MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Anomaly Detection in Manufacturing Systems Using Structured Neural Networks

Liu, J.; Guo, J.; Orlik, P.V.; Shibata, M.; Nakahara, D.; Mii, S.; Takac, M.

TR2018-097 July 13, 2018

Abstract

This paper proposes innovative anomaly detection technologies for manufacturing systems. We combine the event ordering relationship based structuring technique and the deep neural networks to develop the structured neural networks for anomaly detection. The event ordering relationship based neural network structuring process is performed before neural network training process and determines important neuron connections and weight initialization. It reduces the complexity of the neural networks and can improve anomaly detection accuracy. The structured time delay neural network (TDNN) is introduced for anomaly detection via supervised learning. To detect anomaly through unsupervised learning, we propose the structured autoencoder. The proposed structured neural networks outperform the unstructured neural networks in terms of anomaly detection accuracy and can reduce test error by 20%. Compared with popular methods such as one-class SVM, decision trees, and distance-based algorithms, our structured neural networks can reduce anomaly detection misclassification error by as much as 64%.

IEEE World Congress on Intelligent Control and Automation

Anomaly Detection in Manufacturing Systems Using Structured Neural Networks

Jie Liu[1], Jianlin Guo[2], Philip Orlik[2], Masahiko Shibata[3], Daiki Nakahara[3], Satoshi Mii[3], and Martin Takáč[1]

[1]Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA 18015, USA
[2]Mitsubishi Electric Research Laboratories, Cambridge, MA 02139, USA
[3]Mitsubishi Electric Corporation IT R&D Center, Ofuna, Kamakura, Kanagawa, 247-8501, Japan

***Abstract*—This paper proposes innovative anomaly detection technologies for manufacturing systems. We combine the event ordering relationship based structuring technique and the deep neural networks to develop the structured neural networks for anomaly detection. The event ordering relationship based neural network structuring process is performed before neural network training process and determines important neuron connections and weight initialization. It reduces the complexity of the neural networks and can improve anomaly detection accuracy. The structured time delay neural network (TDNN) is introduced for anomaly detection via supervised learning. To detect anomaly through unsupervised learning, we propose the structured autoencoder. The proposed structured neural networks outperform the unstructured neural networks in terms of anomaly detection accuracy and can reduce test error by 20%. Compared with popular methods such as one-class SVM, decision trees, and distance-based algorithms, our structured neural networks can reduce anomaly detection misclassification error by as much as 64%.**

***Keywords*—Anomaly detection, manufacturing system, machine learning, time delay neural network, autoencoder.**

I. Introduction

In manufacturing systems, reducing downtime is critical. Anomaly detection enables predictive maintenance for downtime reduction. Machine learning has been recently applied to detect anomaly in manufacturing processes. Using machine learning, the collected data can be utilized in an automatic learning system, where the specialties of the data can be learned through training. The trained model can detect anomaly in real time data to realize predictive maintenance and downtime reduction. Meanwhile, the new data can also be used to update the learning model.

The manufacturing operations can be divided into process manufacturing and discrete manufacturing. Process manufacturing is primarily concerned with the blending of formulas, where products are generally undifferentiated. One practical approach to detect anomaly in process manufacturing is to use a description of normal operation in terms of the data, define the admissible operating range and detect outliers. One of such methods is anomaly detection in petroleum industry proposed by [1]. On the other hand, discrete manufacturing executes a sequence of operations and produces distinct items. Anomaly may occur if incorrect execution of operations takes place, e.g., an incorrect order of operations. Even in anomalous situation, the measured data may still be in the expected range. Therefore, outlier detection cannot reliably detect anomaly. However, the order of operation execution and time gap between consecutive operation executions become important factors in anomaly detection. [2] provides a anomaly detection method for discrete manufacturing.

Neural networks have achieved success in many applications such as image processing and speech recognition. Neural networks do not require any analytical behavior model and depend on the training data. Using neural networks, additional anomalies that are not obvious from domain knowledge can be detected [3]. This paper presents neural network based anomaly detection techniques for manufacturing systems. We aim to develop anomaly detection methods for both process manufacturing and discrete manufacturing.

