Bayesian Networks for Interpretable Health Monitoring of Complex Systems

Vishnu TV, Narendhar Gugulothu, Pankaj Malhotra
Lovekesh Vig, Puneet Agarwal, Gautam Shroff
TCS Research, New Delhi, India
{vishnu.tv, narendhar.g, malhotra.pankaj, lovekesh.vig, puneet.a, gautam.shroff}@tcs.com

Abstract

A large number of sensors are being installed on machines to capture the operational behavior of machines. The time series data collected from these sensors is then used to monitor the health of machines. We consider the situation where health monitoring of machines by engineers using raw time series visualizations is augmented by a machine learning and visual analytics based system. Machine learning models such as deep recurrent neural networks (RNNs) have been utilized to provide health index (HI) based on sensor data, which is a measure of the degree of degradation of the health of a machine. Such an HI is often opaque and does not explain the intrinsic patterns in the multi-variate time series which form the basis of the model used to detect health degradation. In practice, such an explanation is of great importance to the machine health monitoring engineers. For example, it can provide further insights into the operational behavior of complex machines, and in turn, help the engineers diagnose the root cause of the approaching failure. In this paper, we make an attempt to mitigate this problem, and propose an approach where-in an *HI Estimator* module is followed by an *HI Descriptor* module. The HI Descriptor module is based on Bayesian Networks and visual analytics, and uses the sensor readings and the HI given by HI Estimator to provide an explanation for the cause of poor health. We evaluate our approach on two real-world use cases: the insights provided by HI Descriptor are found to be useful by domain experts.

1 Introduction and Motivation

Deep learning models are able to compute complex and hierarchical representations of data [LeCun *et al.*, 2015]. While this leads to powerful models, the explainability of the output of such models is restricted. On the other hand, models such as Bayesian Networks and Decision Trees are human-interpretable but may not be as powerful as the deep learning models. Recently, LIME [Ribeiro *et al.*, 2016] has been used to obtain local interpretable classifier models to explain the results of more complex models. In this paper, we show how Bayesian Networks can help humans interpret the results of complex deep learning models with applications to health monitoring of complex systems.

Industrial Internet of Things technology is being increasingly adapted by machine manufacturing industry. A large number of sensors are installed on machines to capture their key operational parameters. The data captured is then used to make better informed decisions. For instance, the readings from these sensors are used by engineers in (remote) monitoring centers to monitor the health of machines and detect anomalous readings indicating faulty behavior. This requires monitoring of multiple sensors continuously over a period of time, or less frequent monitoring of aggregate level statistics from sensor data to get actionable insights. An automated system that can aid or augment the manual monitoring process is often desirable, for example, when the number of sensors to be analyzed is large or when the frequency at which data is captured is high.

Recently it has been shown that deep recurrent neural networks (RNNs), Long Short Term Memory (LSTM) Networks [Hochreiter and Schmidhuber, 1997] in particular, are capable of modeling the complex normal behavior of machines based on multi-sensor time series for detecting anomalies and faults [Malhotra *et al.*, 2015; 2016a; Yadav *et al.*, 2015a; Filonov *et al.*, 2016], and prognostics [Malhotra *et al.*, 2016b]. These models yield a score indicatig likelihood of normal behavior at each time instance. Such a likelihood score can be used to obtain a health index (HI). These models are successful for anomaly/fault detection and health monitoring, they lack the capability to provide explanations for the poor health or detected faults. In real-world scenarios where large number of sensors are involved, finding the cause for poor health and relating it to a particular sensor or a subsystem in a complex system is desirable to get actionable insights. For instance, if a machine's health is estimated to be bad, a set of sensors that help to explain the estimated low health can guide engineers to look at the subsystems related to these sensors more closely.

We propose a data-driven health monitoring system with following capabilities: i) *estimate* health by analyzing large number of sensors while taking into account the possible complex operational dependencies between various modules or sub-systems in a machine, ii) *describe* the potential causes

of poor health or faulty operations of the system.

We consider an unsupervised approach[1] to the problem of health estimation where lower values of HI indicate poor health while higher values indicate good health, s.t. *HI Estimator* requires time series data corresponding to normal behavior of machines only for training. (In this work, we consider the HI Estimator based on deep RNNs.) Further, *HI Descriptor* helps to explain the HI given by the estimator by modeling the dependency between HI and sensor readings via a Bayesian Network (BN).

