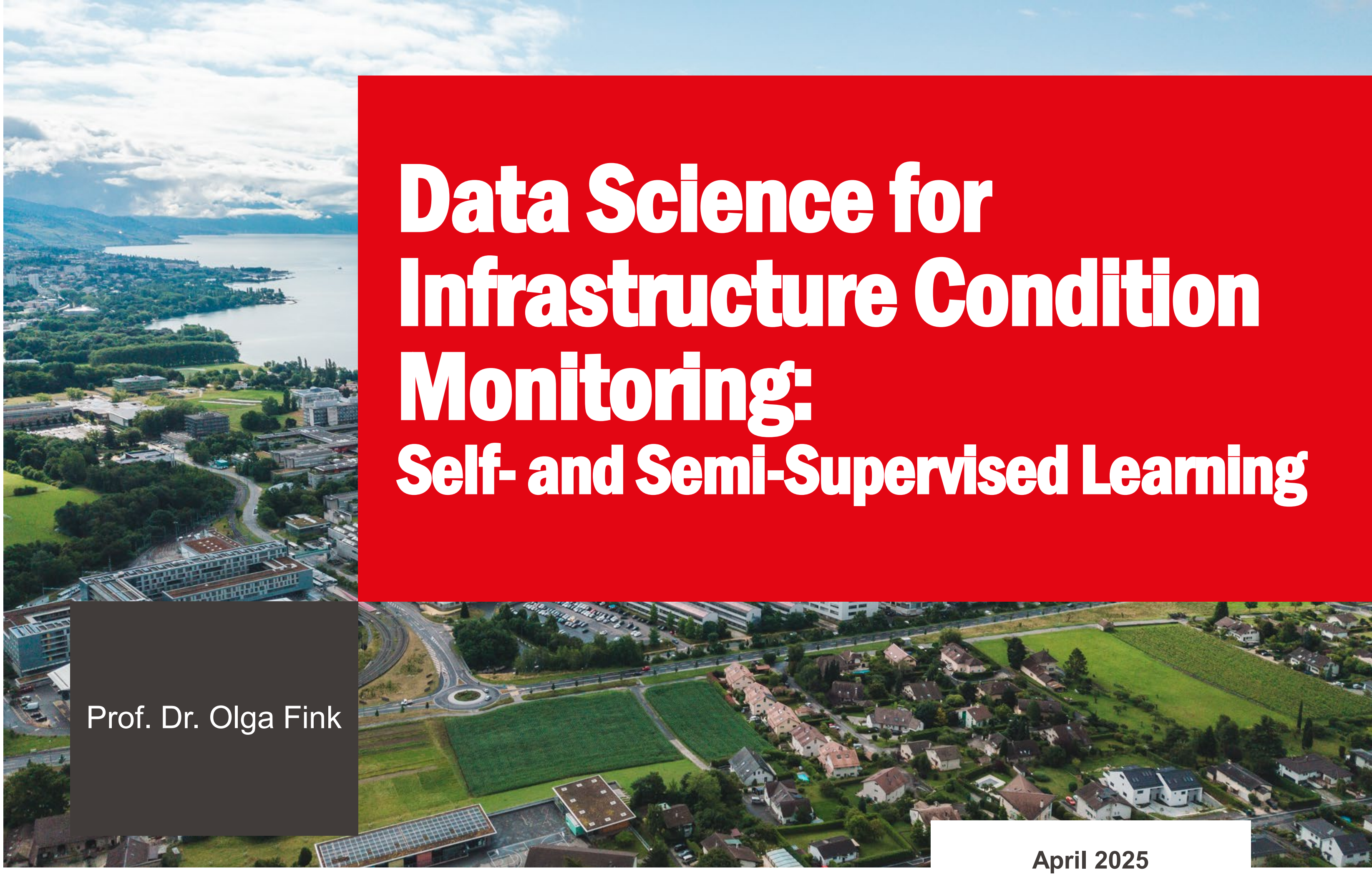


Data Science for Infrastructure Condition Monitoring:

Self- and Semi-Supervised Learning

Prof. Dr. Olga Fink

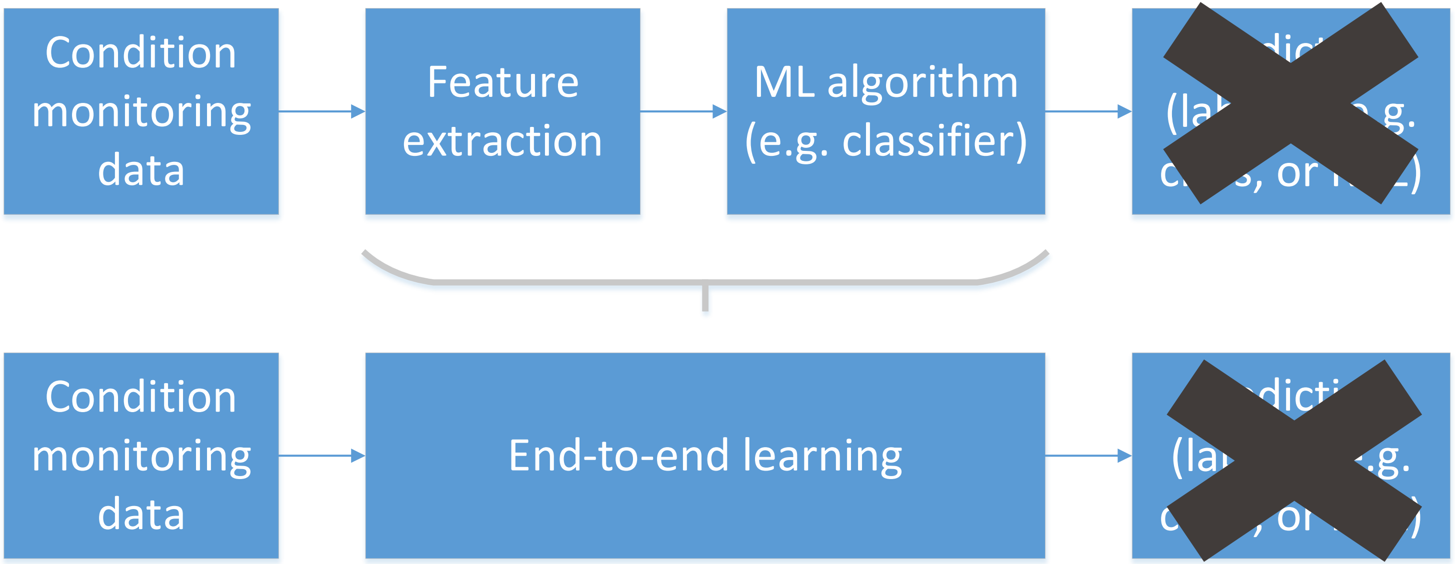
April 2025

École polytechnique fédérale de Lausanne

Unsupervised / Self-supervised learning

Olga Fink

Unsupervised learning



How much information is the machine given during learning?

- **"Pure" Reinforcement Learning (cherry)**
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning (icing)**
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Self-Supervised Learning (cake génoise)**
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**

Source: Y. LeCun

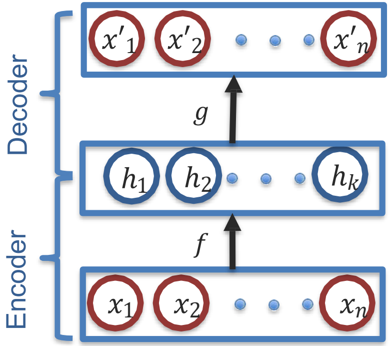
Unsupervised (also self-supervised, predictive) Learning

- We have access to $\{x_1, x_2, x_3, \cdots, x_N\}$ but not $\{y_1, y_2, y_3, \cdots, y_N\}$

- Why would we want to tackle such a task:
  - Extracting interesting information from data
  - Clustering
  - Discovering interesting trend
  - Data compression
  - Learn better representations

Autoencoders

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



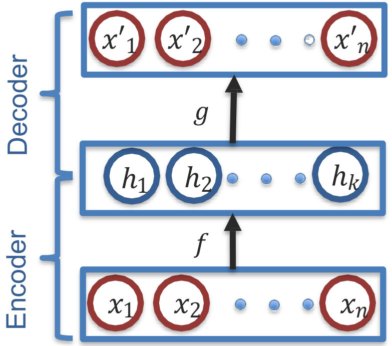
Trivial solution unless:
- Constrain number of units in Layer 2 (learn compressed representation), or
- Constrain Layer 2 to be **sparse**

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

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

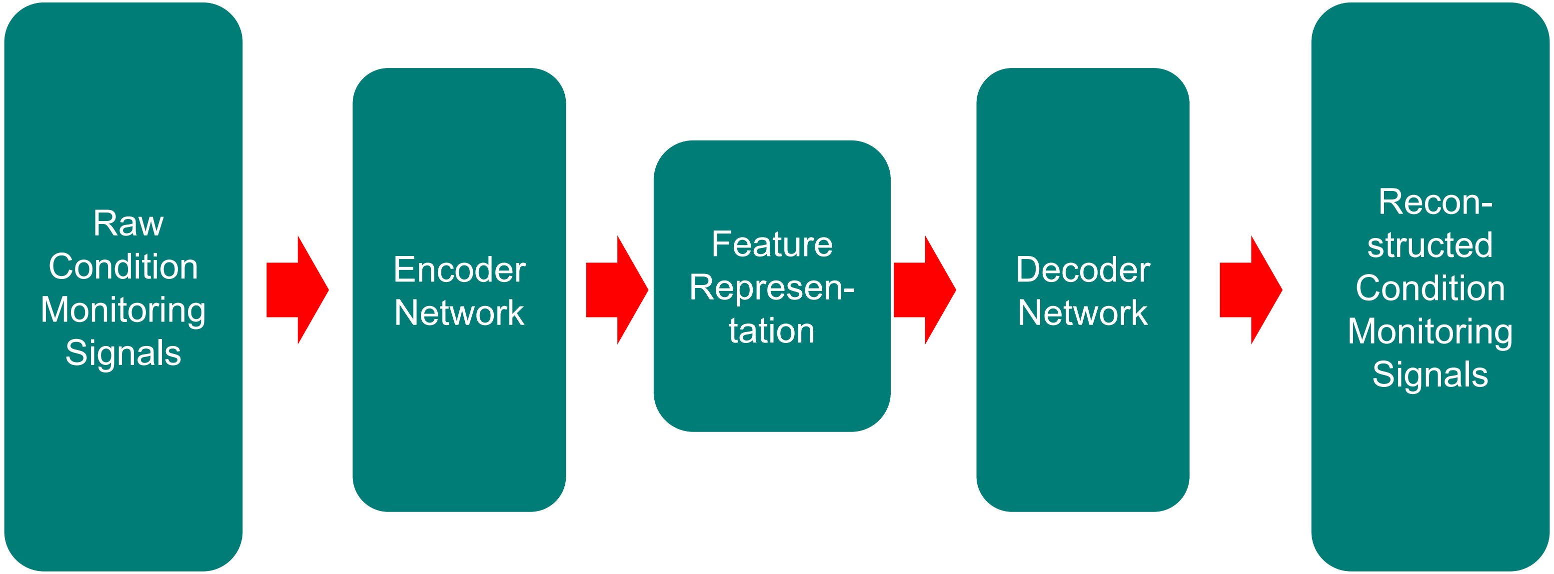
Define a loss function, $\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}})$, and minimize $\mathcal{L}$ with respect to the parameters of $f(\cdot)$ and $g(\cdot)$.

