SNU EE/CS Seminar:
Time-series learning for Manufacturing AI

Sunghee Yun

Chief Applied Scientist
Gauss Labs Inc.

Today

- Why time-series machine learning in manufacturing AI?

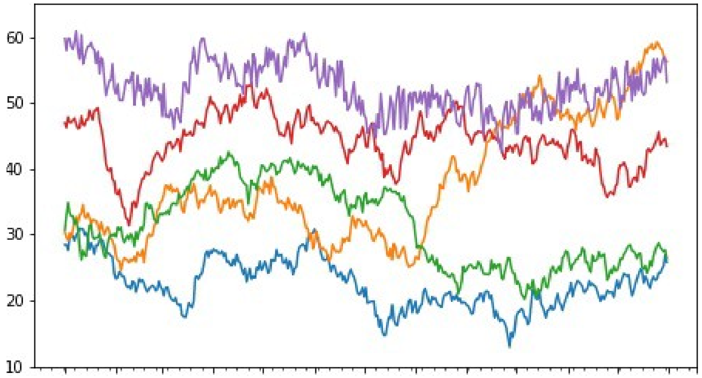
- Machine learning (ML) algorithms for time-series data
  - supervised learning for time-series
  - time-series anomaly detection
  - uncertainty prediction of predictions

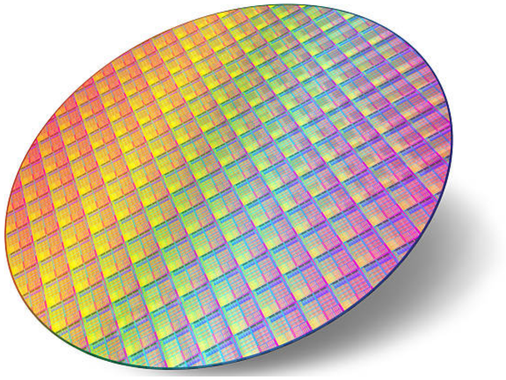
- Time-series learning applications in manufacturing
  - virtual metrology
  - root cause analysis

- Conclusion

Why time-series learning?

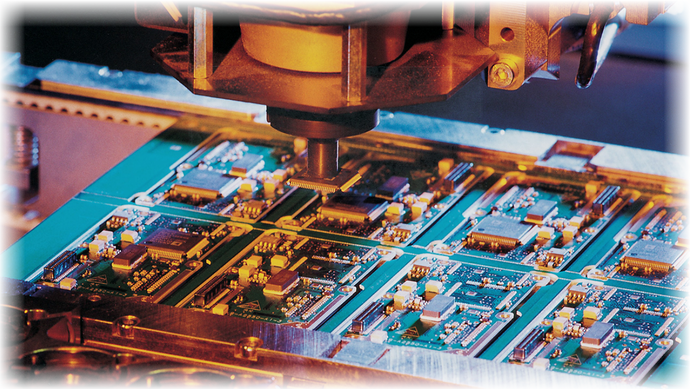
- (almost) all the data coming from manufacturing environment are time-series data
  - sensor data, sound data, process times, material measurement, images, yield, *etc.*
- sheer amount of time-series data is huge
  - tera-scale data per day generated in semiconductor manufacturing lines

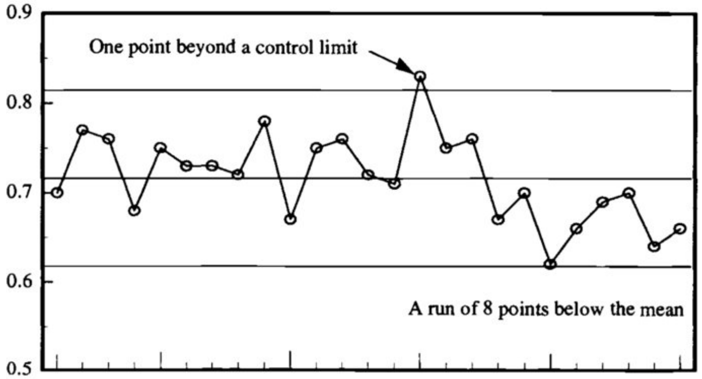




Why time-series learning?