One of the key challenges in applying neural network is to find a suitable and minimal neural network topology, especially for manufacturing systems with large amount of data. A basic approach to find the minimal neural network topology is pruning, which removes portion of neuron connections and/or neurons from the neural network during training process. Even pruning reduces the number of the neural network parameters, it may degrade network performance. To address this issue, we propose an innovative neural network structuring technique that constructs neural network topology based on the event ordering relationship before the training process and improves neural network performance.

The proposed structuring technique is applied to the time delay neural network (TDNN) and the time delay autoencoder to produce the structured TDNN and the structured autoencoder. The structured TDNN is used for anomaly detection via the supervised learning and the structured autoencoder is employed for anomaly detection through the unsupervised learning. Compared with the unstructured neural networks, the structured

neural networks improve anomaly detection accuracy and reduce test error rate.

II. Related Work

The anomaly detection in manufacturing systems has been attracting increased attention among researchers and industry personnels. However, the high complexity of the manufacturing processes and the continuously growing amount of data present challenges to detect anomaly in manufacturing systems.

For process manufacturing, outlier detection methods have been developed. [1] proposes a method for anomaly detection in petroleum industry. The method uses sensor data to find patterns in data that do not conform to a priori expected behavior. [4] uses several classical machine learning approaches to detect outliers in high-dimensional monitoring problems. The anomaly detection strategies have been tested on a real industrial dataset related to a Semiconductor Manufacturing Etching process. [3] proposes the self-learning assistance technique to speed up fault detection of manufacturing processes. The self-learning assistance system identifies relevant relationships by observation of complex manufacturing processes so that failures and anomalies are automatically detected. Several anomaly detection algorithms (distance based approaches, regression models, self-organizing maps and principal component analysis based approaches) are evaluated to detect outliers in complex processes of chemical industry, agricultural harvesting processes and large-scale sorting plants.

For discrete manufacturing, operation order mismatch techniques have been developed. [2] provides an anomaly detection method for discrete manufacturing, in which data are processed to form a stream of discrete events. An event relationship table for normal operations is constructed. This table is then used to detect event order mismatch in the operation of discrete manufacturing. [5] proposes a similar approach, which allows multiple occurrences of an event. This type of anomaly detection methods works well for discrete manufacturing without random operations. However, these methods may face challenge for manufacturing systems with random operations, where the order of the operations can be random. In addition, these methods may also face challenge in scalability. The method proposed in [5] is only validated for small scale of system and the detection error can be as high as 40%. Furthermore, these methods needs to consider the simultaneous event occurrence, which is normal in complex manufacturing system.

The TDNN has been applied to anomaly detection in various systems, especially real time systems. The TDNN is a supervised learning approach and more robust than the classical modeling approaches [6], which describes how to use the TDNN for detecting anomaly in robotic system. [7] uses the TDNN for network intrusion detection to discriminate between normal and abnormal packet flows. The TDNN is applied in the development of failure modes for an insulated gate bipolar transistor to predict the remaining useful life of the power electronic components [8].

Autoencoder is another type of neural networks used for anomaly detection via unsupervised learning. A multi-modal deep autoencoder is employed for anomaly detection and fault disambiguation in large flight data [9]. The autoencoder is also applied to extend classical Principal Component Analysis model to achieve robust, nonlinear and inductive anomaly detection on real world image datasets [10].

III. Neural Networks for Anomaly Detection in Manufacturing Systems

Machine learning has been recently applied to detect anomaly in manufacturing systems. Neural network is the most popular model in machine learning. Neural network can be employed to detect anomaly via both supervised learning and unsupervised learning. We apply time delay neural network architecture to explore the relationship of data signals in time domain.

A. Anomaly Detection with Supervised Neural Nets

For the supervised learning, training data are labeled as either normal or abnormal. We employ the TDNN to detect anomaly, where neural network is the *feedforward* neural network as shown in Fig.1(a). The input layer accepts data signals X and transfers the extracted features through the weight vector W_1 and the activation function to the first hidden layer, which takes output of the input layer and bias $+1$ and transfers the extracted features to next layer. After multiple hidden layers of the feature extraction, neural network reaches to the output layer that has a specific loss function and formulates the corresponding optimization problem. Finally, the output layer produces the output Y.

