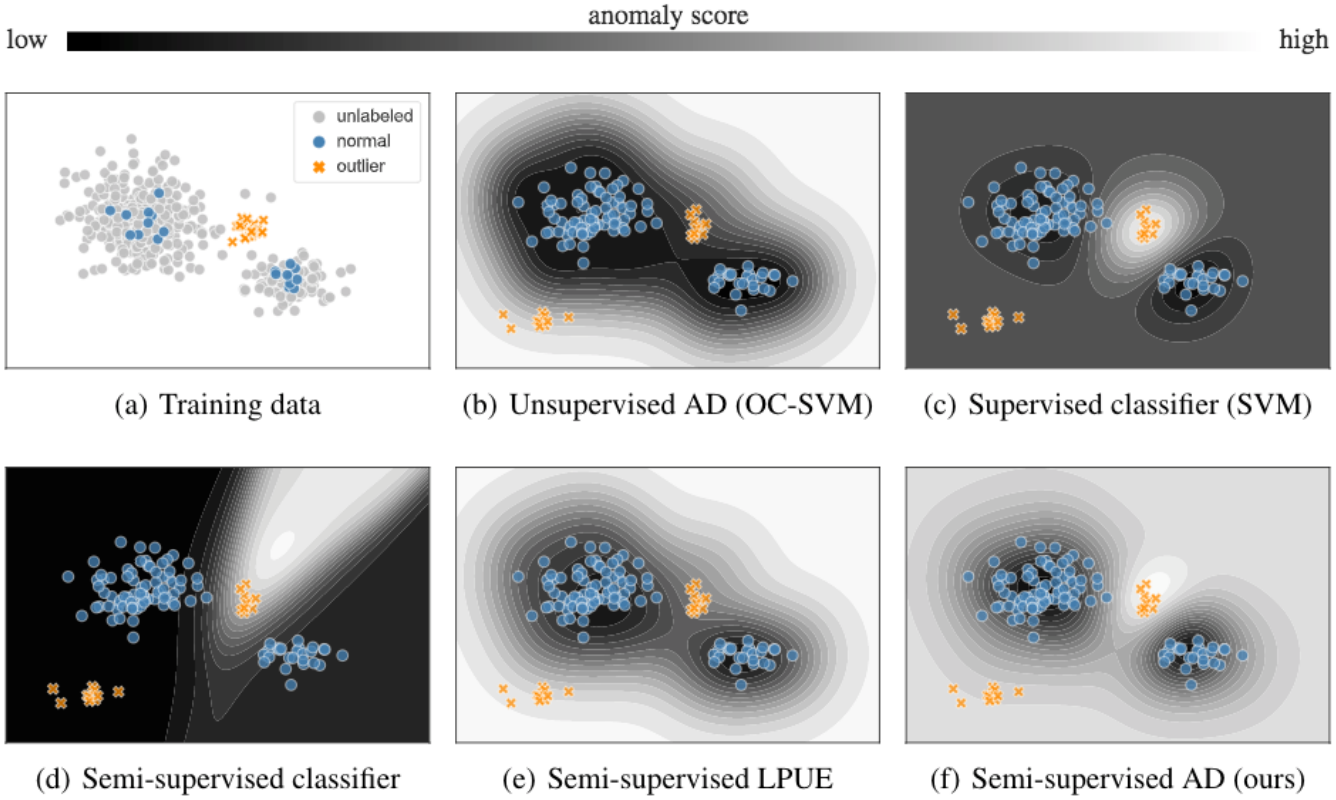
- Extends the the concept of deep SVDD to semi-supervised setups

- Deep SAD combines a few labeled examples (both normal and anomalous) with a larger pool of unlabeled data during training.

$$\min_{\mathcal{W}} \quad \frac{1}{n+m} \sum_{i=1}^{n} \|\phi(\boldsymbol{x}_i; \mathcal{W}) - \boldsymbol{c}\|^2 + \frac{\eta}{n+m} \sum_{j=1}^{m} \left(\|\phi(\tilde{\boldsymbol{x}}_j; \mathcal{W}) - \boldsymbol{c}\|^2\right)^{\tilde{y}_j} + \frac{\lambda}{2} \sum_{\ell=1}^{L} \|\boldsymbol{W}^\ell\|_F^2.$$

Ruff, L., Vandermeulen, R. A., Görnitz, N., Binder, A., Müller, E., Müller, K. R., & Kloft, M. (2019, September). Deep Semi-Supervised Anomaly Detection. In *International Conference on Learning Representations*.

Deep SAD compared to other AD approaches



Ruff, L., Vandermeulen, R. A., Görnitz, N., Binder, A., Müller, E., Müller, K. R., & Kloft, M. (2019, September). Deep Semi-Supervised Anomaly Detection. In *International Conference on Learning Representations*.

Explainability / Interpretability

Most of the time, the algorihtms work really well but sometimes…

A refrigerator filled with lots of food and drinks

Safety and well being

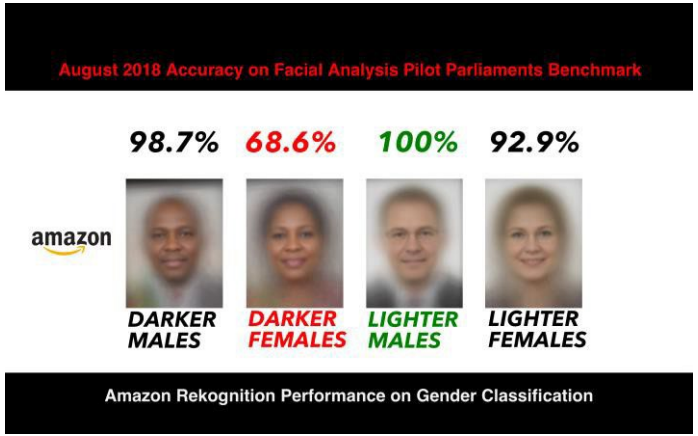
Tesla hit parked police car 'while using Autopilot'

30 May 2018 Share

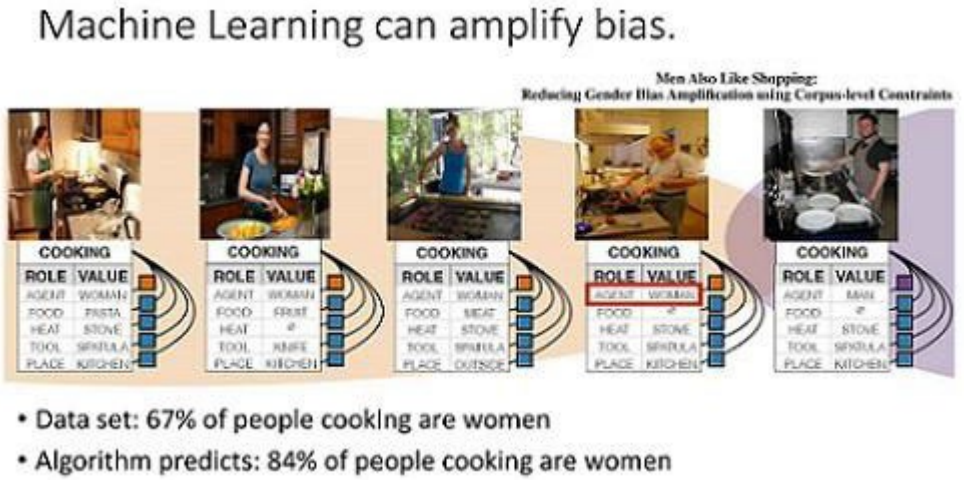
A number of Tesla vehicles have been involved in crashes.

Source: Byron Wallace

Bias in algorithms



https://medium.com/@Joy.Buolamwini/response-racial-and-gender-bias-in-amazon-rekognition-commercial-ai-system-for-analyzing-faces-a289222eeced



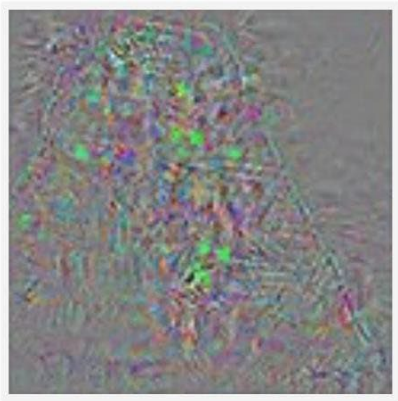
https://www.infoq.com/presentations/unconscious-bias-machine-learning/

Source: Byron Wallace

Adversarial Examples

Original image

Temple (97%)



Perturbations

Adversarial example

Ostrich (98%)

Source: Byron Wallace

Current limitations of the ML algorithms

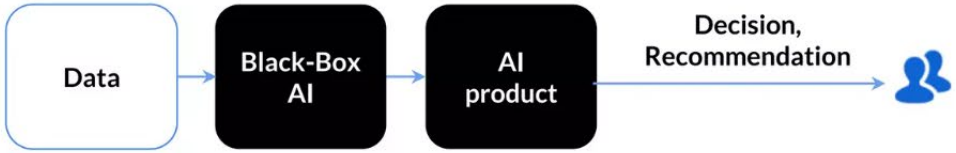
- Interpretability (Explainability, Transparency, Understanding, Trust)
- Physical consistency
- Complex and uncertain data
- Limited labels
- (Computational demand)

What is interpretability?

- Ability to explain or to present a model in understandable terms to humans (Doshi-Velez 2017)

Why explainable AI?

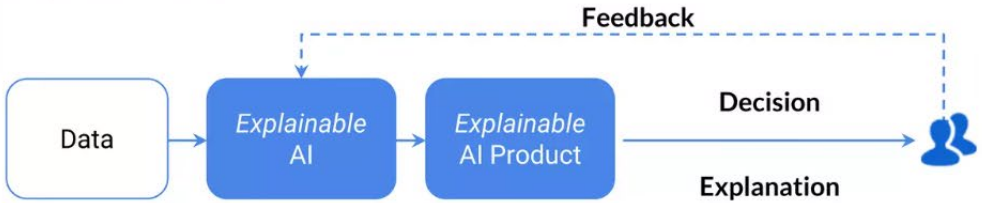
Black Box AI



Confusion with Today's AI Black Box

- Why did you do that?
- Why did you not do that?
- When do you succeed or fail?
- How do I correct an error?

Explainable AI



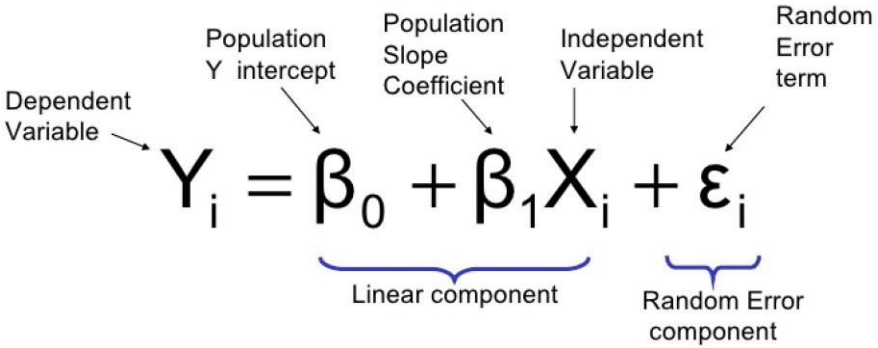
Clear & Transparent Predictions

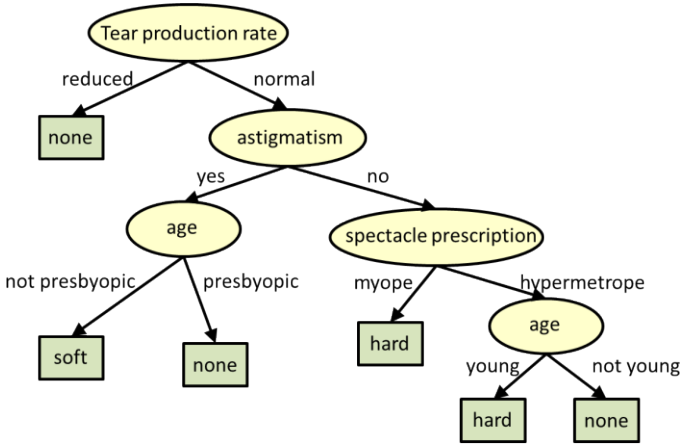
- I understand why
- I understand why not
- I know why you succeed or fail
- I understand, so I trust you

Source: WWW 2020 Tutorial

Simple explainability

- In pre-deep learning models, some models are considered “interpretable”



