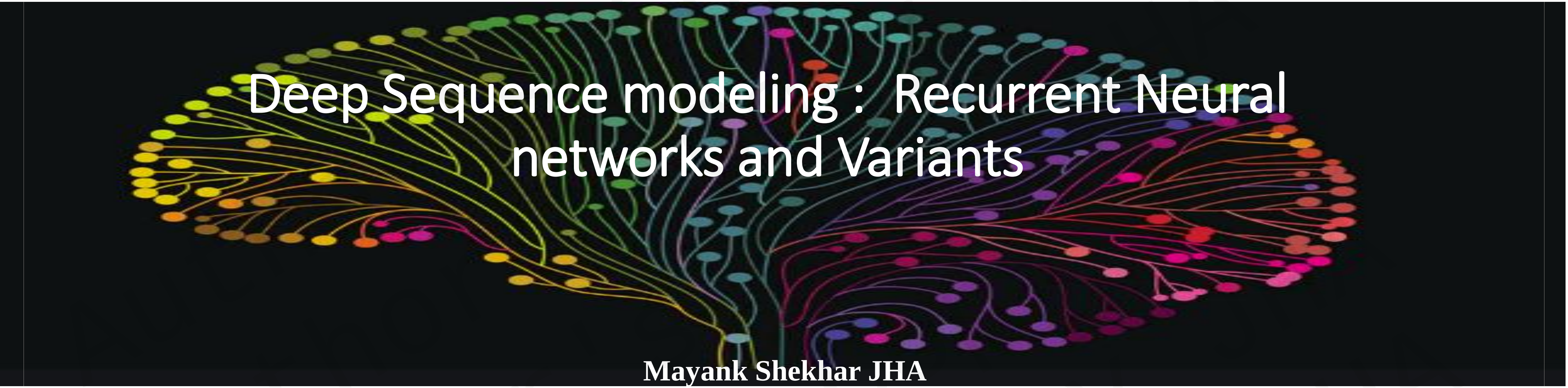


Deep Sequence modeling : Recurrent Neural networks and Variants

Mayank Shekhar JHA

Associate Professor

(Maitre de Conférences)

Automatic Control, Reliability of Systems

[mayank-shekhar.jha{at}univ-lorraine.fr](mailto:mayank-shekhar.jha@univ-lorraine.fr)

Research

Centre de Recherche en Automatique de Nancy (CRAN) UMR 7039,
Faculté des Sciences et Technologies
Boulevard des Aiguillettes - BP 70239 - Bât. 1er cycle 54506 Vandoeuvre-lès-Nancy Cedex France

Teaching

Polytech Nancy (ESSTIN),
2 Rue Jean Lamour 54509 Vandoeuvre-lès-Nancy, Cedex France

Email: mayank-shekhar.jha [at] univ-lorraine.fr

Sequence Modelling

Recurrent Neural Networks

Long Short Term Memory (LSTMs)

Application: Prognostics and Deep Learning

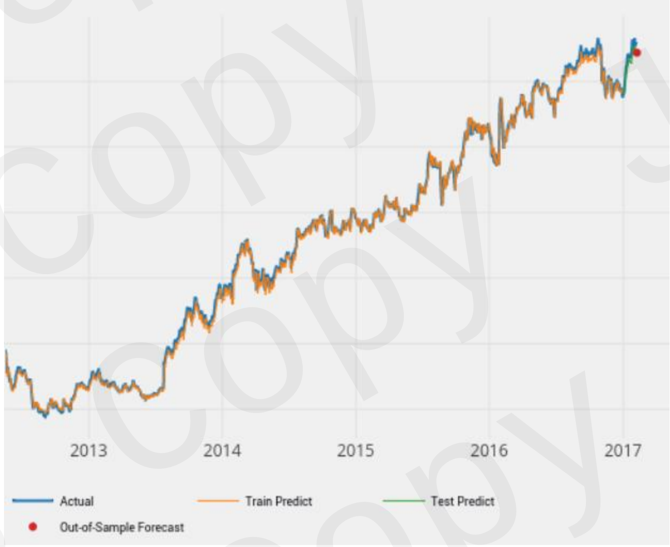
Sequence Modelling

Motivations
Challenges
Some ideas
Design

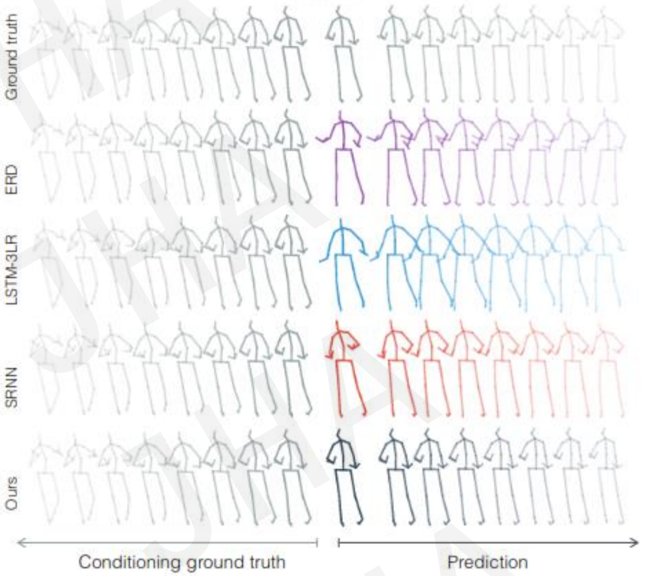
JHA Mayank , Email: mayank-shekhar.jha [at] univ-lorraine.fr

Sequence modelling : Motivations

- Sequential data:
  - time series forecasting,
  - motion prediction (human, self driving cars)
  - sensor data: machine health monitoring/prediction
  - text processing/prediction
  - machine translation



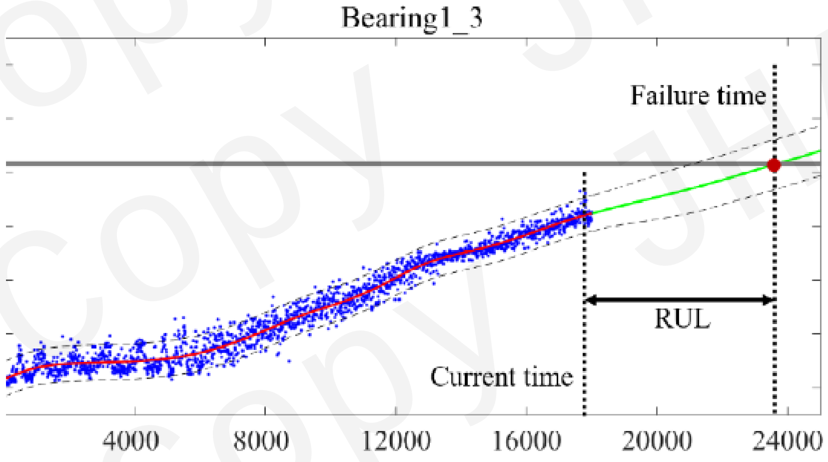
Financial market prediction (Dixon et al.)



Human Motion Prediction

Martinez et al., 2016

*I am from London but I live in Paris and I speak fluent **English.***



Component Failure Prediction
(Yoo et al., 2018)

Sequence modelling : Motivations > Challenges

- Inputs data
  - Variable lengths
  - spatially + temporally dependent
  - ordered
  - output data different length than input (machine translation)

Sequence modelling : Motivations >Challenges > Some ideas

1. Fixed window
   - cannot model long term dependencies
2. Use whole sequence as counts (I occurs 3 times)
   - no learning of order (what followed by what?)
3. Large window length input
   - each has separate parameter
   - learning will not transfer at other places in the sequence.

I am from London but I live in Paris so I speak fluent **……… .**

One hot coding

[00100101000101011000011….] ➡ ???

I live in Paris so I speak fluent

Sequence modelling : Motivations >Challenges > Some ideas

1. Fixed window
   - cannot model long term dependencies
2. Use whole sequence as counts (*I* occurs 3 times)
   - no learning of order (what followed by what?)

3. Large window length input
   - each has separate parameter
   - learning will not transfer at other places in the sequence.

I am from London but I live in Paris so I speak fluent ....

One hot coding

[00100101000101011000011....] ➡ ???

I live in Paris so I speak fluent

- Feed forward NN, not designed to:
  - handle variable data lengths
  - parameter sharing (correlation, temporal dependency…)
  - track long term dependency + order

- CovNets:
  - can share parameters across time but remain shallow.

Sequence modelling : Motivations >Challenges > Some ideas > Design

- Variable length inputs
- Learn long term dependencies
- Learn the order in data
- Share parameters across sequence
- Make predictions (long term) efficiently.

Recurrent Neural Networks

RNNs: Structure

- Recurrence of states. Ex. a dynamical system
- Recursive computation ➔ Computational graph

$$s_t = f(s_{t-1}, w)$$

$$\cdots \xrightarrow{f} s_{t-1} \xrightarrow{f} s_t \xrightarrow{f} s_{t+1} \xrightarrow{f} s_{t+2}$$

$$\begin{aligned} s_3 &= f(s_2, w) \\ &= f(f(s_1, w), w) \end{aligned}$$

RNNs: Structure

$$s_t = f(s_{t-1}, w)$$

- Recurrence of states. Ex. a dynamical system
- Recursive computation ➔ Computational graph
- When system driven by external input,

RNNs: Structure

- Recurrence of states. Ex. a dynamical system
- Recursive computation ➔ Computational graph
- When system driven by external input,

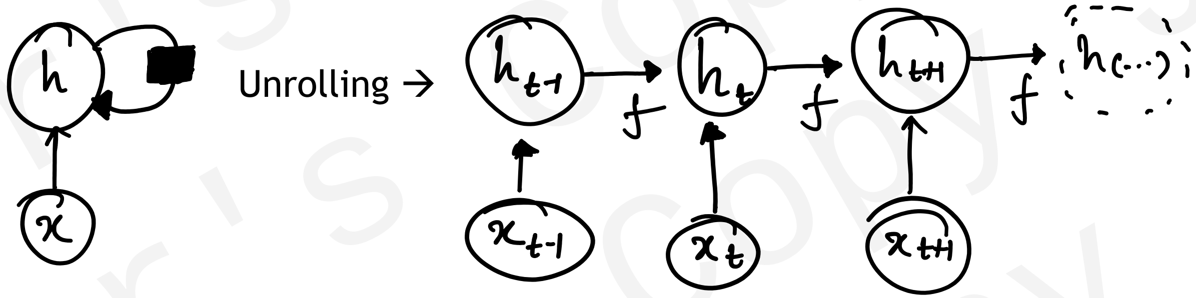
  New state contains information about history.

$$s_t = f(s_{t-1}, w)$$

Rewritten

$$h_t = f(h_{t-1}, x_t, w)$$

RNNs : Output of node fed back into the hidden nodes (recurrent, cyclic structure)



- Captures dependency in input data.
- Same weights at each time step : some weight sharing.

$$h_t = f(h_{t-1}, x_t)$$
$$= \sigma(W^h h_{t-1} + W^x x_{t-1})$$

RNNs: Structure

- Recurrence of states. Ex. a dynamical system
- Recursive computation ➔ Computational graph
- When system driven by external input,

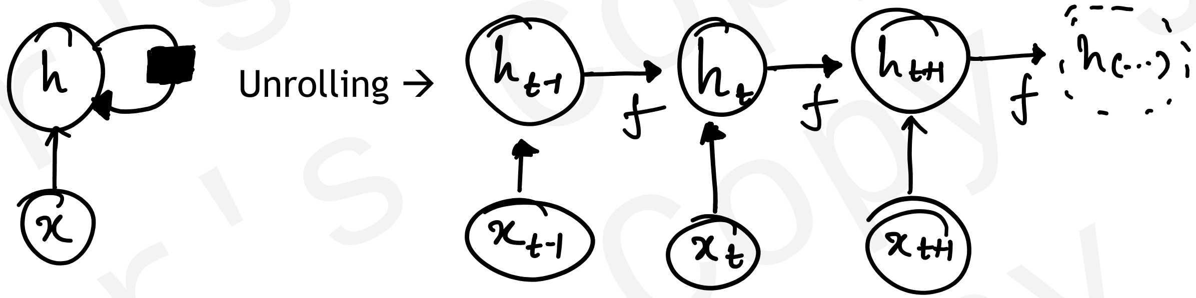
  New state contains information about history.

$$s_t = f(s_{t-1}, w)$$

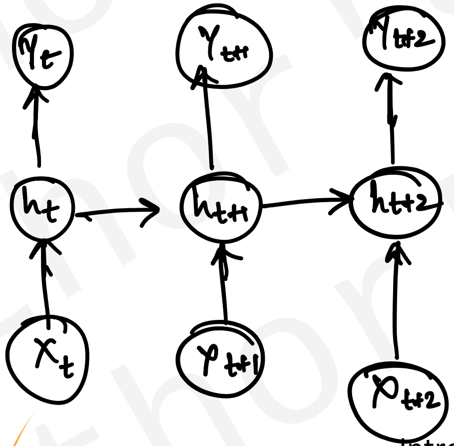
Rewritten

$$h_t = f(h_{t-1}, x_t, w)$$

RNNs : Output of node fed back into the hidden nodes (recurrent, cyclic structure)



- Captures dependency in input data.
- Same weights at each time step : some weight sharing.
- Outputs from RNN:



$$h_t = f(h_{t-1}, x_t)$$

$$= \sigma(W^h h_{t-1} + W^x x_{t-1})$$

$$y_t = g(h_t, x_t)$$

$$= \sigma(W^h h_t)$$

RNNs: Structure

- Recurrence of states. Ex. a dynamical system
- Recursive computation ➔ Computational graph
- When system driven by external input,

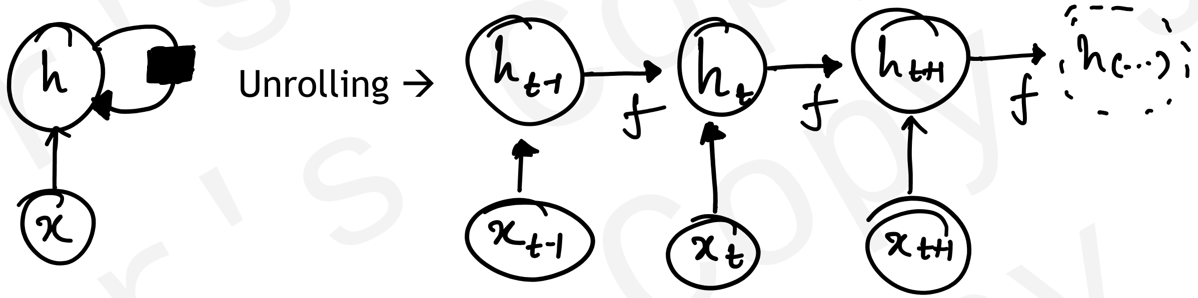
  New state contains information about history.

$$s_t = f(s_{t-1}, w)$$

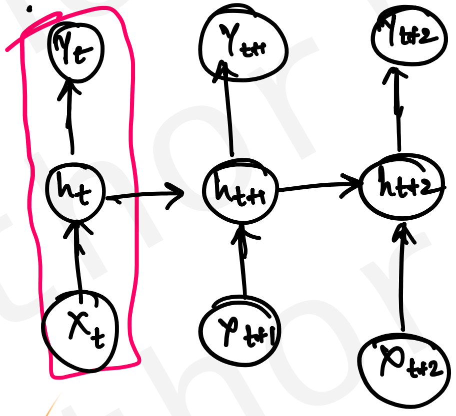
Rewritten

$$h_t = f(h_{t-1}, x_t, w)$$

RNNs : Output of node fed back into the hidden nodes (recurrent, cyclic structure)