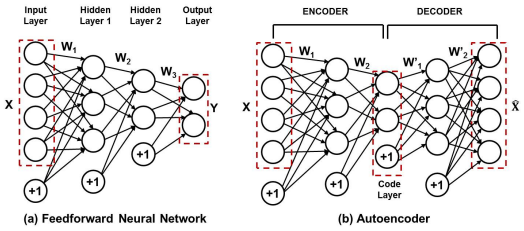

Fig. 1: Time Delay Neural Network and Autoencoder.

Mathematically, the neural network model can be interpreted as chains of functions where each successor f as a neuron at layer l can be propagated by its predecessors g_i at layer $l-1$ with weights as

$$f(\mathbf{w}, x) = a(\sum_i w_i g_i(x)),$$

where a is the predefined activation function. Usually, at the output layer, instead of using an activation function,

a specific cost function is employed to deal with the corresponding task. The cost function is generally data-independent. The final neural network problem can be summarized as the following finite-sum optimization problem:

$$\min_{w\in\mathbb{R}^d} \left\{ F(x) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} l(\mathbf{w}; x_i) \right\}, \tag{1}$$

where as a regression/classification problem via supervised learning, we are predicting/classifying the time instance $x^{(t)}$ using the previous time instances with specific window, with the cost function as

$$l(w; x) = f(x^{(t)}, h(\mathbf{w}; x^{(t-1)}, ...)), \tag{2}$$

where $h(\cdot)$ denotes the value function of the whole neural network.

B. Anomaly Detection with Unsupervised Autoencoder

The manufacturing data may be collected under normal operation condition only since anomaly rarely happens in manufacturing system or the anomalous data are difficult to collect. Under this circumstance, the data are usually not labeled and therefore, the unsupervised learning techniques are required. In this case, we apply the time delay *autoencoder* to detect anomaly.

The *autoencoder* is the special neural network used to learn a representation of a set of data, i.e., to reconstruct the input data with the encoder and the decoder composed of a single or multiple hidden layers as shown in Fig. 1(b), where the encoder and the decoder are feedforward neural networks, *X* is the input data and $\hat{X}$ is the reconstructed data. The compressed features appear in the middle layer is usually called the code (layer) in the network structure, where as a regression problem, the loss function (2) is replaced by

$$\begin{aligned} l(w; x) &= f(x^{(t)}, \hat{x}^{(t)}) = f(x^{(t)}, h(\mathbf{w}; x^{(t)})) \\ &= \tfrac{1}{2}\|x - h(\mathbf{w}; x^{(t)})\|^2. \end{aligned} \tag{3}$$

IV. Event Ordering Relationship Based Neural Network Structure

The problem of determining the proper size of neural network is important. Even though the fully connected neural network can learn its weights through training, reasonably reducing the complexity of the network can improve the performance and has a potential of reduction in computational cost.

One popular approach for tackling this problem is commonly known as pruning and it consists of training a larger than necessary network and then removing unnecessary weights and/or neurons. The pruning is time consuming and may also degrade performance. This paper proposes an innovative event ordering relationship based neural network structuring method. Instead of removing unnecessary weights and neurons during training process, we determine neural network structure before the training process. Our approach can reduce training time and improve accuracy. More precisely, we pre-process the training data to find the event ordering relationship, which is used to determine important neuron connections. We then remove the unimportant connections and the isolated neurons or set weights of the unimportant connections to 0 before neural network training.

For processing manufacturing, an event can represent abnormal status such as measured data being out of admissible operating range. For discrete manufacturing, an event may indicate an execution of the operation. In a manufacturing system, tens to hundreds of sensors are used to collect data periodically or aperiodically. We call data measurement from a specific sensor as a signal, e.g., a voltage sensor measures voltage signal. A sensor may measure multiple data signals.