- manufacturing application is about one of the following:
  - prediction of time-series values - virtual metrology, yield prediction
  - anomaly detection on time-series data - root cause analysis, yield analysis
  - classification of time-series values - equipment anomaly alarms





Machine Learning (ML) techniques for time-series data

Time-series data

- definition of times-series:

$$x : T \rightarrow \mathbf{R}^n \text{ where } T = \{\ldots, t_{-2}, t_{-1}, t_0, t_1, t_2, \ldots\} \subseteq \mathbf{R}$$

- example: material measurements: when $n = 3$

$$x(t) = \left[\begin{array}{c} \text{thickness}(t) \\ \text{some\_index}(t) \\ \text{feature\_size}(t) \end{array}\right]$$

- for supervised learning, we define two time series

$$x : T \rightarrow \mathbf{R}^n \text{ and } y : T \rightarrow \mathbf{R}^m$$

Time index

- time index does not have to be *time* index
- more general defintion

$$x : T \to \mathbf{R}^n \text{ where } T = \{\ldots, s_{-2}, s_{-1}, s_0, s_1, s_2, \ldots\}$$

  where $\cdots < s_{-1} < s_0 < s_1 < \cdots$ defines *an* ordering ($e.g.$, total order)

- for example, $x(s)$ and $y(s)$ can represent the features and target values for a processed material, s, where they are not measured at the same time
- throughout this talk, though, we will use time-index

Supervised learning for time-series

- canonical problem:

$$\begin{array}{rl} \text{predict} & y(t_k) \\ \text{given} & x(t_k), x(t_{k-1}), \ldots \text{ and } y(t_{k-1}), y(t_{k-2}), \ldots \end{array}$$

- lots of methods exist depending on assumptions of the data
  - for example, if we assume joint probability distribution of the data, we can have optimal solutions in certain criteria
- however, in this talk, we will *not* make such assumptions

Problem formulation

- canonical problem formulation:

$$\begin{array}{ll}\text{minimize} & \sum_{k=0}^{K} l(y(t_k), \hat{y}(t_k)) \\ \text{subject to} & \hat{y}(t_k) = g(x(t_k), x(t_{k-1}), \ldots, y(t_{k-1}), y(t_{k-2}), \ldots)\end{array}$$

  where $l : \mathbf{R}^m \times \mathbf{R}^m \to \mathbf{R}_+$ is loss function and $g : \mathcal{D} \to \mathbf{R}^m$
  where $\mathcal{D} = \mathbf{R}^n \times \mathbf{R}^n \times \cdots \mathbf{R}^m \times \mathbf{R}^m \times \cdots$

- we will use shortened notation for the predictor: $g : \cdot \to \mathbf{R}^m$

Machine learning (ML) solution candidates

- ignore temporal dependency and try to predict $y(t_k)$ from $x(t_k)$
  - supervised learing such as random forest, partial least squares, DL, *etc.*
- use sequential learning methods
  - recurrent neural network (RNN), LSTM, *etc.*
  - Transformer-like approach using attention mechanism

Difficulties with manufacturing applications

- for many manufacturing applications
  - concept drifts exist:
    - $p(x(t_k), x(t_{k-1}), \ldots)$ changes over time
    - $p(y(t_k)|x(t_k), x(t_{k-1}), \ldots, y(t_{k-1}), y(t_{k-2}), \ldots)$ changes over time
  - hence, traditional off-line training *doesn't* work!
  - DL-type algorithms do not work, either, because
    - data got stale very quickly
    - hence, data hungry DP do not work

Prediction based on expert advice

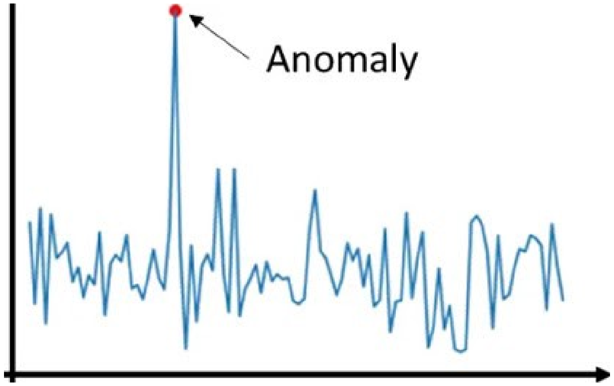
- assume p experts: $f_{i,k} : \cdot \to \mathbf{R}^m$ $(i = 1, 2, \ldots, p)$ for each time step, t_k
  - $f_{i,k}$ can be classical statistical learning, deep neural net, *etc.*
- model predictor at time step k, $g_k : \cdot \to \mathbf{R}^m$ as weighted sum of experts:

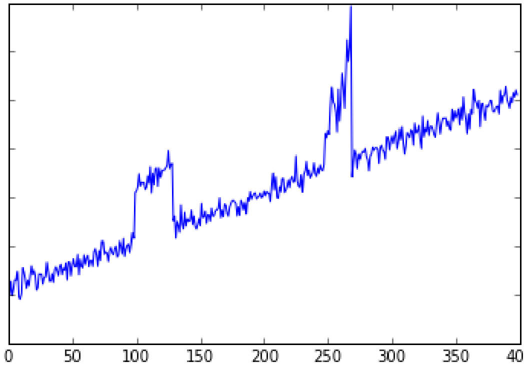
$$g_k = w_{1,k} f_{1,k} + w_{2,k} f_{2,k} + \cdots + w_{p,k} f_{p,k} = \sum_{i=1}^{p} w_{i,k} f_{i,k}$$

- algorithm:
  - predict $y(t_k)$, i.e., $\hat{y}(t_k) = g_k(\cdots)$ given current and past x's and past y's
  - observe $y(t_k)$
  - update weights $w_{1,k+1}, w_{2,k+1}, \ldots, w_{p,k+1}$ based on $\hat{y}(t_k) - y(t_k)$
  - repeat these steps

Time-series anomaly detection

- three types of anomaly detection: given time-series $x : T \to \mathbf{R}^n$
    - point anomaly: find k such that $x(t_k)$ is considerably different from most of the other data
    - segment anomaly: find k_1 and k_2 such that time-series segment $x(t_k)|_{k=k_1}^{k_2}$ is considerably different from most of the other data
    - sequence anomaly: given $x_1, \ldots, x_n : T \to \mathbf{R}$, find x_i such that it is considerably different from the other time-series, *i.e.*, x_j $(j \neq i)$





Time-series segment anomaly detection algorithm

- one method investigated using classification: given $x(t_j)|_{j=k}^{k-l+1}$, (segment of length l)
  - training:
    - choose one classifier, c, and p feature extractors (or transformers): f_i
    - extract p features by applying extractors: $y_{i,k} = f_i\left(x(t_j)|_{j=k-l+1}^{k}\right)$
    - train the classifier, c, with training data: $(y_{1,k}, 1)$, $(y_{2,k}, 2)$, . . . , $(y_{p,k}, p)$,
  - inferencing:
    - given new segment $x(t_j)|_{j=k-l+1}^{k}$, apply c to the extracted features.
    - if they are substantically different from $(1, 2, \ldots, p)$, declare it's anomaly
    - here "difference" quantified by some *anomaly score*, $e.g.$, KL divergence or entropy
  - c can be any classifier, $e.g.$, LSTM, $etc.$

Prediction of uncertainty of prediction

- every point prediction is wrong!
  - $\mathbf{Prob}(\hat{Y}_k = Y_k) = 0$

  no matter how accuracy the predictor is

- more importantly, want to know how reliable our prediction is

- we call this “model uncertainty estimation (MUE)”

Model uncertainty estimation (MUE)

- multiple ways to measure this:
  (1) probability of true value falling into an interval: for fixed $a > 0$

$$\mathbf{Prob}(|Y_k - \hat{Y}_k| < a) = \mathbf{Prob}(Y_k \in (\hat{Y}_k - a, \hat{Y}_k + a))$$

  (2) predictive distribution size: find $a > 0$ such that

$$\mathbf{Prob}(|Y_k - \hat{Y}_k| < a) = 95\%$$

  (3) distribution of Y_k: find PDF of Y_k

- solving (3) readily solves (1) and (2)

MUE for expert-based online learning

- reminder: online learning method based on expert advice is given by

$$g_k = w_{1,k} f_{1,k} + w_{2,k} f_{2,k} + \cdots + w_{p,k} f_{p,k} = \sum_{i=1}^{p} w_{i,k} f_{i,k}$$

- uncertainty for $f_{i,k}$ modeled by distribution parameterized by $\theta_{i,k}$, *i.e.*, $p(\gamma; \theta_{i,k})$; γ is random variable
- we first evaluate the predictive distribution

$$p_{i,k}(y(t_k); x(t_k)) = \int p(y; x(t_k), \gamma) p(\gamma; \theta_{i,k}) d\gamma$$

- problem to solve: evaluate distribution of g_k given those of $f_{i,k}$

MUE for expert-based online learning

- independent case: if $p_{1,k}, \ldots, p_{p,k}$ are (statistically) independent, then PDF of $g_k(x(t_k))$ can be calculated by

$$\frac{p_{1,k}(y/w_{1,k}; x(t_k))}{w_{1,k}} \star \cdots \star \frac{p_{p,k}(y/w_{p,k}; x(t_k))}{w_{p,k}}$$