- Captures dependency in input data.
- Same weights at each time step : some weight sharing.
- Outputs from RNN:



$$h_t = f(h_{t-1}, x_t)$$

$$= \sigma(W^h h_{t-1} + W^x x_{t-1})$$

$$y_t = g(h_t, x_t)$$

$$= \sigma(W^h h_t)$$

RNNs: Structure

- Recurrence of states. Ex. a dynamical system
- Recursive computation ➔ Computational graph
- When system driven by external input,

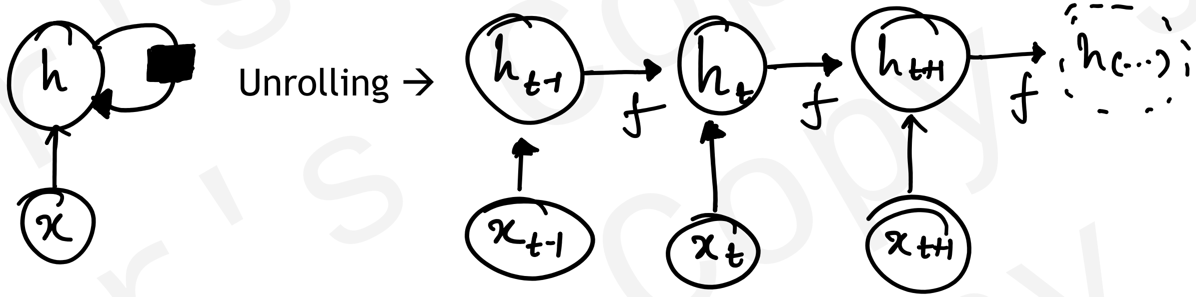
  New state contains information about history.

$$s_t = f(s_{t-1}, w)$$

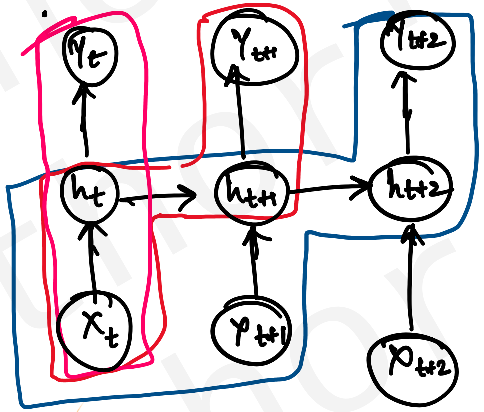
Rewritten

$$h_t = f(h_{t-1}, x_t, w)$$

RNNs : Output of node fed back into the hidden nodes (recurrent, cyclic structure)



- Captures dependency in input data.
- Same weights at each time step : some weight sharing.
- Outputs from RNN:



$$h_t = f(h_{t-1}, x_t)$$
$$= \sigma(W^h h_{t-1} + W^x x_{t-1})$$

$$y_t = g(h_t, x_t)$$
$$= \sigma(W^h h_t)$$

RNNs: Structure

- Depending upon application:
  - h needs to be rich,
  - capture all historical trends {cyclicity, seasonality, trend, fluctuations, global/local}

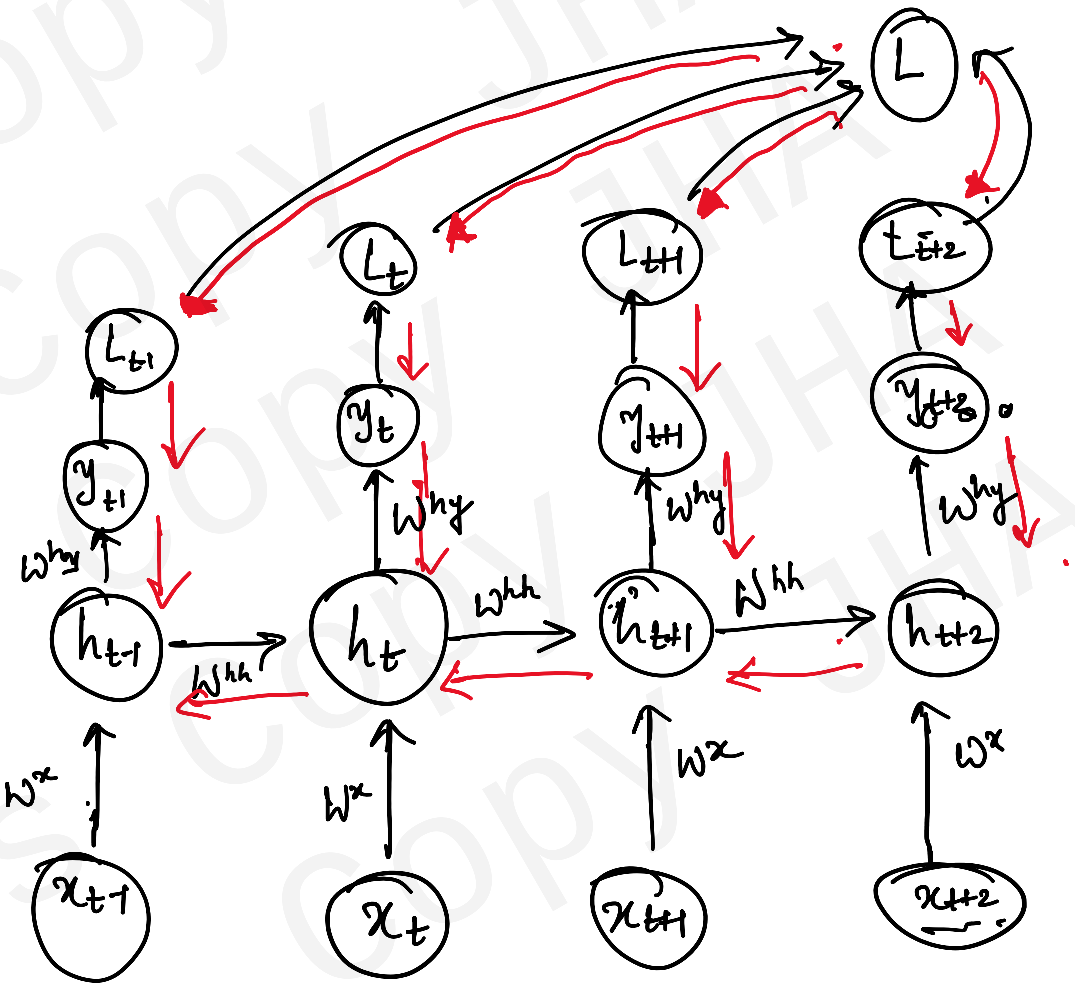
- Advantage:
  - learnt model has same size (regardless of input size)
  - possible to use same transition function f

RNNs: Structure

- Depending upon application:
  - *h* needs to be rich,
  - capture all historical trends {cyclicity, seasonality, trend, fluctuations, global/local}

- Advantage:
  - learnt model has same size (regardless of input size)
  - possible to use same transition function *f*

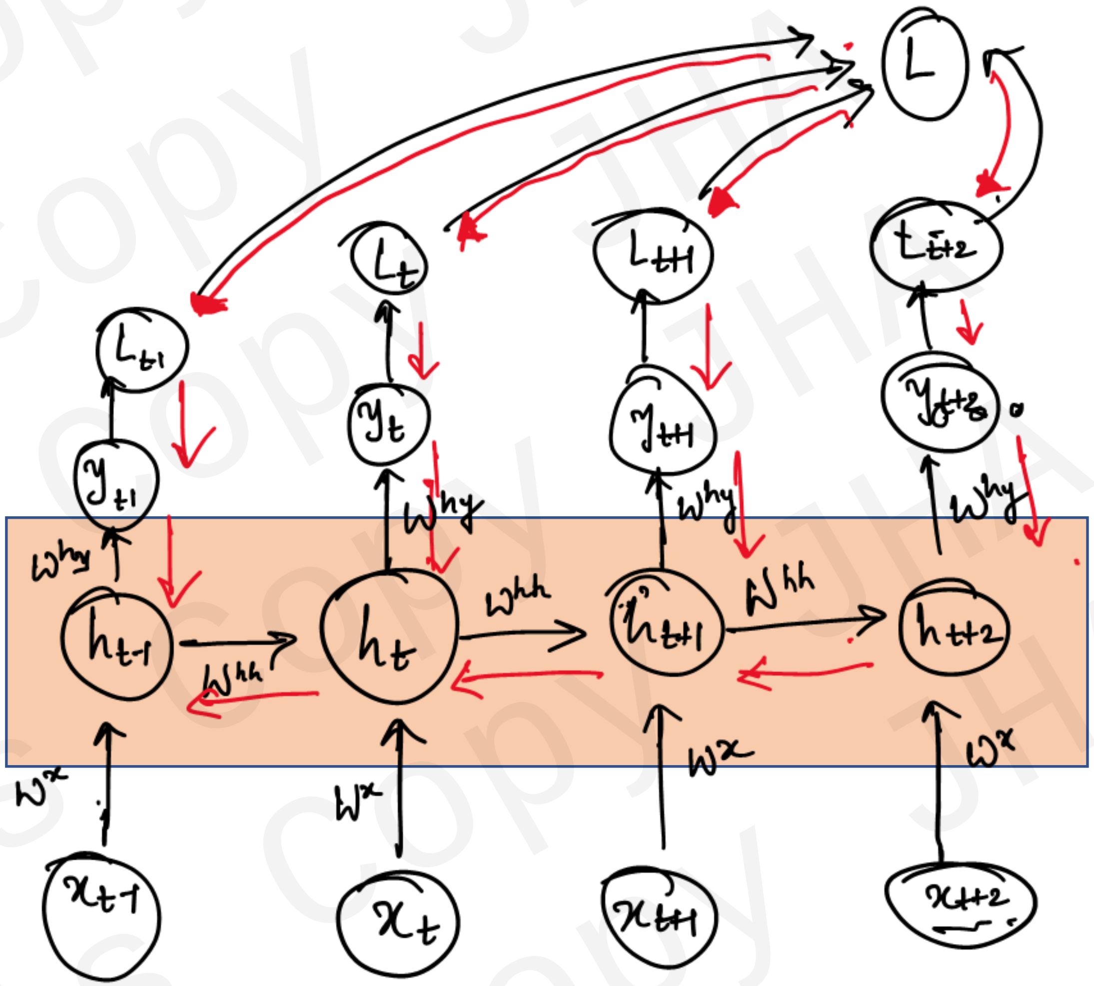
- Learning → Back-propagation through time (BPTT)
  - errors calculated/back-propagated over time = BP over unrolled network
  - gradients calculated in time.
  - Training slower than MLP:
    - repeated multiplication of weights in sequence length
    - repeated product of derivative of activation function.



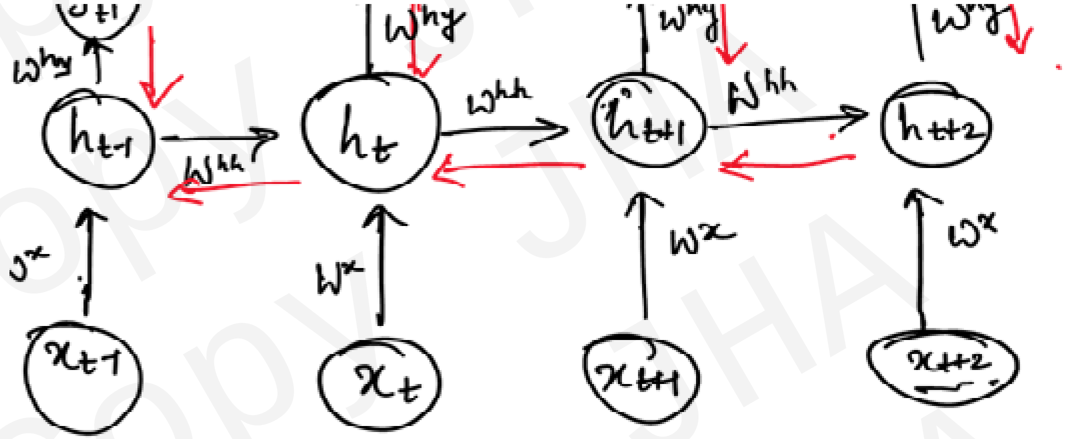
Challenges:

Vanishing gradients: Many values <<1

- activation gradient products
- small weights
- negligible gradient ➔ negligible learning.



Challenges:



Vanishing gradient problem: Many values <<1

- activation gradient products
- small weights
- negligible gradient ➔ negligible learning.

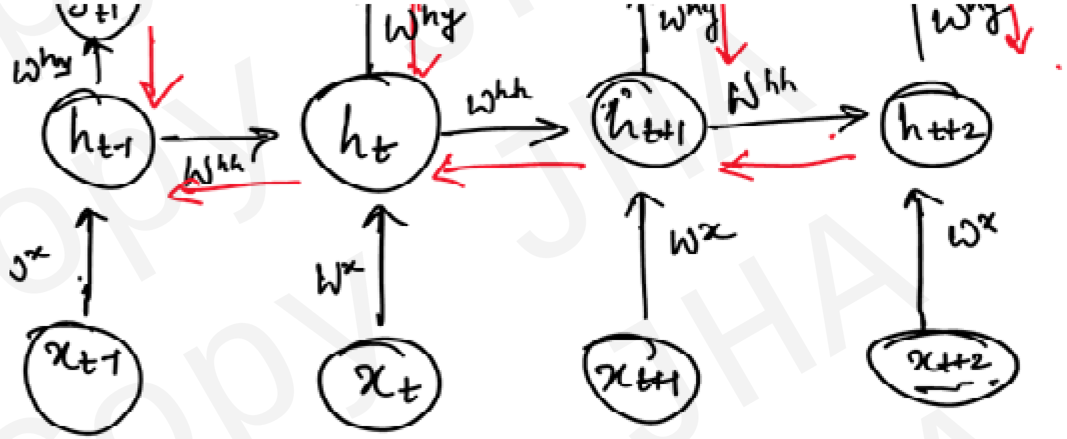
Long range Learning:

- hidden units modify with new information
- vanishing gradient problem ➔ new information not preserved over long ranges.
    - time series forecasting: seasonality etc.

Challenges:

Vanishing gradient problem: Many values <<1

- activation gradient products
- small weights
- negligible gradient ➔ negligible learning.



Long range Learning:

- hidden units modify with new information
- vanishing gradient problem ➔ new information not preserved over long ranges.
  - time series forecasting: seasonality etc.
  - machine translation: relation of first word to context
  - prognostics: prediction of state of health at long time range

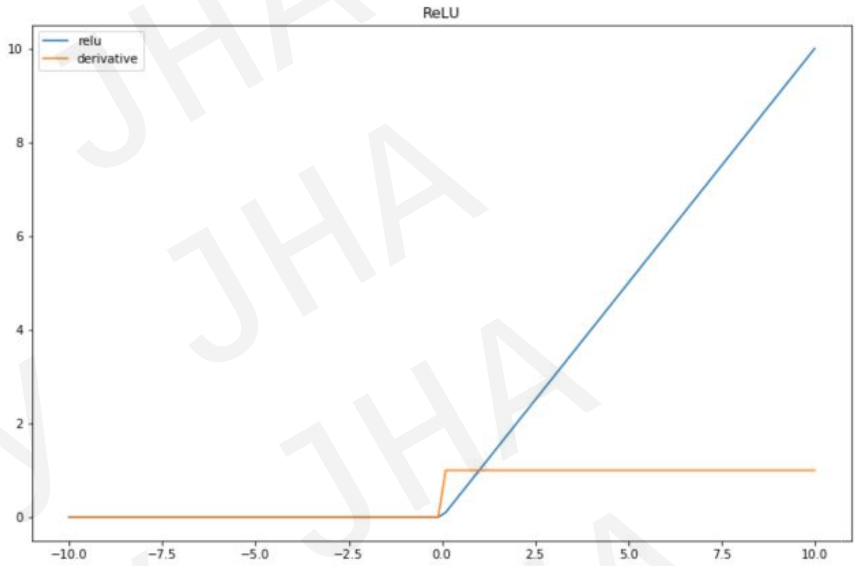
I am from London but I live in Paris so I speak fluent **……..**

Prediction Drift:

- next step prediction ➔ recurrence of *h* learnt
- long range prediction → recurrence of *h* over multiple steps
- error cumulation over multiple time steps

Solution:

- Efficient parameter initialization
- Non-saturating activation functions: ReLU, Leaky ReLu…
- Gradient clipping



- Gated Cells:
  - "control" the information flow
  - allow more useful information, forget non-useful information…
  - track information through many time steps to filter out the useless ones.