For a data signal, we define an event as signal value change from one level to another level. The signal changes can be either out of admissible range or in admissible range. More specifically, an event is defined as $E = \{S, ToS, T\}$, where S represents data signal that results in the event, ToS indicates type of event for signal S, e.g., for a binary switch signal, $0 \rightarrow 1$ means event ON and $1 \rightarrow 0$ indicates event OFF, and T is the time at which the event occurred. Therefore, an event may correspond to a normal operation execution or an anomalous incident in the system. Before training neural network, the training data are processed to extract events for each signal. These events are used to build the event ordering relationship (EOR). Assume there are M signals $\{S_i\}_1^M$, which generate N events. According to event occurring time, arrange these events into an event sequence $\{E_i\}_1^N$. Assume event sequence $\{E_i\}_1^N$ contains K distinct events $\{\hat{E}_i\}_1^K$, where $\hat{E}_i$ has a format of {S, ToS}. The EOR has following format:

	$\hat{E}_1$	$\hat{E}_2$	$\hat{E}_3$	...	$\hat{E}_K$
$\hat{E}_1$	0	e_{12}	e_{13}	...	e_{1K}
$\hat{E}_2$	e_{21}	0	e_{23}	...	e_{2K}
$\hat{E}_3$	e_{31}	e_{32}	0	...	e_{3K}
⋮	⋮	⋮	⋮	⋮	⋮
$\hat{E}_K$	e_{K1}	e_{K2}	e_{K3}	...	0

where e_{ij} $(i \neq j)$ is initialized to 0. During EOR construction process, e_{ij} is increased by 1 for each event pair $\{\hat{E}_i, \hat{E}_j\}$ occurrence in event sequence $\{E_i\}_1^N$. If events $\hat{E}_i$ and $\hat{E}_j$ occur at same time, both e_{ij} and e_{ji} are increased by $\frac{1}{2}$. Therefore, e_{ij} $(i \neq j)$ indicates the number of times event $\hat{E}_j$ following event $\hat{E}_i$. A larger e_{ij} indicates that event $\hat{E}_j$ tightly follows event $\hat{E}_i$, a smaller e_{ij} implies that event $\hat{E}_j$ loosely follows event $\hat{E}_i$, and $e_{ij} = 0$ indicates that event $\hat{E}_j$ never follows event $\hat{E}_i$. If both e_{ij} and e_{ji} are greater than zero, events

$\hat{E}_i$ and $\hat{E}_j$ can occur in either order.

The EOR can be used to construct a $M \times M$ signal connection matrix (SCM) as

$$SCM = I + \begin{bmatrix} 0 & c_{12} & c_{13} & \dots & c_{1M} \\ c_{21} & 0 & c_{23} & \dots & c_{2M} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{M1} & c_{M2} & c_{M3} & \dots & 0 \end{bmatrix},$$

where I is the identity matrix, c_{ij} $(i \neq j)$ represents the number of times events of signal S_j following events of signal S_i. Therefore, $c_{ij} = 0$ indicates event of signal S_j never follows event of signal S_i. A higher value of c_{ij} indicates that signal S_j tightly depends on signal S_i in the sense that the change of signal S_i may cause the change of signal S_j with a greater probability. On the other hand, a lower value of c_{ij} implies that signal S_j loosely depends on signal S_i. A threshold C_{TH} can be defined for neural network connection configuration such that if $c_{ij} \geq C_{TH}$, then signal S_i can be considered to impact signal S_j.

We use a three signal system to illustrate the structured TDNN construction. Assume three signals are $\{S_1, S_2, S_3\}$, time delay window is 2 and $C_{TH} = 1$. We use s_{i0} and s_{i1} $(1 \leq i \leq 3)$ to denote measurements of signal S_i at time t and time $t-1$, which correspond to the nodes S_{i0} and S_{i1} $(1 \leq i \leq 3)$ in the neural network. Fig.2 shows fully connected TDNN and the structured TDNN constructed using signal connection matrix. We structure the input layer and the first hidden layer of the neural network because these two layers have the most influence on the topology of the neural network. Other layers and bias nodes are described in section III-A. It can be seen that the number of nodes at input layer is 6, i.e., number of signals $\times$ time delay window size, and the number of nodes at the first hidden layer is 3, i.e., number of signals. The objective of the first hidden layer node configuration is to concentrate features related to a specific signal to a single hidden node. For the fully connected TDNN, there are total 18 connections. Using the signal connection matrix in Fig.2, 18 connections are reduced to 10 connections in the structured TDNN. For example, $c_{12} = 1 = C_{TH}$ indicates that signal S_1 may impact signal S_2. Therefore, collections from S_{10} and S_{11} to H_{12} are important because H_{12} is used to collect information for signal S_2. On the other hand, $c_{13} = 0 < C_{TH}$ indicates that connections from S_{10} and S_{11} to H_{13} are unimportant and can be removed from neural network (pruning). An alternative option is to set weights of the unimportant connections to 0 if those connections are not removed.