We introduce an *explainability index* (EI) that quantifies the effect of each sensor on the HI through the change in distribution of the readings a sensor takes over time between predicted high HI and low HI ranges. Similar to learning local interpretable classifier models around the predictions [Ribeiro *et al.*, 2016], we build a localized *BN* around the low HI regions of the time series in order to explain which sensors are the likely reasons for low HI. Through experiments on real-world datasets, we show that once a complex temporal model is able to detect faults, it suffices to build a simpler explainable BN model to find the sensors that help to detect the fault.

The rest of the paper is organized as follows: In Section 2, we describe the proposed health monitoring system. In Section 3, we provide implementation details of the health monitoring system on two real-world use cases. We provide a review of related research in Section 4, and conclude in Section 5.

2 Interpretable Health Monitoring System

We propose a health monitoring system with following modules:

1. **HI Estimator**: This module ingests multi-sensor time series data from a machine and returns a HI for the machine at each time instance. A high HI indicates good health while a low HI indicates poor health or faulty behavior. This module is designed to capture the possibly complex temporal and pointwise correlations across sensors. We describe this in detail in Section 2.1 by using RNN based models as an example of HI Estimation.

2. **HI Descriptor**: This module ingests the HI obtained from HI Estimator and the multi-sensor input, and provides possible explanations for the low HI values in terms of the set of sensors that contribute the most to low HI. This module captures the dependencies between HI and sensor readings via a BN. Further, these dependencies and the amount of change in each sensor with respect to HI can be interactively studied via a visual analytics based user interface. We describe this module in detail in Section 2.2.

The process depicting the proposed health monitoring system with HI Estimator and HI Descriptor modules is shown in Figure 1.

[1]It is important to consider that the number of failure instances is often small to learn a supervised model that can differentiate normal and faulty/anomalous behavior.

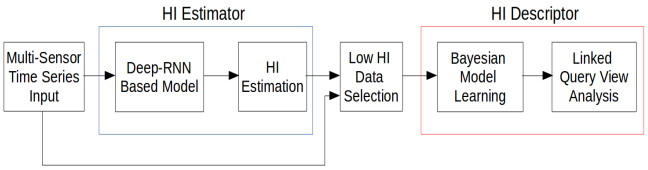


Figure 1: Health Monitoring System

2.1 HI Estimation using RNNs

An LSTM network is trained on the task of prediction or reconstruction of the time series corresponding to normal/healthy machine behavior. This ensures that once trained, the network is expected to predict/reconstruct the time series corresponding to normal behavior well but expected to perform badly on the prediction or reconstrucion task on time series corresponding to abnormal behavior. We leverage the prediction or reconstruction error to estimate the health of the machine. We consider two variants of deep RNN based on LSTM to model the normal behavior: i) LSTM-AD with an LSTM network as a time series prediction model [Malhotra *et al.*, 2015], ii) LSTM-ED with an encoder-decoder pair as a time series reconstruction model [Malhotra *et al.*, 2016a] [2]. Refer Appendix A for further details.

Consider the multi-sensor data to be a multivariate time series $\mathbf{x}_i = \{\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}, ..., \mathbf{x}_i^{(l)}\}$ corresponding to ith instance of a machine, where l is the length of the time series, each point $\mathbf{x}_i^{(t)} \in \mathbf{R}^m$ in the time series is an m-dimensional vector with each dimension corresponding to a sensor. We train a model on the multi-sensor time series taken from healthy state of a machine to predict or reconstruct the time series. The LSTM based models are trained to minimize the squared error between the original time series $\mathbf{x}_i$ and the estimated time series $\hat{\mathbf{x}}_i$ given by $\mathbf{e}_i^{(t)} = \mathbf{x}_i^{(t)} - \hat{\mathbf{x}}_i^{(t)}$ over all training instances, i.e., minimizing $\sum_{i=1}^{N}\sum_{t=1}^{l} \|\mathbf{e}_i^{(t)}\|^2$, where N is the total number of time series in training set.

The error vectors corresponding to the healthy behavior are assumed to follow a Normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. The parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ can be obtained using Maximum Likelihood Estimation method over time series in the training set. Once $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are learned, the HI is computed as follows:

$$h_i^{(t)} = log\left(c \cdot exp\left(-\frac{1}{2}(\mathbf{e}_i^{(t)} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{e}_i^{(t)} - \boldsymbol{\mu})\right)\right) \quad (1)$$

where $c = \frac{1}{\sqrt{(2\pi)^d |\boldsymbol{\Sigma}|}}$ and d is the dimension of error vector. If required, machine instance i can be classified into healthy or unhealthy class at time t if $h_i^{(t)} > \tau$, where τ is a tunable parameter that can be set by a domain expert (or learned in a supervised manner, refer [Malhotra *et al.*, 2015] for details).