Long Short Term Memory (LSTMs)

JHA Mayank , Email: mayank-shekhar.jha [at] univ-lorraine.fr

LSTMs (Hochreiter & Schmidhuber 1997)

Gated RNNs: let selective information through

$$h_t = f(h_{t-1}, x_t)$$
$$= \sigma(W^h h_{t-1} + W^x x_{t-1})$$

cleverly designed

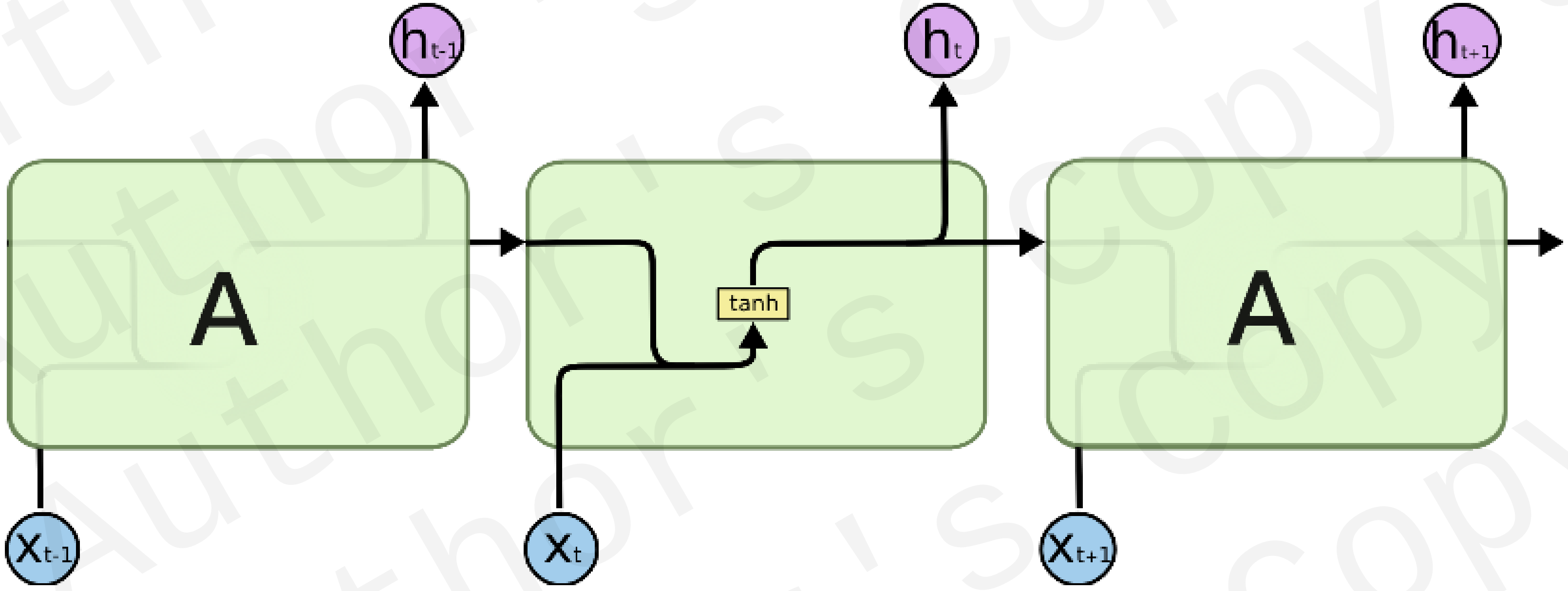
LSTMs (Hochreiter & Schmidhuber 1997)

Gated RNNs: let selective information through

$$h_t = f(h_{t-1}, x_t)$$
$$= \sigma(W^h h_{t-1} + W^x x_{t-1})$$

cleverly designed

RNNS:



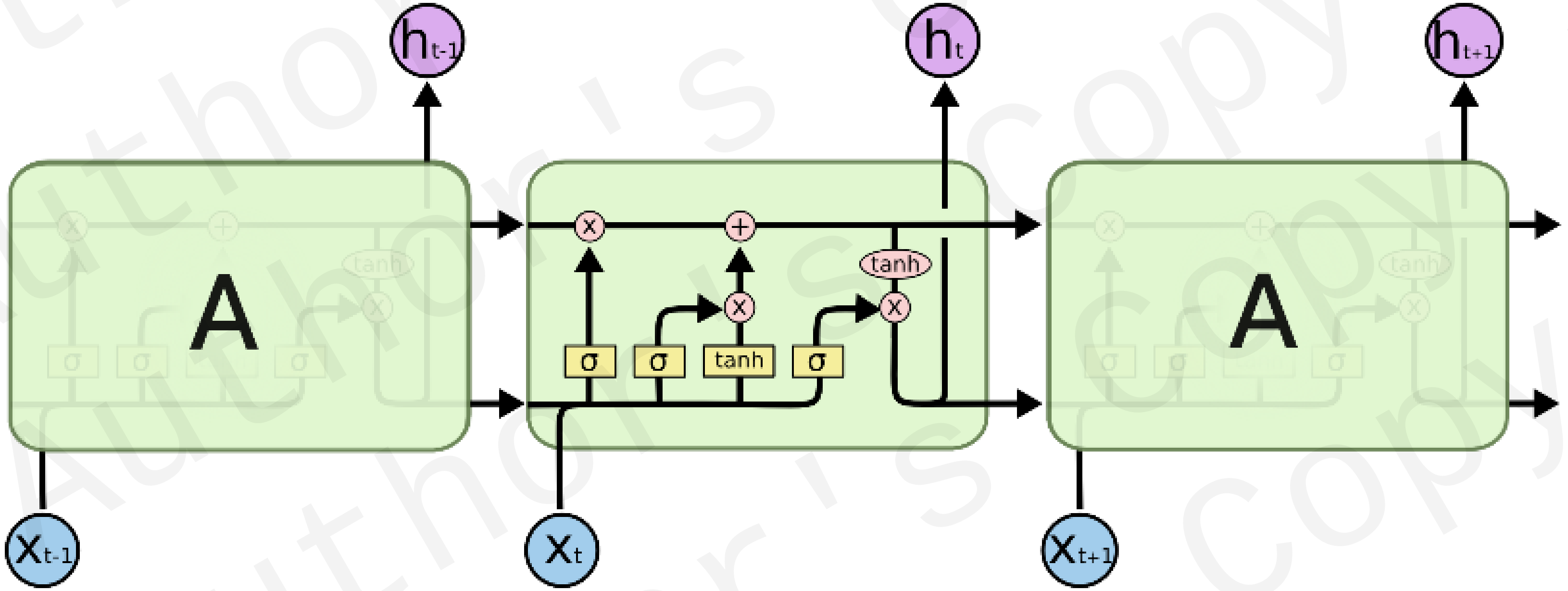
LSTMs (Hochreiter & Schmidhuber 1997)

Gated RNNs: let selective information through

$$h_t = f(h_{t-1}, x_t)$$
$$= \sigma(W^h h_{t-1} + W^x x_{t-1})$$

cleverly designed

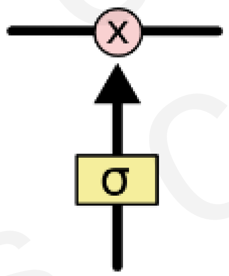
LSTMs:



LSTMs (Hochreiter & Schmidhuber 1997)

Gated RNNs: let selective information through

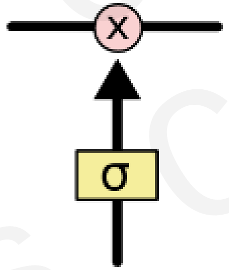
Gates:



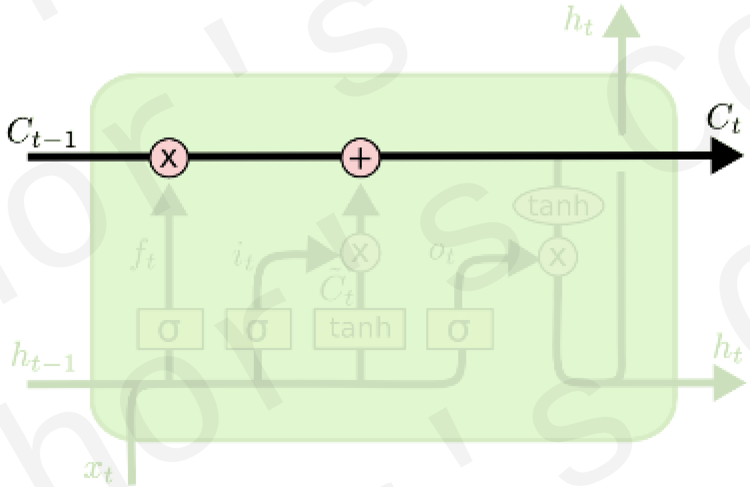
LSTMs (Hochreiter & Schmidhuber 1997)

Cell state: let selective information through

Gates:



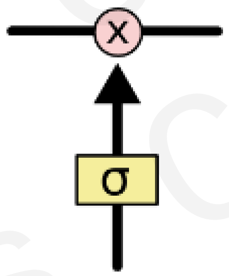
Cell state : Information highway.



LSTMs (Hochreiter & Schmidhuber 1997)

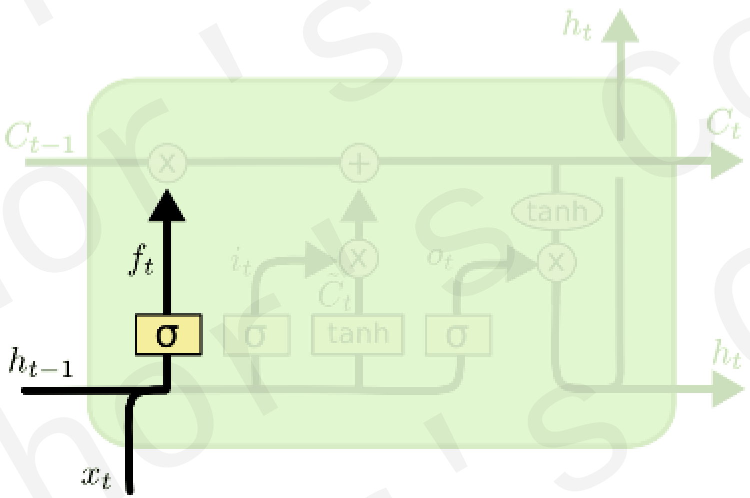
Cell state: let selective information through

Gates:



Cell state : Information highway.

1. Forget:

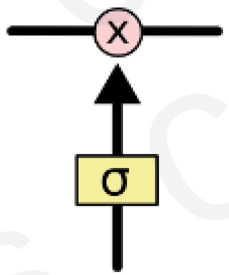


$$f_t = \sigma(W^f[h_{t-1}, x_t] + b_f)$$

LSTMs (Hochreiter & Schmidhuber 1997)

Cell state: let selective information through

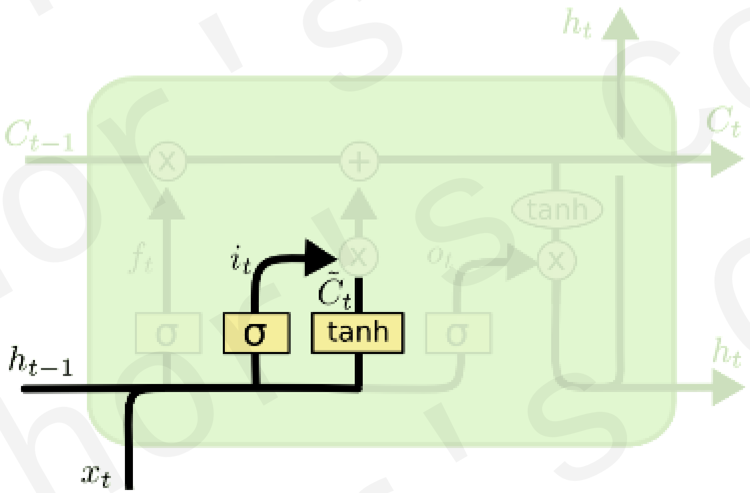
Gates:



Cell state : Information highway.

1. What to Forget:

2. what to Store:



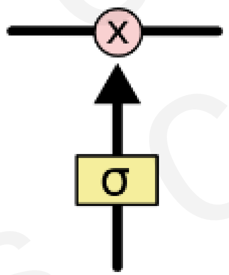
$$i_t = \sigma(W^i[h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W^C[h_{t-1}, x_t] + b_C)$$

LSTMs (Hochreiter & Schmidhuber 1997)

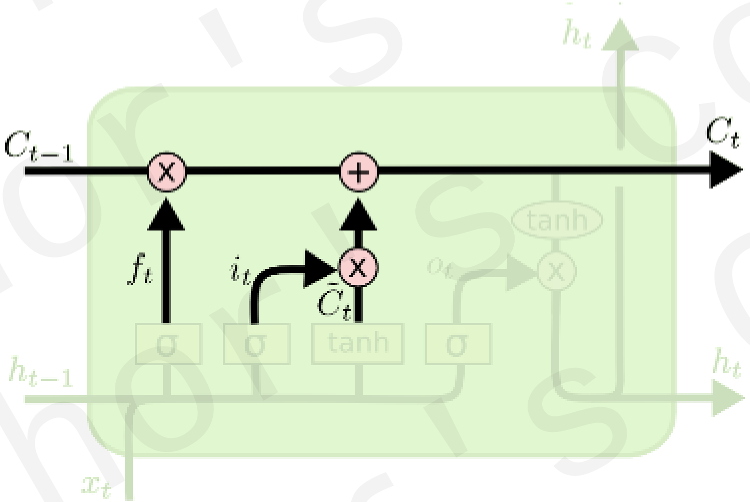
Cell state: let selective information through

Gates:



Cell state : Information highway.

1. What to Forget:
2. what to Store:
3. Update old cell state:

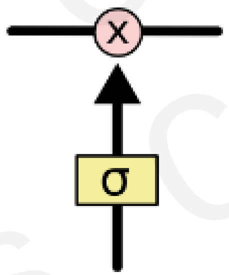


$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

LSTMs (Hochreiter & Schmidhuber 1997)

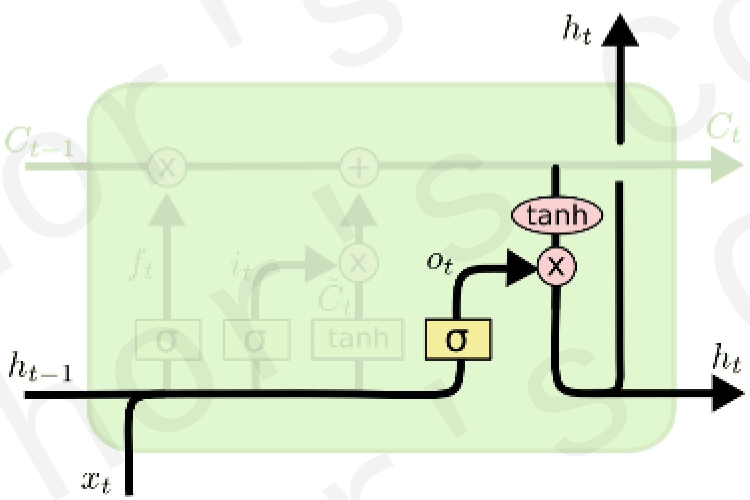
Cell state: let selective information through

Gates:



Cell state : Information highway.