To construct the structured autoencoder, we employ the untied weights. The input layer and the first hidden layer of autoencoder can be constructed similarly as the structured TDNN. The rest of autoencoder layers and bias nodes are described in section III-B.

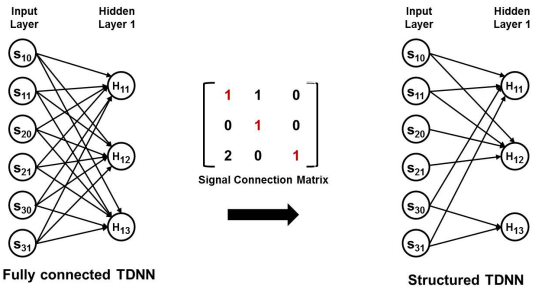

Fig. 2: Event Ordering Relationship Based TDNN Structure.

We should note that except for the structured networks we proposed above, there are many extensions to the structured networks. Instead of only detecting the immediate relations as the matrix relation between the 1st and 2nd layer, we can also detect indirect relations, i.e., we form the matrix with a delay of 2. For example, if we have $S_i => S_j => S_k => ...$ as a relation, instead of setting $e_{ij} = 1$ we should set $e_{ik} = 1$, and therefore, we can essentially form another matrix relations. By reasonably assembling these relations into the neural networks, we can have structured networks with more than one relation. However, both implementations and theory of the framework needs more caveats, and in this paper we focus on the simple networks with only one structured relation.

V. Neural Network Learning and Optimization

For the supervised learning with the time delay feedforward neural network, we formulate anomaly detection into classification problem. For this purpose, we apply classic *Sigmoid* activation function and the cost function is *Cross-Entropy* along with *Softmax Function*. A weight decay regularizer is also applied.

For the unsupervised learning with the time delay autoencoder, we prefer overfitting and omit the regularization. We also use *Sigmoid* activation function. However, the *Least Square* cost function is used.

For both learning cases, we employ the popular *stochastic gradient descent (SGD)* based optimizers, which covers a large class of algorithms. The momentum SGD [11] is one of the most widely used variant. Other variants include data-dependent AdaGrad [12], ADAM [13], SVRG [14], and SARAH [15]. All these optimizers have comparable performance and we present results based on Momentum SGD.

VI. Numerical Experiment Results

In order to validate the proposed anomaly detection methodologies, we performed numerical experiment with the data collected from real manufacturing production processes. We conducted the experiments using the unstructured neural networks and the structured neural networks. We compared the performance of the

structured neural networks with the unstructured neural networks and other popular methods.

We implemented all anomaly detection methods using Python TensorFlow. However, our current TensorFlow implementation does not realize the pruning. As a result, the structured neural networks are implemented as fully connected neural networks, in which the unimportant weights are initialized to 0 as described in section IV.

A. Data Description

We collected data using 151 sensors. The format of the data is digital. Data are collected every 10ms. At the beginning of data collection, an unique index is assigned to represent the collection time. The index is then increased by 1 for each new collection. One training data set is collected under normal operation condition and it contains 406,701 samples for all sensors. Two test data sets are collected under abnormal condition, where the anomaly happens after a specific time. The corresponding problematic index can be seen in Table I.

Test Data Set	Index Range	Anomalous Index
Set 1	21302 – 54161	49303
Set 2	72853 – 105817	100854

TABLE I: Statistics of Test Data Sets.

B. TDNN and Structured TDNN

We conduct anomaly detection experiments via supervised learning using TDNN and structured TDNN. We aim to detect whether a test data set is normal or not. If the test data set is abnormal, the anomaly occurrence time, i.e., the anomalous index, is detected. To perform supervised learning, we use training data set and test data set 1 for training and use test data set 2 for testing. We have validated various network structures and the performance is robust. We present results of the network, where time delay window is 5 and both TDNN and structured TDNN come with a layer structure of 755-151-50-2.