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

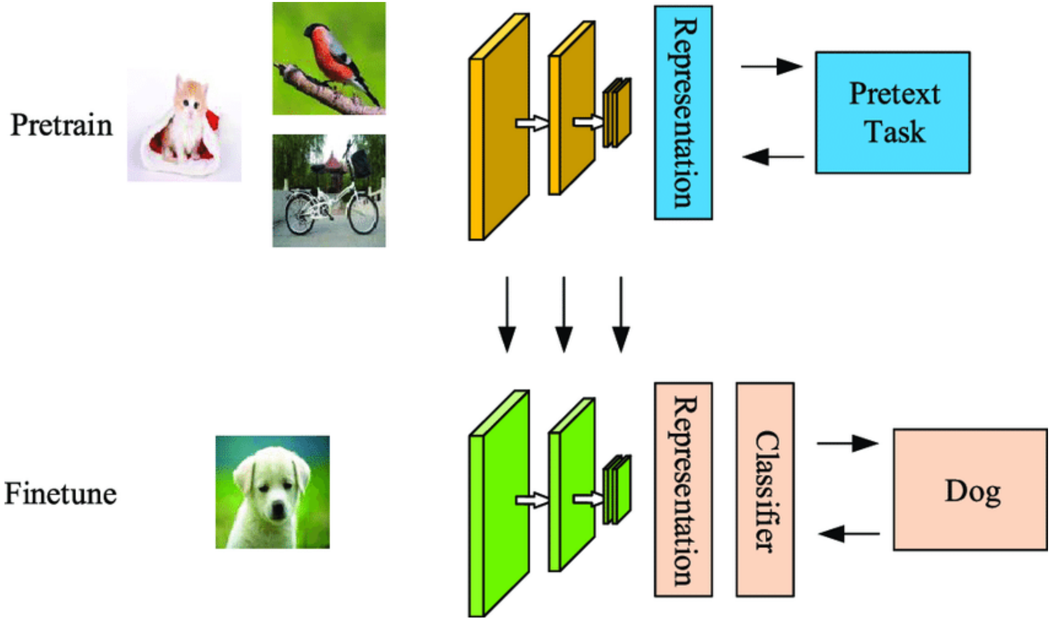
Learning features from raw condition monitoring data



Self-supervised learning

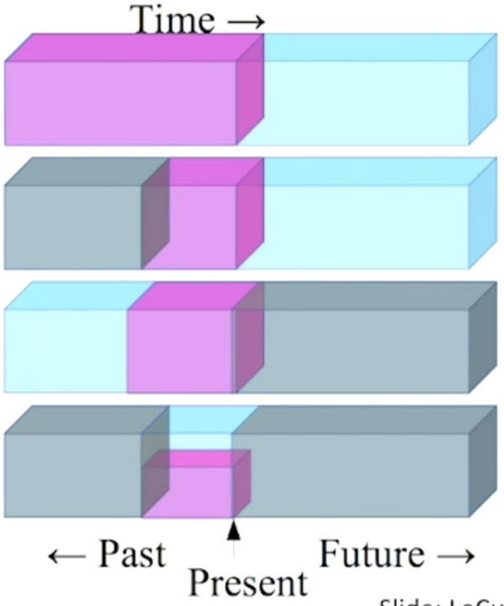
- Pretext task → important strategy for learning data representations under self-supervised mode
- Self-defined pseudo-labels
- Pseudo-labels automatically generated based on the attributes found in the unlabeled data

Pretext tasks



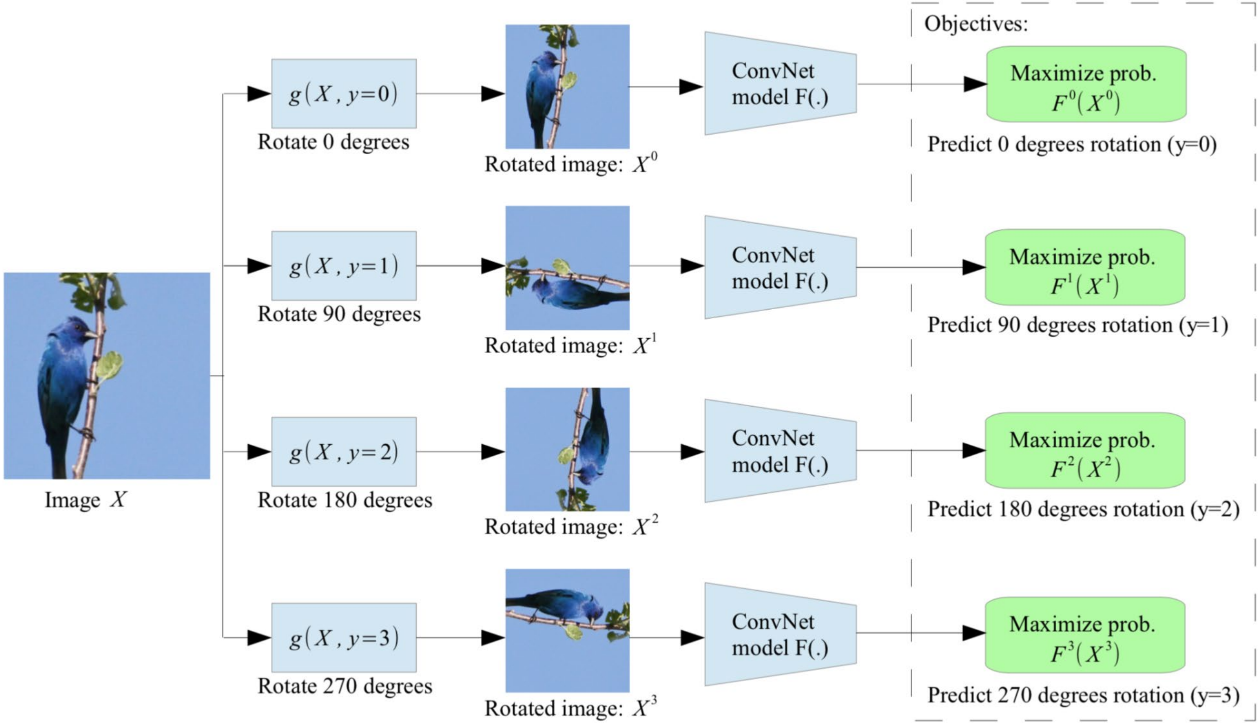
Self-supervised learning tasks (time series)

- Predict any part of the input from any other part.
- Predict the future from the past.
- Predict the future from the recent past.
- Predict the past from the present.
- Predict the top from the bottom.
- Predict the occluded from the visible
- Pretend there is a part of the input you don't know and predict that.



Slide: LeCun

Example: rotation



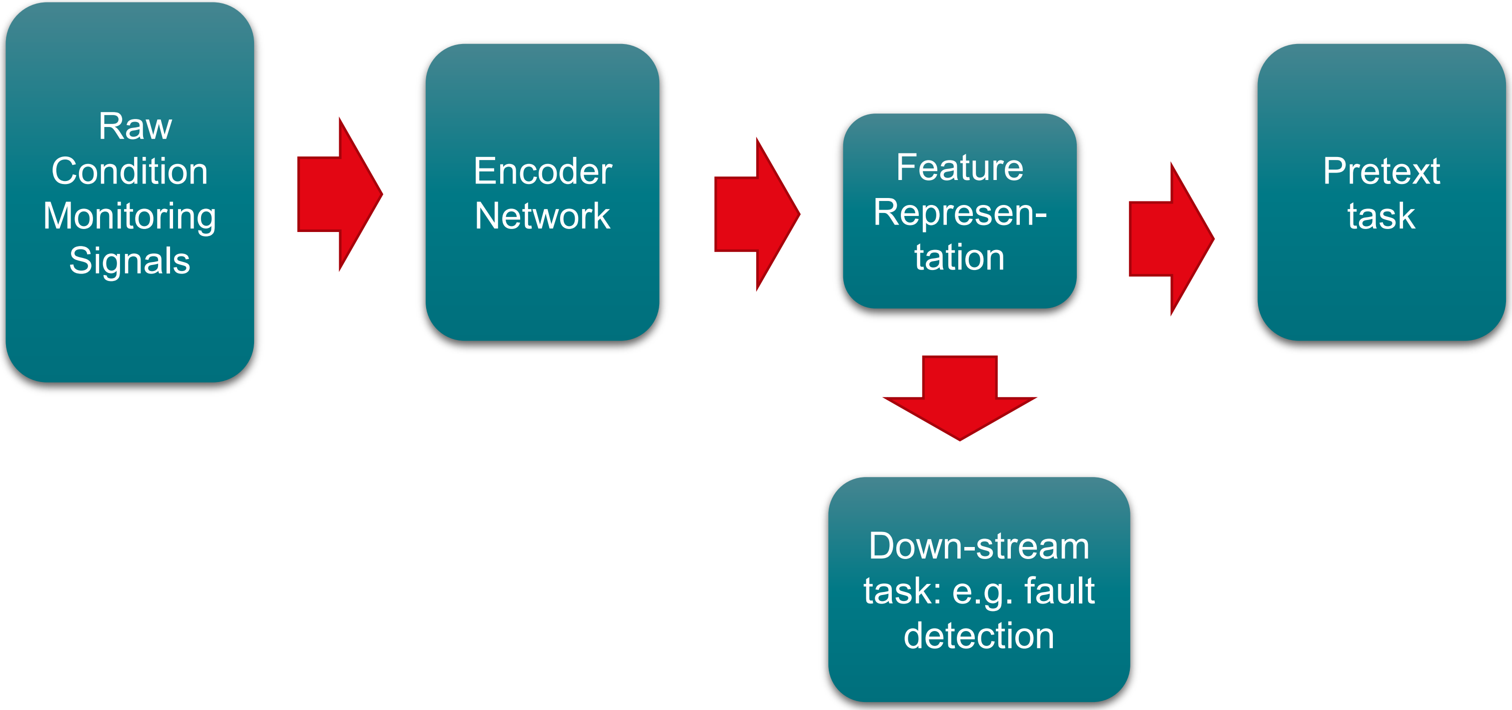
Gidaris et al. 2018

Important pretext tasks

- color transformations
- geometric transformations
- context-based tasks
- cross-modal-based tasks