1. What to Forget:
2. What to Store:
3. Update old cell state:
4. Generate output:



$$o_t = \sigma(W^o[h_{t-1}, x_t] + b_o)$$

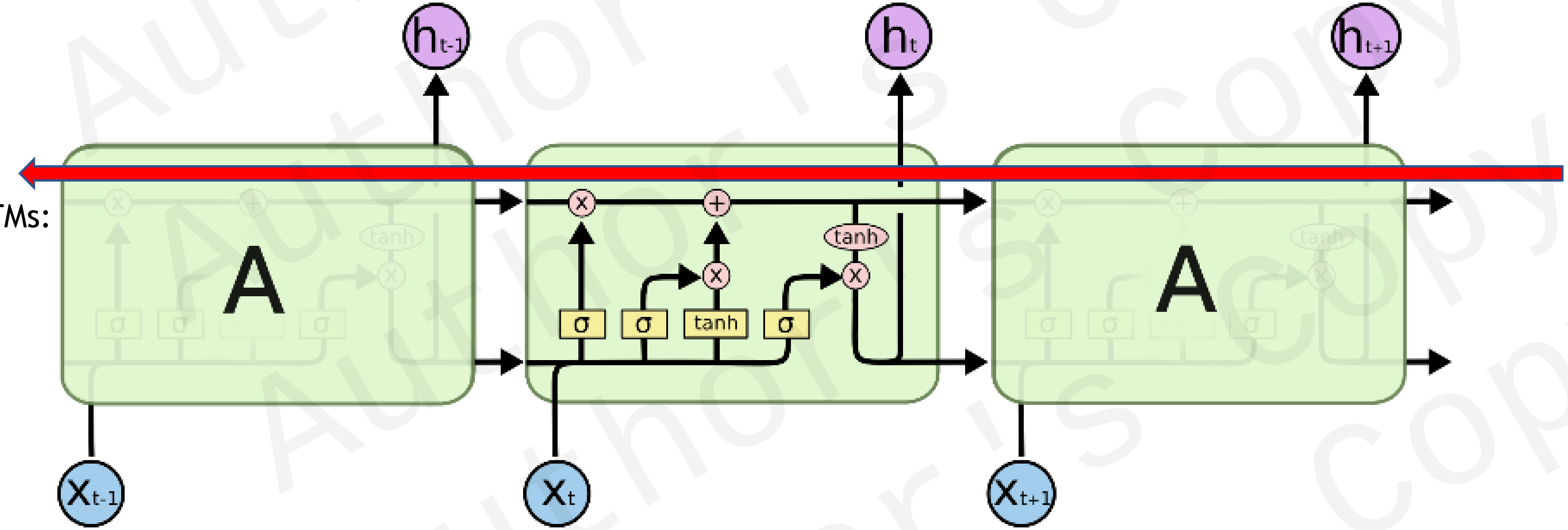
$$h_t = o_t \odot \tanh(C_t)$$

LSTMs (Hochreiter & Schmidhuber 1997)

Gated RNNs: let selective information through

Backpropagation: Uninterrupted gradient flow
Learning:
Faster than RNNs,
Long range dependency conserved..

LSTMs:



$$f_t = \sigma(W^f[C_{t-1}, h_{t-1}, x_t] + b_f)$$

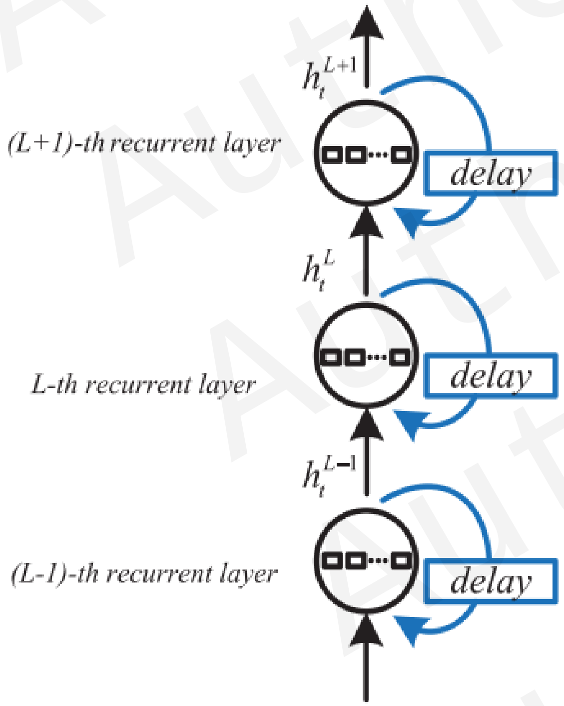
$$i_t = \sigma(W^i[C_{t-1}, h_{t-1}, x_t] + b_i)$$

$$o_t = \sigma(W^o[C_{t-1}, h_{t-1}, x_t] + b_o)$$

LSTM Variants:

- Peephole connections
- Gated Recurrent Units (GRUs) (Cho et al. 2014)
- etc.

Deep (Stacked) LSTMs (Fernández, Graves, & Schmidhuber,2007):



LSTM Variants:

- Peephole connections
- Gated Recurrent Units (GRUs) (Cho et al. 2014)
- etc.

Deep (Stacked)LSTMs (Fernández, Graves, & Schmidhuber,2007):

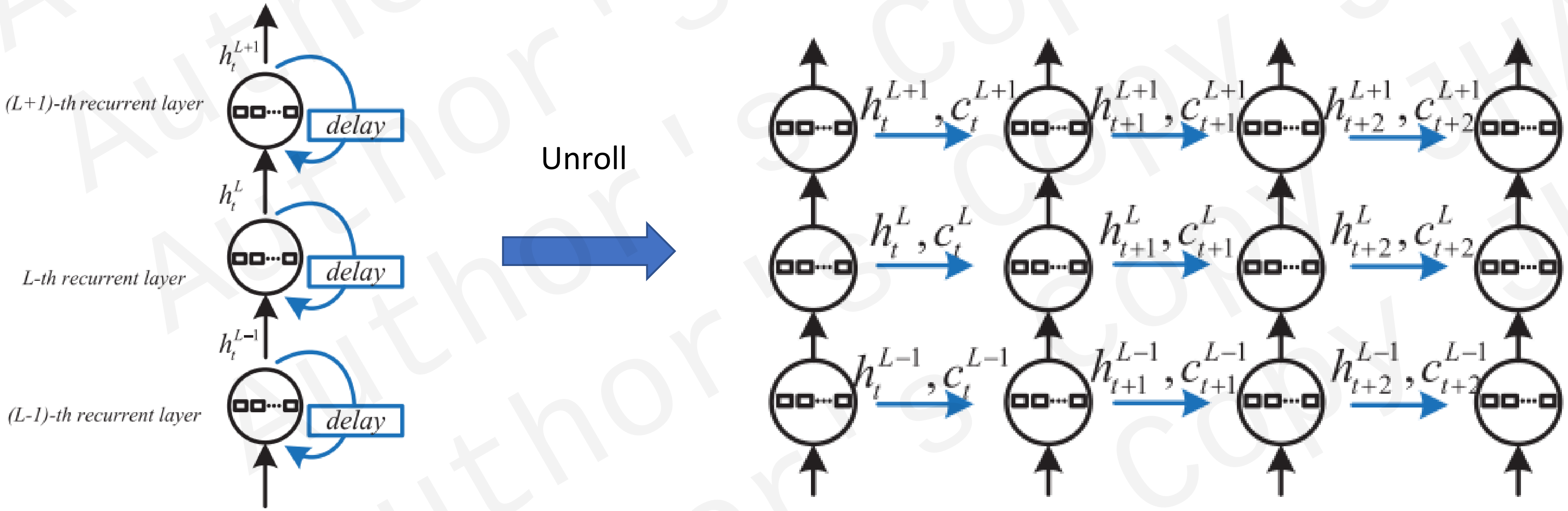


Image credits: Fernández, Graves, & Schmidhuber,2007

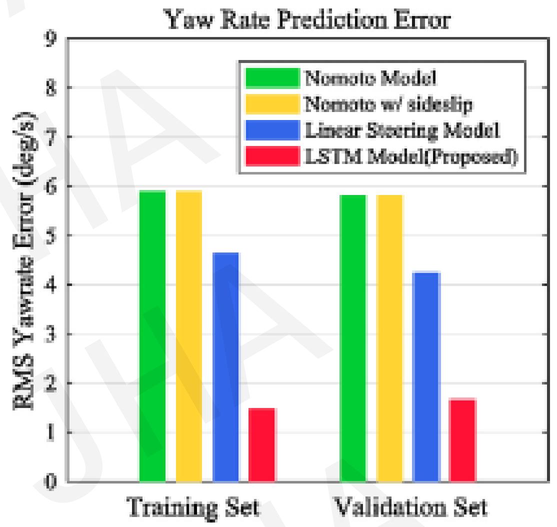
Deep LSTMs

- Advantages over RNNs:
  - Learn long term dependencies easily.
  - Avoid vanishing gradient problem through easy information flow.

- Replaced RNNs for Identification of Non-linear systems (dynamical systems).
  - Benchmarking performance LSTM > RNN > MLP > CNN (different datasets/ factors)

(A Richard et al. 2019)

Deep LSTMs



- Advantages over RNNs:
  - Learn long term dependencies easily.
  - Avoid vanishing gradient problem through easy information flow.

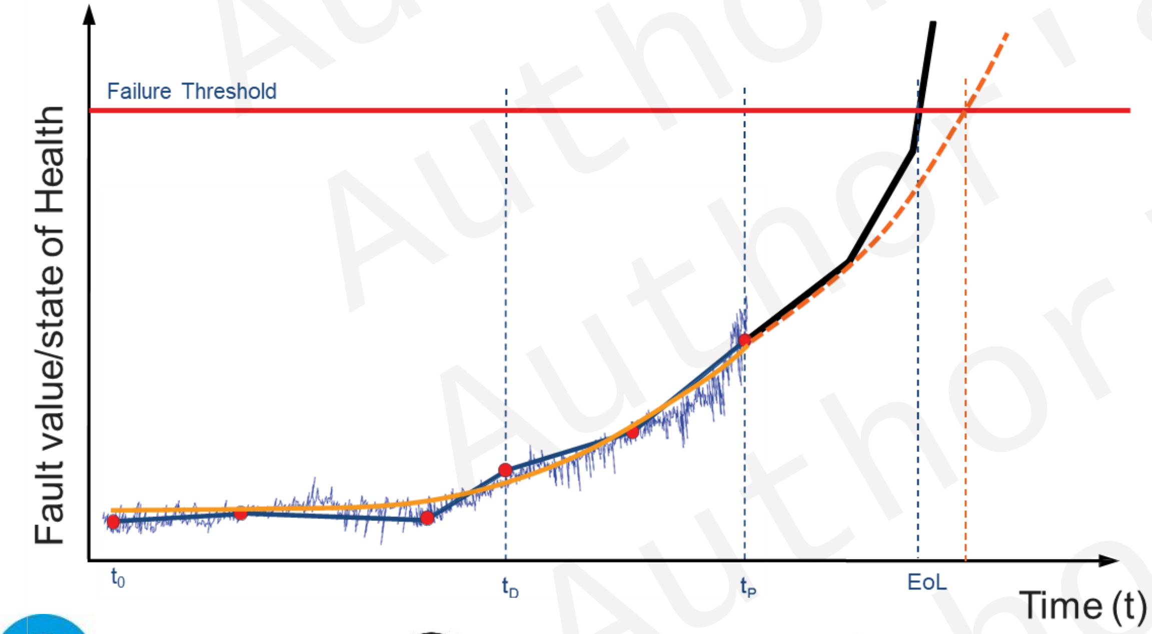
- Replaced RNNs for Identification of Non-linear systems (dynamical systems).
  - Benchmarking performance LSTM > RNN > MLP > CNN (different datasets/ factors) (A Richard et al. 2019)
  - Dynamic Model identification using Deep-LSTM : complex coupled non-linear effects Unmanned surface Vehicles (Woo et al, 2018) etc.
  - Learning Inverse dynamics for Robot control (Liu et al. 2019)
- Forecasting non-linear, non-stationary time series data : Short–long horizon.
- Limitations:
  - Deep LSTMs very long to train : 1000 sequence data → 1000 gradients to calculate!
  - Incoming New data → Perturbs the existing learning.

- Is this state of the art? → NO! surpassed by Attention-based mechanisms (2015) for LSTM (efficient learning, long range predictions,…)

Application:
Prognostics and Deep Learning

System degradation

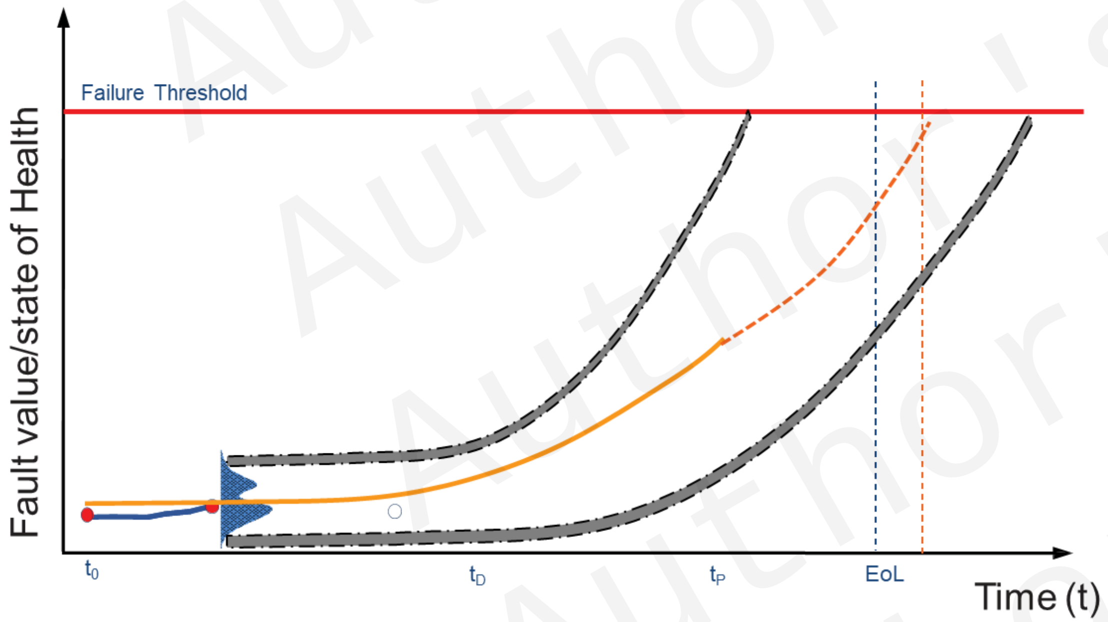
- Machines (dynamical systems) degrade with:
  - time
  - operational load cycles
  - operational conditions etc.