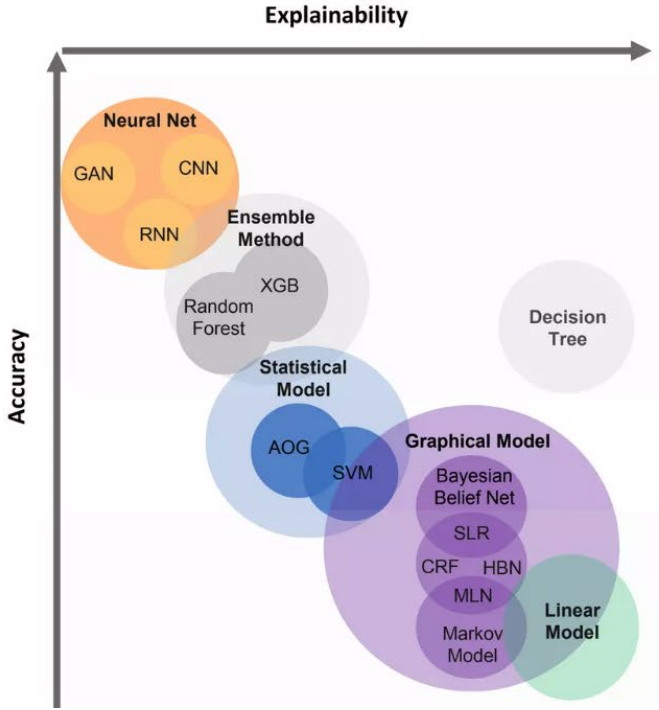


Source: Byron Wallace

Accuracy vs. Explainability

Learning

- Challenges:
  - Supervised
  - Unsupervised learning
- Approach:
  - Representation Learning
  - Stochastic selection
- Output:
  - **Correlation**
  - **No causation**



Interpretability

- Non-Linear functions
- Polynomial functions
- Quasi-Linear functions

Source: WWW 2020 Tutorial

Different ways to achieve interpretability

- Build interpretability into the model (e.g. by fusing physical models and machine learning or learning the underlying physics explicitly)
- Post-hoc approach to interpretability → trying to explain given models and their output

Some properties of Interpretations

- **Faithfulness** - how to provide explanations that accurately represent the true reasoning behind the model's final decision.
- **Plausibility** – Is the explanation correct or something we can believe is true, given our current knowledge of the problem ?
- **Understandable** – Can I put it in terms that end user without in-depth knowledge of the system can understand ?
- **Stability** – Do similar instances have similar interpretations ?

Source: Byron Wallace

Different ways to achieve interpretability

- Build interpretability into the model (e.g. by fusing physical models and machine learning or learning the underlying physics explicitly)
- Post-hoc approach to interpretability → trying to explain given models and their output

Evaluating Interpretability [Doshi-Velez 2017]

- Application level evaluation – Put the model in practice and have the end users interact with explanations to see if they are useful .
- Human evaluation – Set up a Mechanical Turk task and ask non- experts to judge the explanations
- Functional evaluation – Design metrics that directly test properties of your explanation.

Global vs Local

- **Do we explain individual prediction ?**

- Example –

- Heatmaps
- Rationales

- **Do we explain entire model?**

- Example –

- Linear Regression
- Decision Trees

Source: Byron Wallace

Inherent vs Post-hoc

- Is the explainability built into the model ?

- Examples:

- Linear Regression
- Decision Trees

- Is the model black-box and we use external method to try to understand it ?

- Examples:

- Heatmaps (Some forms)

Source: Byron Wallace

Model based vs Model Agnostic

- **Can it explain only few classes of models?**

- Examples:

- Decision Trees
- Attention
- Gradients (Differentiable Models only)

- **Can it explain any model ?**

- Examples:

- LIME – Locally Interpretable Model Agnostic Explanations
- SHAP – Shapley Values

Source: Byron Wallace

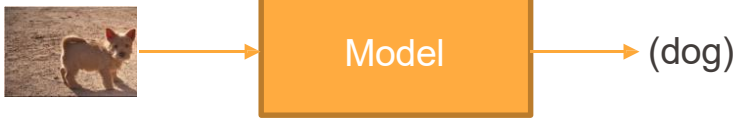
Different approaches to explain the behavior of ML approaches post-hoc

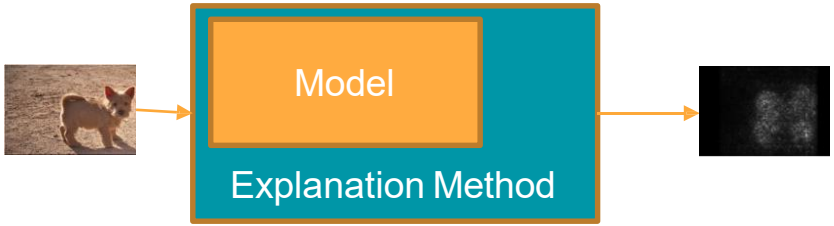
- Explaining with Surrogates
- Explaining with local perturbations
- Propagation-Based Approaches (Leveraging Structure)
- Meta-explanations

W. Samek and K. R. Müller, “Towards Explainable Artificial Intelligence,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11700 LNCS, Springer Verlag, 2019, pp. 5–22.

Saliency Based Methods

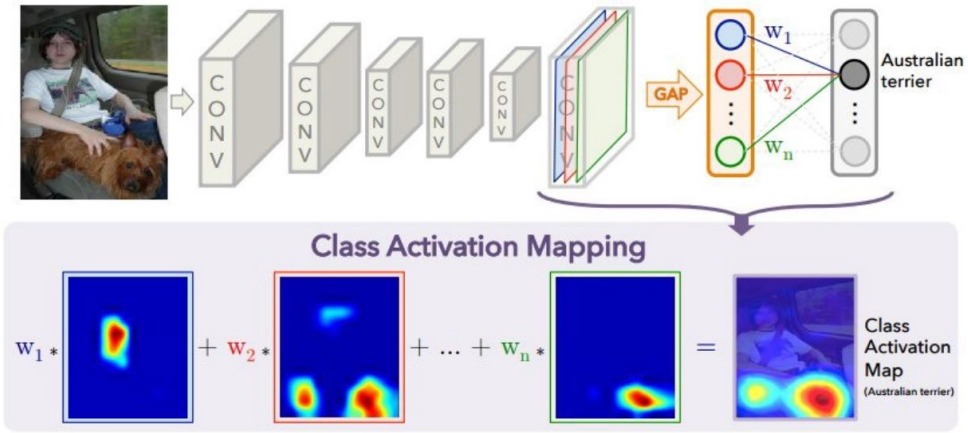
- Heatmap based visualization
- Need differentiable model in most cases
- Normally involve gradient





Source: Byron Wallace

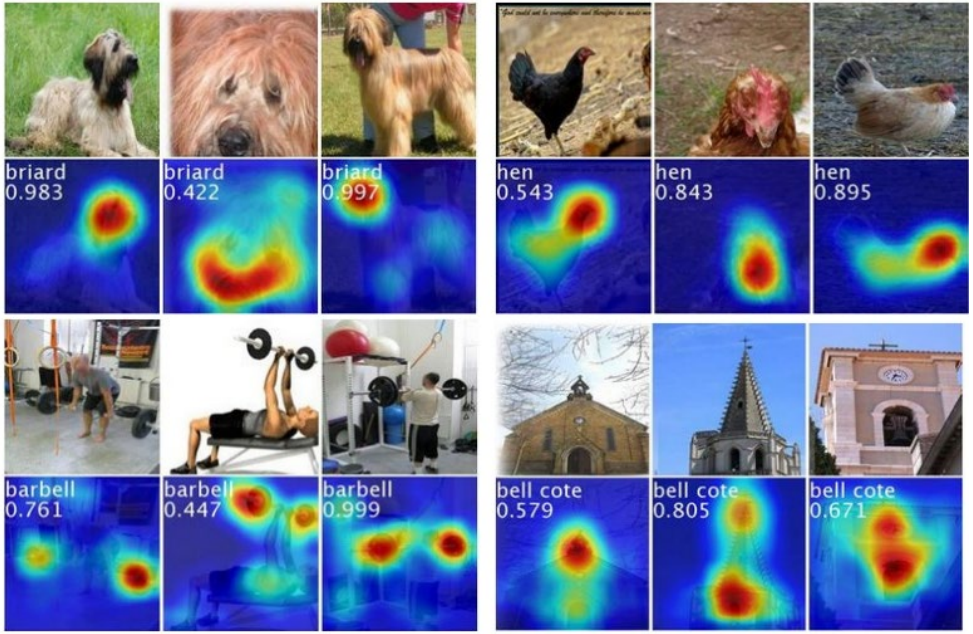
Class Activation Mapping



- GAP: Global average pooling
- We can identify the importance of the image regions by projecting back the weights of the output layer on the convolutional feature maps obtained from the last Convolution Layer.

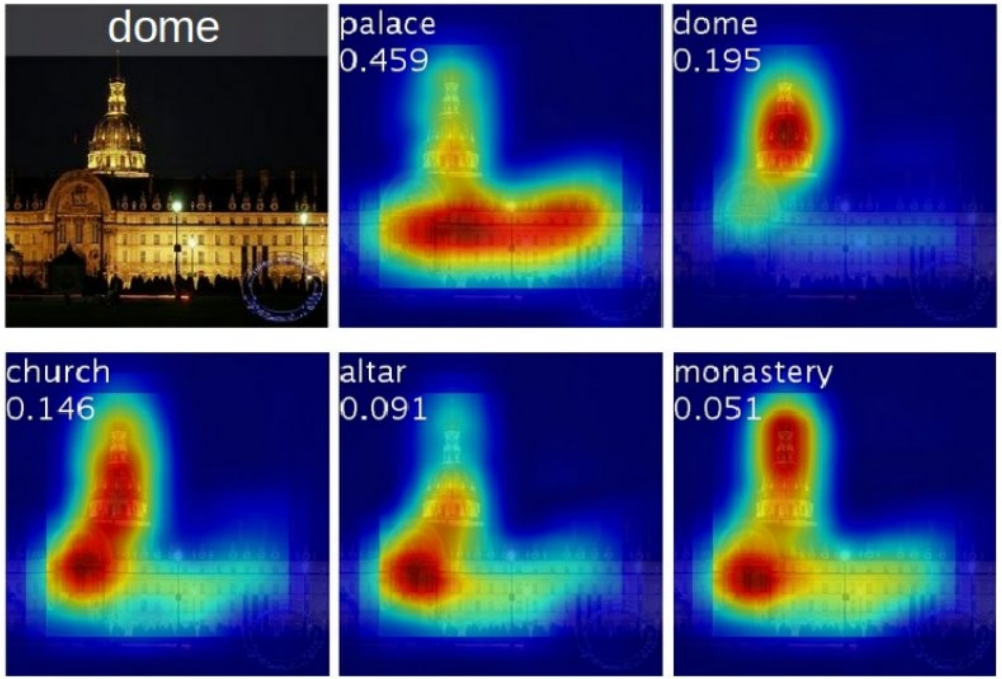
Zhou, Bolei, et al. "Learning deep features for discriminative localization." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

Class Activation Mapping: examples

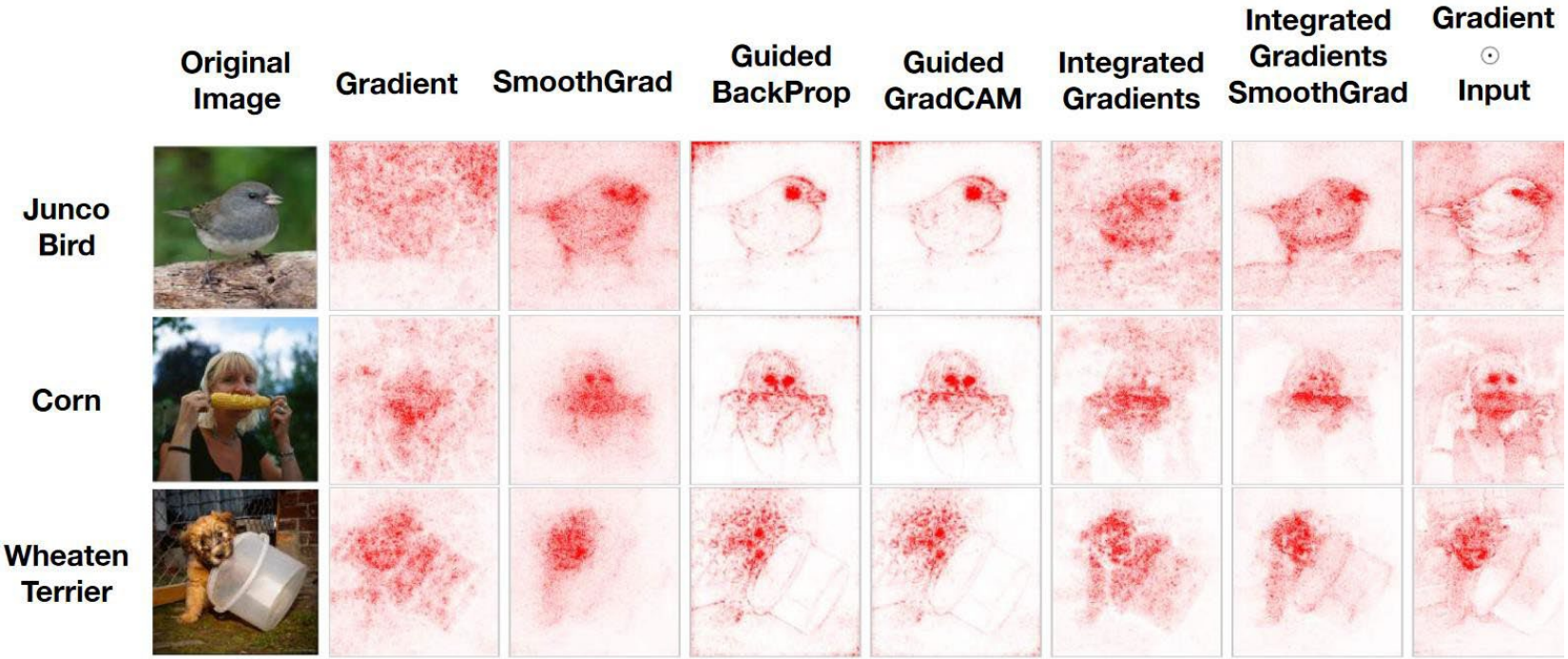


Zhou, Bolei, et al. "Learning deep features for discriminative localization." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

Class Activation Mapping



Zhou, Bolei, et al. "Learning deep features for discriminative localization." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.