EPFL

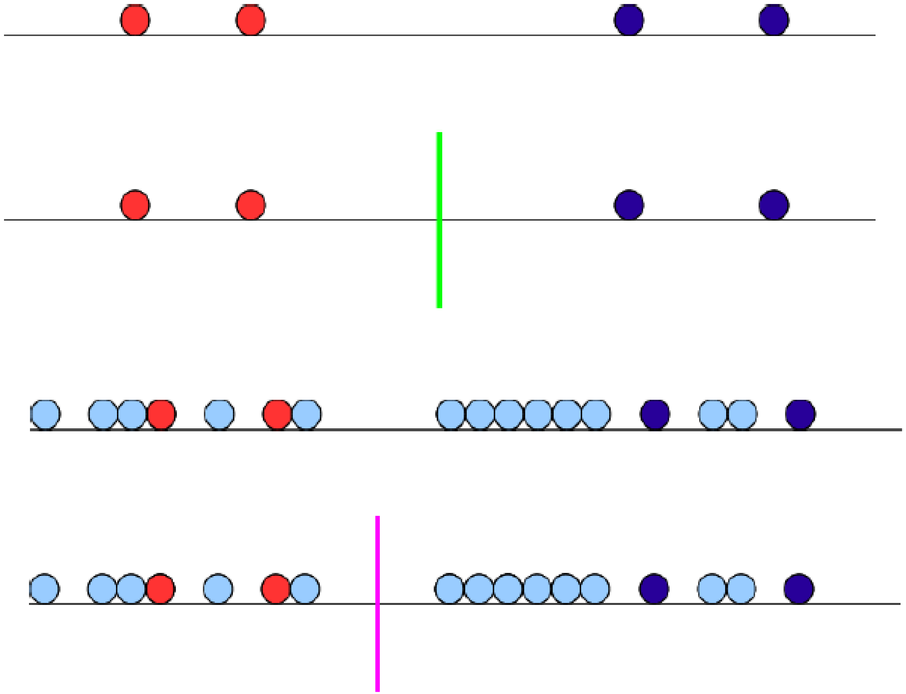
Basic idea of self-supervised learning



Semi-supervised learning

Olga Fink

Why/How Might Unlabeled Data Help?



Source: Piyush Rai 2011

Semi-supervised learning

- Semi-supervised learning leverages the available unlabeled data to improve the performance of the supervised learning task.
- Different concepts have been proposed for semi-supervised learning tasks
  - generative models
  - graph-based methods
  - transductive methods
- A further possibility to distinguish the different semi-supervised learning approaches is to differentiate between those based on
  - consistency regularization
  - entropy minimization
  - traditional regularization

Benefits of Semi-Supervised Learning:

- **Cost-effective:** Collecting labeled data can be time-consuming and expensive, especially for large datasets. Semi-supervised learning allows for the use of large amounts of unlabeled data, which can be collected relatively easily and inexpensively, to train the model.
- **Improved accuracy:** By incorporating both labeled and unlabeled data into the training process, those models can learn patterns and relationships within the data that may not be easily visible from the labeled data alone. This can lead to improved accuracy compared to models trained solely on labeled data.
- **Better generalization:** They tend to generalize better to new data compared to models trained solely on labeled data. This is because the models are able to learn more about the underlying structure of the data by incorporating both labeled and unlabeled data into the training process.
- **Flexibility**: It is a flexible approach that can be used in a variety of different applications, including image classification, natural language processing, and more.

Limitations of Semi-Supervised Learning

- **Labeled data limitations:** The effectiveness of semi-supervised learning models is dependent on the quality and quantity of the labeled data available. If the labeled data is limited or of poor quality, the model may not perform as well.
- **Model selection:** Selecting the right model for a semi-supervised learning problem can be challenging, as different models may perform better or worse depending on the specific problem and dataset.
- **Evaluation difficulty:** Evaluating the performance of that kind of model can be challenging, as the available labeled data may be limited and it can be difficult to determine the effectiveness of the model in making predictions for new data.

Assumptions for different methods

- **Continuity assumption** → objects near each other are likely to share the same group or label +data points that are part of the same cluster are more likely to share the same label
- **Cluster assumptions** →data points that are part of the same cluster are more likely to share the same label
- **Manifold assumptions** → high-dimensional data lie on a low-dimensional manifold → the learning algorithm should respect the manifold structure → learning should primarily happen on the manifold
- **Smoothness assumption:** if two points in a high-dimensional space are close to each other, then so should be their outputs

The Learning Problem

- Using both labeled and unlabeled data to build better learners, than using each one alone

input instance x, label y

learner $f : \mathcal{X} \mapsto \mathcal{Y}$

labeled data $(X_l, Y_l) = \{(x_{1:l}, y_{1:l})\}$

unlabeled data $X_u = \{x_{l+1:n}\}$, available during training

usually $l \ll n$

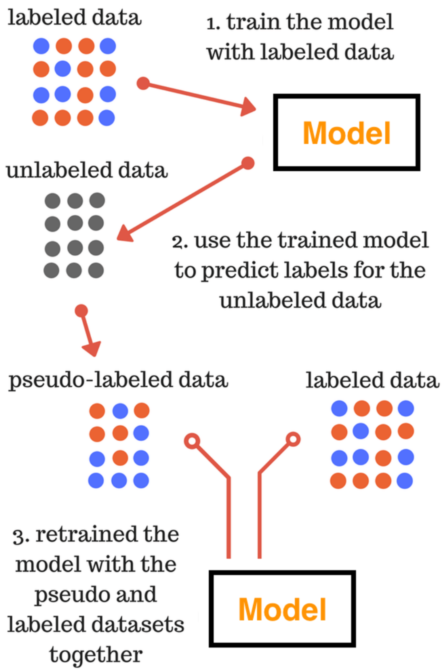
test data $X_{test} = \{x_{n+1:}\}$, not available during training

Source: Piyush Rai 2011

Pseudo-labeling

Olga Fink

Pseudo-labeling



Source: Potrimba, 2022

Self-Training

Olga Fink

Self-Training

Input: labeled data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, unlabeled data $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$.

1. Initially, let $L = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ and $U = \{\mathbf{x}_j\}_{j=l+1}^{l+u}$.
2. Repeat:
3. Train f from L using supervised learning.
4. Apply f to the unlabeled instances in U.
5. Remove a subset S from U; add $\{(\mathbf{x}, f(\mathbf{x})) | \mathbf{x} \in S\}$ to L.

Self-training is a *wrapper* method

- the choice of learner for f in step 3 is left completely open
- good for many real world tasks like natural language processing
- but mistake by f can reinforce itself

Source: Piyush Rai 2011

EPFL

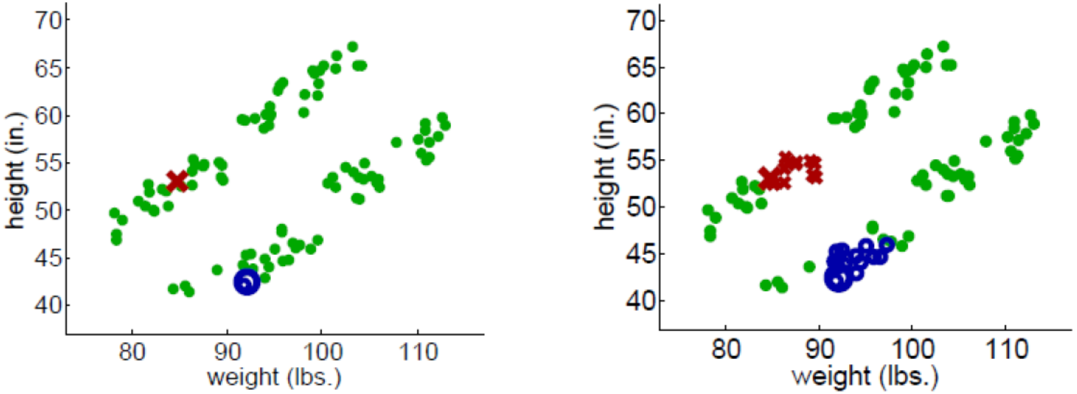
Learning algorithm (k-nearest neighbors)

Input: labeled data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, unlabeled data $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$, distance function $d()$.

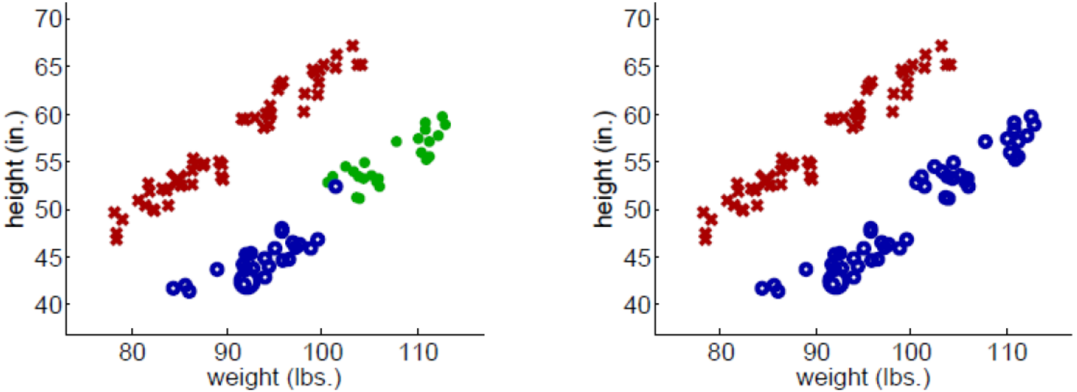
1. Initially, let $L = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ and $U = \{\mathbf{x}_j\}_{j=l+1}^{l+u}$.
2. Repeat until U is empty:
3. Select $\mathbf{x} = \mathrm{argmin}_{\mathbf{x} \in U} \min_{\mathbf{x}' \in L} d(\mathbf{x}, \mathbf{x}')$.
4. Set $f(\mathbf{x})$ to the label of $\mathbf{x}$'s nearest instance in L. Break ties randomly.
5. Remove $\mathbf{x}$ from U; add $(\mathbf{x}, f(\mathbf{x}))$ to L.