JHA Mayank , Email: mayank-shekhar.jha [at] univ-lorraine.fr

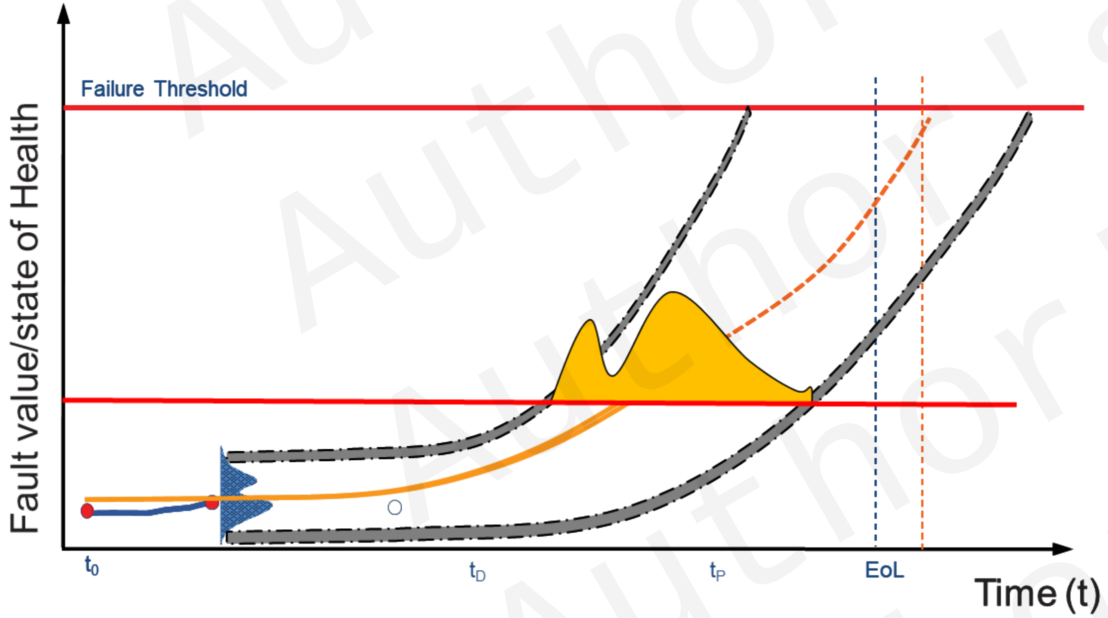
Prognostics

- Prognostics:
  - Estimate (state of health) → identification of degradation model.



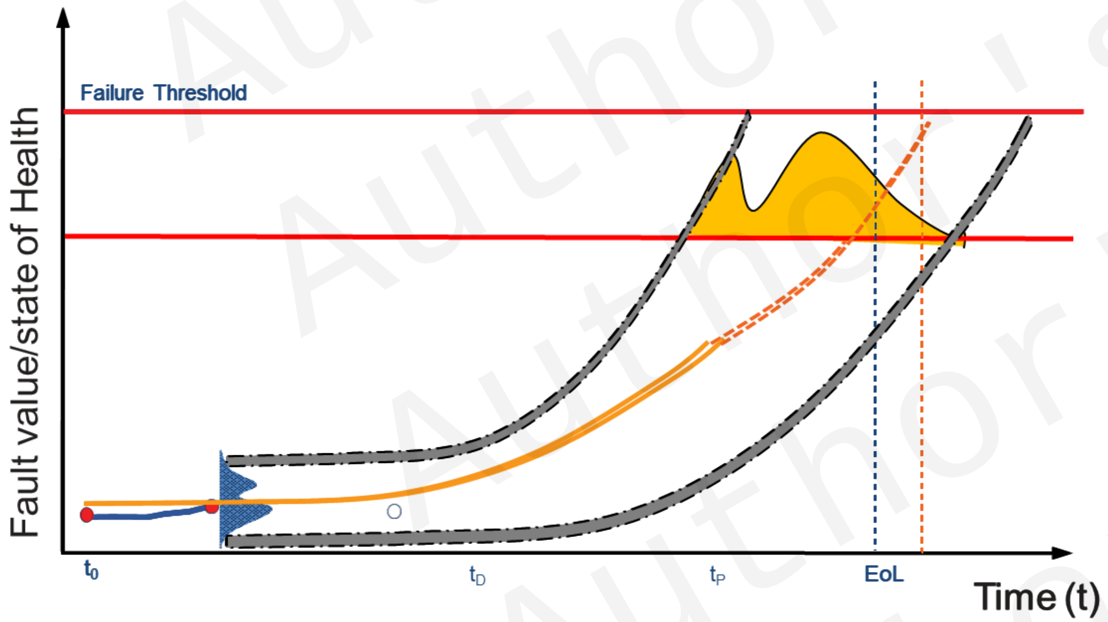
Prognostics

- Prognostics:
  - Estimate (state of health) → identification of degradation model.
  - Prediction of future health + Remaining Useful Life (RUL)



Prognostics

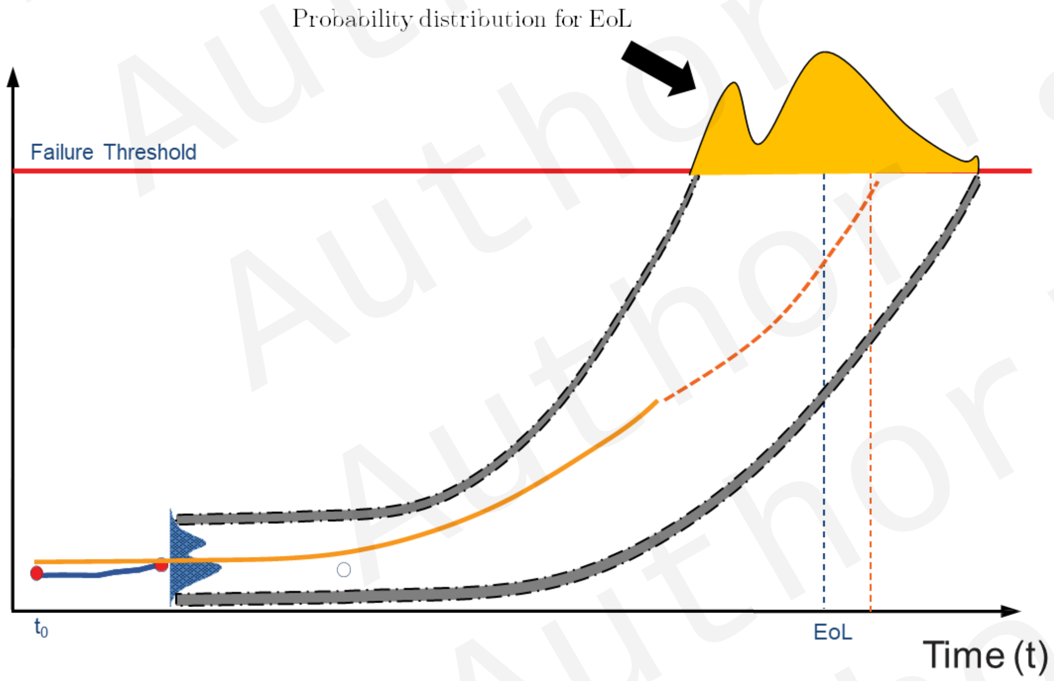
- Prognostics:
  - Estimate (state of health) → identification of degradation model.
  - Prediction of future health + Remaining Useful Life (RUL)



JHA Mayank , Email: mayank-shekhar.jha [at] univ-lorraine.fr

Prognostics

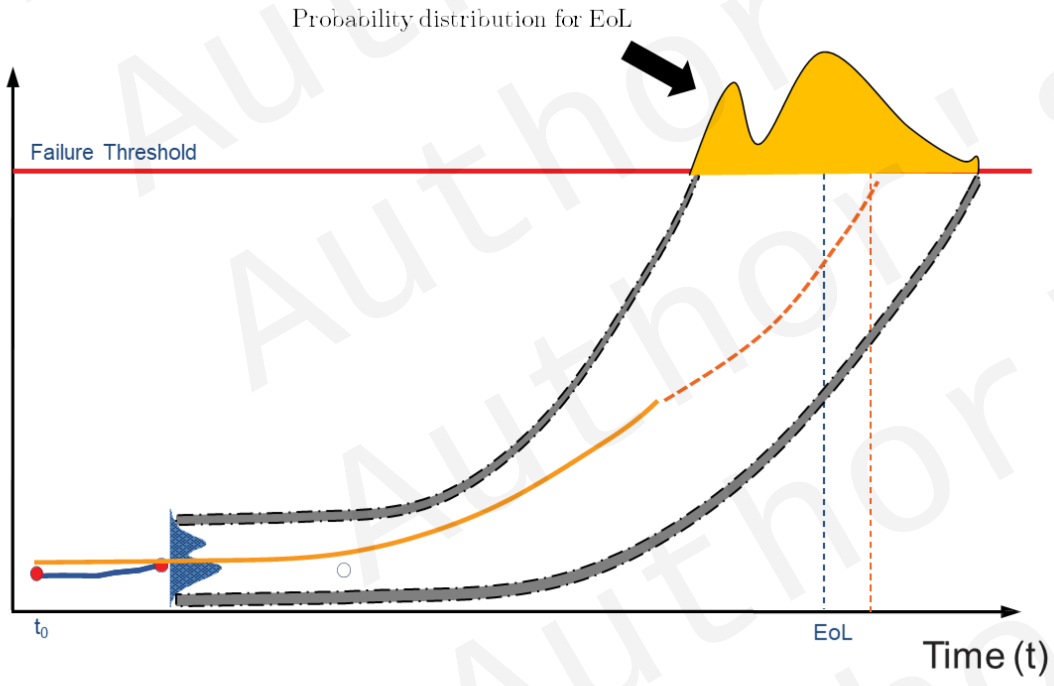
- Prognostics:
  - Estimate (state of health) → identification of degradation model.
  - Prediction of future health + Remaining Useful Life (RUL)

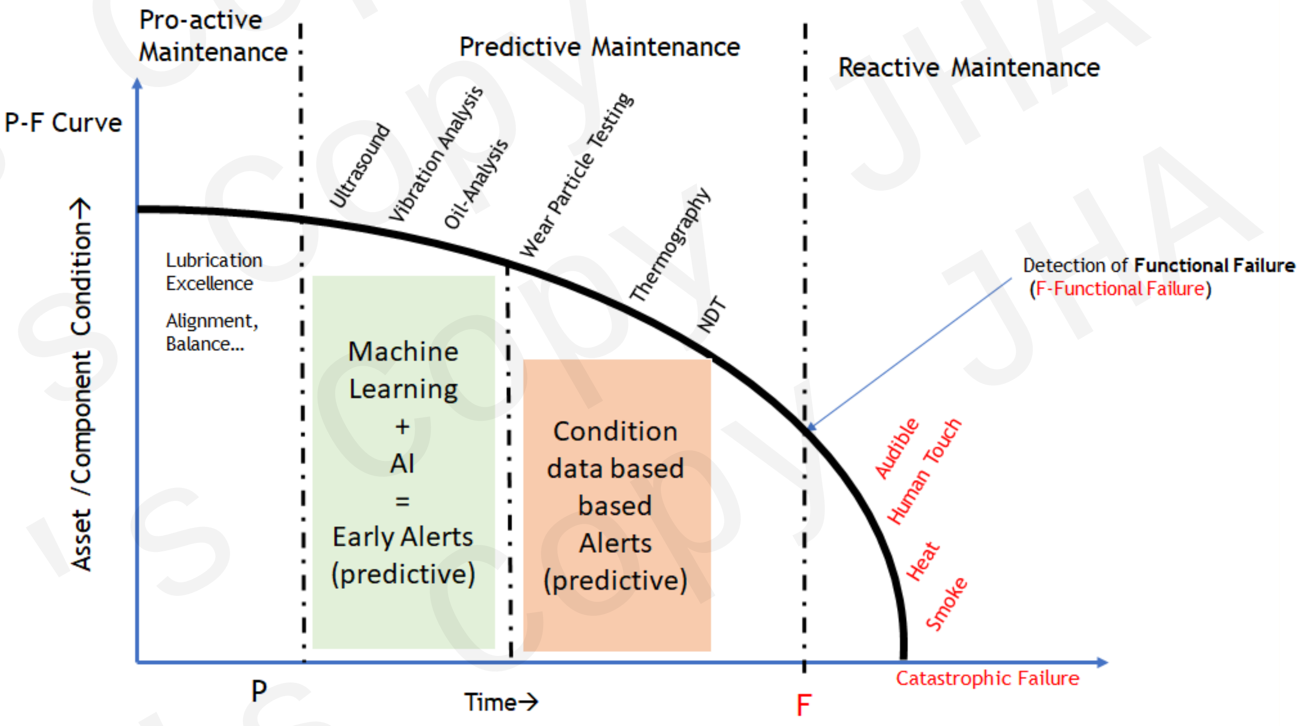


JHA Mayank , Email: mayank-shekhar.jha [at] univ-lorraine.fr

Prognostics

- Prognostics:
  - Estimate (state of health) → identification of degradation model.
  - Prediction of future health + **Remaining Useful Life (RUL)**
  - Evaluate: Decision “when failure occurs ???” “what maintenance strategy”





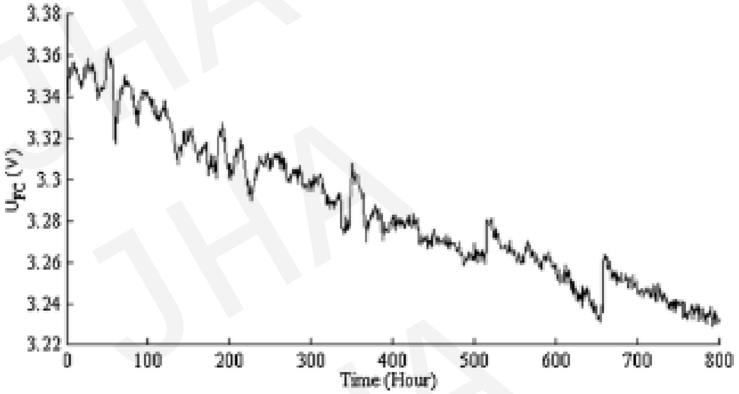
JHA Mayank , Email: mayank-shekhar.jha [at] univ-lorraine.fr

Degradation Data

- Degradation:
  - unknown, non-linear varying dynamics
  - sensor data: non-stationary process → trend, seasonality, cyclic etc.
  - depends on qualitative+ quantitative factors.

Degradation Data

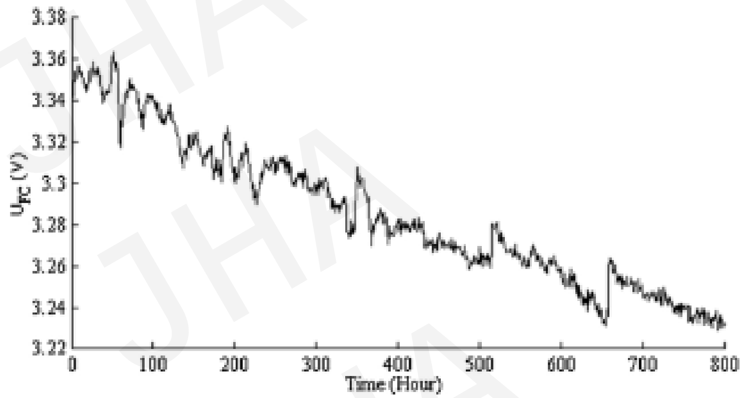
- Degradation:
  - unknown, non-linear varying dynamics
  - sensor data: non-stationary → trend, seasonality, cyclic etc.
  - depends on qualitative+ quantitative factors.