[2]LSTM-AD can be used when time series are predictable or quasi-predictable. LSTM-ED can be used for unpredictable time series, for example, where lots of manual controls are involved.

2.2 Interactive HI Description

The time series of HI values obtained from the HI Estimator are used to find windows with large number of low HI values. For each such window, a localized BN is used to find the sensors whose behavior changed the most when compared to window(s) from the recent past with large number of high HI values. The sensors exhibiting large change are likely to be cause of drop in HI. More specifically, if a large fraction of points in time window w_A of length w are estimated to have low HI s.t. $h_i^{(t)} \leq \tau$ for $t \in [t_A - w + 1, t_A]$, we first find the most recent window w_N of length w of normal behavior in the past s.t. $h_i^{(t)} > \tau$ for $t \in [t_N - w + 1, t_N]$ with $t_N \leq t_A - w$. The time series data and the HI values from w_A and w_N are then used to learn a localized BN. The learned BN is used to obtain the Explainability Index (EI) for each sensor that quantifies the contribution of each sensor to low HI values in w_A. An interactive visual analytics technique Linked Query View [Yadav *et al.*, 2015b] is used to qualitatively analyze the change in the behavior of each sensor based on the BN.

Bayesian Network based Explainability Index

Consider a discrete random variable H corresponding to HI, and a set of m discrete random variables $\{S_1, S_2, ..., S_m\}$ correponding to the m sensors. We model the dependence between the sensors and HI via a Bayesian Network with $m + 1$ nodes. The network models the joint distribution $P(S_1, S_2, ..., S_m, H)$ of the set of random variables $X = \{S_1, S_2, ..., S_m, H\}$. (In practice, since we are only interested in modeling the dependence between each sensor and the health index HI, a naive Bayes model with H being the parent node and each S_i being a child node can be assumed. In other cases, the network structure can be given by domain experts.)

A random variable $X_i \in X$ is considered to have k possible outcomes $[b_i^1, b_i^2, ..., b_i^k]$ corresponding to k discretized bins for the range of values the variable can take. An m-dimensional vector of sensor readings $\mathbf{x}^{(t)}$ and health index $h^{(t)}$ for every time instant t in windows w_N and w_A yield one observation for the set of random variables $X = \{S_1, S_2, ..., S_m, H\}$. The marginal probability distribution for S_i is given by $P(S_i) = [p_i^1, p_i^2, ..., p_i^k]$, here p_i^j is the probability of j^{th} outcome of S_i. Given a range of values for HI, i.e. certain outcomes for H, the conditional probability distribution for S_i is given by $P(S_i|H) = [\hat{p}_i^1, \hat{p}_i^2, ..., \hat{p}_i^k]$. The change in the distribution of the random variable S_i conditioned on outcomes of H corresponding to high HI, i.e. $P(S_i\ |H_{>\tau})$ and low HI, i.e. $P(S_i\ |H_{\leq\tau})$, is used to quantify the effect of the ith sensor on HI. Considering $P(S_i\ |H_{>\tau})$ and $P(S_i\ |H_{\leq\tau})$ as vectors in $\mathbf{R}^k$, we quantify the change in terms of an Explainability Index (EI) given by:

$$E(S_i) = ||P(S_i\ |H_{>\tau}) - P(S_i\ |H_{\leq\tau})|| \qquad (2)$$

where $||\cdot||$ is the L_2-norm. Higher the value for $E(S_i)$, higher is the effect of sensor S_i on the HI.

Linked Query Views for interactive visual analytics

The health monitoring system further leverages an interactive user interface named Linked Query View (LQV) to analyze the dependencies obtained from BN. For example, queries such as: "what happens to the distribution of variable S_i when H is low?" can be executed, and results based on Bayesian inference over the learned BN can be seen in real-time in form of changes in the histogram (e.g., refer Figure 4c). It supports visualizations to analyze and query the probability distributions of variables in X via 1D and 2D histograms. For example, Figure 6b shows a 1D histogram for the distribution of H, and Figure 6a shows joint distribution of two sensors via a heatmap. A range of bins in the histograms or cells in the heatmaps can be interactively selected to condition the distribution of other variables. The conditional query can be based on single or multiple sensors. LQV primarily contains the following three different views:

1. *Original view*: This view shows the histograms and heatmaps of sensors, and is used to execute a query on a range of values of one or more selected sensors.
2. *Updated view*: Once a query is made on selected sensors, conditional probability distributions of other sensors are shown via updated histograms and heatmaps.
3. *Compare view*: This view shows the difference or change in the histograms/heatmaps of sensors which have been updated after a query.