Source: Piyush Rai 2011

Self-Training: A Good Case

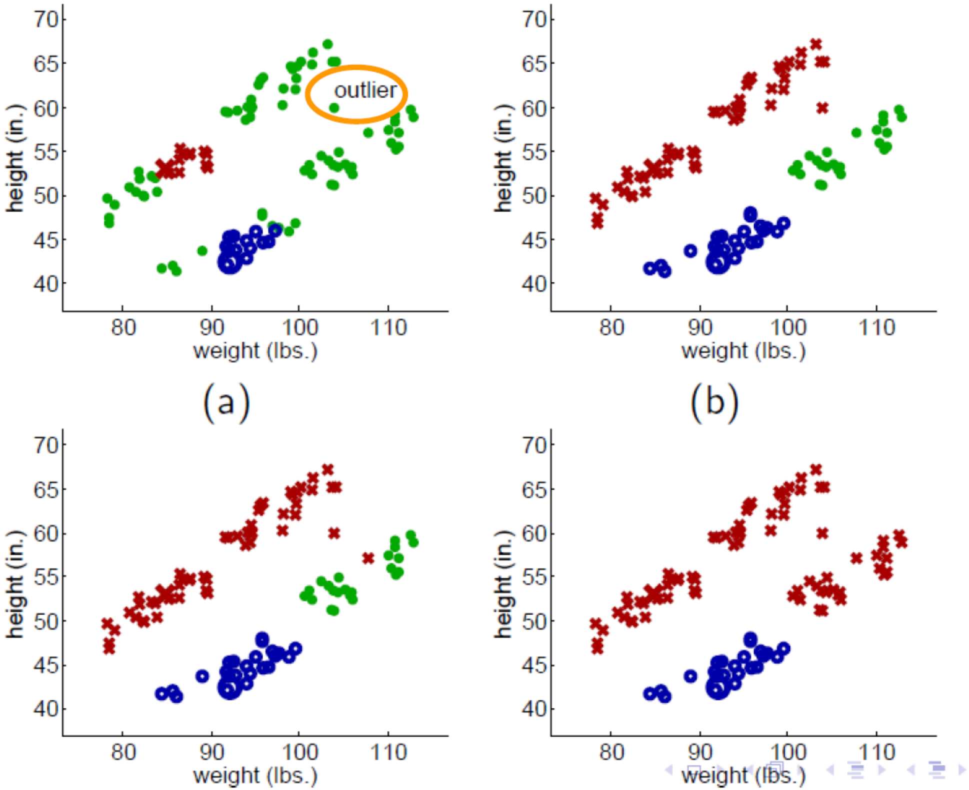


(a) Iteration 1 (b) Iteration 25



Source: Piyush Rai 2011

Self-Training: A bad Case



Source: Piyush Rai 2011

Co-Training

14.04.25

Olga Fink

Co-Training

- Given: Labeled data $\{\mathbf{x}_i, y_i\}_{i=1}^{L}$, unlabeled data $\{\mathbf{x}_j\}_{j=L+1}^{L+U}$
- Each example has 2 views: $\mathbf{x} = [\mathbf{x}^{(1)} \quad \mathbf{x}^{(2)}]$
- How do we get different views?
- Naturally available (different types of features for the same object)
  - Webpages: view 1 from page text; view 2 from social tags
  - Images: view 1 from pixel features; view 2 from Fourier coefficients
- ... or by splitting the original features into two groups

- Assumption: Given sufficient data, each view is good enough to learn from Co-training: Utilize both views to learn better with fewer labeled examples Idea: Each view teaching (training) the other view
- Technical Condition: Views should be conditionally independent
- Intuitively, we don't want redundancy between the views

Source: Piyush Rai 2011

Co-Training

- Given labeled data L and unlabeled data U
- Create two labeled datasets L_1 and L_2 from L using views 1 and 2
- Learn classifier $f^{(1)}$ using L_1 and classifier $f^{(2)}$ using L_2
- Apply $f^{(1)}$ and $f^{(2)}$ on unlabeled data pool U to predict labels
  - Predictions are made only using their own set (view) of features
- Add K most confident predictions $((x, f^{(1)}(x))$ of f_1 to L_2
- Add K most confident predictions $((x, f^{(2)}(x))$ of f_2 to L_1
- Note: Absolute margin could be used to measure confidence
- Remove these examples from the unlabeled pool
- Re-train $f^{(1)}$ using L_1, $f^{(1)}$ using L_2
- Like self-training but two classifiers teaching each other
- Finally, use a voting or averaging to make predictions on the test data

Source: Piyush Rai 2011

Multi-view Learning

Olga Fink

Multi-view Learning

- A general class of algorithms for semi-supervised learning
- Based on using multiple views (feature representations) of the data Co-training is a special type of multi-view learning algorithm
- General Idea: Train multiple classifiers, each using a different view
- Modus Operandi: Multiple classifiers must agree on the unlabeled data How might it help learn better?
  - Learning is essentially searching for the best classifier
  - By enforcing agreement among classifiers, we are reducing the search space
  - ⇒ hope is that the best classifier can be found easily with little labeled data
- For test data, these multiple classifiers can be combined
- E.g., voting, consensus, etc.
- Backed by a number of theoretical results

Source: Piyush Rai 2011

Cluster-and-Label Approach

Olga Fink

Cluster-and-Label Approach

Input: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l), \mathbf{x}_{l+1}, \ldots, \mathbf{x}_{l+u}$,
a clustering algorithm $\mathcal{A}$, a supervised learning algorithm $\mathcal{L}$

1. Cluster $\mathbf{x}_1, \ldots, \mathbf{x}_{l+u}$ using $\mathcal{A}$.
2. For each cluster, let S be the labeled instances in it:
3. Learn a supervised predictor from S: $f_S = \mathcal{L}(S)$.
4. Apply f_S to all unlabeled instances in this cluster.

Output: labels on unlabeled data $y_{l+1}, \ldots, y_{l+u}$.

- Finally train a supervised learner on the entire labeled data
- Assumption: Clusters coincide with decision boundaries
  - Poor results if this assumption is wrong

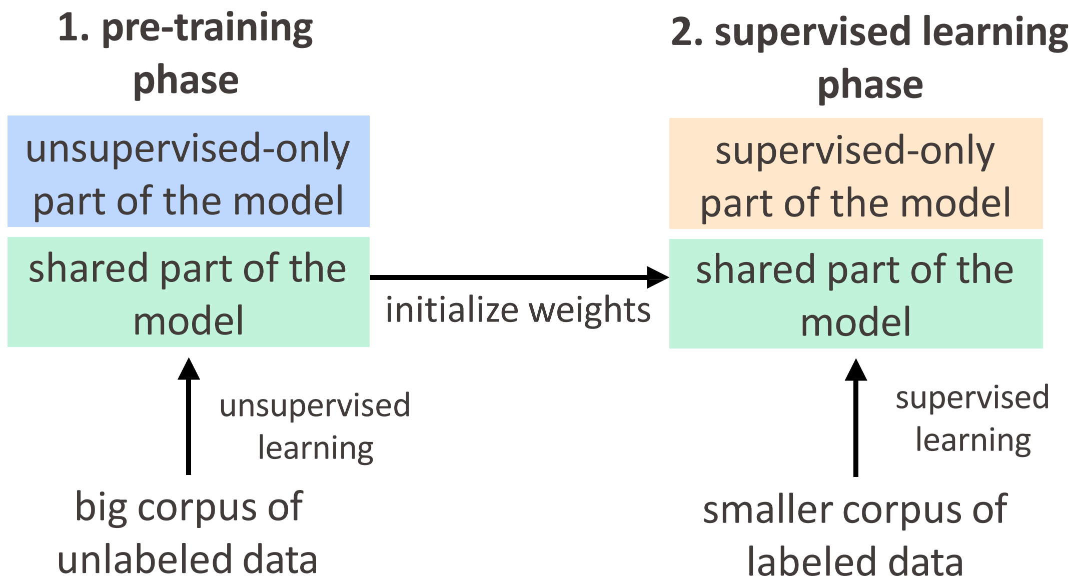
Source: Piyush Rai 2011

Pre-training

Olga Fink

Pre-training

- First train an unsupervised model on unlabeled data
- Then incorporate the model's learned weights into a supervised model and train it on the labeled data
  - Optional: continue fine-tuning the unsupervised weights.



Source: Clark.2019

Why does pre-training work?

- ”Smart” initialization for the model
- More meaningful representations in the model

Supervised learning: have to learn everything from “raw” input

supervised NN

↑

input

Pre-training: supervised part gets more useful representations as input

supervised NN

↑

pre-trained NN

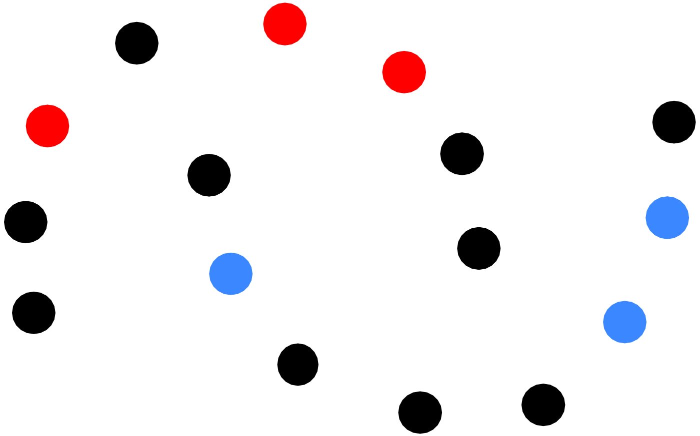
↑

input

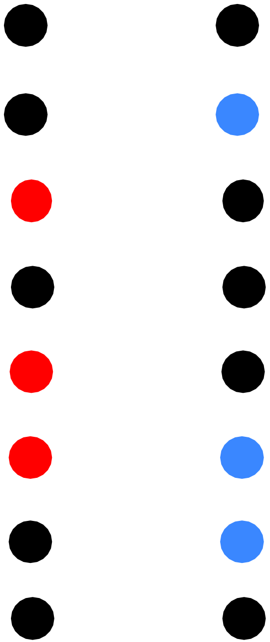
Source: Clark.2019

Why does pre-training work?