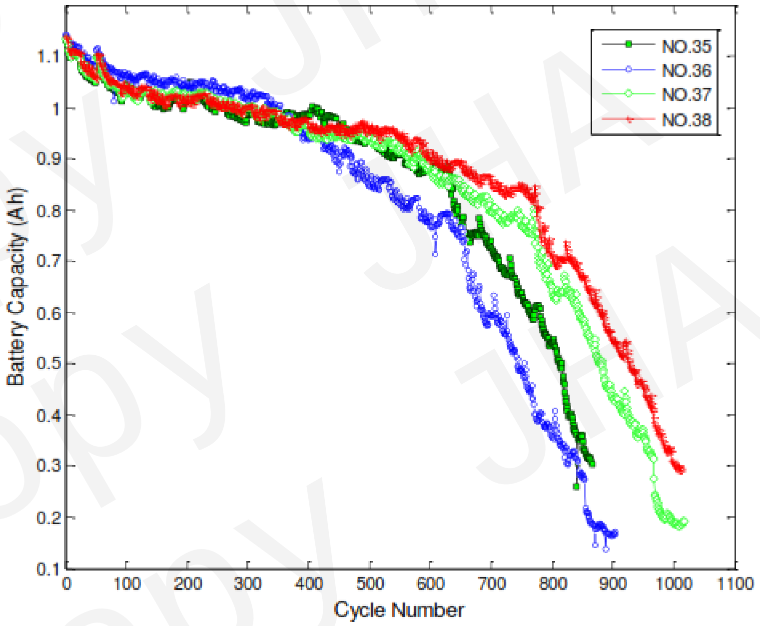
PEM Fuel Cell degradation (Jha et al. 2016)

Degradation Data

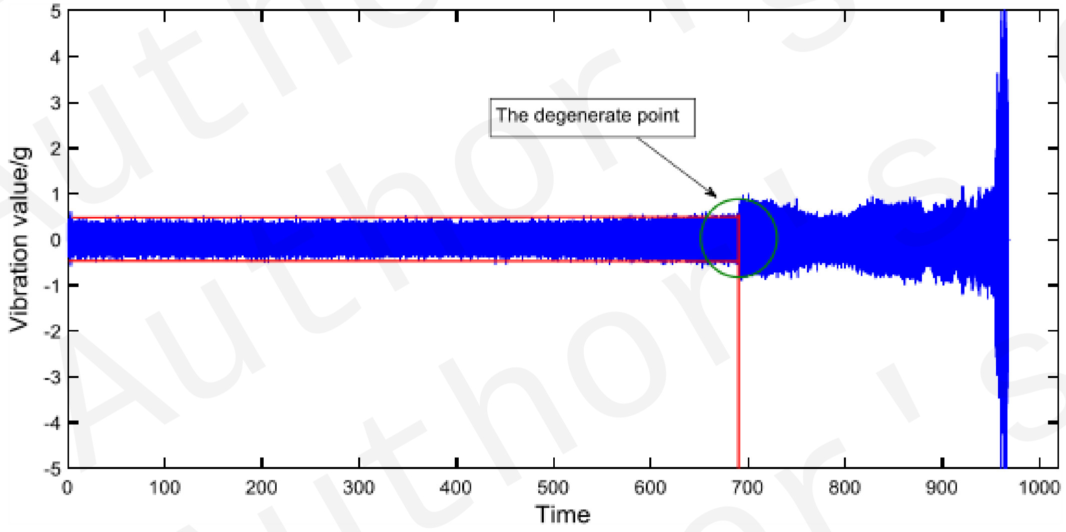
- Degradation:
  - unknown, non-linear varying dynamics
  - sensor data: non-stationary → trend, seasonality, cyclic etc.
  - depends on qualitative+ quantitative factors.



PEM Fuel Cell degradation (Jha et al. 2016)



Lithium-ion battery degradation,
Center for Advanced Life Cycle Engineering (CALCE)
in University of Maryland (He W., Williard N., Osterman M., & Pecht M., 2011)

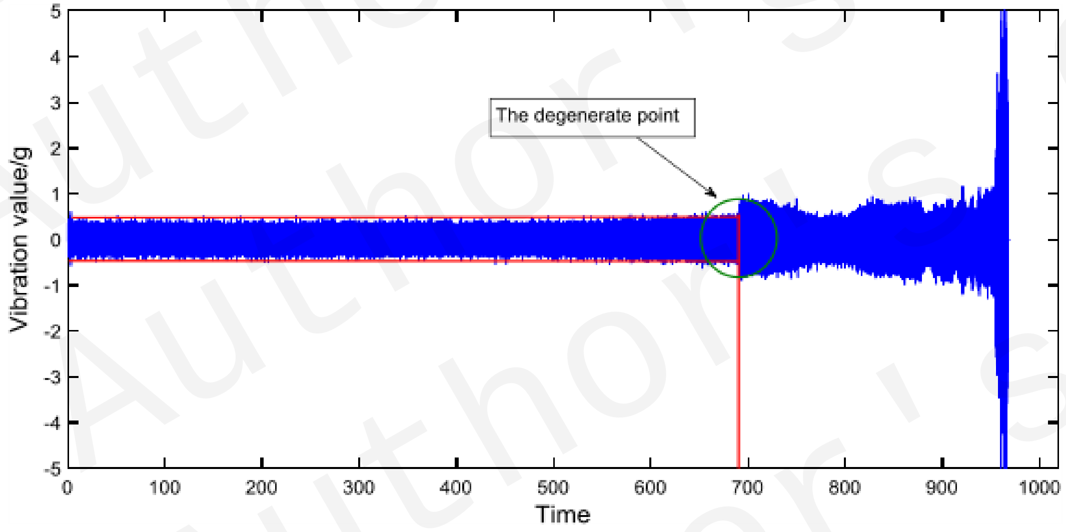


Roller bearing degradation (PRONOSTIA platform)

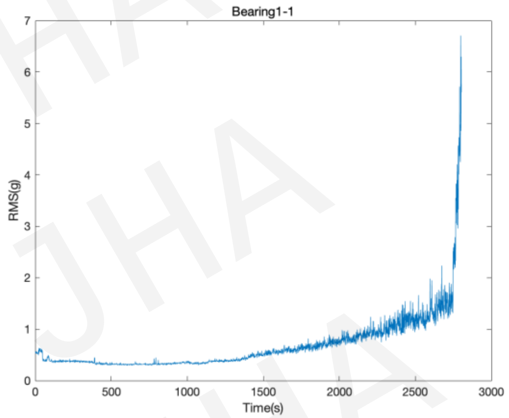
Degradation Data

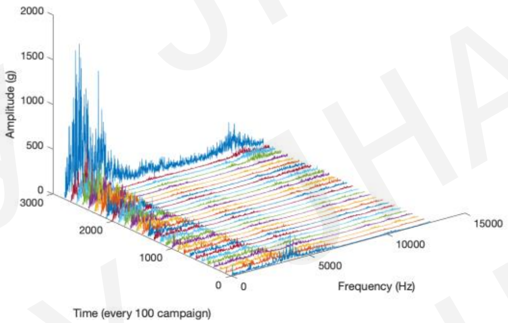
- Degradation:
    - unknown, non-linear varying dynamics
    - sensor data: non-stationary process → trend, seasonality, cyclic etc.
    - depends on qualitative+ quantitative factors.

- Raw degradation data → Hidden features / representation:
    - Spatially varying
    - Temporally varying
    - Multimodal characteristics



Roller bearing degradation (PRONOSTIA platform)





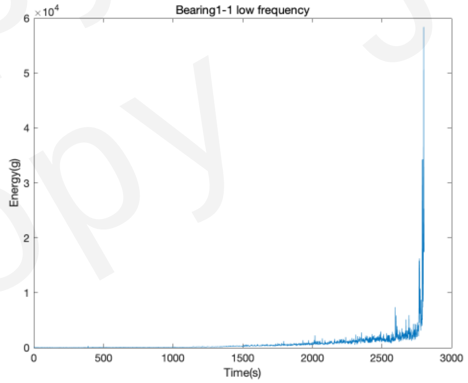


Photo: Report of Jha

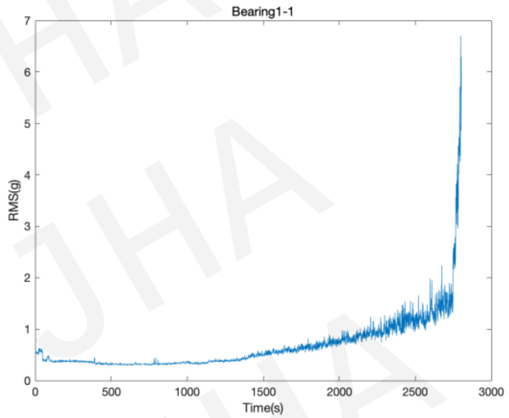
Degradation Data

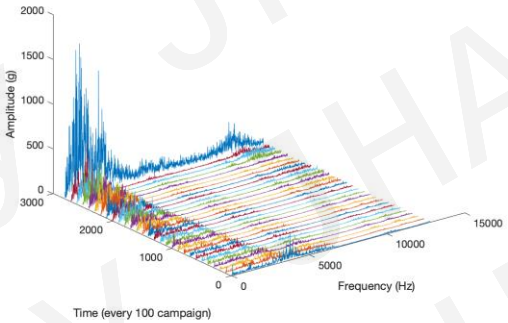
- Degradation:
  - unknown, non-linear varying dynamics
  - sensor data: non-stationary process → trend, seasonality, cyclic etc.
  - depends on qualitative+ quantitative factors.

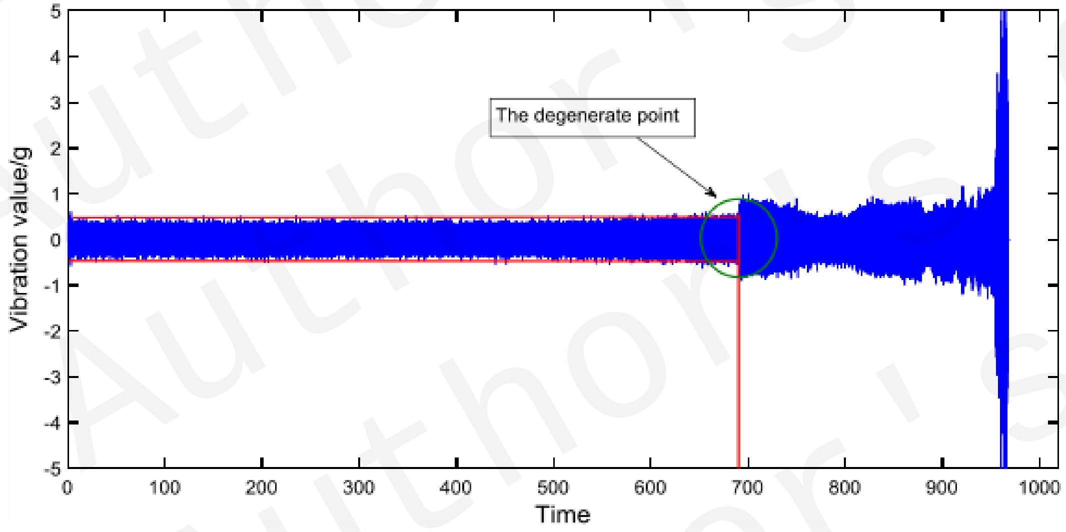
Deep LSTMs

- Raw degradation data → Hidden features / representation:
  - Spatially varying
  - Temporally varying
  - Multimodal characteristics

CNNs







Roller bearing degradation (PRONOSTIA platform)

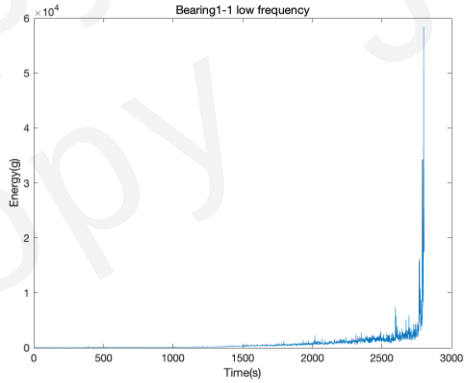
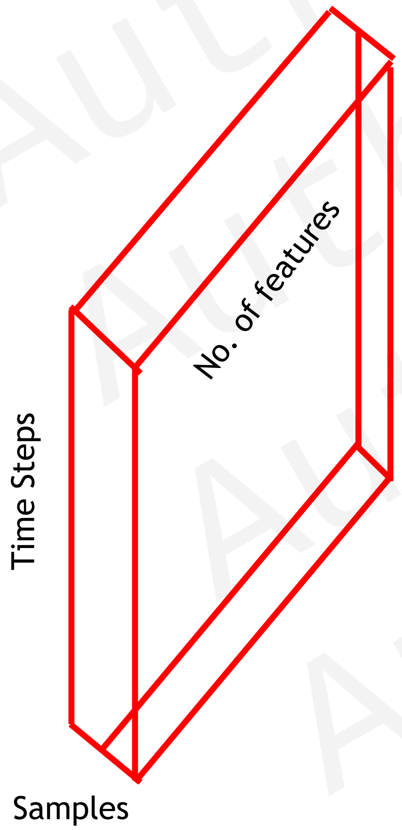


Photo: Report of Jha

Deep LSTMs for Prognostics

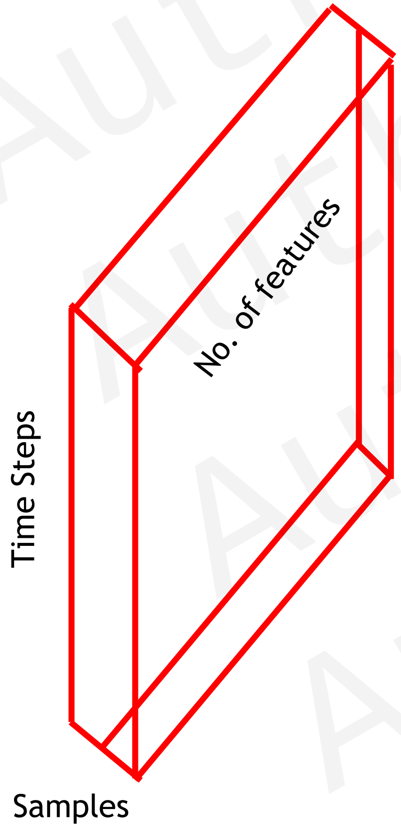
Basic Architecture



3D- Input

Deep LSTMs for RUL prediction

Basic Architecture



3D- Input

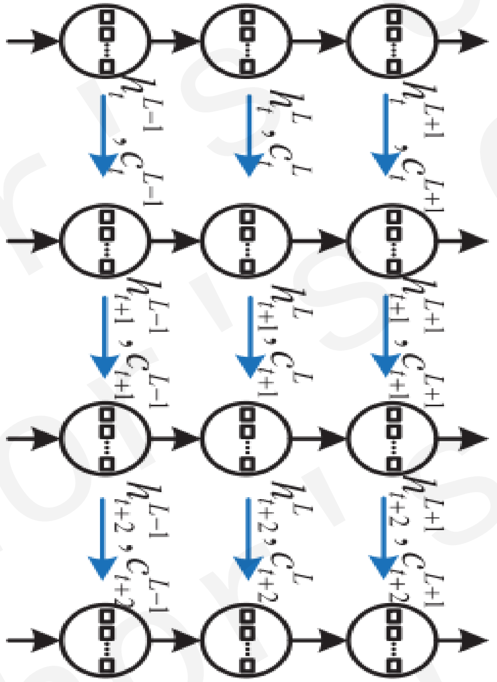
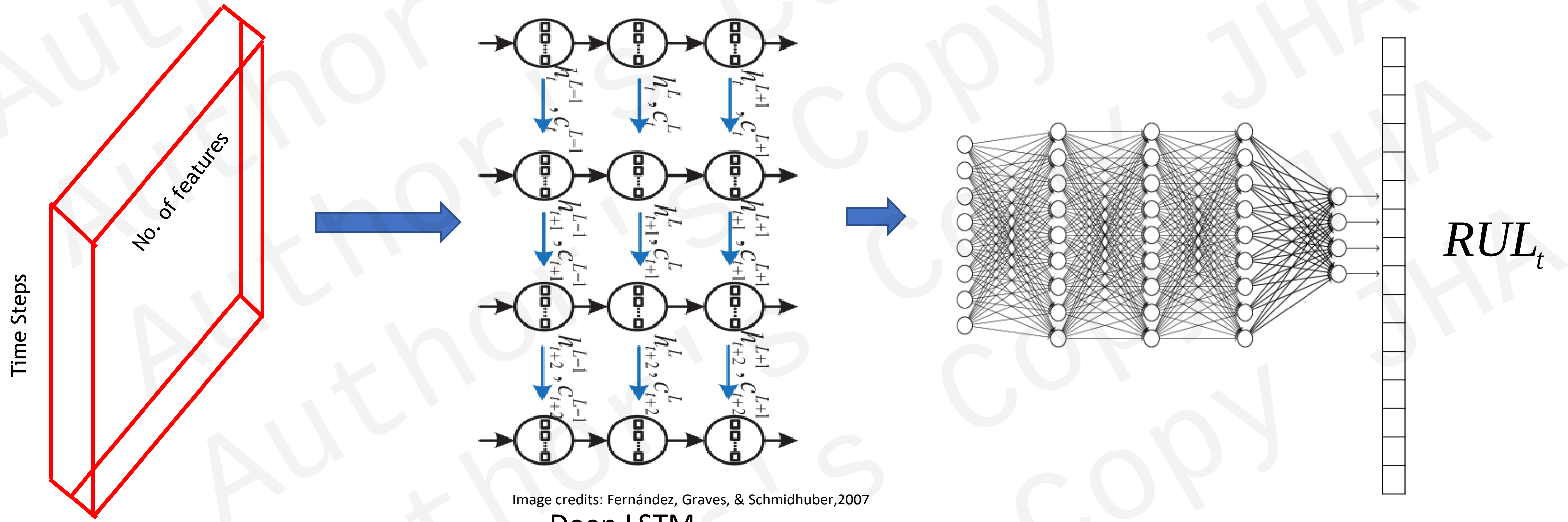