[Adebayo et al 2018]

Saliency Example - Gradients

$$f(x): R^d \rightarrow R$$

$$E(f)(x) = \frac{df(x)}{dx}$$

Source: Byron Wallace

Saliency Example – Leave-one-out

$$f(x): R^d \rightarrow R$$

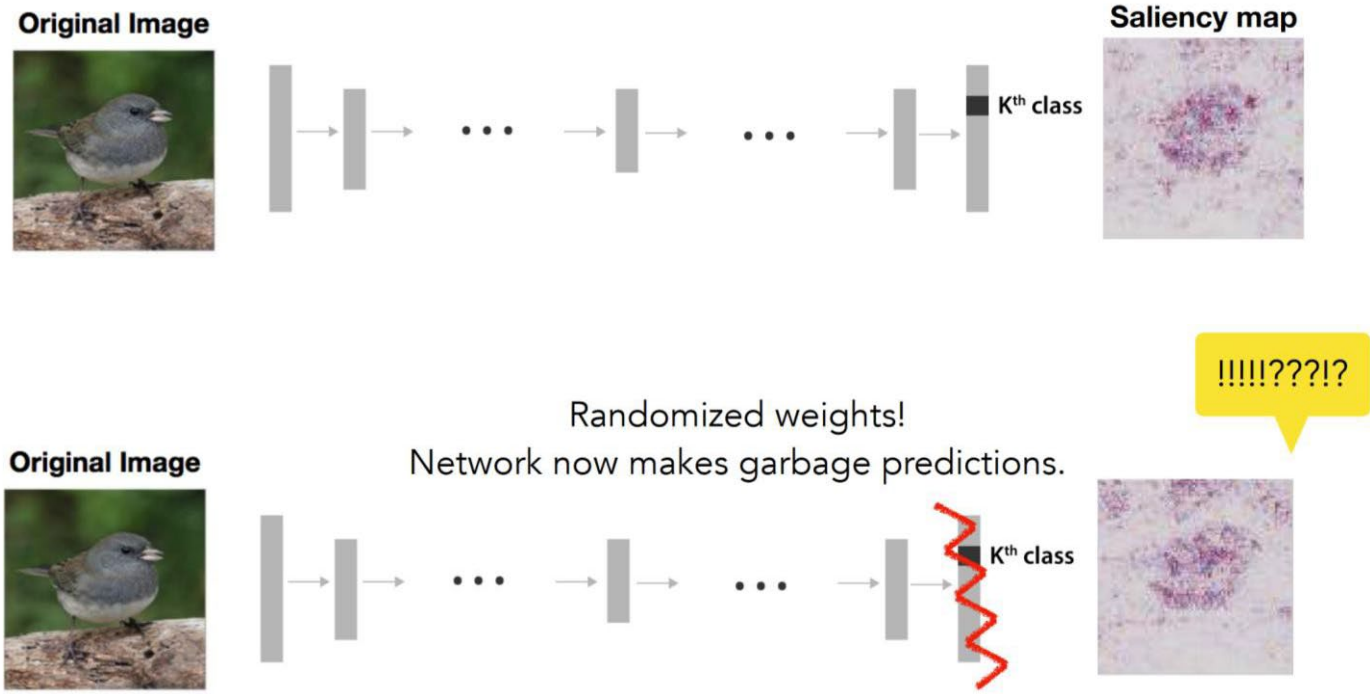
$$E(f)(x)_i = f(x) - f(x\backslash \mathrm{i})$$

How to remove ?

1. Zero out pixels in image
2. Remove word from the text
3. Replace the value with population mean in tabular data

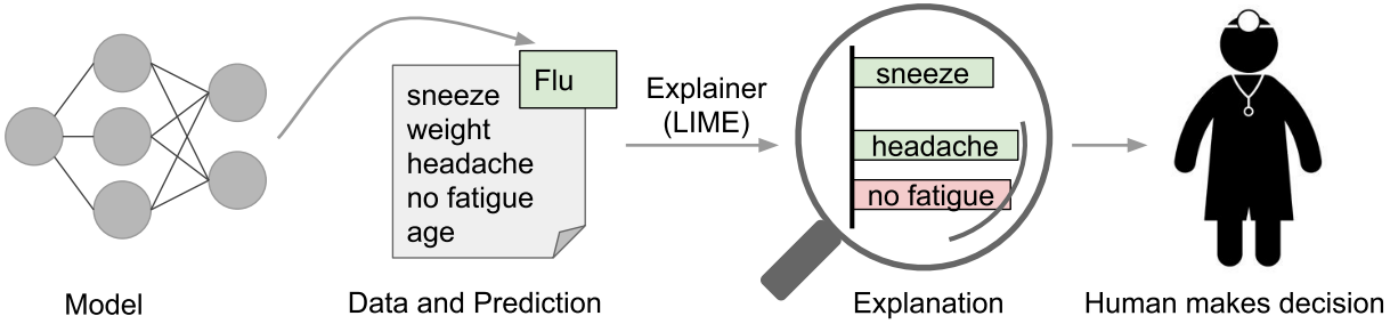
Source: Byron Wallace

Sanity check:
When prediction changes, do explanations change?



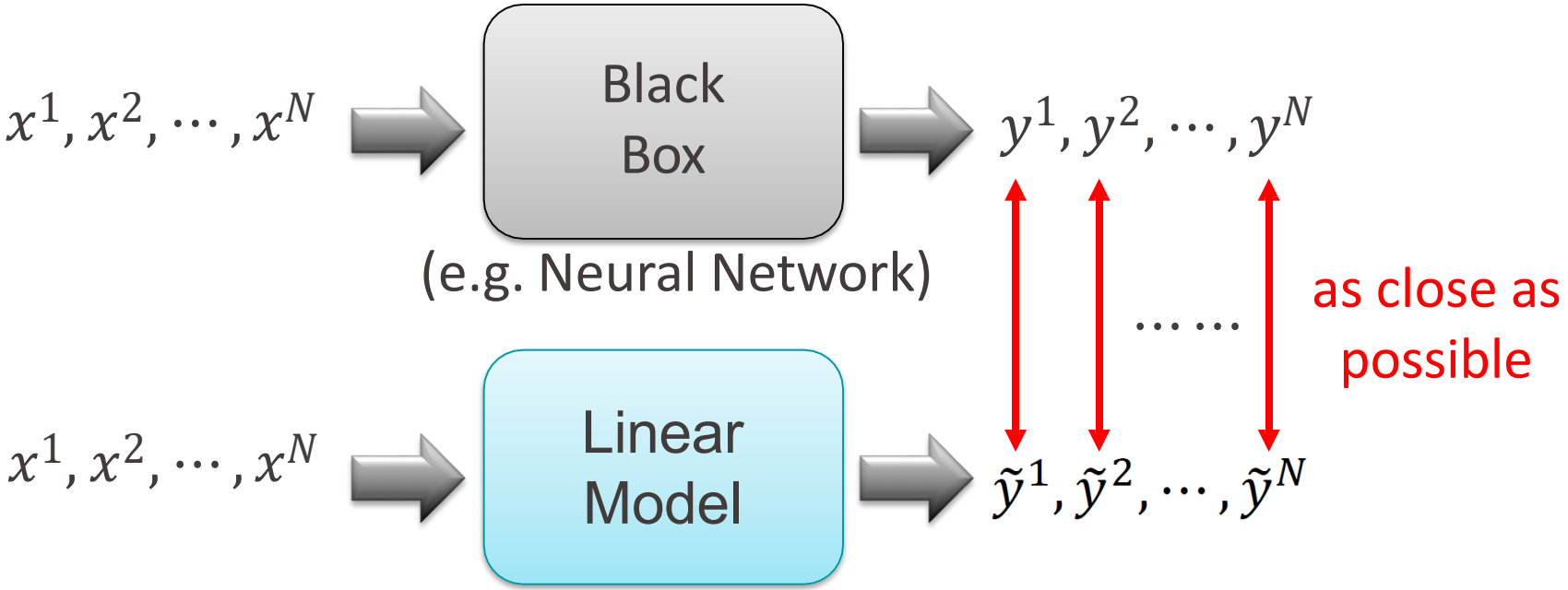
Source: Julius Adebayo)

Local Interpretable Model-Agnostic Explanations (LIME): basic idea



M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?' Explaining the predictions of any classifier," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, vol. 13-17-Augu, pp. 1135–1144.

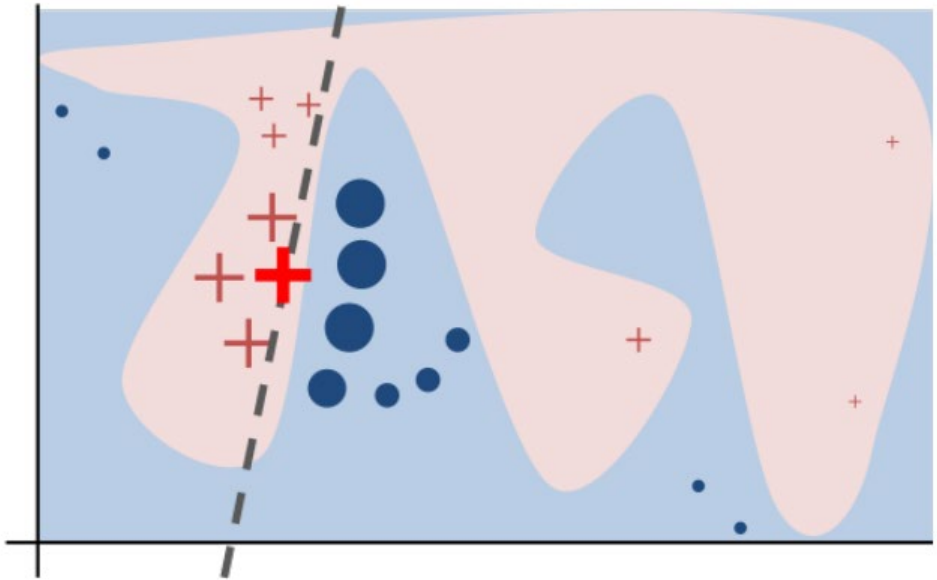
LIME – locally interpretable model agnostic



Can't do it globally of course, but locally ? Main Idea behind LIME

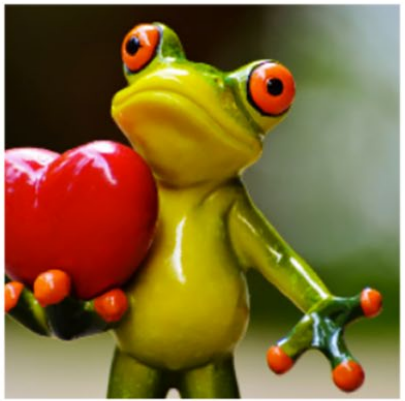
Source: Hung-Yi Lee

LIME: Toy example of the basic concept



M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?' Explaining the predictions of any classifier," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, vol. 13-17-Augu, pp. 1135–1144.