original representation space

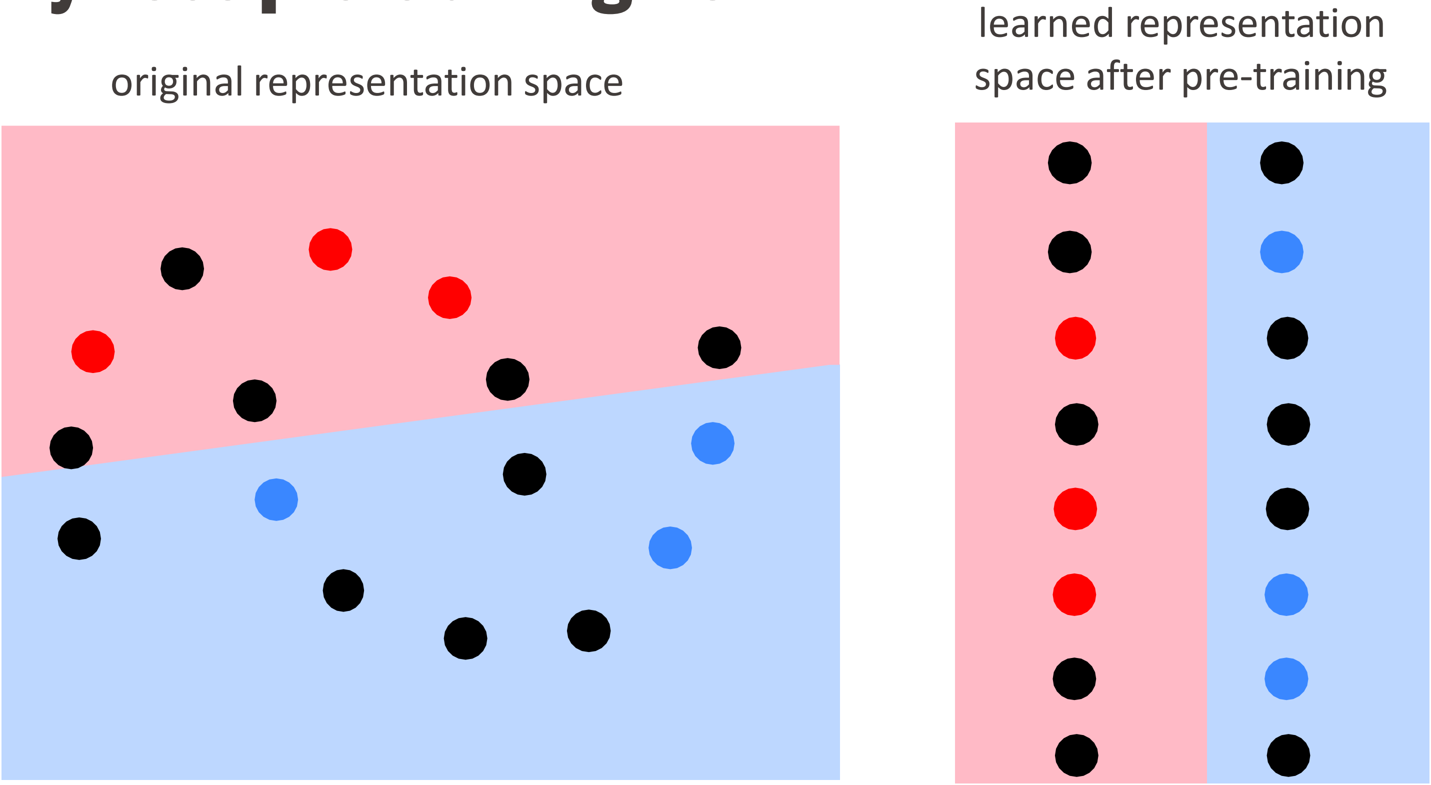


learned representation space after pre-training



Source: Clark.2019

Why does pre-training work?



Supervised part of the model has a much easier job after pre-training

Source: Clark.2019

Olga Fink

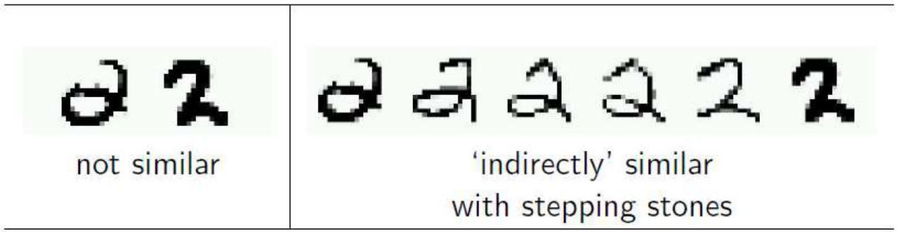
Graph Based Semi-supervised Learning

Olga Fink

Graph Based Semi-supervised Learning

- Graph based approaches exploit the property of label smoothness
- Idea: Represent each example (labeled/unlabeled) as vertices of some graph
- Idea: The labels should vary smoothly along the graph
- ⇒ Nearby vertices should have similar labels
- This idea is called Graph-based Regularization

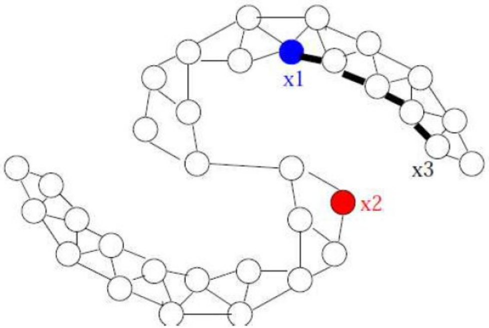
Handwritten digits recognition with pixel-wise Euclidean distance



Source: Piyush Rai 2011

Graph Regularization

- Nodes: $X_l \cup X_u$
- Edges: similarity weights computed from features, e.g.,
  - k-nearest-neighbor graph, unweighted (0, 1 weights)
  - fully connected graph, weight decays with distance $w = \exp\left(-\|x_i - x_j\|^2/\sigma^2\right)$
  - ϵ-radius graph
- Assumption Instances connected by heavy edge tend to have the same label.



Source: Piyush Rai 2011

Graph Regularization

- Assume the predictions on the entire data L∪U to be defined by function *f*
- Graph regularization assumes that the function *f* is smooth
- ⇒ Similar examples *i* and *j* should have similar predictions f_i and f_j
- Graph regularization optimizes the following objective:

$$\min_f \sum_{i \in \mathcal{L}} (y_i - f_i)^2 + \lambda \sum_{i,j \in \mathcal{L},\mathcal{U}} w_{ij}(f_i - f_j)^2$$

- First part is minimizing the loss on labeled data, second part ensures smoothness of labels of labeled and unlabeled data
- ⇒ Minimization makes f_i and f_j to be very similar if w_{ij} is large
- λ is a trade-off parameter
- Several variants and ways to solve the above problem

Source: Piyush Rai 2011

Consistency regularization

Olga Fink

Consistency Regularization

- Add noise to the student's inputs

$$J(\theta) = CE(p(y|x_j, \theta), p(y|x_j + \eta, \theta))$$

Soft target

Model learns to produce target even when noise is added to its input

- Where η is a vector with a random direction and a small magnitude ϵ

Source: Clark.2019

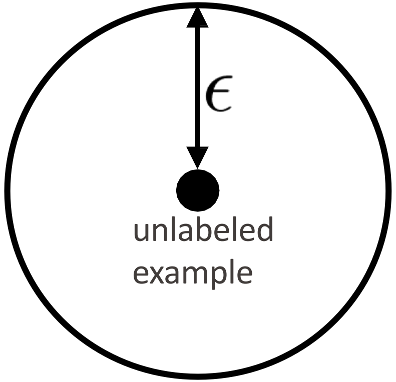
Consistency Regularization

- Add noise to the student's inputs

$$J(\theta) = CE(p(y|x_j, \theta), p(y|x_j + \eta, \theta))$$

  - Where η is a vector with a random direction and a small magnitude ϵ

- Train the model so a bit of noise doesn't mess up its predictions
- Equivalently, the model must give consistent predictions to nearby data points

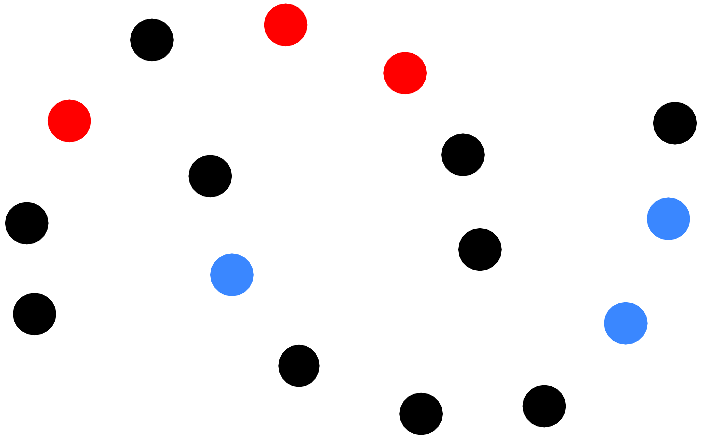


The model is trained to give the same prediction for any point in the circle

"distributional smoothing"