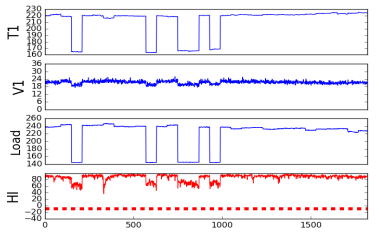
For a single variable, LQV uses blue bars to represent bins of a histogram in Original and Updated views, while it uses blue bars to show the original histograms and red bars to show the updated histograms in Compare view. For analyzing two variables at once via heatmaps, Compare view is shown as a heatmap of differences (e.g., refer Figure 6d). We use a cold to warm colour scale for this. Cells for which the probability increases (differences are negative) are colored in shades of blue, while those for which the probability decreases (differences are positive) are plotted in shades of red. Cells for which the distribution does not change are shown in white and those which are not populated in the Original view are shown in grey.

3 Experimental Results

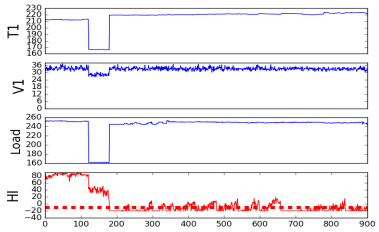
We evaluate our approach on two real-world datasets, namely, Turbomachinery and Engine datasets. For Turbomachinery dataset, we demonstrate how the Explainability Index (EI) $E(S_i)$ for a sensor S_i (refer Equation 2) can be used to arrive at a subset of sensors that explain the faulty operation of the machine. On Engine dataset, we further demonstrate how HI Estimator and LQV can be used to analyze the sensor behavior to get insights about the possible causes for low HI. For learning the BN models, we discretize sensor values to $k = 10$ bins. The performance of HI Estimator module is evaluated in terms of Precision, Recall and F_β-score for detecting faulty behavior and differentiating it from normal behavior. We take $\beta = 0.1$ to give higher importance to Precision over Recall as all points on a day labeled as "Faulty" may not exhibit faulty behavior.

3.1 Turbomachinery Dataset

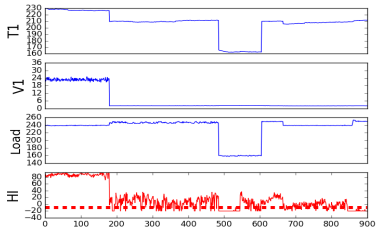
This dataset contains readings from 58 sensors such as temperature, pressure, and vibration recorded for 6 months of



(a) Normal operation



(b) Temp-C1 fault



(c) Vibration fault

Figure 2: Turbomachinery: Sample HI Estimator results. HI values are clipped to -20.0 for visual clarity.

Table 1: HI Estimator: Precision, Recall, $F_{0.1}$-Score for Normal vs Faulty Classification

Dataset	Model	Architecture	Precision	Recall	$F_{0.1}$-score
Engine	LSTM-AD	25 units,1 layer	0.94	0.12	0.89
Turbomachinery	LSTM-ED	500 units, 1 layer	0.96	0.41	0.94

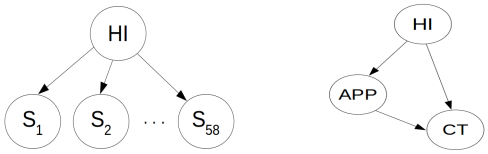
operation. These sensors capture behavior of different components such as bearing and coolant of the turbomachinery. The turbomachinery is controlled via an automated control system having multiple controls making the sensor readings change frequently, and hence, unpredictable. We, therefore use LSTM-ED for HI estimation. We train the LSTM Encoder Decoder to reconstruct all sensors, i.e., $m = 58$. We provide performance details of HI Estimator and HI Descriptor on three types of faults related to: i) abnormal temperature fluctuations in component C_1 (Temp-C_1), ii) abnormal temperature fluctuations in component C_2 (Temp-C_2), iii) abnormal vibration readings.

HI Estimator performance evaluation: Table 1 shows the performance of HI Estimator for classifying normal and faulty behavior. We denote the most relevant sensor for Temp-C_1 and Vibration faults by T_1 and V_1, respectively. Figure 2 shows sample time series for normal and faulty behavior for sensors T_1, V_1, Load, and HI. The red dotted line in HI subplot is the classification threshold τ (refer Section 2.1). We can observe that while HI is consistently high for normal behavior, it drops below τ for faulty behavior.

Table 2: Turbomachinery: HI Descriptor Results Summary

Fault Type	No. of Instances	Explained Instances	Avg. Rank
Temp-C_1	3	3	1.0
Temp-C_2	1	1	1.0
Vibration	6	3	3.0
Total	**10**	**7**	**2.2**

Next, we show that once the HI values are available from the LSTM-ED temporal model to detect faults, it suffices to build a simpler BN model to find the relevant sensors carrying the fault signature.