LIME: divide images into interpretable components (contiguous superpixels)

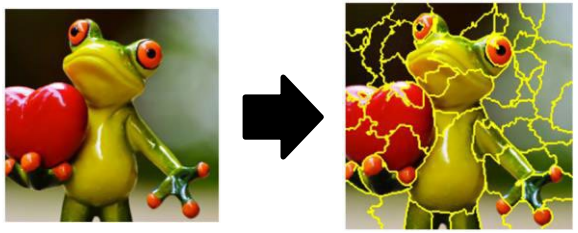


Original Image

Interpretable Components

M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?' Explaining the predictions of any classifier," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, vol. 13-17-Augu, pp. 1135–1144.

LIME — Image



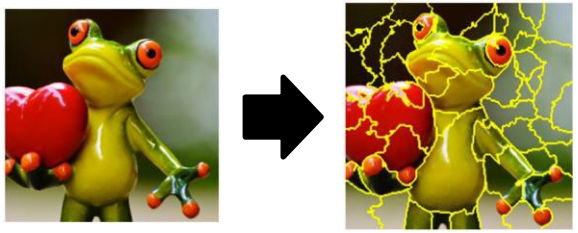
1. Given a data point you want to explain
2. Sample at the nearby - Each image is represented as a set of superpixels (segments).

- Randomly delete some
- segments.

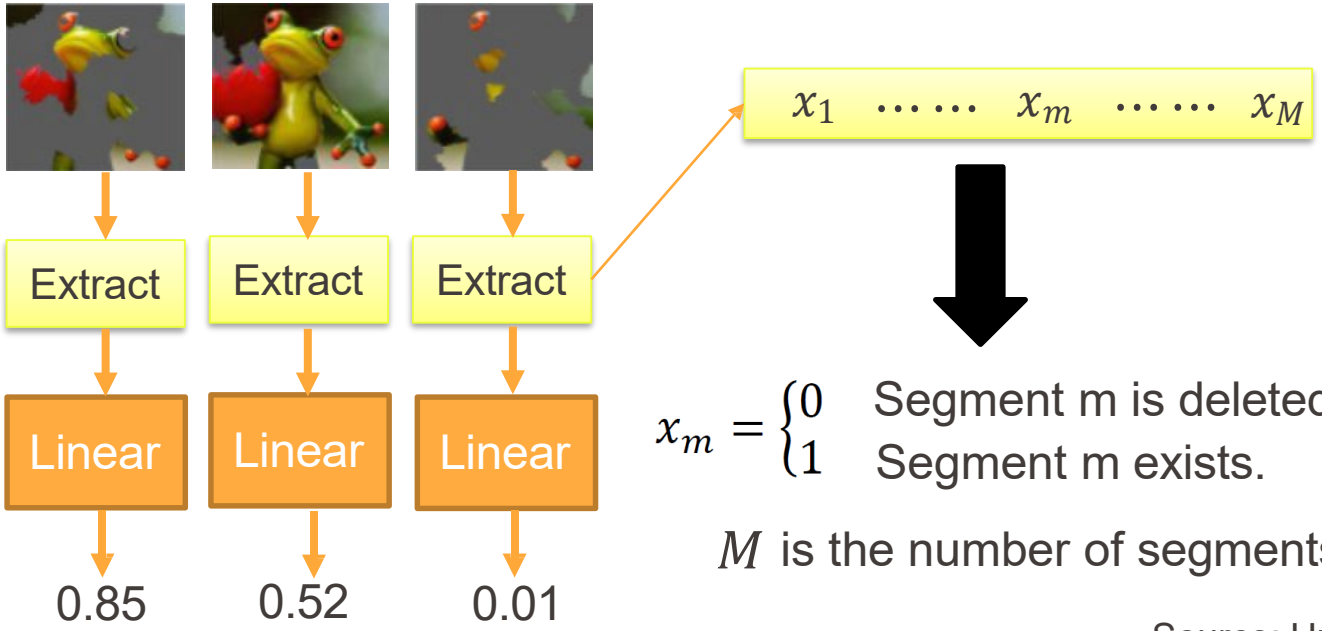
Compute the probability of “frog” by black box

Source: Hung-Yi Lee

LIME — Image



- 3. Fit with linear (or interpretable) model

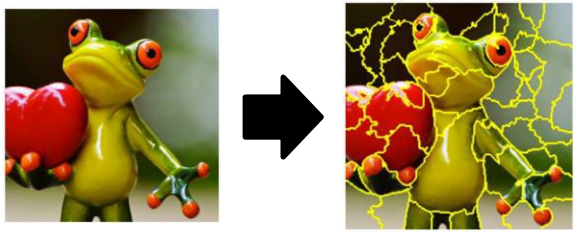


$$x_m = \begin{cases} 0 & \text{Segment m is deleted.} \\ 1 & \text{Segment m exists.} \end{cases}$$

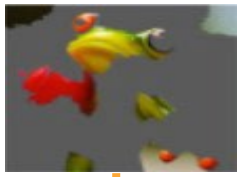
M is the number of segments.

Source: Hung-Yi Lee

LIME — Image



- 4. Interpret the model you learned



$$y = w_1 x_1 + \cdots + w_m x_m + \cdots + w_M x_M$$

$$x_m = \begin{cases} 0 & \text{Segment m is deleted.} \\ 1 & \text{Segment m exists.} \end{cases}$$

M is the number of segments.

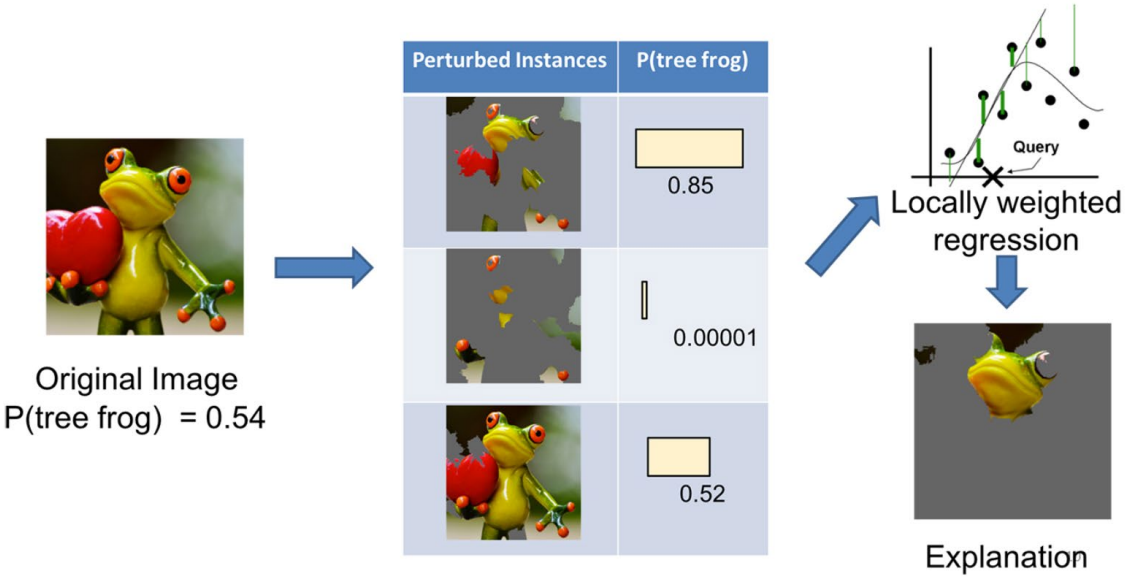
If $w_m \approx 0$ ➡ segment m is not related to “frog”

If w_m is positive ➡ segment m indicates the image is “frog”

If w_m is negative ➡ segment m indicates the image is not “frog”

Source: Hung-Yi Lee

LIME: perturbation + local models



M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?' Explaining the predictions of any classifier," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, vol. 13-17-Augu, pp. 1135–1144.

LIME: underlying algorithm

Algorithm 1 Sparse Linear Explanations using LIME

Require: Classifier f, Number of samples N
Require: Instance x, and its interpretable version x'
Require: Similarity kernel π_x, Length of explanation K

$\mathcal{Z} \leftarrow \{\}$
for $i \in \{1, 2, 3, ..., N\}$ **do**
  $z'_i \leftarrow sample\_around(x')$
  $\mathcal{Z} \leftarrow \mathcal{Z} \cup \langle z'_i, f(z_i), \pi_x(z_i) \rangle$
end for
$w \leftarrow$ K-Lasso$(\mathcal{Z}, K)$ ▷ with z'_i as features, $f(z)$ as target
return w

Match interpretable model to black box

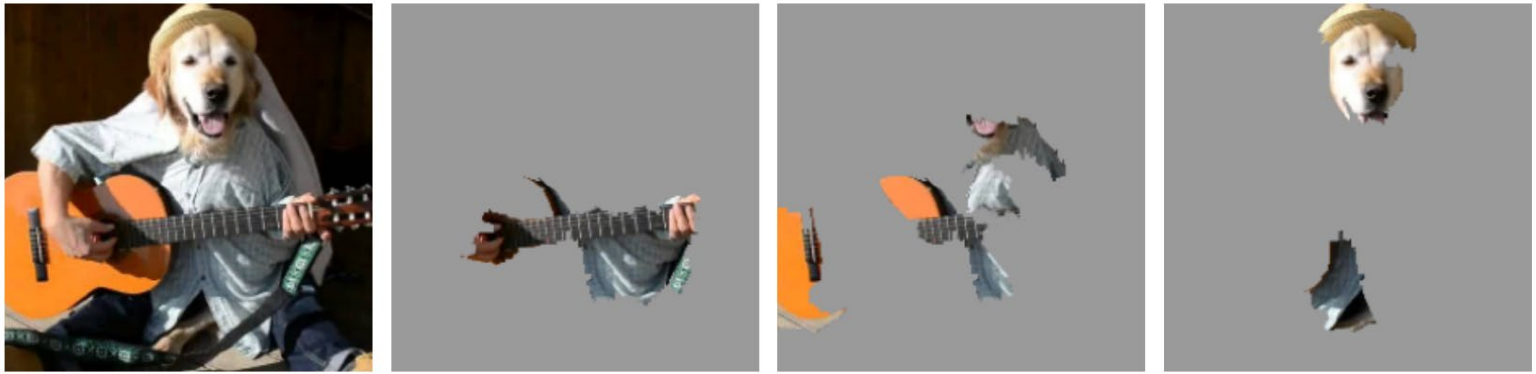
Control complexity of the model

$$\xi(x) = \underset{g \in G}{\operatorname{argmin}} \; \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

$$\mathcal{L}(f, g, \pi_x) = \sum_{z, z' \in \mathcal{Z}} \pi_x(z) \left(f(z) - g(z')\right)^2$$

M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?' Explaining the predictions of any classifier," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, vol. 13-17-Augu, pp. 1135–1144.

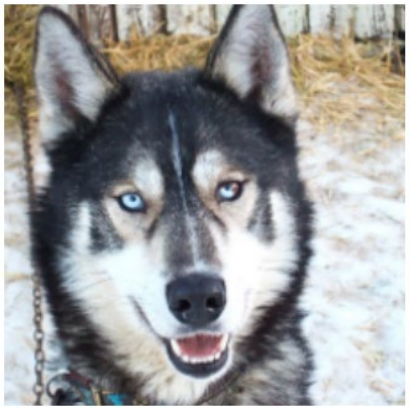
LIME: Example



(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?' Explaining the predictions of any classifier," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, vol. 13-17-Augu, pp. 1135–1144.

LIME: bringing trust («Husky vs Wolf)



(a) Husky classified as wolf

(b) Explanation

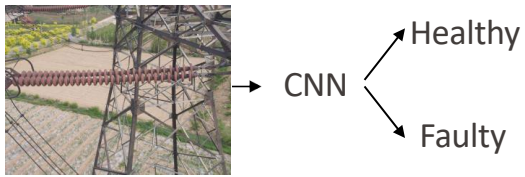
	Before	After
Trusted the bad model	10 out of 27	3 out of 27
Snow as a potential feature	12 out of 27	25 out of 27

M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?' Explaining the predictions of any classifier," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, vol. 13-17-Augu, pp. 1135–1144.