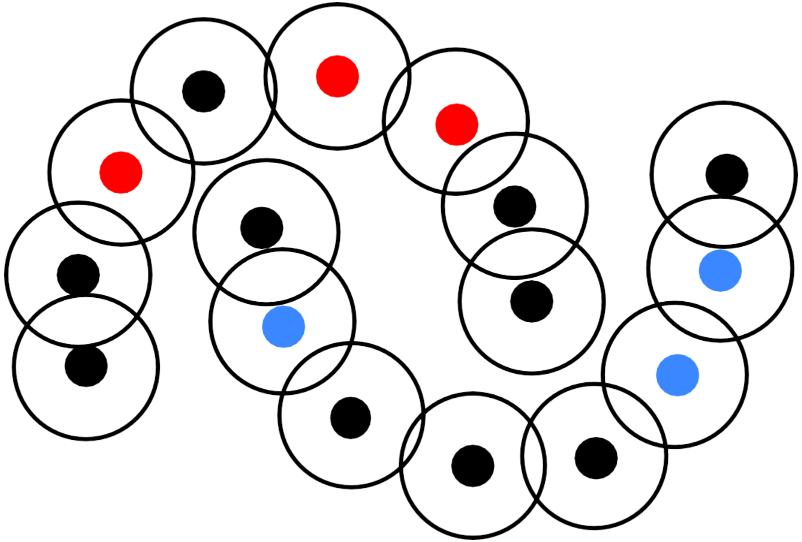
Source: Clark.2019

Consistency Regularization: Example



Source: Clark.2019

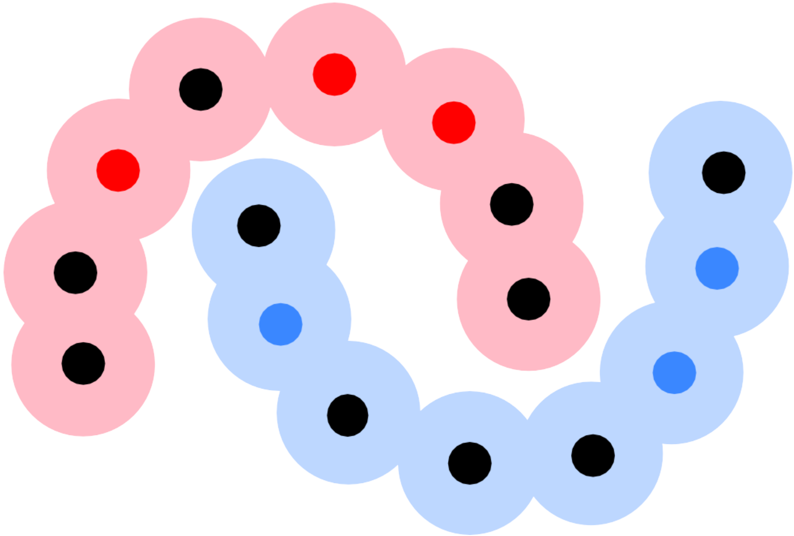
Consistency Regularization: Example



Model should produce the same predictions everywhere in the circles -> overlapping circles should have the same prediction

Source: Clark.2019

Consistency Regularization: Example



Decision boundary will look like this

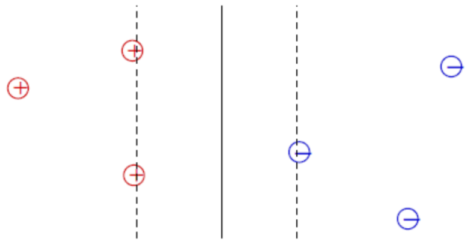
Source: Clark.2019

Semi-supervised SVMs(S3VMs)

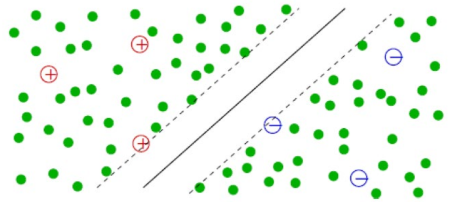
Olga Fink

Semi-supervised SVMs (S3VMs)

- SVMs



- Semi-supervised SVMs(S3VMs) = Transductive SVMs (TSVMs)



- Assumption: unlabeled data from different classes are separated by large margin
- Idea: The decision boundary shouldn’t lie in the regions of high density

Source: Xiaojin Zhu 2007

S3VM objective function

How to incorporate unlabeled points?

- Assign putative labels sign$(f(x))$ to $x \in X_u$
- sign$(f(x))f(x) = |f(x)|$
- The hinge loss on unlabeled points becomes

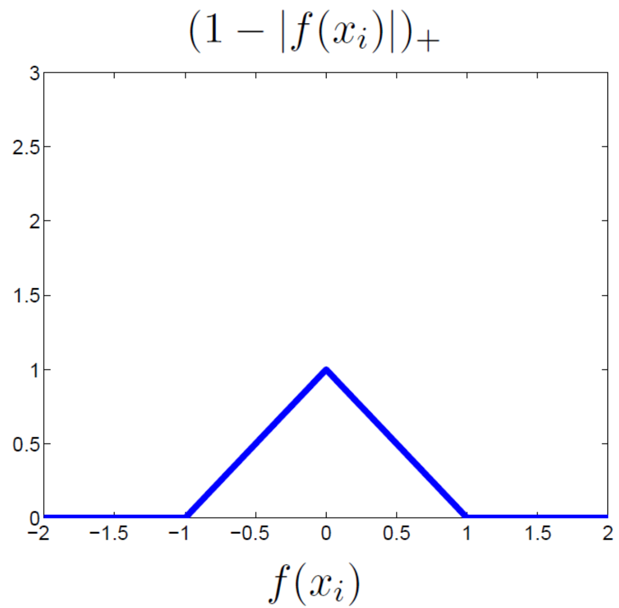
$$(1 - y_i f(x_i))_+ = (1 - |f(x_i)|)_+$$

S3VM objective:

$$\min_f \sum_{i=1}^{l} (1 - y_i f(x_i))_+ + \lambda_1 \|h\|^2_{\mathcal{H}_K} + \lambda_2 \sum_{i=l+1}^{n} (1 - |f(x_i)|)_+$$

Source: Xiaojin Zhu 2007

The hat loss on unlabeled data



Prefers $f(x) \geq 1$ or $f(x) \leq -1$, i.e., unlabeled instance away from decision boundary $f(x) = 0$.

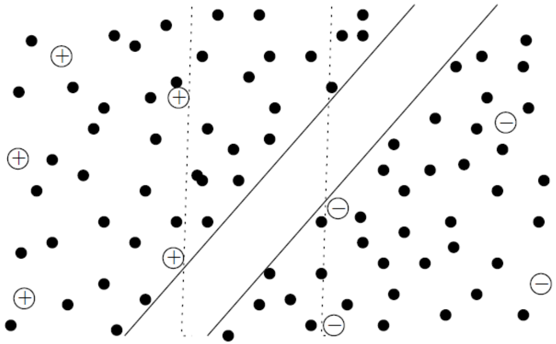
Source: Xiaojin Zhu 2007

Avoiding unlabeled data in the margin

S3VM objective:

$$\min_f \sum_{i=1}^{l} (1 - y_i f(x_i))_+ + \lambda_1 \|h\|^2_{\mathcal{H}_K} + \lambda_2 \sum_{i=l+1}^{n} (1 - |f(x_i)|)_+$$

the third term prefers unlabeled points outside the margin. Equivalently, the decision boundary $f = 0$ wants to be placed so that there is few unlabeled data near it.

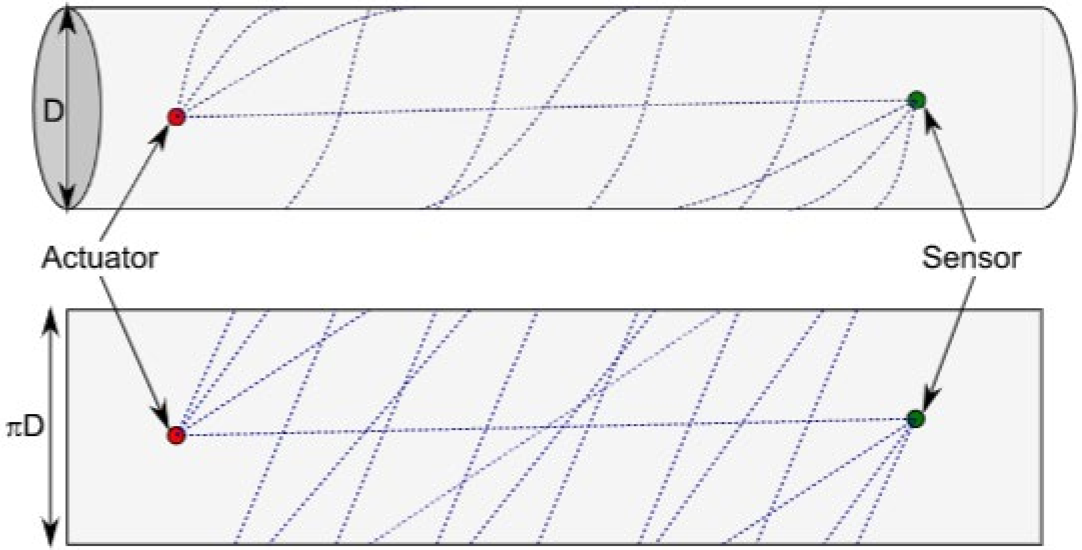


Source: Xiaojin Zhu 2007

Semi-Supervised Learning: examples

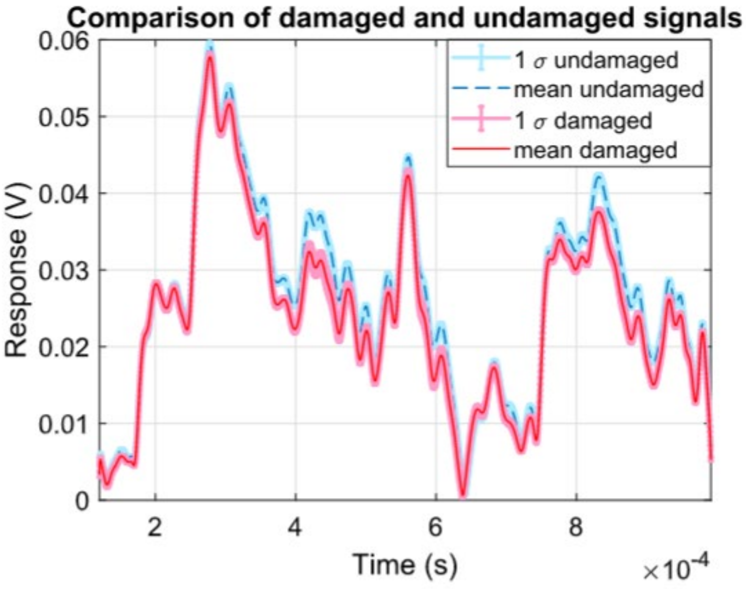
Olga Fink

Helical guided ultrasonic waves on a pipe



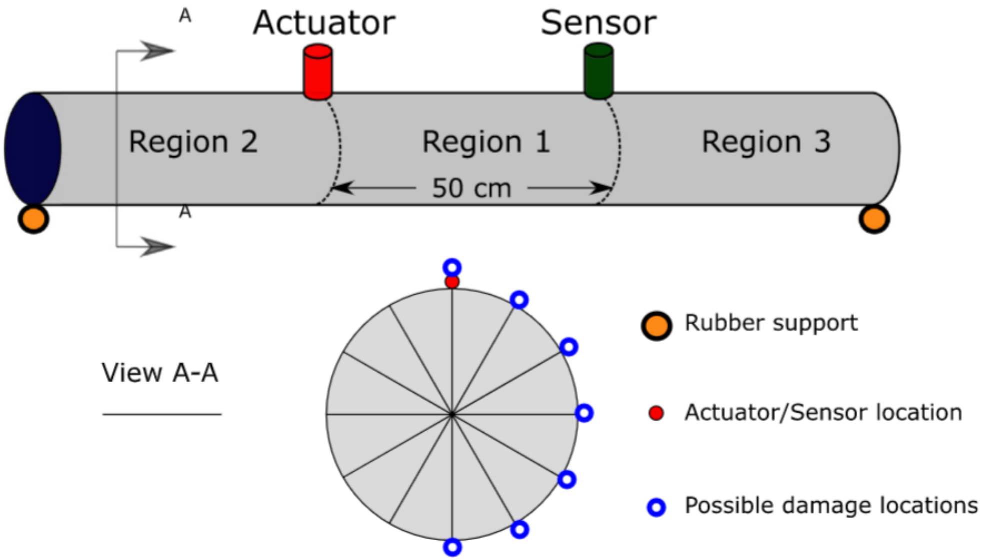
Sen, D., Aghazadeh, A., Mousavi, A., Nagarajaiah, S., Baraniuk, R., & Dabak, A. (2019). Data-driven semi-supervised and supervised learning algorithms for health monitoring of pipes. *Mechanical Systems and Signal Processing*, *131*, 524-537.