(a) Turbomachinery data (b) Engine data

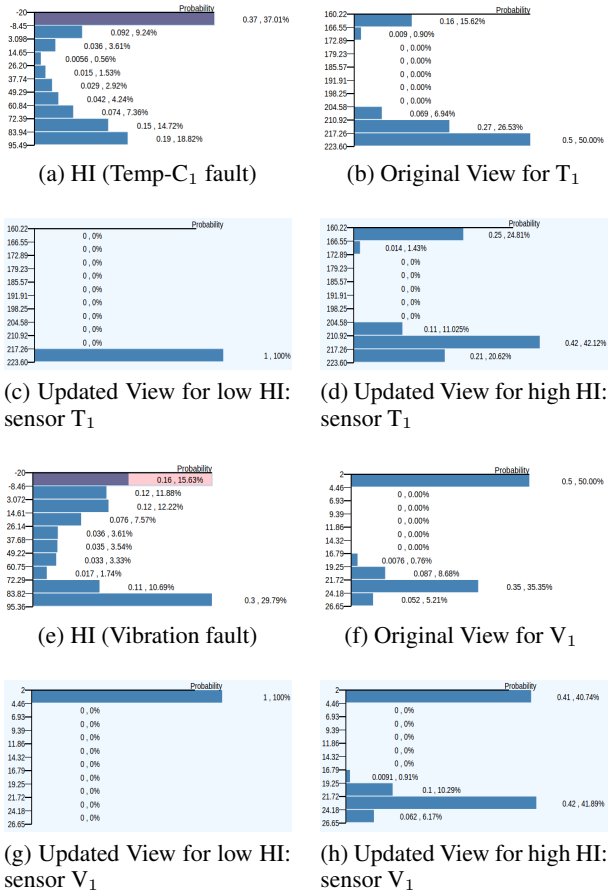
Figure 3: Bayesian Networks considered for HI Descriptor

HI Descriptor for explaining low HI regions: The BN structure used to analyze sensor behavior in the regions of low HI is shown in Figure 3a. For learning the BN, we consider normal (w_N) and faulty (w_A) windows of length $w = 720$ (corresponding to 12 hours of operation), s.t. at least 70% of points in w_A have HI below τ. To find the most relevant sensor carrying the fault signature, we rank the sensors from 1 to 58 s.t. the sensor with highest EI gets rank 1 while sensor with lowest EI gets rank 58. A fault instance is considered to be explained by the HI Descriptor, if the most relevant sensor for the fault type gets the highest rank based on EI. Table 2 shows the results for the three fault types where all the instances of Temp-C_1 and Temp-C_2, and 3 out of 6 vibration related faults could be explained by the highest ranked sensor. For the remaining three instances, we found that operating conditions for the faulty window w_A and the corresponding normal window w_N were different leading to incorrect explanations. Thus for these cases, the ranks for the most relevant sensor were 2, 6, and 7.

In Figure 4, we show that the distribution of the most relevant sensor changes significantly across the normal and faulty operating conditions. This change in distribution is captured using EI to find the most relevant sensor. Figures 4a and 4b show the *Original Views* of HI and temperature sensor T_1, respectively, for one of the faults related to Temp-C_1. Figures 4c and 4d show the *Updated views* for sensor T_1 under low HI and high HI condition, respectively. The results for one of the instances of vibration fault are shown in Figures 4e-4h.

3.2 Engine Dataset

This dataset contains readings from 12 sensors, recorded for $\approx$ 3 years of engine operation. The sensor readings in this dataset are quasi-predictable and depend on an external manual control, namely, Accelerator Pedal Position (APP). We, therefore, use LSTM-AD based HI Estimator for this dataset. We use all sensors as input to LSTM-AD s.t. $m = 12$, and predict two of the sensors: APP and Coolant Temperature



(a) HI (Temp-C_1 fault) (b) Original View for T_1

(c) Updated View for low HI: sensor T_1 (d) Updated View for high HI: sensor T_1

(e) HI (Vibration fault) (f) Original View for V_1

(g) Updated View for low HI: sensor V_1 (h) Updated View for high HI: sensor V_1

Figure 4: LQV: Turbomachinery. Red shaded region in HI graphs corresponds to low HI regions.