- Gaussian case: $p_{1,k}, \ldots, p_{p,k}$ are Gaussians with correlation coefficient matrixa R, *i.e.*,

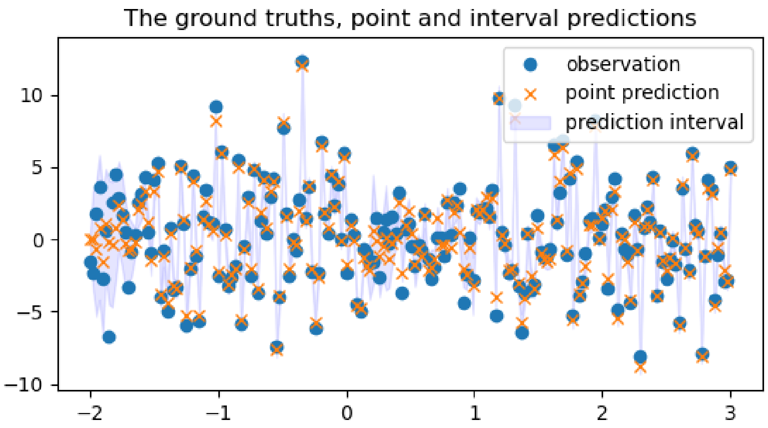
$$p_{i,k} \sim \mathcal{N}(\mu_{i,k}(x(t_k)), \sigma_{i,k}(x(t_k))^2)$$
$$R = \begin{bmatrix} 1 & \rho_{1,2} & \rho_{1,3} & \cdots & \rho_{1,p} \\ \rho_{1,2} & 1 & \rho_{2,3} & \cdots & \rho_{2,p} \\ \rho_{1,3} & \rho_{2,3} & 1 & \cdots & \rho_{3,p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho_{1,p} & \rho_{2,p} & \rho_{3,p} & \cdots & 1 \end{bmatrix} \in \mathbf{R}^{p\times p}$$

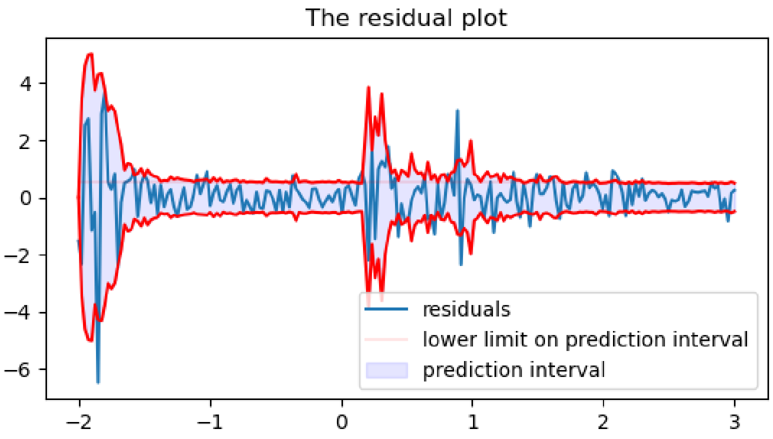
- then g_k is also Gaussian

$$\mathcal{N}(w_k^T \mu_k(x(t_k)), w_k^T \,\mathbf{diag}(\sigma_k(x(t_k)))R\,\mathbf{diag}(\sigma_k(x(t_k)))w_k)$$

where

$$\begin{aligned}
w_k &= \begin{bmatrix} w_{1,k} & \cdots & w_{p,k} \end{bmatrix}^T \in \mathbf{R}^p \\
\mu_k(x(t_k)) &= \begin{bmatrix} \mu_{1,k}(x(t_k)) & \cdots & \mu_{p,k}(x(t_k)) \end{bmatrix}^T \in \mathbf{R}^p \\
\sigma_k(x(t_k)) &= \begin{bmatrix} \sigma_{1,k}(x(t_k)) & \cdots & \sigma_{p,k}(x(t_k)) \end{bmatrix}^T \in \mathbf{R}^p
\end{aligned}$$