Fig. 3 shows training cost and test error, where data sampling rate 1s, 100ms and 10ms indicate 1 of 100 data samples, 1 of 10 data samples and all data samples used, respectively. TDNN and structure TDNN have similar training cost and test error, which stabilize around 600 data passes and improve as data sampling rate increases. For 100ms data sampling rate, TDNN exhibits unusual peaks.

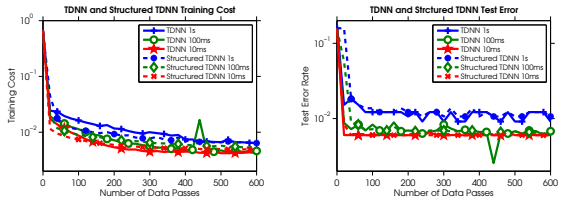

Fig. 3: TDNN and Structured TDNN Training Cost and Test Error.

Both TDNN and structured TDNN classify test data set 2 as anomalous. The anomalous indices detected by both methods are larger than actual anomalous index 100854, i.e., the anomaly occurrence times detected by both methods are later than actual anomaly occurrence time. For 100ms data sampling rate, the anomalous index detected by TDNN is 101094 and the anomalous index detected by structured TDNN is 101084. It indicates that anomaly occurrence time detected by TDNN is 2.4 seconds later than actual anomaly occurrence time and anomaly occurrence time detected by structured TDNN is 2.3 seconds later than actual anomaly occurrence time, i.e., structured TDNN detects the anomaly 100 milliseconds early than the TDNN. Therefore, structured TDNN is more accurate than TDNN. Both TDNN and structured TDNN have similar training time, e.g., about 28 minutes for 100ms data sampling rate.

C. Autoencoder and Structured Autoencoder

We also conduct anomaly detection experiments via the unsupervised learning using autoencoder and structured autoencoder. We apply time delay autoencoder with the untied weights. With a time delay window 5, both autoencoder networks come with a layer structure of 755-151-50-755. In this case, we only use training data set for training and two test data sets for testing. We aim to detect if each test data set is normal or not. For anomalous test data set, the anomaly occurrence time is detected. Both test data sets are detected as anomalous. Both networks have detected same anomalous index. For example, for 10ms data sampling rate, the detected anomalous index is 49413 for test data set 1 and is 101061 for test data set 2. These indices represent the detected anomaly occurrence times that are accurate to within 2 seconds.

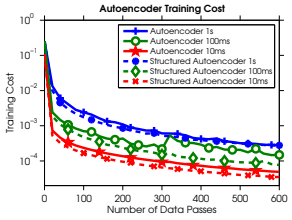

Fig. 4: Autoencoder and Structured Autoencoder Training Cost.

Fig. 4 shows training cost for autoencoder and the structured autoencoder. The training costs stabilize around 600 data passes, which means that the training process is trustful and reliable. It can be seen that the structured autoencoder has lower training cost than autoencoder for all data sampling rates. The corresponding training time is about 18% higher than the TDNN training time. For 100ms data sampling rate, Fig. 5 shows the squared error between the recovered signals and the original signals for both autoencoder and the structured autoencoder after 600 data passes. For both test data sets, we

can see that autoencoder has higher false alarms than the structured autoencoder. The structured autoencoder improves the total test error from 2.31% to 1.79%, i.e. a 20% improvement.

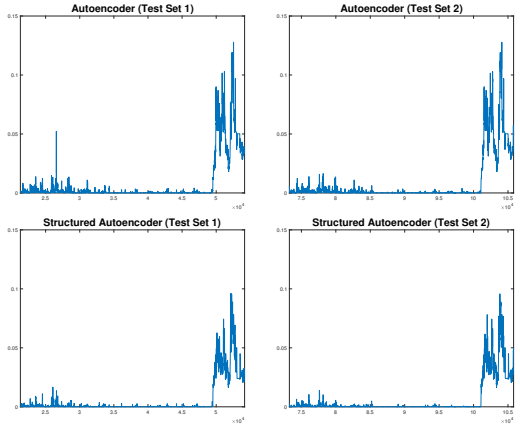

Fig. 5: Test Error of Autoencoder and Structured Autoencoder.