Image credits: Fernández, Graves, & Schmidhuber,2007

Deep LSTM
+ dropout schemes

Deep LSTMs for RUL prediction

Basic Architecture: LSTMs: Temporal features + FNNs: Map features in RULs



Samples

3D- Input

Image credits: Fernández, Graves, & Schmidhuber,2007

Deep LSTM
+ dropout schemes

Fully connected Layer

Target Vector

Deep LSTMs for RUL prediction

- Degradation data➔ Time Series sequence ➔ segmented into sliding windows.
- Each sliding window is assigned a target RUL value [Zeng et al, 2017]

$X = [X_1, X_2, \ldots, X_t, \ldots X_{T-1}]$ to estimate RUL_{T-1}

$X = [X_1, X_2, \ldots, X_t, \ldots X_{T-2}]$ to estimate RUL_{T-2}

Deep LSTMs for RUL prediction

- Degradation data➔ Time Series sequence ➔ segmented into sliding windows.
- Each sliding window is assigned a target RUL value [Zeng et al, 2017]

$X = [X_1, X_2, \ldots, X_t, \ldots X_{T-1}]$ to estimate RUL_{T-1}

$X = [X_1, X_2, \ldots, X_t, \ldots X_{T-2}]$ to estimate RUL_{T-2}

Many variants exist!

$[X_t, X_{t-1}, \ldots, X_{t-d+1}], \epsilon \mathbb{R}^d$

$[\mathrm{RUL}_{t+L}, \mathrm{RUL}_{t+L+1}, \ldots, \mathrm{RUL}_n]$

Training tuples:

$([X_t, X_{t-1}, \ldots, X_{t-d+1}], \mathrm{RUL}_{t+L})$

$\widehat{\mathrm{RUL}_{t+L}} = \phi(X_t, X_{t-1}, \ldots, X_{t-d+1})$

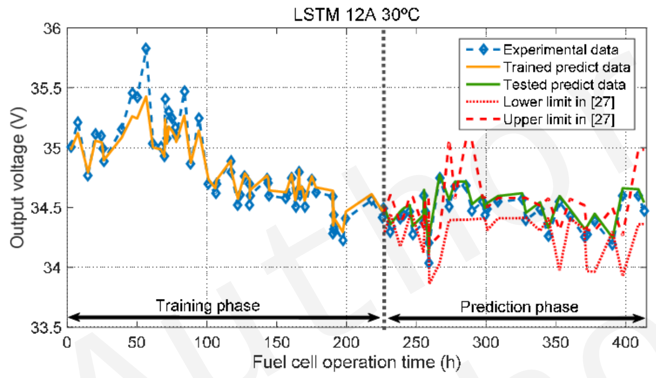
Loss Calculation : Error based cost function

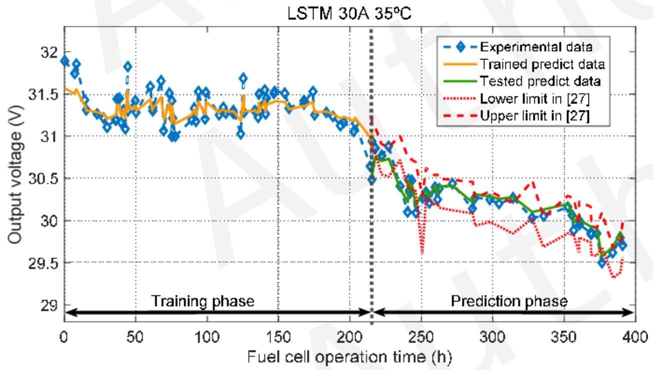
$$J = \sum_t \|(RUL_{est}^t - RUL_{calc}^t\|^2$$

Some issues:

- Independent Windows → to assure assumption of i.i.d
- Dependent windows → claim more realistic.

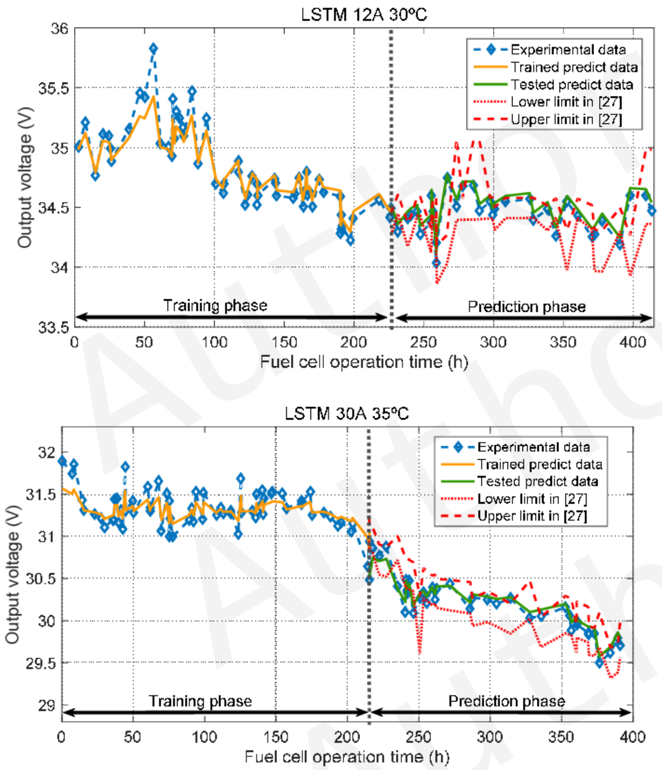
Some applications:



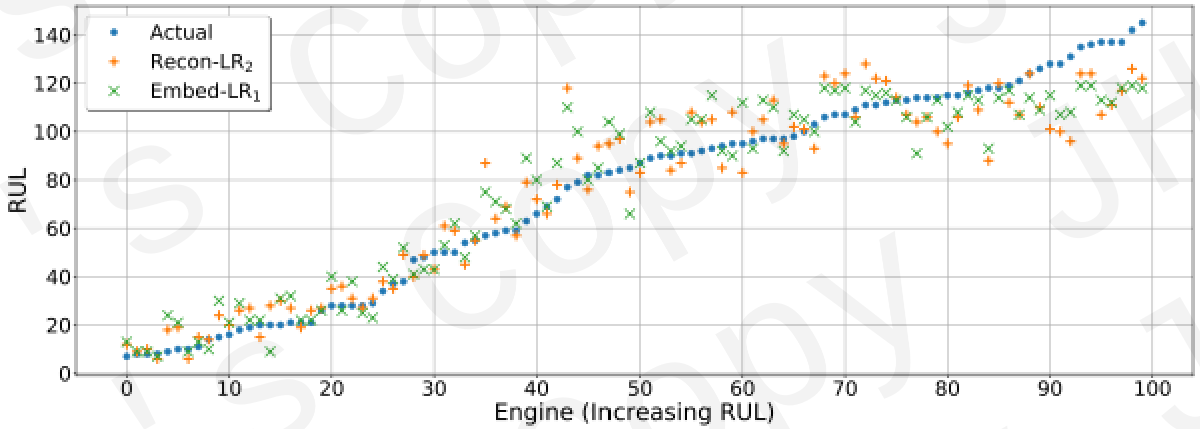


PEM Fuel Cell degradation

Some applications:

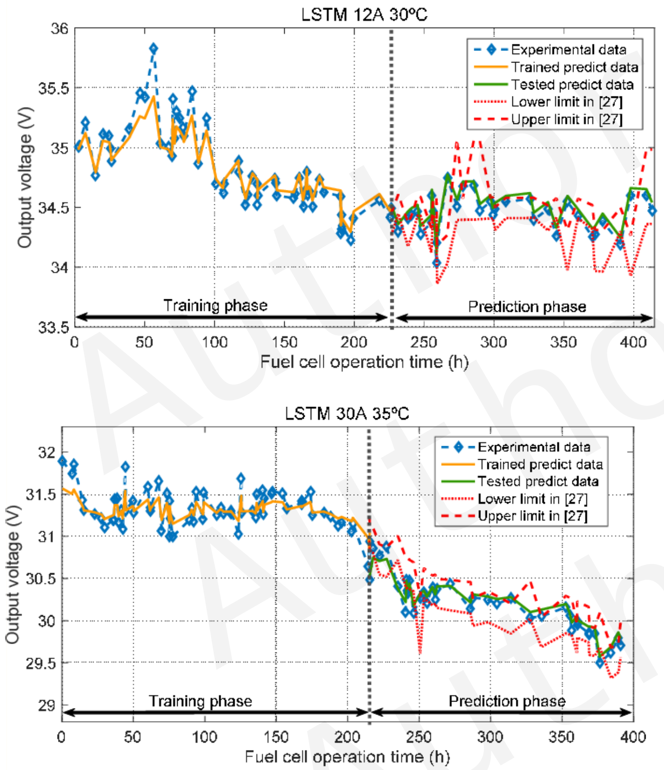


PEM Fuel Cell degradation

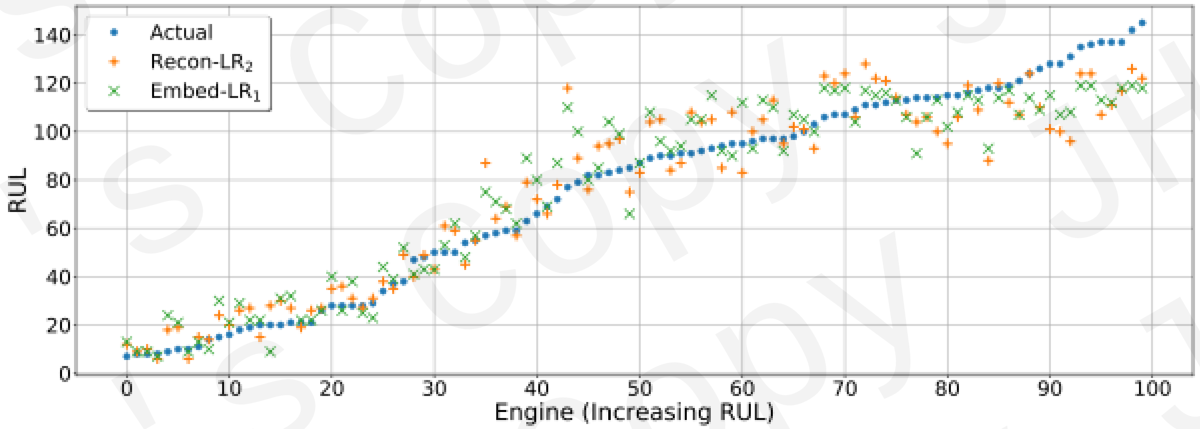


[Gugulothu et al 2017]

Some applications:



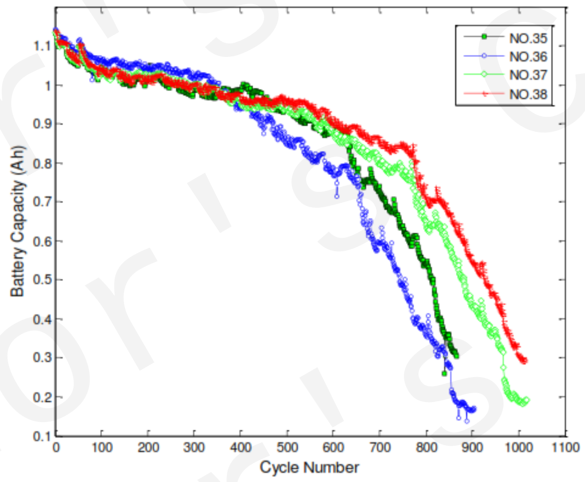
PEM Fuel Cell degradation



[Gugulothu et al 2017]

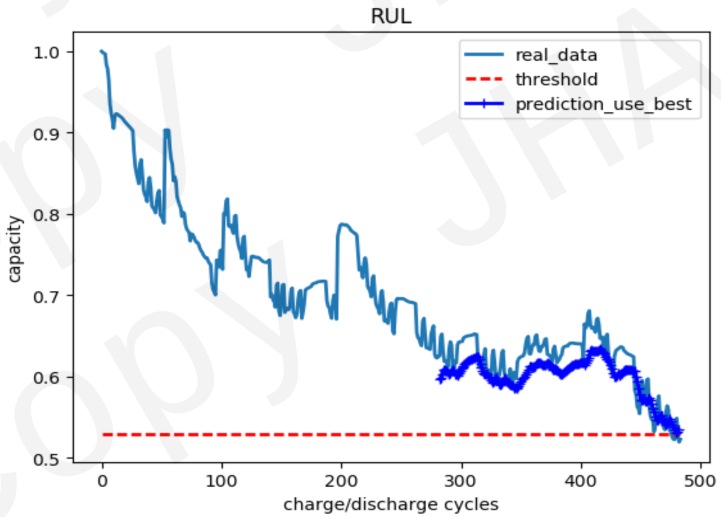
Engine prognostics (NASA) : CMAPSS
'Commercial Modular Aero-Propulsion System Simulation'
[Zhang et al, 2017]

- unknown non-linear dynamics,
- non-stationary (multi modal degradation,
- multiple modes of degradation)



Lithium-ion battery RUL prediction

(He W., Williard N., OstermanM., & Pecht M., 2011)



CNNs for Prognostics

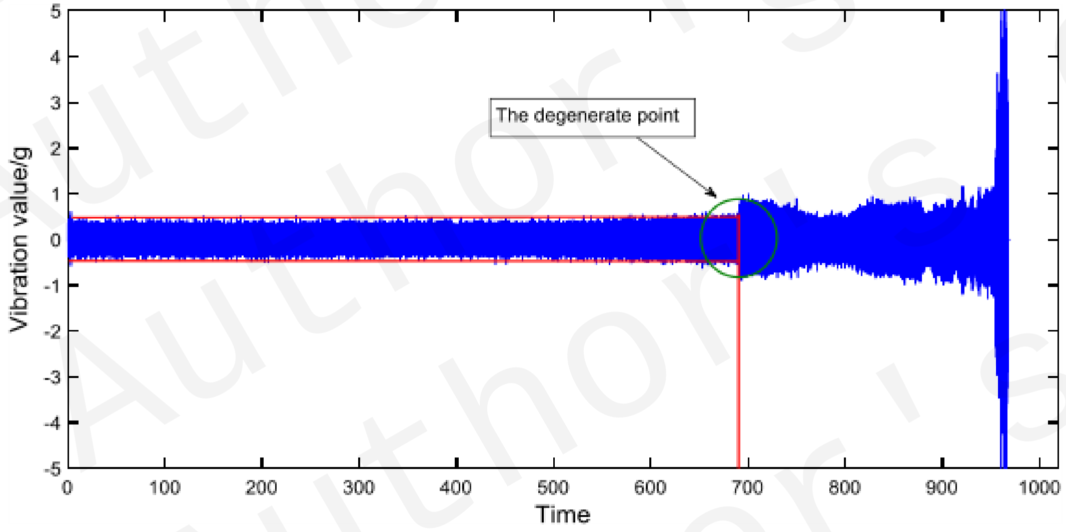
- LSTMs: good sequence learning

but good input sequence needs to be provided!!