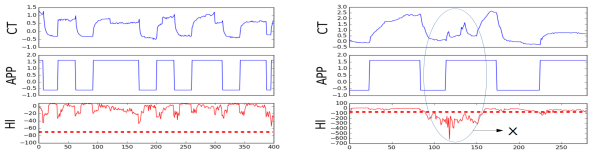
(CT), and use LQV to get insights into the reasons for estimated low HI. The low HI regions found correspond to three instances of abnormal CT.

HI Estimator performance evaluation: Table 1 shows the performance of HI Estimator. Figures 5a and 5b show the time series plots for CT, APP, and HI for samples of normal and faulty regions in the data, respectively.

HI Descriptor for explaining low HI regions: We model the dependence between HI and sensors using a BN as shown in Figure 3b. The joint distribution of APP and CT (represented as a heatmap), and the distribution of HI are shown in Figures 6a and 6b, respectively.

From domain knowledge, it is known that high APP leads to high CT, while low APP leads to low CT over time with a certain time lag where transient behavior is observed. Any time window over which APP and CT do not exhibit such a temporal correlation is considered faulty. We show how BN based HI Descriptor on top of HI Estimator is able to corroborate this knowledge and suggest the reasons for faulty regions (sample shown in region marked X in Figure 5b) using LQV:

1. Figure 6a shows the *Original view* of the joint distribution of APP and CT. The Figure clearly suggests that when APP is high, CT is also high (marked as A), and when APP is low, CT is low (marked as B). The cells marked A and B cover the highest percentage of data and correspond to high HI.
2. We next condition the joint distribution of APP and CT on the low HI regions by interactively selecting low HI bars as shown in red in Figure 6b. The *Updated view* thus obtained is shown in Figure 6c. Here, *the highest probability bin is shown by a marker C, where APP is low but CT is high*. This suggests that when HI is low, the machine is indeed in a faulty operation state.
3. In *Compare view* in Figure 6d, cells D and E correspond to maximum decrease and maximum increase in joint probability of APP and CT, respectively. This indicates that the number of points corresponding to healthy state (cell D) decrease and those indicating poor health (cell E) increase when HI is low.



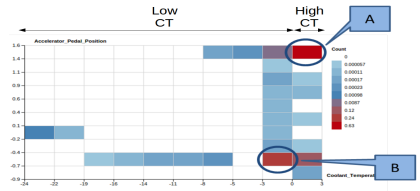
(a) Normal operation (b) Faulty operation

Figure 5: Engine: Sample HI Estimator Results

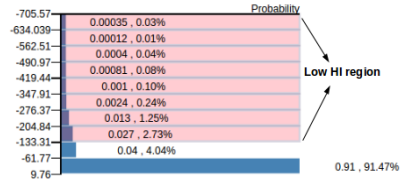
4 Related Work

Explainability and interpretability of complex machine learning models, especially deep learning models is an open research problem [Lipton, 2016]. Several attempts such as the ones reviewed in [Vellido *et al.*, 2012] have been made to address this challenge by making inherently interpretable machine learning models. [Baehrens *et al.*, 2010] explained how local explanation vectors (local gradients) play pivot role in predicting the label for a datapoint using non-linear classifiers. [Kulesza *et al.*, 2015] propose multinomial Naive Bayes to explain machine learning model for text data. [Shwartz-Ziv and Tishby, 2017] suggest Information Planes based on Information Bottleneck principle to explain the internal working of Deep Neural Networks. Further, [Ribeiro *et al.*, 2016] propose LIME to explain the predictions of classifiers by learning locally interpretable models. To the best of our knowledge, our work is the first attempt to explain the real-valued temporal outputs of a deep learning model with the help of locally built Bayesian Networks. Our work can be seen as an extension of [Ribeiro *et al.*, 2016] to build locally interpretable simplified models to explain the outputs of a black-box machine learning model for multivariate time series data where output is also a real-valued time series.

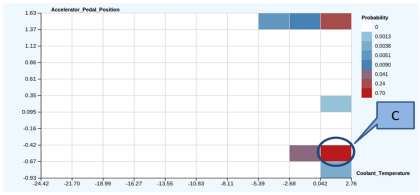
Association rule mining and exception rules mining have been proposed in [Saikia *et al.*, 2014] to find the conditions that imply certain sensor behaviors, and then also find exceptions to such conditions. This work directly attempts to explain the multi-sensor data with the help of rules and exceptions. [Letham *et al.*, 2015] proposed a generative Bayesian rule list model using decision lists to interpret classification models. [Sanchez *et al.*, 2015] used simple logic rules and



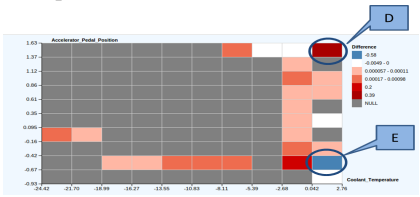
(a) Original View. X-axis:CT, Y-axis:APP



(b) Selected Query on HI histogram



(c) Updated View. X-axis:CT, Y-axis:APP



(d) Compare View. X-axis:CT, Y-axis:APP

Figure 6: LQV: Engine data (best viewed when magnified)

BN to explain descriptive representations of matrix factorization. Such approaches do not address the challenge of temporal aspects of the multi-sensor data and are likely to miss important temporal patterns in the data. On the other hand, our approach is capable of capturing temporal as well as point-wise dependencies across sensors while trying to interpret the behavior around regions of poor health.