D. Comparison with the State-of-the-Art

To verify the proposed anomaly detection methods broadly, we compare our structured autoencoder to TDNN Regression (TDNNR) and other popular methods that can be classified as three main methodologies, i.e., distance-based method (Local Outlier Factor), tree-based method (Isolation Forest) and kernel method (One-Class SVM). We well-tuned all the models in our experiments and present the best results based on our current datasets. It can be observed from the Table II, the structured autoencoder performs the best among all considered methods; while isolation forest and LOF work well. With the structured autoencoder, the misclassification error is reduced by 64% compared to the best state-of-the-art method (LOF).

Methods	Test Errors
Isolation Forest	5.65%
Local Outlier Factor (LOF)	5.01%
One-Class SVM	68.12%
TDNNR	14%
Autoencoder	2.31%
Structured Autoencoder	1.79%

TABLE II: Performance of Different Anomaly Detection Methods.

VII. Conclusion

We have developed a novel neural network architecture with the structure constructed from the event ordering relationship to detect anomaly in the manufacturing processes. As far as we know, they are the first time delay neural network and autoencoder that enforce a sparsity based on the event ordering relationship. With the numerical comparison between vanilla TDNN and structured TDNN, vanilla autoencoder and structured autoencoder as well as structured autoencoder and other popular anomaly detection methods, we demonstrate advantages of the structured neural network over vanilla neural network and other methodologies.

References

[1] L. Marti, N. Sanchez-Pi, J. Molina, and A. Garcia, "Anomaly detection based on sensor data in petroleum industry applications," http://www.mdpi.com/journal/sensors, pp. 2774–2797, 2015.

[2] D. Nikovski and M. Jones, "Method for anomaly detection in discrete manufacturing processes," 2015. [Online]. Available: https://www.google.com/patents/US20150277416

[3] S. Windmann, A. Maier, O. Niggemann, and et al, "Big data analysis of manufacturing processes," in *12th European Workshop on Advanced Control and Diagnosis*. ACD, 2015.

[4] G. Susto, M. Terzi, and A. Beghi, "Anomaly detection approaches for semiconductor manufacturing," in *27th International Conference on Flexible Automation and Intelligent Manufacturing*, 2017.

[5] L. Allen and D. Tilbury, "Anomaly detection using model generation for event-based systems without a preexisting formal model," *IEEE Transactions on Systems, Man, and CyberneticsPart A: Systems and Humans*, vol. 42, no. 3, pp. 654–668, 2012.

[6] B. Jakimovski, *Biologically Inspired Approaches for Locomotion, Anomaly Detection and Reconfiguration for Walking Robots*. Springer-Verlag Berlin Heidelberg, 2011.

[7] B.-D. Kang, J.-W. Lee, J.-H. Kim, and et al, "An intrusion detection system using principal component analysis and time delay neural network," in *7th International Workshop on Enterprise Networking and Computing in Healthcare Industry*, 2005.

[8] A. Alghassi, M. Samie, and S. Perinpanayagam, "Stochastic rul calculation enhanced with tdnn-based igbt failure modeling," *IEEE Transactions on Reliability*, vol. 65, no. 2, pp. 558–573, 2015.

[9] K. K. Reddy, S. Sarkar, V. Venugopalan, and M. Giering, "Anomaly detection and fault disambiguation in large flight data: A multi-modal deep auto-encoder approach," in *Annual Conference of the Prognostics and Health Management Society*, 2016.

[10] R. Chalapathy, A. K. Menon, and S. Chawla, "Robust, deep and inductive anomaly detection," in *European Conference On Machine Learning and Principles and Practice of Knowledge Discovery*, 2017.

[11] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, 1964.

[12] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2014.

[14] J. Konečnỳ, J. Liu, P. Richtárik, and M. Takáč, "Mini-batch semi-stochastic gradient descent in the proximal setting," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 2, pp. 242–255, 2016.

[15] L. M. Nguyen, J. Liu, K. Scheinberg, and M. Takáč, "Stochastic recursive gradient algorithm for nonconvex optimization," *arXiv preprint arXiv:1705.07261*, 2017.