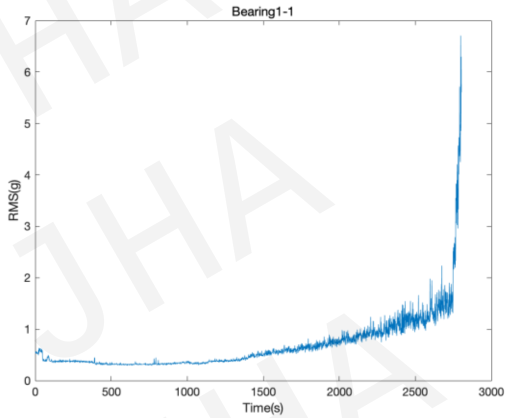
- Feature extraction needs domain knowledge.
- Labelled data → difficult !

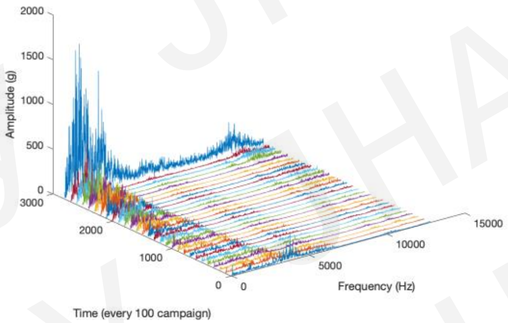
- CNNs → Hidden features / representation of sequence:
  - Spatially varying
  - Temporally varying
  - Multimodal characteristics

CNNs



Roller bearing degradation (PRONOSTIA platform)





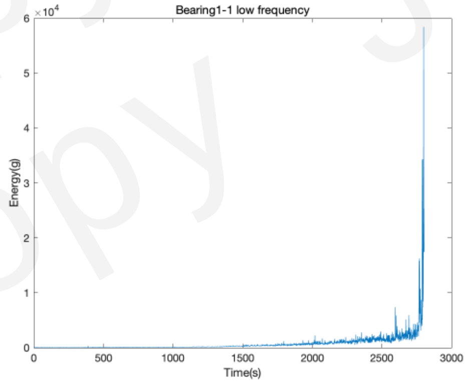
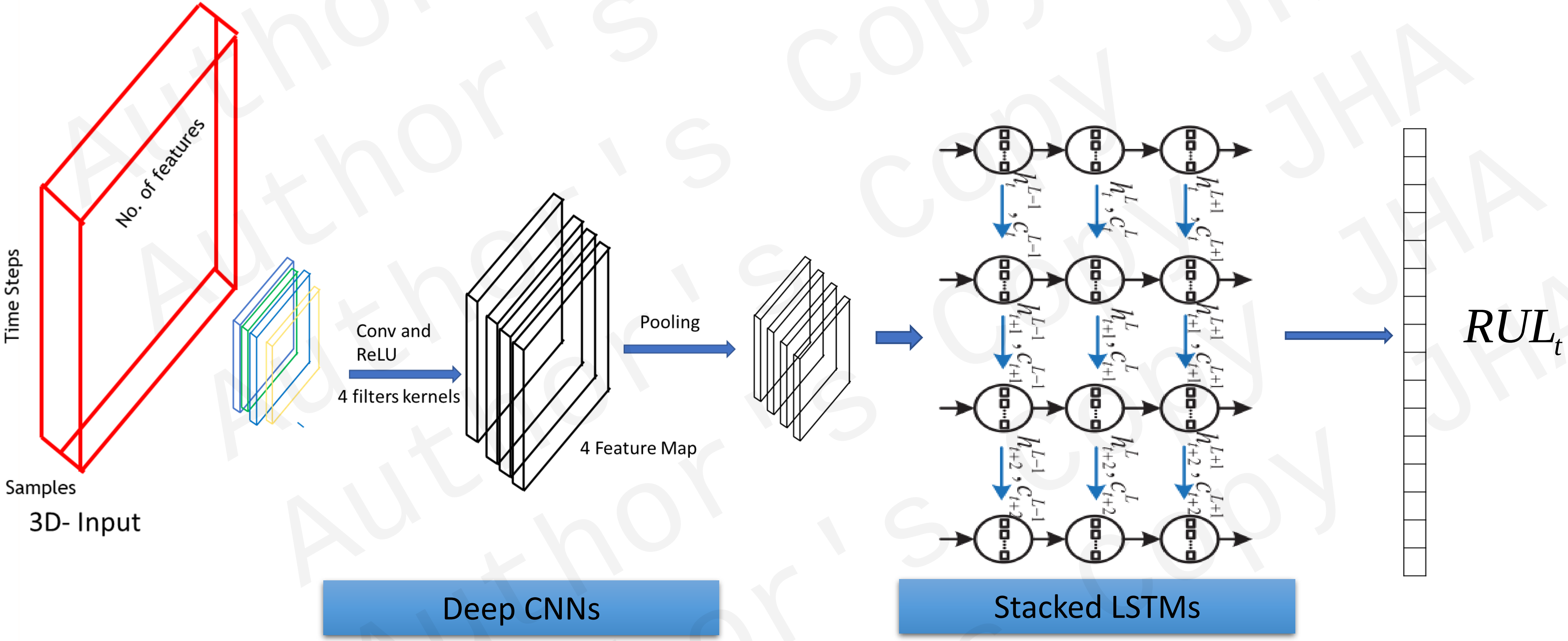


Photo: Report of Jha

JHA Mayank , Email: mayank-shekhar.jha [at] univ-lorraine.fr

CNNs for Prognostics

- CNNs → Traditionally, 2D-3D structured data for face/object recognition
- Prognostics → 3D structured topology for sequence data



CNNs for Prognostics

- Automatically learn feature representation, hidden multimodal distributions

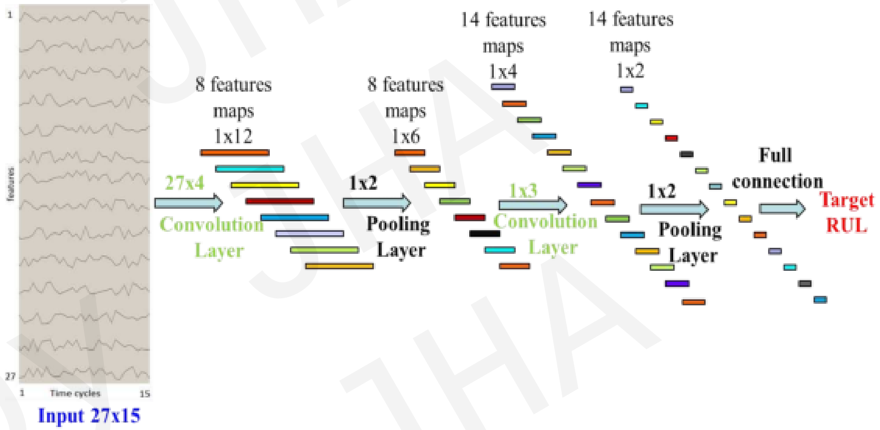
[Liu et al., 2017] [Jing et al., 2017] [Li et al., 2018]

&

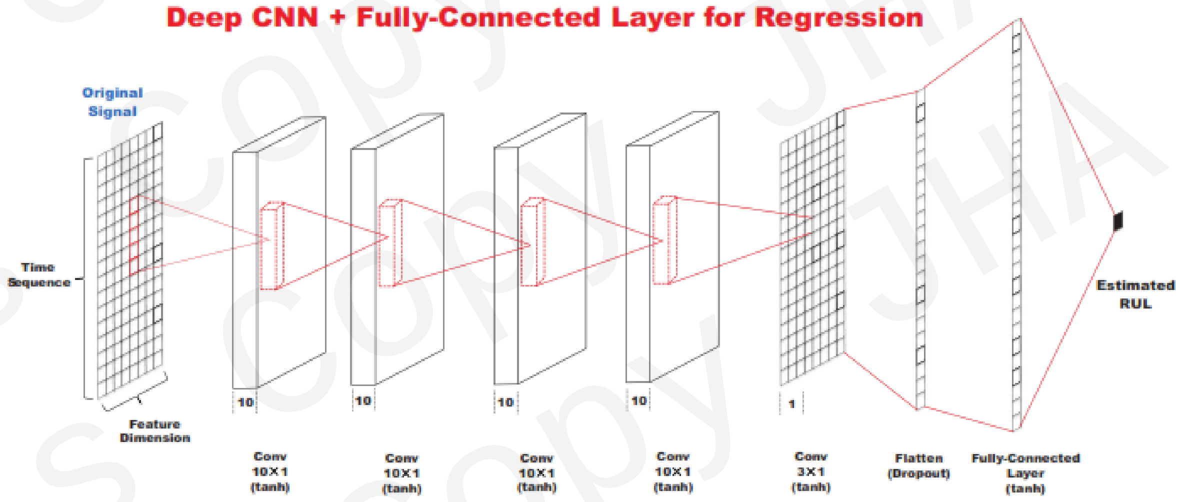
- Efficient learning with multi-variate sequential (time series) data.

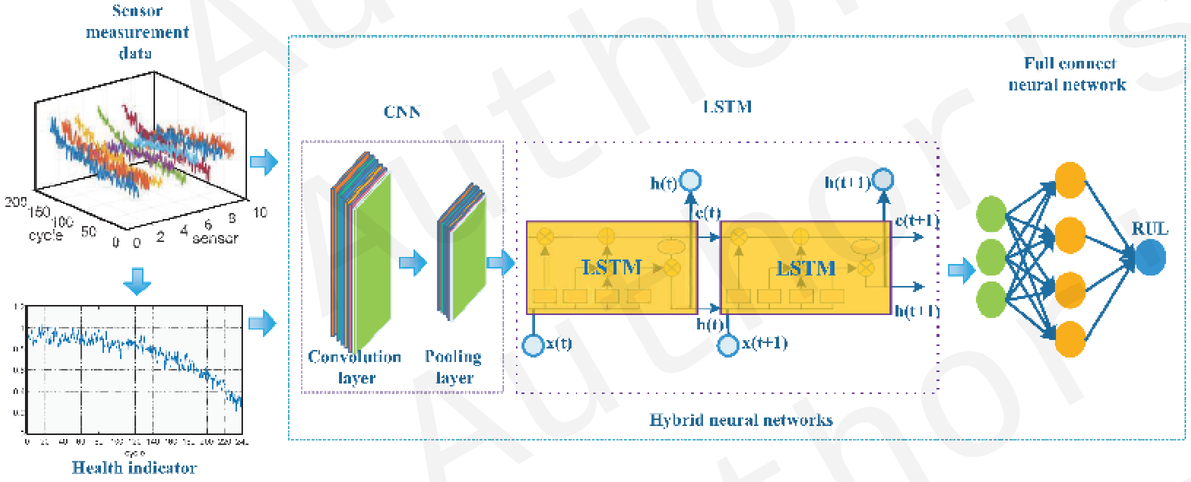
[Babu et al., 2016]

- Hybrid structure



[Babu et al., 2016]





[Liu et al., 2017]

[Kong et al. 2019]

Bibliography

- Learning algorithms for classification: A comparison on handwritten digit recognition. Neural networks Stat Mech Perspect 261:276
- Simonyan K, Zisserman A (2015) VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION. ICLR 75:398–406. doi: 10.2146/ajhp170251
- Lin M, Chen Q, Yan S (2013) Network In Network. 1–10. doi: 10.1109/ASRU.2015.7404828
- He K, Zhang X, Ren S, Sun J (2015) Deep Residual Learning for Image Recognition. Multimed Tools Appl 77:10437–10453. doi: 10.1007/s11042-017-4440-4
- Khan, A., Sohail, A., Zahoora, U. *et al.* A survey of the recent architectures of deep convolutional neural networks. *Artif Intell Rev* **53,** 5455-5516 (2020)
- Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. pp 249–256
- Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.
- Richard, A., Mahé, A., Pradalier, C., Rozenstein, O., & Geist, M. (2019). A Comprehensive Benchmark of Neural Networks for System Identification.
- Woo, J., Park, J., Yu, C., & Kim, N. (2018). Dynamic model identification of unmanned surface vehicles using deep learning network. *Applied Ocean Research*, *78*, 123-133.

Bibliography

- Rueckert, E., Nakatenus, M., Tosatto, S., & Peters, J. (2017, November). Learning inverse dynamics models in o (n) time with lstm networks. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)* (pp. 811-816). IEEE.
- Fernández, S., Graves, A., & Schmidhuber, J. (2007b). An application of recurrent neural networks to discriminative keyword spotting. In *Proceedings of the International Conference on Artificial Neural Networks (pp. 220–229). Berlin: Springer*
- Liu, N., Li, L., Hao, B., Yang, L., Hu, T., Xue, T., & Wang, S. (2019). Modeling and simulation of robot inverse dynamics using LSTM-based deep learning algorithm for smart cities and factories. *IEEE Access*, *7*, 173989-173998.
- Gugulothu, N., TV, V., Malhotra, P., Vig, L., Agarwal, P., & Shroff, G. (2017). Predicting remaining useful life using time series embeddings based on recurrent neural networks. *arXiv preprint arXiv:1709.01073*.
- *Systems*. Springer, Cham, 2017. 233-270Jha, Mayank-Shekhar; 2017. "Algorithm Architectures for Intelligent Communications, Control and Monitoring Systems, **Rolls Royce University Technology Centre** Sheffield."
- Jha, Mayank Shekhar, et al. "Particle filter based hybrid prognostics of proton exchange membrane fuel cell in bond graph framework." *Computers & Chemical Engineering* 95 (2016): 216-230.
- Jha, Mayank S., Geneviève Dauphin-Tanguy, and B. Ould-Bouamama. "Particle Filter Based Integrated Health Monitoring in Bond Graph Framework." *Bond Graphs for Modelling, Control and Fault Diagnosis of Engineering* .

Bibliography

- Zheng, S., Ristovski, K., Farahat, A., & Gupta, C. (2017, June). Long short-term memory network for remaining useful life estimation. In *2017 IEEE international conference on prognostics and health management (ICPHM)* (pp. 88-95). IEEE.
- Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. In *European conference on computer vision* (pp. 818-833). Springer, Cham.