The ground truths, point and interval predictions


The residual plot

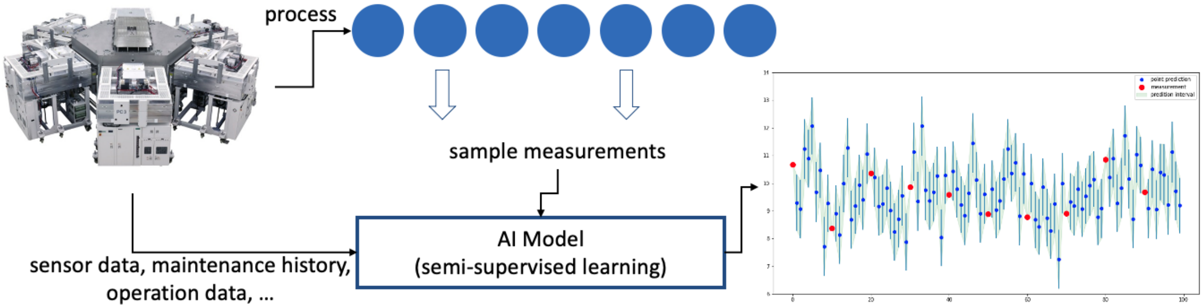
Time-series Learning Applications in Manufacturing

Virtual metrology (VM)

- in many cases, we cannot measure all processed materials for fundamental reasons
  - measurement equipment is too expensive
  - no room in the factory for many measurement equipment
  - measuring every materials hinders production speed inducing low throughput
- thus, we do sampling (with very low smapling rate)
  - in semiconductor manufacturing line, avarage sampling rate is less than 1%
- problem: we want to predict the measurement of unmeasured material using indirect signals such as
  - sensor data, maintenance history, operation data, . . .

VM

- difficulties
  - concept drift/shift due to maintenance
  - data becomes stale quickly
- *online learning method based on expert advice* is used for the solution
- MUE provides the uncertainty level of our prediction
  - process engineers can judge when they can trust the predictions
  - we can monitor performance degradation



process

sample measurements

AI Model
(semi-supervised learning)

sensor data, maintenance history,
operation data, ...

Applications of VM

- why do we even develop VM?
- focus on the values we deliver to out customers; want VM to be used for
  - process control, $e.g.$, feedback control
  - detecting equipment out-of-control status
  - detecting root caues for yield drop
  - predicting (future) yield

Different error measures depending on VM applications

- mean-square-error (MSE) for run-to-run control (where $\mathcal{K}$ is test index set)

$$\mathbf{E}\,\|Y - \hat{Y}\|^2 \simeq \frac{1}{|\mathcal{K}|} \sum_{k\in\mathcal{K}} \|y(t_k) - \hat{y}(t_k)\|^2$$

- mean-p-norm-error (MPE) for anomaly detection (with $p > 2$)

$$\mathbf{E}\,\|Y - \hat{Y}\|_p^p \simeq \frac{1}{|\mathcal{K}|} \sum_{k\in\mathcal{K}} \|y(t_k) - \hat{y}(t_k)\|_p^p$$

- R-squared (R^2)

$$1 - \frac{\mathbf{E}\,\|Y - \hat{Y}\|^2}{\mathbf{E}\,\|Y - \mathbf{E}\,Y\|^2} \simeq 1 - \frac{\sum_{k\in\mathcal{K}} \|y(t_k) - \hat{y}(t_k)\|^2}{\sum_{k\in\mathcal{K}} \|y(t_k) - \bar{y}\|^2}$$

Root cause analysis by anomaly detection

- background: statistical process control (SPC)
  - conventional old method used in manufacturing (since 1950's)
  - monitor measurement and alert when things go wrong
  - things go wrong defined by rules; examples:
    - measument out of $(\mu - 3\sigma, \mu + 3\sigma)$,
    - three consecutive measurements out of $(\mu - 2\sigma, \mu + 2\sigma)$
- our problem: when SPC alarm goes off, find the responsible (chamber in) equipment

Root cause analysis by anomaly detection

- two methods exist: (1) segment anomaly detection and (2) sequence anomaly detection
- two types of data exist: (1) sensor data and (2) processed material measurement data
- problems: given time-series data $x_e(t_0), x_e(t_1), \ldots$ for each entity $e \in E$ (entity refers to equipment, chamber, station, $etc.$)
  - find entity e that shows abnormal behavior using segment anomaly detection
  - find entiry e that is different from other entities using sequence anomaly detection

Conclusion

- time-series learning and anomaly detection occur at various places in manufacturing AI applications
- concept drift and data noise make them very challenging, but have working solutions
- solutions: time-series supervised learning, time-series anomaly detection, model uncertainty estimation
- lots of applications exist
  - virtual metrology, root cause analysis, yield prediction, failure pattern analysis, predictive maintenance, *etc.*