A comparison between envelope of mean damaged and mean undamaged response signals



Sen, D., Aghazadeh, A., Mousavi, A., Nagarajaiah, S., Baraniuk, R., & Dabak, A. (2019). Data-driven semi-supervised and supervised learning algorithms for health monitoring of pipes. *Mechanical Systems and Signal Processing*, *131*, 524-537.

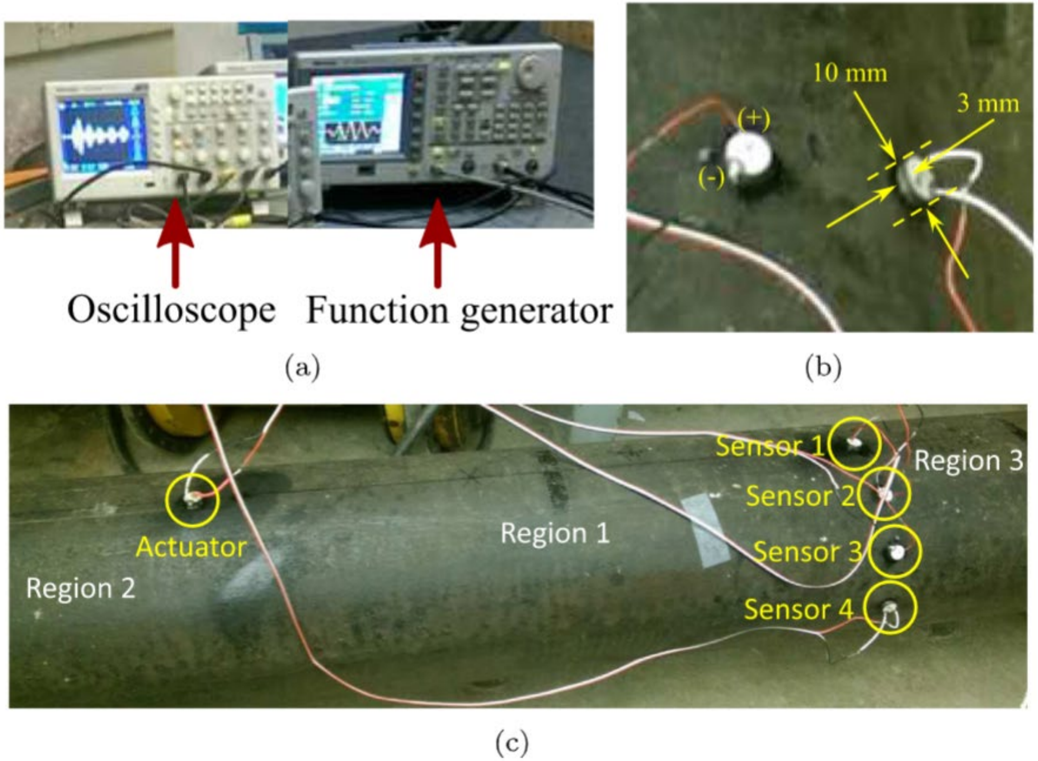
The general setup of the pipes



Pipe label	Material	Length (m)	Outer diameter (cm)	Thickness (mm)
A	Cast iron	1.23	16	5.2
B	Cast iron	3.05	16	6.73

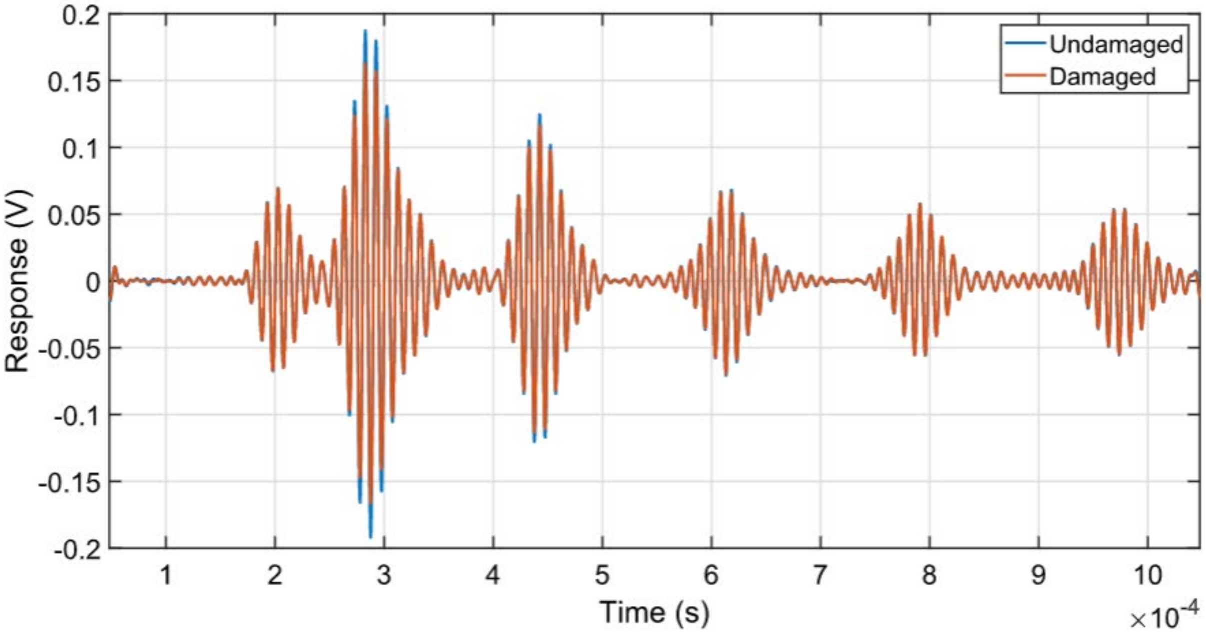
Sen, D., Aghazadeh, A., Mousavi, A., Nagarajaiah, S., Baraniuk, R., & Dabak, A. (2019). Data-driven semi-supervised and supervised learning algorithms for health monitoring of pipes. *Mechanical Systems and Signal Processing*, *131*, 524-537.

The experimental setup



Sen, D., Aghazadeh, A., Mousavi, A., Nagarajaiah, S., Baraniuk, R., & Dabak, A. (2019). Data-driven semi-supervised and supervised learning algorithms for health monitoring of pipes. *Mechanical Systems and Signal Processing*, *131*, 524-537.

A typical undamaged and damaged signal



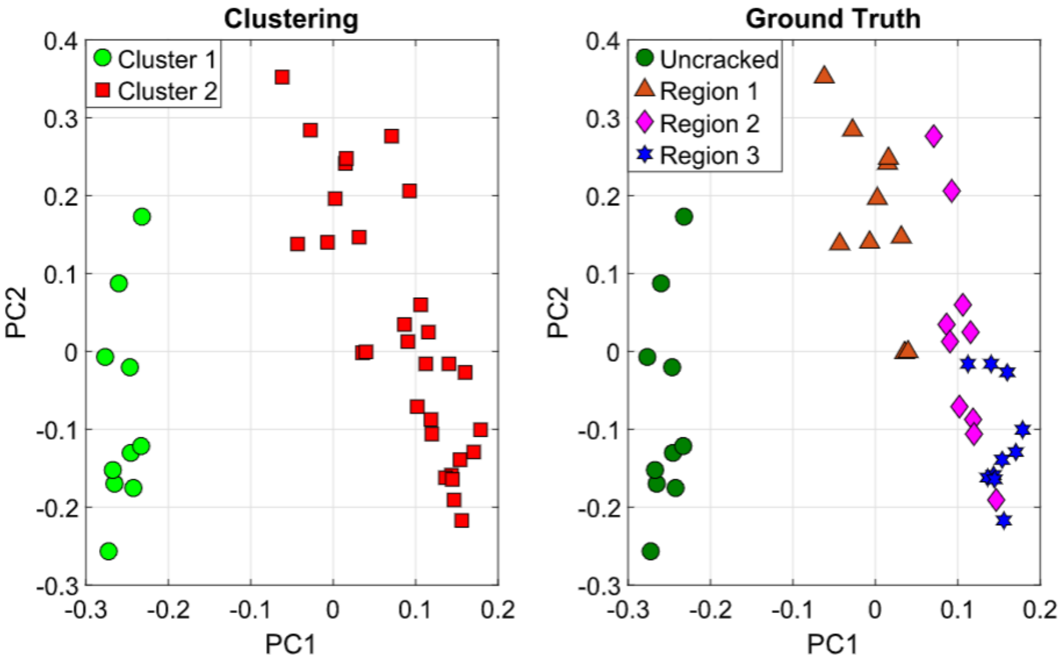
Sen, D., Aghazadeh, A., Mousavi, A., Nagarajaiah, S., Baraniuk, R., & Dabak, A. (2019). Data-driven semi-supervised and supervised learning algorithms for health monitoring of pipes. *Mechanical Systems and Signal Processing*, *131*, 524-537.