Examples

Detecting defective insulators

- XAI on Power Grid Insulators:
  - Fault detection ➔ binary classifier
  - When faulty, apply XAI to show which part of the image led to the decision,
  - highlighting the defect region



Interpretability

Fault localization

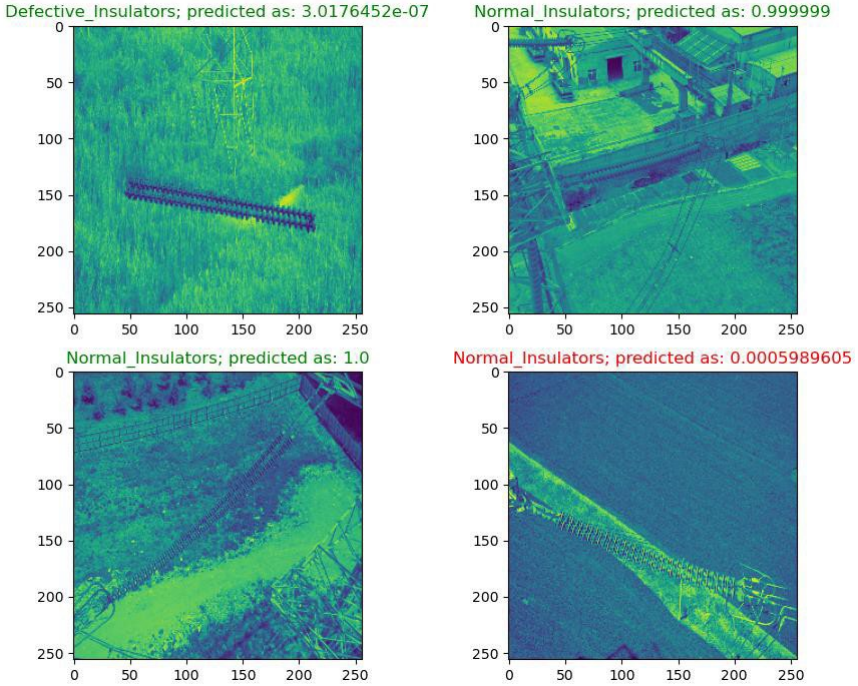
Evaluation of the prediction results

Test Accuracy = 96.4%

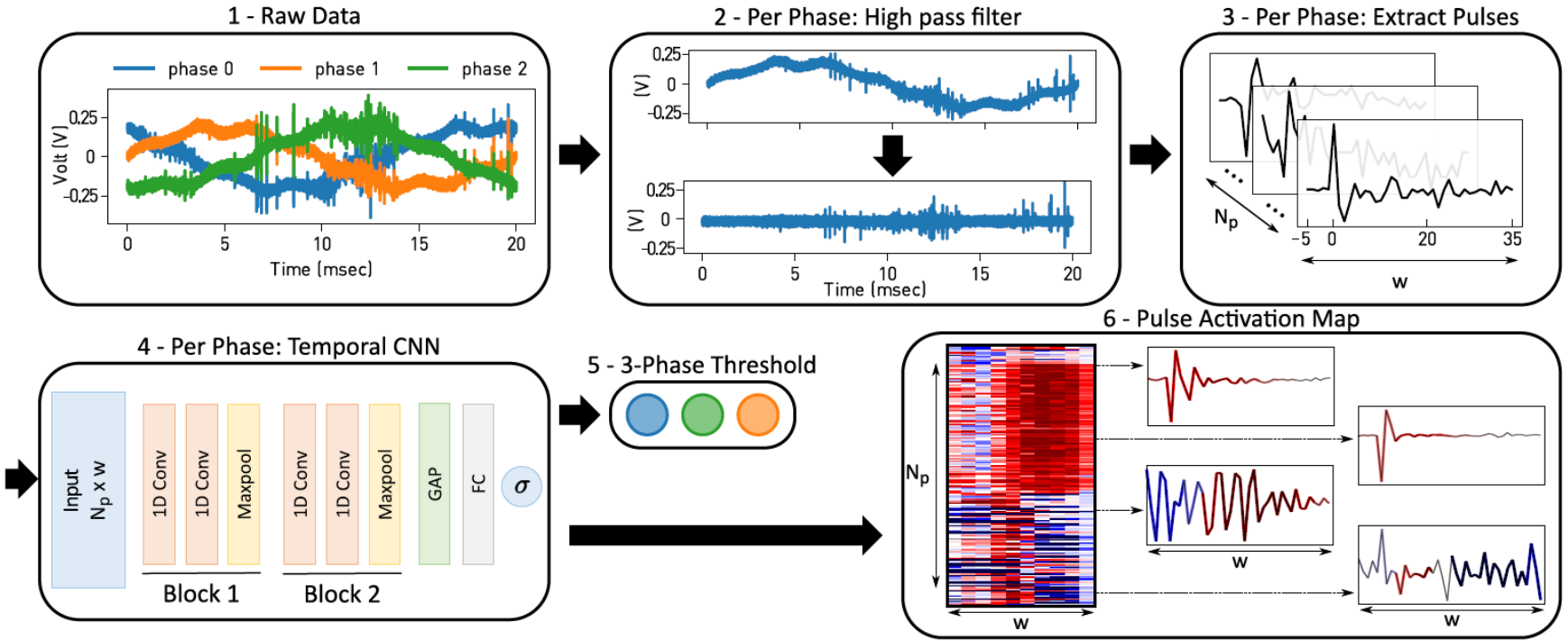
FPR = 3.6 %

FNR= 3.8 %

Test Samples Examples
(defective ➔ 0; Normal ➔1)

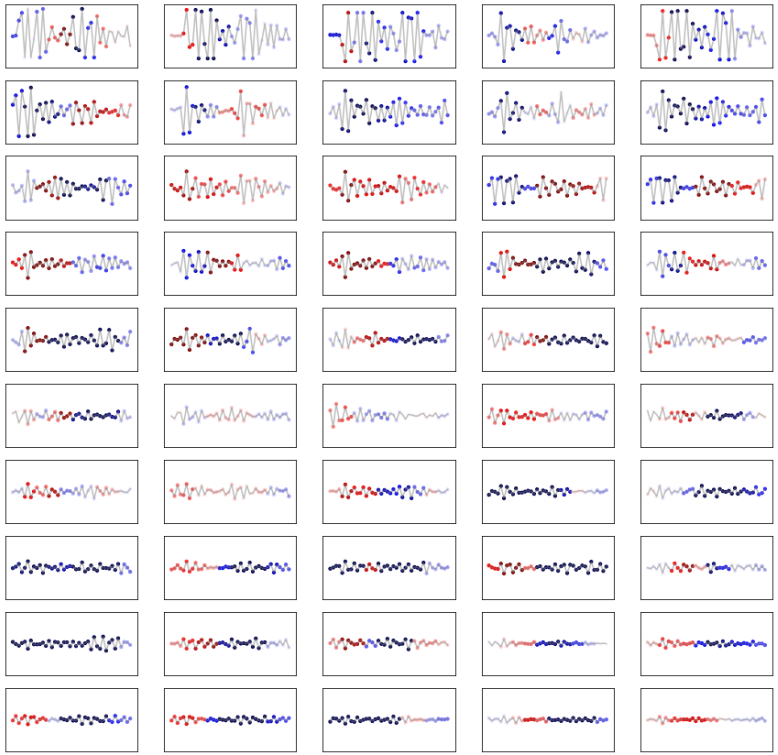


Interpretable Detection of Partial Discharge in Power Lines with Deep Learning → Framework



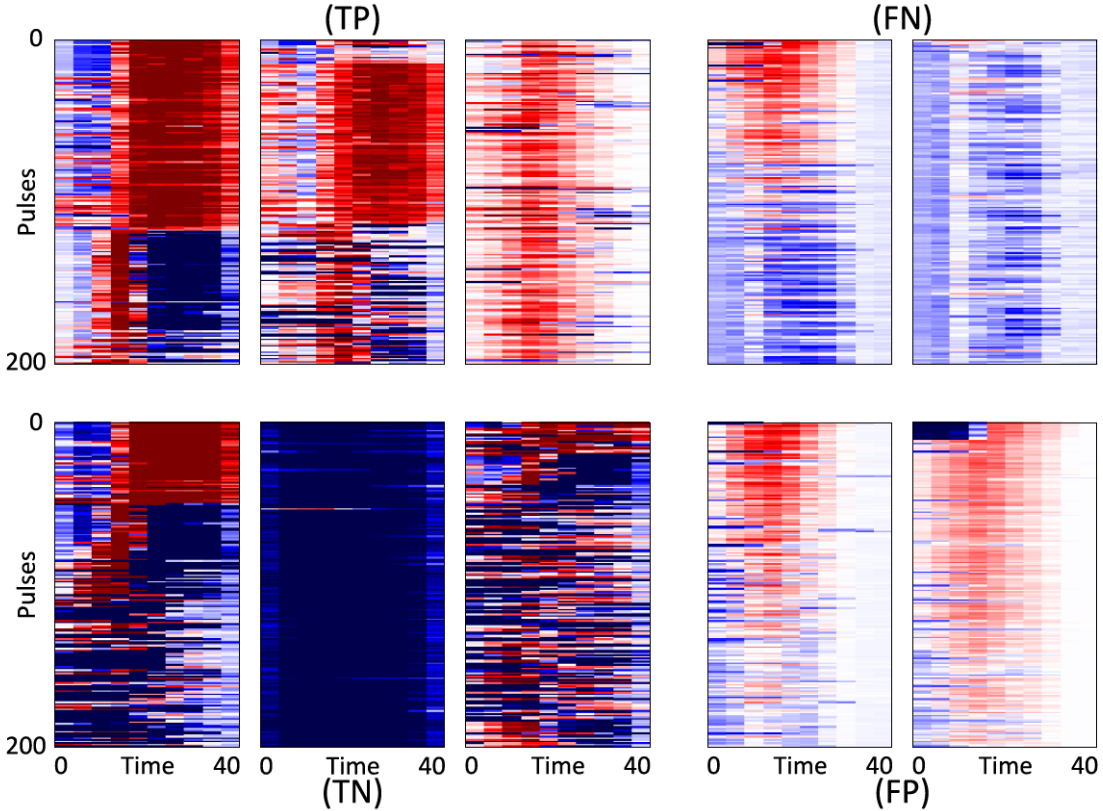
Michau, Gabriel, Chi-Ching Hsu, and Olga Fink. "Interpretable Detection of Partial Discharge in Power Lines with Deep Learning." Sensors 21.6 (2021): 2154.

Pulse activations



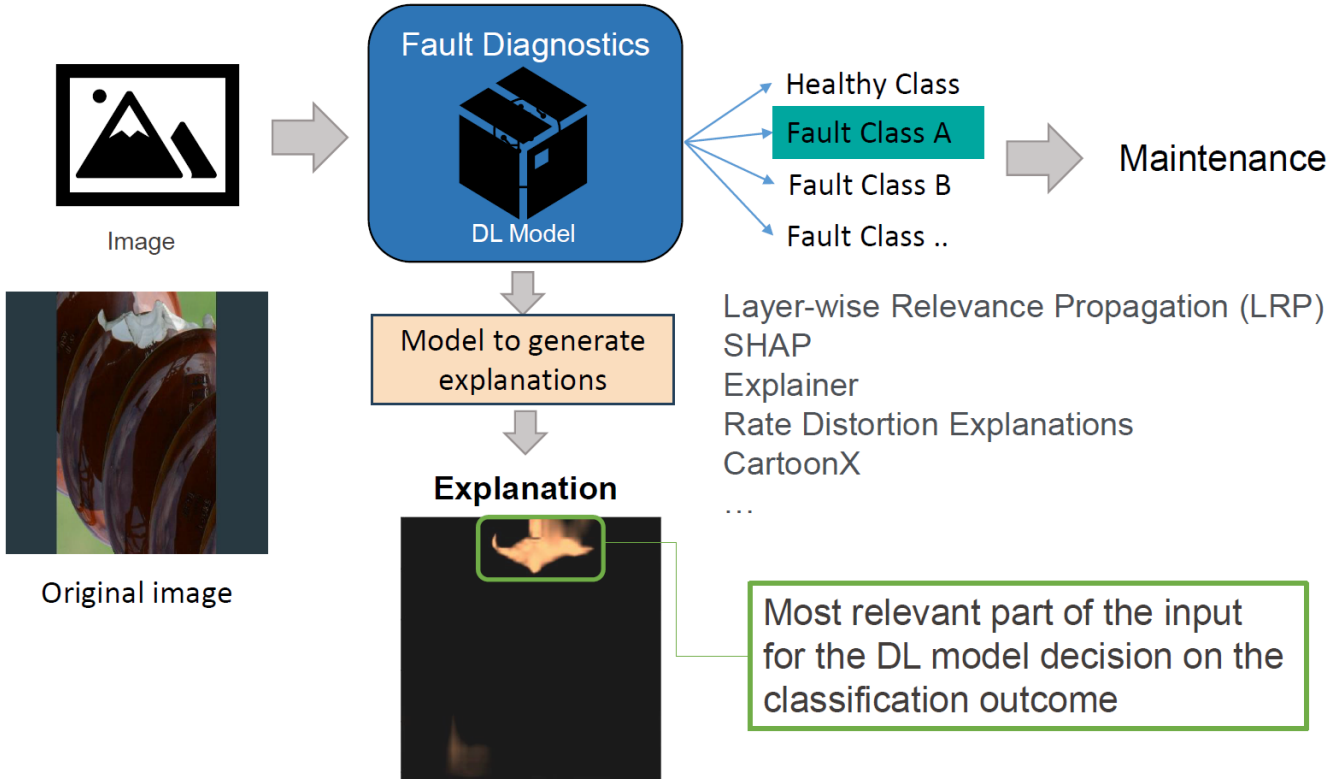
Michau, Gabriel, Chi-Ching Hsu, and Olga Fink. "Interpretable Detection of Partial Discharge in Power Lines with Deep Learning." Sensors 21.6 (2021): 2154.