5 Discussion

We address a common challenge faced when using complex machine learning models for critical applications: "How to explain the outputs of machine learning models?". We have proposed an approach to address this in the context of machine health monitoring systems using multi-sensor time series data from Industrial IOT systems. An approach using a complex temporal model based on recurrent neural networks (RNNs) as health estimator, and a human-interpretable simple Bayesian Network (BN) supported by visual Linked Query Views to explain the results of health estimator is proposed. The localized BN model is shown to be useful to interpret the results given by RNN on two real-world scenarios. The results given by our approach are found to be useful by domain-experts for interpreting the results. Further, our approach can be easily extended to explain HI from any HI Estimator module.

Acknowledgments

We would like to thank the anonymous reviewers for valuable feedback to help improve the paper. We thank T Joel for corroborating the results and providing useful domain knowledge. We also thank Mudit Singh and Shashank Joshi for helping with the experiments.

References

[Baehrens *et al.*, 2010] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, et al. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(Jun):1803–1831, 2010.

[Filonov *et al.*, 2016] Pavel Filonov, Andrey Lavrentyev, and Artem Vorontsov. Multivariate industrial time series with cyber-attack simulation: Fault detection using an lstm-based predictive data model. *NIPS Time Series Workshop 2016, arXiv preprint arXiv:1612.06676*, 2016.

[Hochreiter and Schmidhuber, 1997] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Kulesza *et al.*, 2015] Todd Kulesza, Margaret Burnett, Weng-Keen Wong, and Simone Stumpf. Principles of explanatory debugging to personalize interactive machine learning. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, pages 126–137. ACM, 2015.

[LeCun *et al.*, 2015] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[Letham *et al.*, 2015] Benjamin Letham, Cynthia Rudin, Tyler H McCormick, et al. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, 9(3):1350–1371, 2015.

[Lipton, 2016] Zachary C Lipton. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490*, 2016.

[Malhotra *et al.*, 2015] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Long short term memory networks for anomaly detection in time series. In *23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2015.

[Malhotra *et al.*, 2016a] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Lstm-based encoder-decoder for multi-sensor anomaly detection. In *Anomaly Detection Workshop at 33rd International Conference on Machine Learning (ICML 2016)*. CoRR, abs/1607.00148, 2016, https://arxiv.org/abs/1607.00148, 2016.

[Malhotra *et al.*, 2016b] Pankaj Malhotra, Vishnu TV, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Multi-sensor prognostics using an unsupervised health index based on lstm encoder-decoder. In *1st ACM SIGKDD Workshop on Machine Learning for Prognostics and Health Management, San Francisco, CA, USA, 2016*. CoRR, abs/1607.00148, 2016. URL http://arxiv.org/abs/1607.00148, 2016.

[Ribeiro *et al.*, 2016] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM, 2016.

[Saikia *et al.*, 2014] S Saikia, G Shroff, P Agarwal, A Srinivasan, A Pandey, and G Anand. Exploratory data analysis using alternating covers of rules and exceptions. In *Proceedings of the 20th International Conference on Management of Data*, pages 105–108. Computer Society of India, 2014.

[Sanchez *et al.*, 2015] Ivan Sanchez, Tim Rocktaschel, Sebastian Riedel, and Sameer Singh. Towards extracting faithful and descriptive representations of latent variable models. *AAAI Spring Syposium on Knowledge Representation and Reasoning (KRR): Integrating Symbolic and Neural Approaches*, 2015.

[Shwartz-Ziv and Tishby, 2017] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.

[Vellido *et al.*, 2012] Alfredo Vellido, José David Martín-Guerrero, and Paulo JG Lisboa. Making machine learning models interpretable. In *ESANN*, volume 12, pages 163–172. Citeseer, 2012.

[Yadav *et al.*, 2015a] Mohit Yadav, Pankaj Malhotra, Lovekesh Vig, K Sriram, and Gautam Shroff. Ode-augmented training improves anomaly detection in sensor data from machines. In *NIPS Time Series Workshop*. CoRR, abs/1605.01534, 2016. URL http://arxiv.org/abs/1605.01534, 2015.