Algorithm

- Step 1: Acquire response signals (damaged or undamaged) from a pitch-catch set up.
- Step 2: Apply hierarchical clustering with complete linkage and Euclidean distance as the difference measure with number of clusters set to two.
- Step 3: Search for the labeled undamaged data in the clusters.
- Step 4: Assign the cluster with the undamaged label where the labeled data is present. This essentially leads to the knowledge about existence of damage

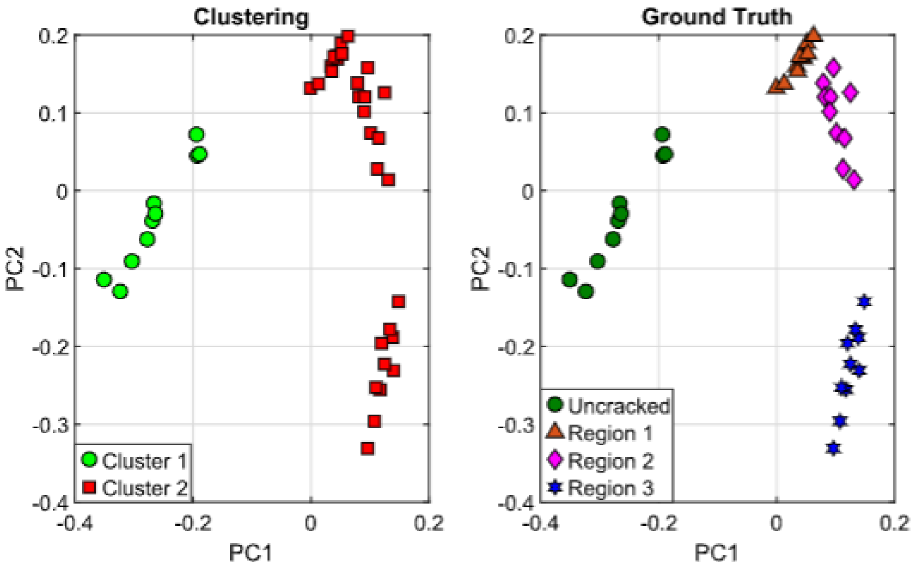
Sen, D., Aghazadeh, A., Mousavi, A., Nagarajaiah, S., Baraniuk, R., & Dabak, A. (2019). Data-driven semi-supervised and supervised learning algorithms for health monitoring of pipes. *Mechanical Systems and Signal Processing*, *131*, 524-537.

Results obtained from the proposed semi-supervised algorithm (Pipe A)



Sen, D., Aghazadeh, A., Mousavi, A., Nagarajaiah, S., Baraniuk, R., & Dabak, A. (2019). Data-driven semi-supervised and supervised learning algorithms for health monitoring of pipes. *Mechanical Systems and Signal Processing*, *131*, 524-537.

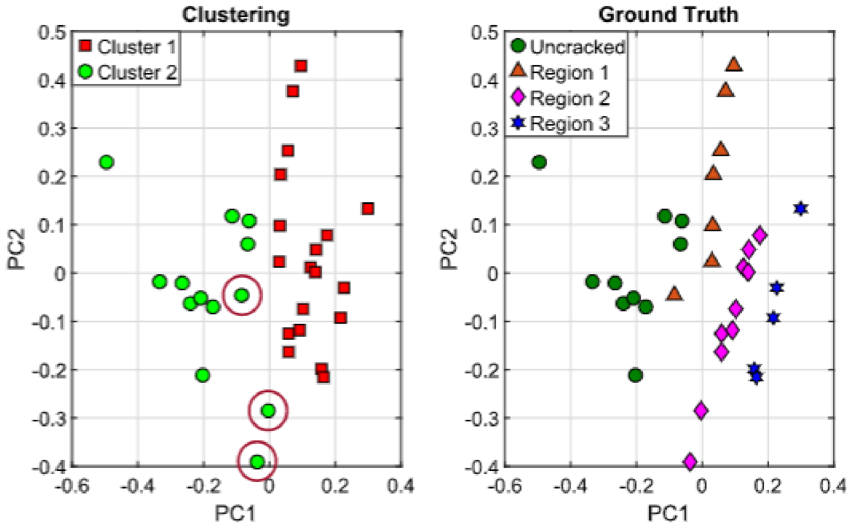
Results obtained from the proposed semi-supervised algorithm (Pipe B)



(a) 30°

Sen, D., Aghazadeh, A., Mousavi, A., Nagarajaiah, S., Baraniuk, R., & Dabak, A. (2019). Data-driven semi-supervised and supervised learning algorithms for health monitoring of pipes. *Mechanical Systems and Signal Processing*, *131*, 524-537.

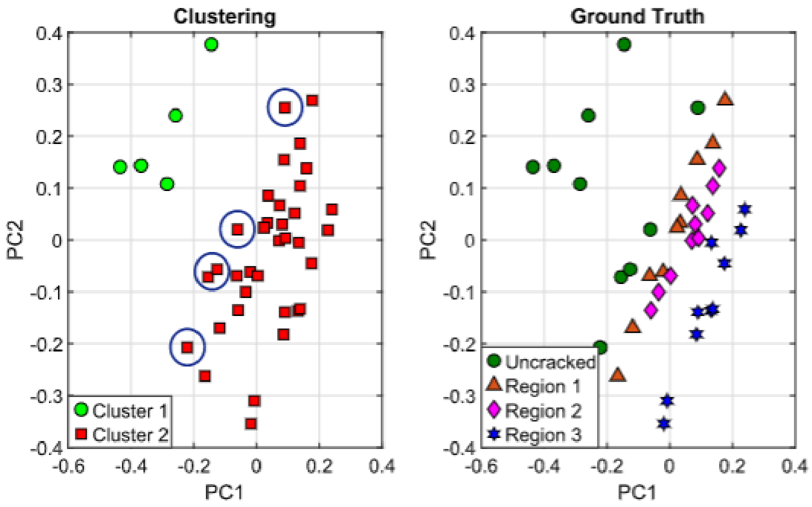
Results obtained from the proposed semi-supervised algorithm (Pipe B)



(d) 120°

Sen, D., Aghazadeh, A., Mousavi, A., Nagarajaiah, S., Baraniuk, R., & Dabak, A. (2019). Data-driven semi-supervised and supervised learning algorithms for health monitoring of pipes. *Mechanical Systems and Signal Processing*, *131*, 524-537.

Results obtained from the proposed semi-supervised algorithm (Pipe B)



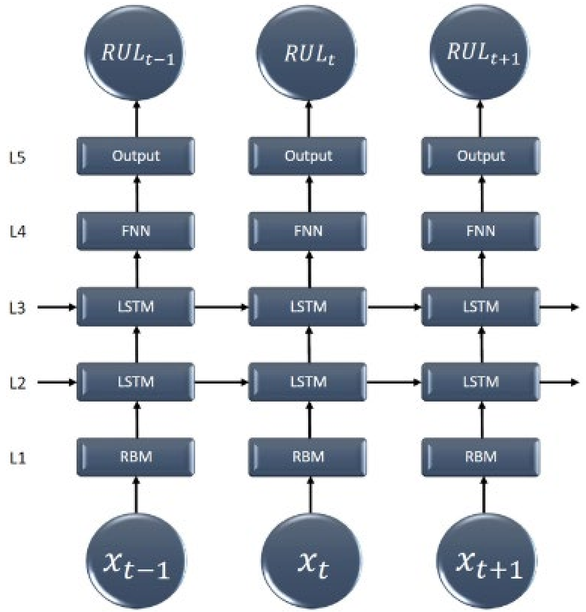
(e) 150°

Sen, D., Aghazadeh, A., Mousavi, A., Nagarajaiah, S., Baraniuk, R., & Dabak, A. (2019). Data-driven semi-supervised and supervised learning algorithms for health monitoring of pipes. *Mechanical Systems and Signal Processing*, *131*, 524-537.

Semi-Supervised Learning: RUL example 2

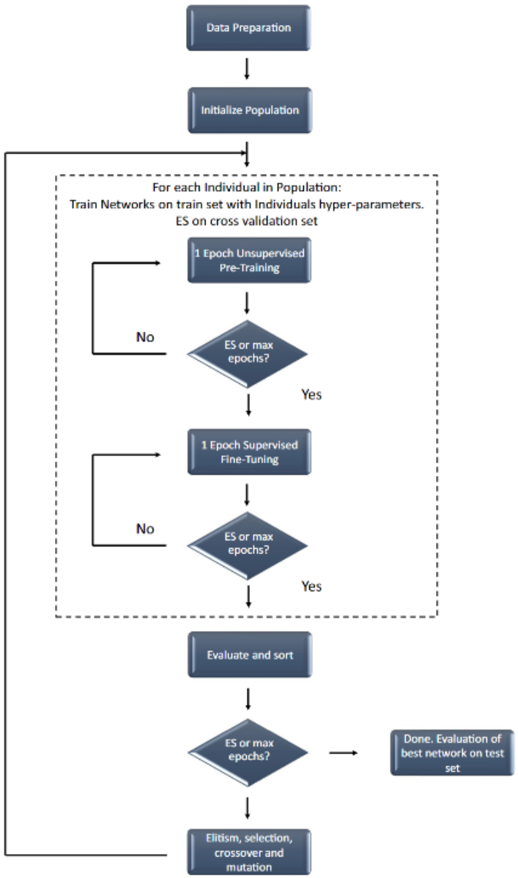
Olga Fink

Proposed architecture: Restricted Boltzmann Machines used for pre-training



Ellefsen, André Listou, et al. "Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture." *Reliability Engineering & System Safety* 183 (2019): 240-251.

Hyperparameter optimization



Ellefsen, André Listou, et al. "Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture." *Reliability Engineering & System Safety* 183 (2019): 240-251.

Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture

The proposed semi-supervised deep architecture with and without unsupervised pre-training on subset FD004 when the labeled training data is reduced from 100% to 10%. Improvement $= (1 - \frac{\text{Semi} - \text{supervised}}{\text{Supervised}})$.

RMSE	100%	80%	60%	40%	20%	10%
Semi-supervised with 100% training features in the pre-training stage	22.66	23.04	24.07	25.46	30.26	34.19
Supervised	23.62	23.45	24.14	26.40	30.27	34.90
Improvement	4.06%	1.75%	0.29%	3.56%	0.03%	2.03%
S	100%	80%	60%	40%	20%	10%
Semi-supervised with 100% training features in the pre-training stage	2840	3175	3576	5522	9562	22,476
Supervised	3234	3427	3650	6536	15,612	27,138
Improvement	12.18%	7.35%	2.03%	15.51%	38.75%	17.18%
Average training time per epoch (s)	100%	80%	60%	40%	20%	10%
Pre-training stage	7.08	7.08	7.08	7.08	7.08	7.08
Fine-tuning procedure	34.14	28.97	22.39	15.2	9.74	5.93

Ellefsen, André Listou, et al. "Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture." *Reliability Engineering & System Safety* 183 (2019): 240-251.