Pulse Activation Maps (PAM)



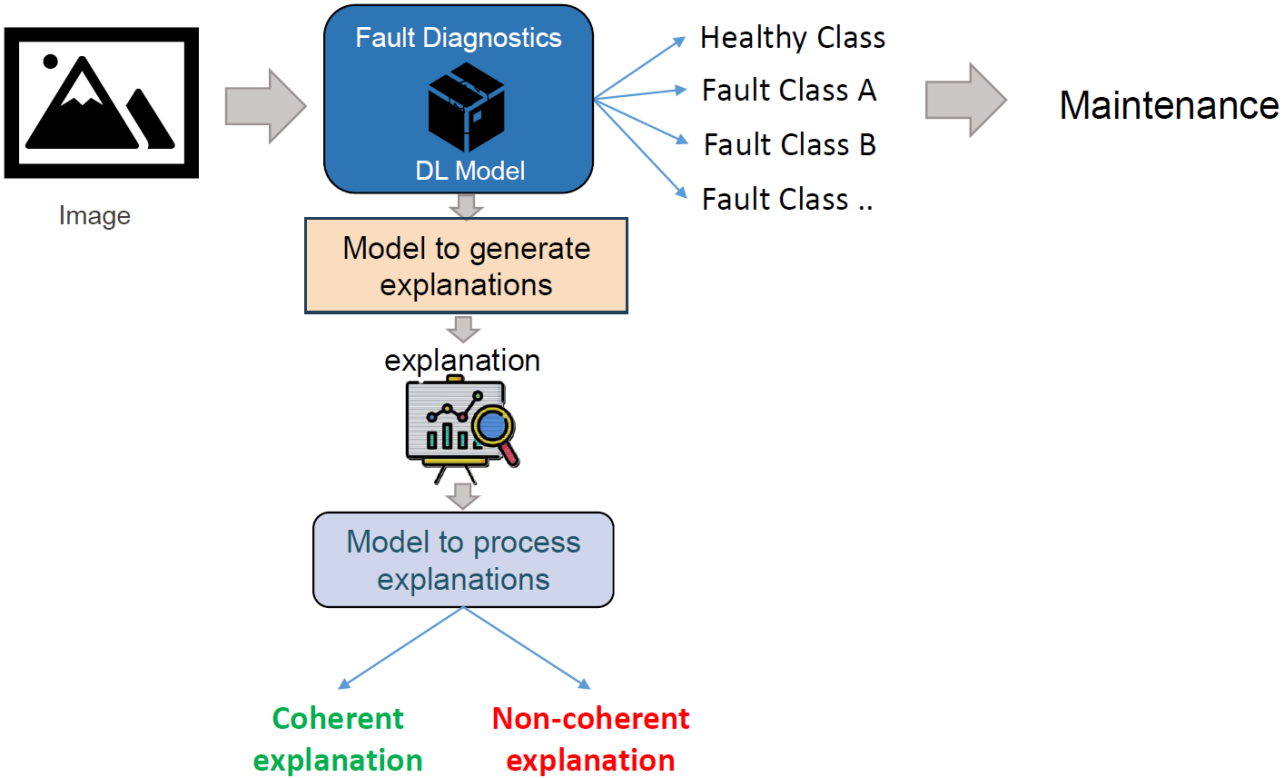
Michau, Gabriel, Chi-Ching Hsu, and Olga Fink. "Interpretable Detection of Partial Discharge in Power Lines with Deep Learning." Sensors 21.6 (2021): 2154.

Processing explanations



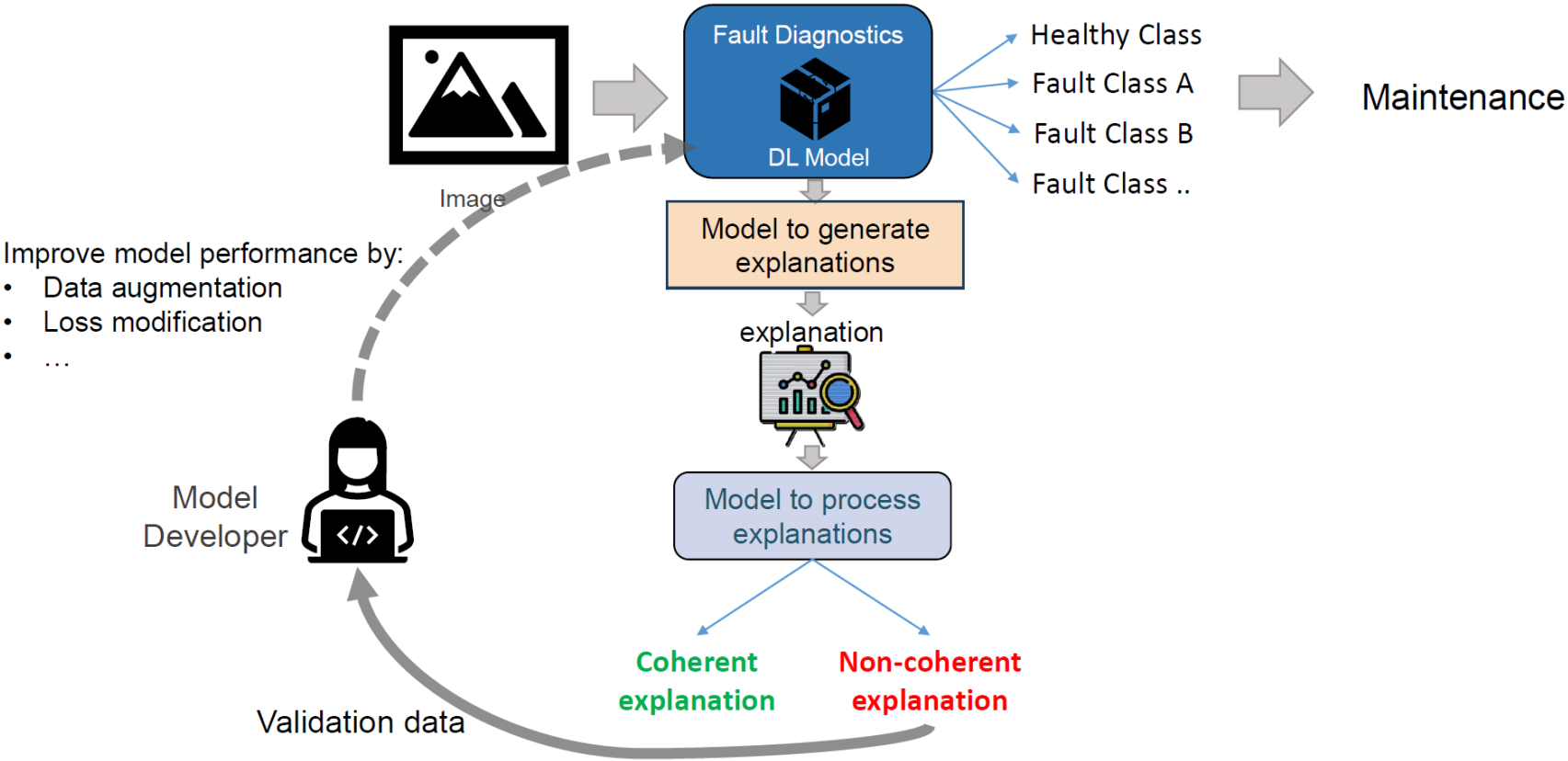
Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

How can explanations be used?



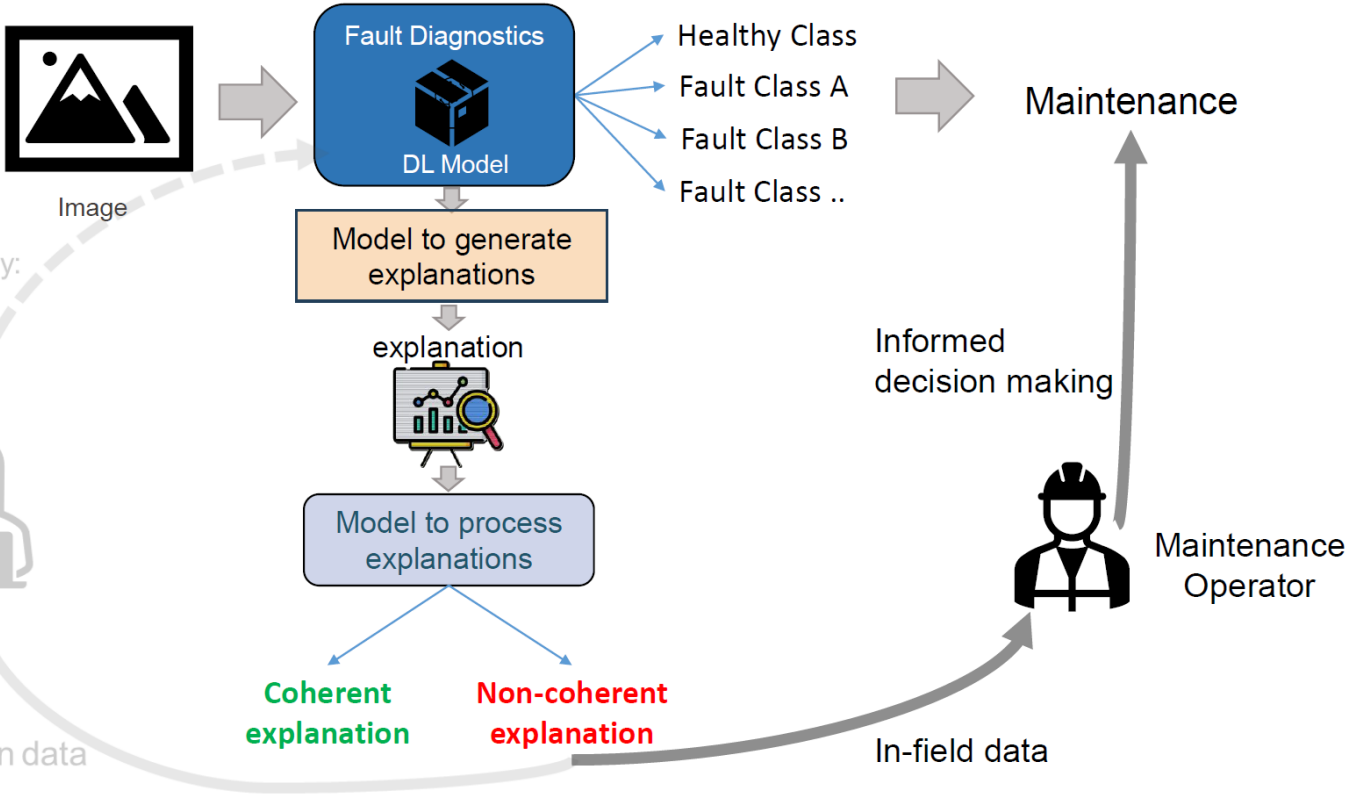
Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