[Yadav *et al.*, 2015b] Surya Yadav, Gautam Shroff, Ehtesham Hassan, and Puneet Agarwal. Business data fusion. In *Information Fusion (FUSION), 2015 18th International Conference on*. IEEE, 2015.

[Zaremba *et al.*, 2014] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014.

Appendix A

Health Index Estimation using LSTMs

The HI Estimator uses LSTM recurrent unit based neural network architectures, namely LSTM-AD [Malhotra *et al.*, 2015] and LSTM-ED [Malhotra *et al.*, 2016a]. Long Short Term Memory (LSTM) is a type of recurrent hidden unit which is capable of remembering and passing relevant information over a large number of steps in a sequence. A basic LSTM unit [Zaremba *et al.*, 2014] uses the input x_t, the hidden state activation h_{t-1}, and memory cell activation c_{t-1} to compute the hidden state activation h_t at time t as defined in Equations 3-5. It uses a combination of a memory cell c, input gate i, forget gate f, and output gate o to decide if the input needs to be remembered (using input gate), when the previous memory needs to be retained (forget gate), and when the memory content needs to be output (using output gate). Consider $T_{n_1,n_2} : R^{n_1} \rightarrow R^{n_2}$ as an affine transform of the form $x \mapsto Wx + b$ for matrix W and vector b of appropriate dimensions.

$$\begin{pmatrix} i_t \\ f_t \\ o_t \\ g_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ tanh \end{pmatrix} T_{m+n,4n} \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix} \quad (3)$$

Here $\sigma(x) = \frac{1}{1+e^{-x}}$ and $tanh(x) = 2\sigma(2x) - 1$. The operations σ and $tanh$ are applied elementwise. The four equations from the above simplified matrix notation are read as: $i_t = \sigma(W_1 x_t + W_2 h_{t-1} + b_i)$, etc. Here $x_t \in R^m$, and all others $i_t, f_t, o_t, g_t, h_t, c_t \in R^n$, such that m is the input dimension and n is the number of LSTM units in the hidden layer. The updated hidden state h_t is computed as follows:

$$c_t = f_t c_{t-1} + i_t g_t \quad (4)$$

$$h_t = o_t tanh(c_t) \quad (5)$$

The elements of matrix $T_{m+n,4n}$ are the learnable parameters of the neural network and are learnt using the backpropagation algorithm.

LSTM-AD based HI Estimator: At time t, LSTM-AD predicts the time series $\{\mathbf{x}_i^{(t+1)}, ..., \mathbf{x}_i^{(t+p)}\}$ corresponding to next p time steps given the time series $\{\mathbf{x}_i^{(1)}, ..., \mathbf{x}_i^{(t)}\}$ till time t. The *error vector* $\mathbf{e}_i^{(t)}$ at time t is given by concatenation of error vectors corresponding to all predictions of $\mathbf{x}_i^{(t)}$, such that $\mathbf{e}_i^{(t)} = [\mathbf{e}_{i1}^{(t)}, \mathbf{e}_{i2}^{(t)}, ..., \mathbf{e}_{ip}^{(t)}]$, where $\mathbf{e}_{ij}^{(t)}$ is the difference between $\mathbf{x}_i^{(t)}$ and its value as predicted at time $t - j$. The HI is obtained from the error vector $\mathbf{e}_i^{(t)}$ as in Equation 1. Refer [Malhotra *et al.*, 2015] for further details on LSTM-AD.

LSTM-ED based HI Estimator: LSTM-ED ingests the time series $\{\mathbf{x}_i^{(1)}, ..., \mathbf{x}_i^{(t)}\}$ and reconstructs it to yield $\{\hat{\mathbf{x}}_i^{(1)}, ..., \hat{\mathbf{x}}_i^{(t)}\}$ via RNN encoder-decoder. The encoder RNN network captures the relavant information in the time series and returns a fixed-dimensional vector embedding for the time series. The decoder RNN uses this embedding to reconstruct the time series. The error vector $\mathbf{e}_i^{(t)}$ at time t is given by $\mathbf{e}_i^{(t)} = |\mathbf{x}_i^{(t)} - \hat{\mathbf{x}}_i^{(t)}|$, where $|.|$ returns elementwise absolute value. Similar to LSTM-AD, the HI is obtained from the error vector $\mathbf{e}_i^{(t)}$ as in Equation 1. Refer [Malhotra *et al.*, 2016a] for further details on LSTM-ED.