Model improvement



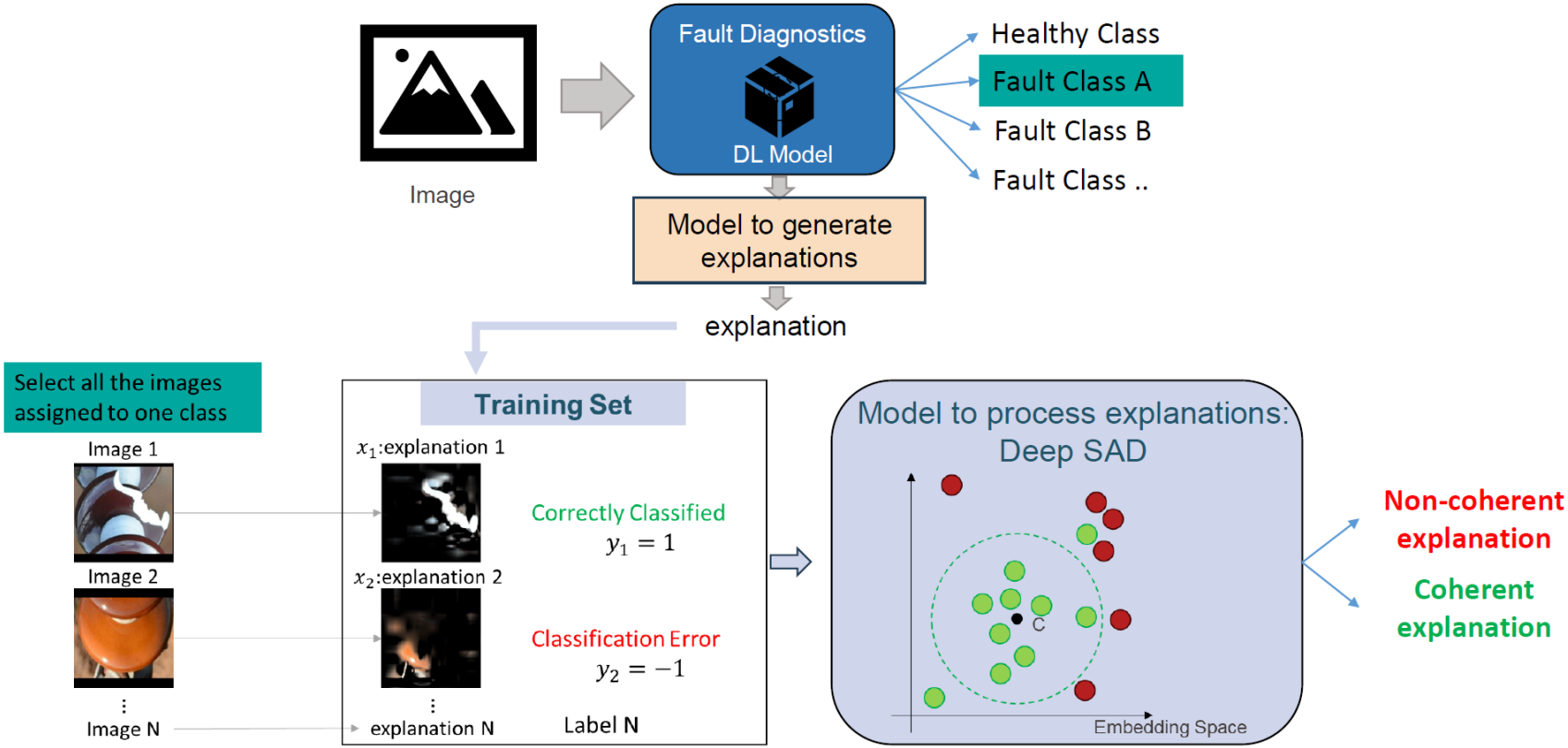
Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

Informed decision making



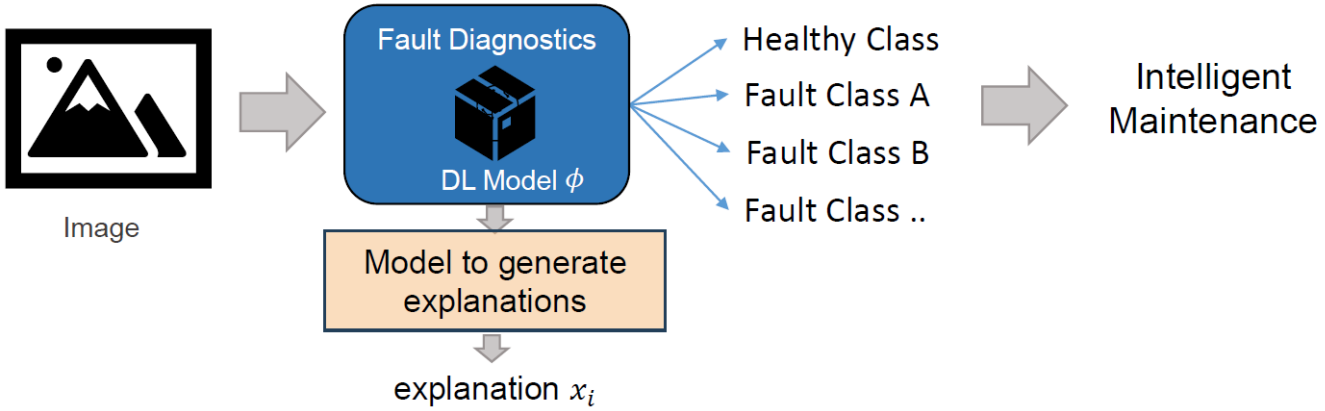
Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

Proposed method for automating the evaluation



Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

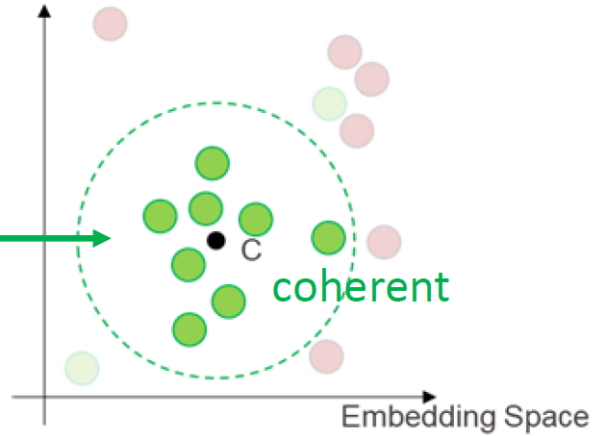
Proposed method: detecting unusual explanations



Embedding space definition: loss function

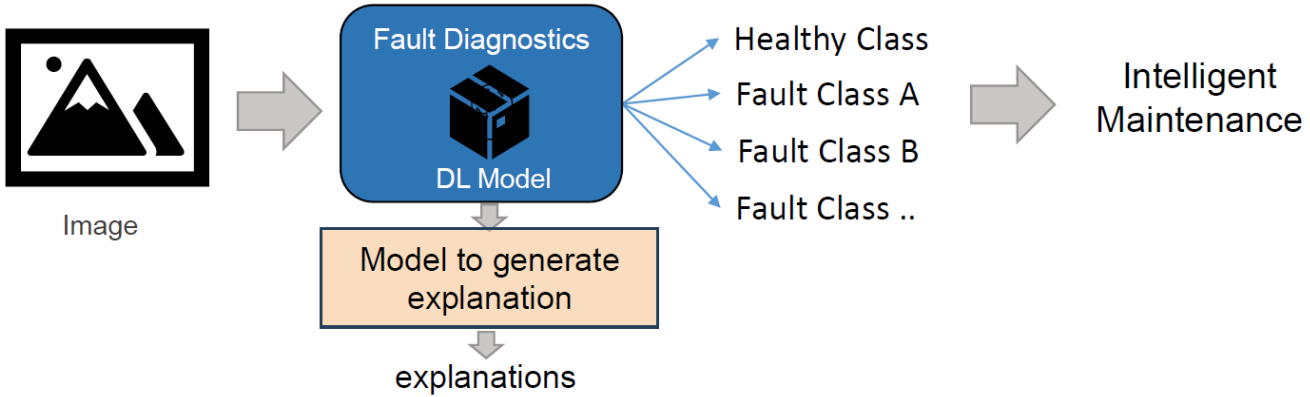
- Correct classifications: minimize the distance from the centre

$$\min_{\mathcal{W}} \frac{1}{n+m} \sum_{i=1}^{n} \|\phi(x_i; \mathcal{W}) - c\|^2$$



Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

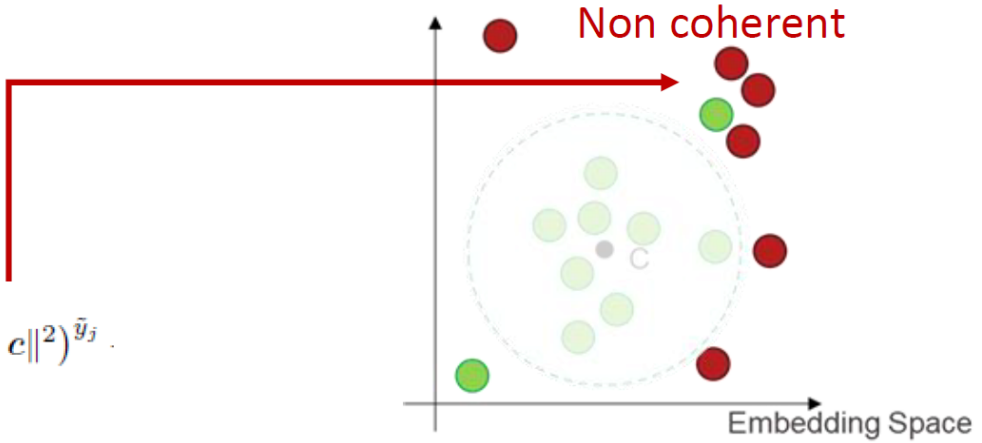
Proposed method: detecting unusual explanations



Embedding space definition: loss function

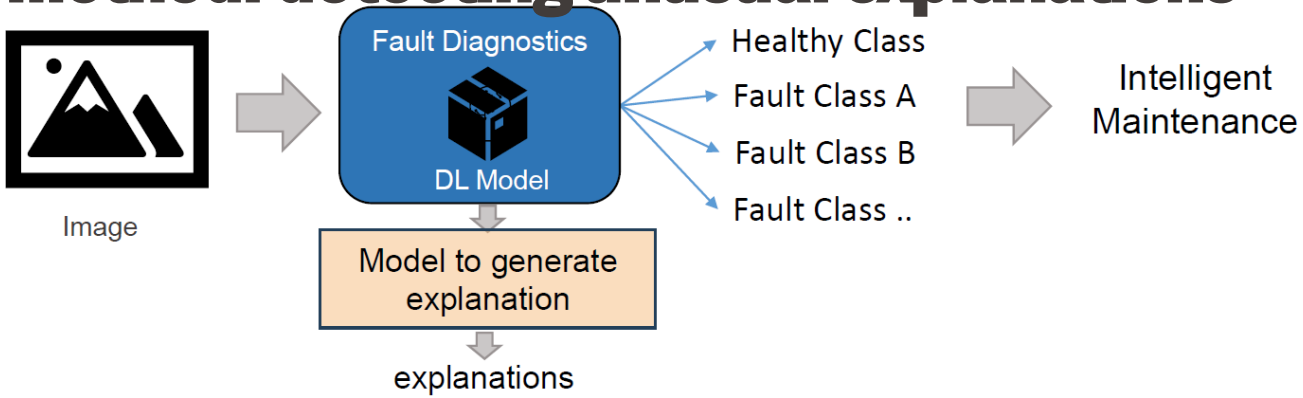
- Classifications errors: maximize the distance from the centre (hyperparameter η)

$$\min_{\mathcal{W}} \frac{1}{n+m}\sum_{i=1}^{n}\|\phi(x_i;\mathcal{W})-c\|^2+\frac{\eta}{n+m}\sum_{j=1}^{m}\left(\|\phi(\tilde{x}_j;\mathcal{W})-c\|^2\right)^{\tilde{y}_j}.$$



Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

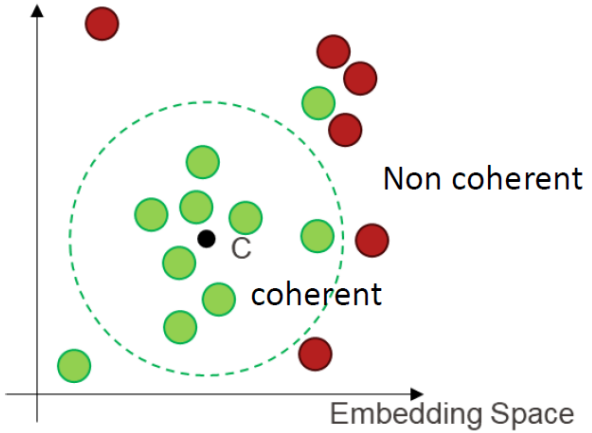
Proposed method: detecting unusual explanations



Embedding space definition: loss function

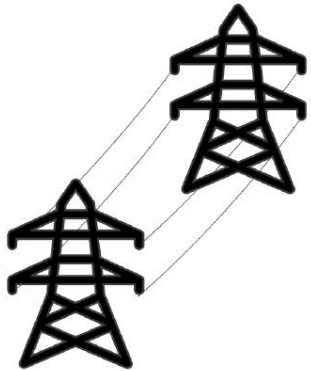
Regularization term to avoid overfitting (hyperparameter λ)

$$\min_{\mathcal{W}} \frac{1}{n+m}\sum_{i=1}^{n}\|\phi(\boldsymbol{x}_i;\mathcal{W})-\boldsymbol{c}\|^2+\frac{\eta}{n+m}\sum_{j=1}^{m}\left(\|\phi(\tilde{\boldsymbol{x}}_j;\mathcal{W})-\boldsymbol{c}\|^2\right)^{\tilde{y}_j}+\frac{\lambda}{2}\sum_{\ell=1}^{L}\|\boldsymbol{W}^{\ell}\|_F^2.$$



Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

Case study: infrastructure monitoring

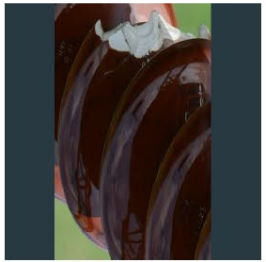


Power Grid

Critical Components:

Insulators

Drones

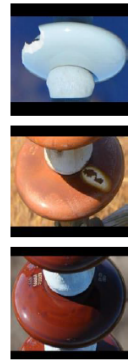


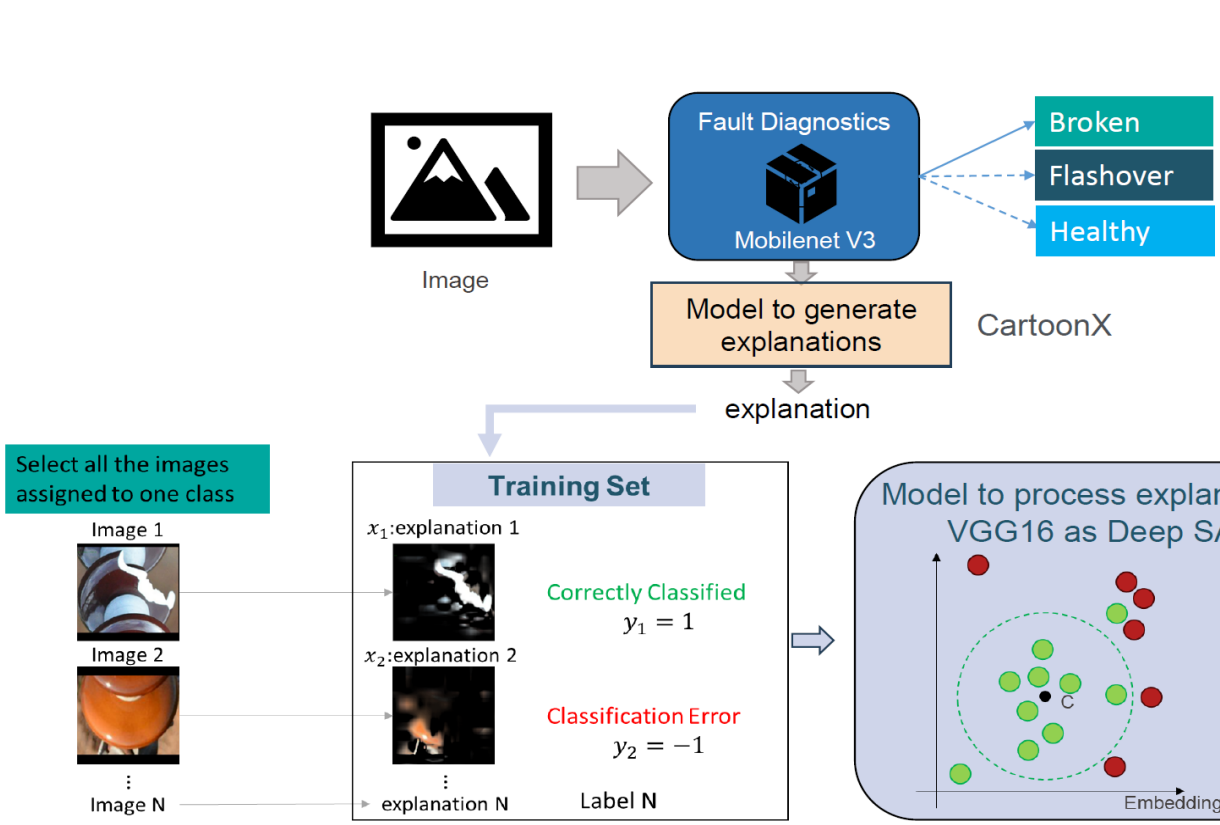
Shells' images

e.g. swiss power grid:
6700 km long → 12 000 pylons

Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

Application of the methodology





Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

Application of the methodology

Image

Fault Diagnostics

DL Model

Broken

Flashover

Healthy

Correct!

Model to generate explanations

explanation

Model developer is guided for model improvement

e.g. data augmentation

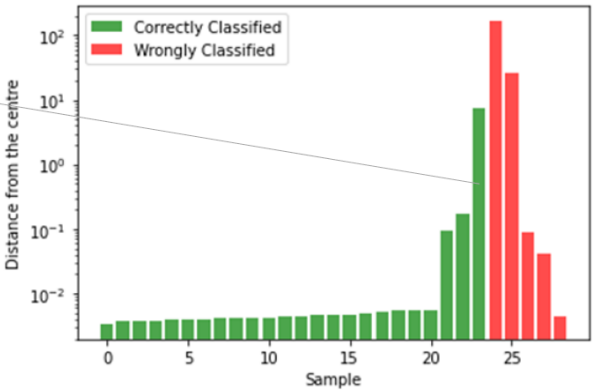
short-cut!

Non coherent

Deep SAD to process explanations

Embedding Space

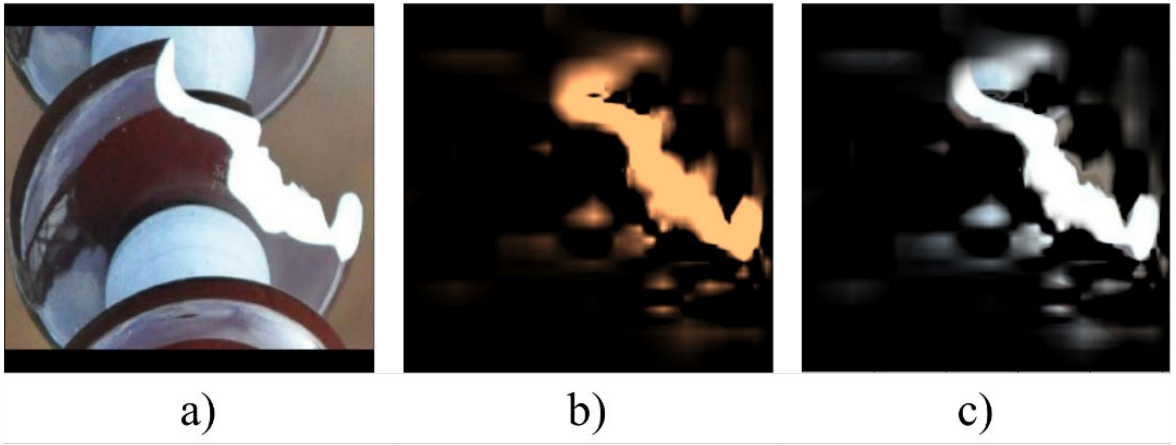
Validation set



Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

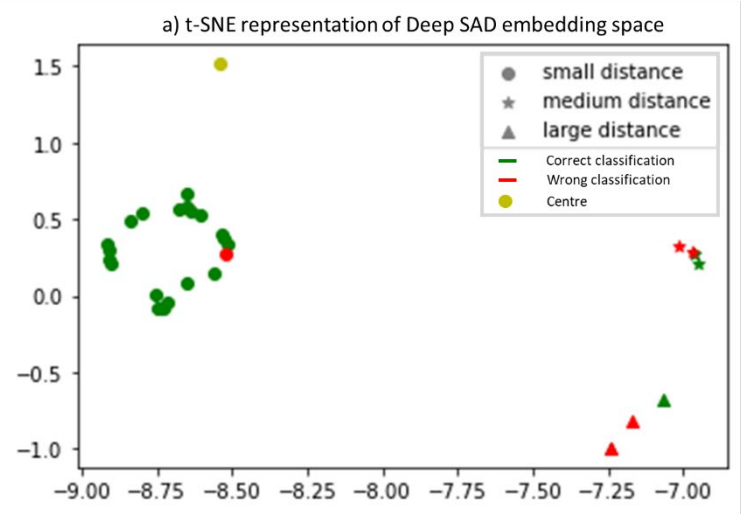
Examples

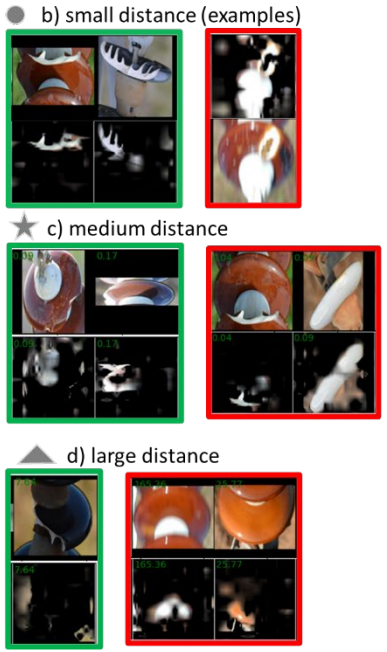
Example of one broken shell and its explanation



Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

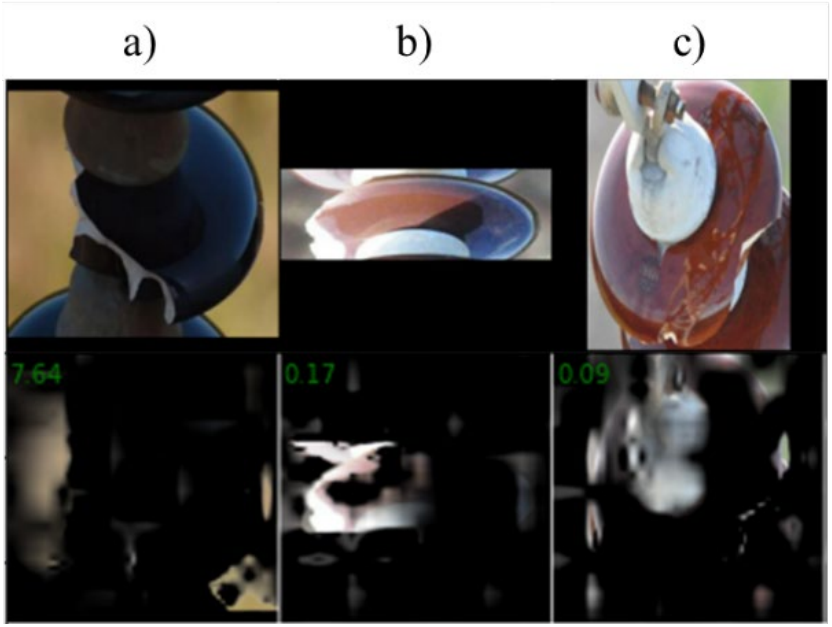
Coherent vs. Non-coherent explanations





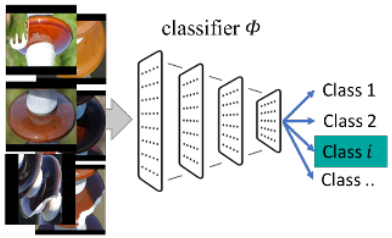
Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

Examples

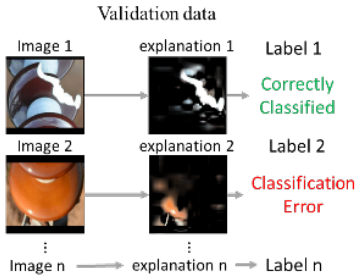


Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation

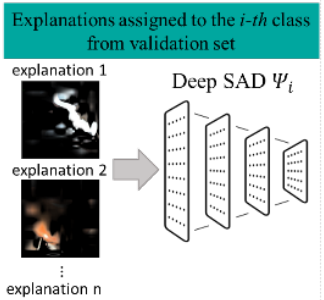
Summary



Step 1. Train the supervised classifier Φ

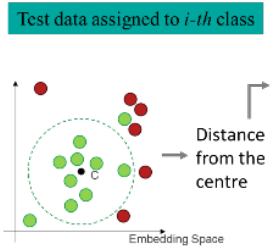


Step 2. Compute explanations and assign binary label for correct/incorrect classifications

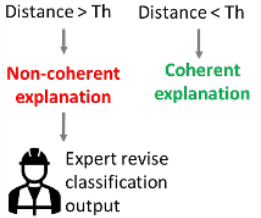


Step 3. Use labelled explanations to train Deep SAD model Ψ_i for the i-th class

For each i-th class



Step 4. Apply Ψ_i to test data explanations and map them in the embedding space



Step 5. Apply threshold to spot non-coherent explanations and ask expert to revise them

Floreale, G., P. Baraldi, E. Zio, O. Fink: Processing Explanations to Improve Performance and Enhance Trustworthiness of DL Models for Fault Diagnostics in Large Infrastructures, in preparation