International Journal of Production Research

ISSN: 0020-7543 (Print) 1366-588X (Online) Journal homepage: https://www.tandfonline.com/loi/tprs20

Big data driven jobs remaining time prediction in discrete manufacturing system: a deep learning-based approach

Weiguang Fang, Yu Guo, Wenhe Liao, Karthik Ramani & Shaohua Huang

To cite this article: Weiguang Fang, Yu Guo, Wenhe Liao, Karthik Ramani & Shaohua Huang (2019): Big data driven jobs remaining time prediction in discrete manufacturing system: a deep learning-based approach, International Journal of Production Research, DOI: 10.1080/00207543.2019.1602744

To link to this article: https://doi.org/10.1080/00207543.2019.1602744

Published online: 12 Apr 2019.

International Journal of Production Research, 2019
https://doi.org/10.1080/00207543.2019.1602744

Big data driven jobs remaining time prediction in discrete manufacturing system: a deep learning-based approach

Weiguang Fang[a,b*], Yu Guo[a], Wenhe Liao[a,c], Karthik Ramani[b] and Shaohua Huang[a]

[a]*School of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, People's Republic of China;* [b]*School of Mechanical Engineering, Purdue University, West Lafayette, IN, USA;* [c]*School of Mechanical Engineering, Nanjing University of Science and Technology, Nanjing, People's Republic of China*

(*Received 4 September 2018; accepted 25 March 2019*)

Implementing advanced big data (BD) analytic is significant for successful incorporation of artificial intelligence in manufacturing. With the widespread deployment of smart sensors and internet of things (IOT) in the job shop, there is an increasing need for handling manufacturing BD for predictive manufacturing. In this study, we conceive the jobs remaining time (JRT) prediction during manufacturing execution based on deep learning (DL) with production BD. We developed a procedure for JRT prediction that includes three parts: raw data collection, candidate dataset design and predictive modelling. First, the historical production data are collected by the widely deployed IOT in the job shop. Then, the candidate dataset is formalised to capture various contributory factors for JRT prediction. Further, a DL model named stacked sparse autoencoder (S-SAE) is constructed to learn representative features from high dimensional manufacturing BD to make robust and accurate JRT prediction. Our work represents the first DL model for the JRT prediction at run time during production. The proposed methods are applied in a large-scale job shop that is equipped with 44 machine tools and produces 13 types of parts. Lastly, the experimental results show the S-SAE model has higher accuracy than previous linear regression, back-propagation network, multi-layer network and deep belief network in JRT prediction.

Keywords: big data; job shop; jobs remaining time prediction; stacked sparse autoencoder; deep learning

1. Introduction

Advances in big data (BD) and Internet of things (IOT) have driven the implementation of smart manufacturing (Kusiak 2017a; Lee, Kao, and Yang 2014). The radio frequency identification (RFID) technique enables the connectivity among work in process (WIP), machine tools, materials, workers and other physical assets in production line (Ding and Jiang 2018). Due to networked RFID in more and more modern factories (Zhong et al. 2015), the manufacturing BD are generated at a high velocity. Meanwhile, with the advent of BD, deep learning (DL), as one of the most currently remarkable machine learning algorithms, plays an increasing important role in BD solutions (Zhang et al. 2018; Wang et al. 2018), including building advanced predictive models for production management and control (Kusiak 2017b).

On the other hand, the increasing level of customisation in combination with increased product complexities lead to the manufacturing firms to shift from make-to-stock (MTS) to make-to-order (MTO) production mode (Gansterer 2015). In this new scenario, assigning a applicable due date, estimating an accurate order completion time and forecasting a reliable cycle time are all crucial to meet the customers' demands. Regarding this, previous studies put a lot of emphasis on predicting the due date (Nguyen et al. 2014; Yao et al. 2015), order completion time (Wang and Jiang 2017; Gansterer 2015) and cycle time (Chen and Wang 2013; Wang and Zhang 2016), which are very helpful for decision-making in production planning and order acceptance.

In the future manufacturing, dynamic business and engineering process require factory to respond flexibly to disturbances in manufacturing processes (Brettel et al. 2014). Thus, for the management insight of manufacturing system, it is essential to acquire real-time progress information of dispatched orders to balance the differences between original production schedule and actual manufacturing processes. The jobs remaining time (JRT) represents the remaining time required to complete the rest of jobs in the order, which is instantaneous prediction under the real-time conditions. Predicting the JRT is the dynamic response to the ever-changing production status, which enables manager to foresee the deviation of

*Corresponding author. Email: fang_weiguang@hotmail.com, fang200@purdue.edu production planning and actual execution, and trigger the decision-making of real-time online scheduling in manufacturing system.

In ideal conditions, the JRT would be same with the planned completion time of remaining jobs as original production schedule. However, it is commonly seen that the actual completion time is not in accordance with the planned time due to unexpected production disturbances. The disturbances in production can be divided into two categories (Zengqiang and Mingxin 2009; Liu et al. 2014; Wang and Jiang 2018): (i) dominant disturbances (such as machine tools breakdown, arrival of urgent order, large volumes of unqualified WIPs) which shut down the original production schedule and result in the large fluctuation of JRT. (ii) Recessive disturbances (such as the deviation of WIP arrival time, processing time and preparation time) which do not directly shut down the original schedule, instead, change the production plan little by little and bring about the small fluctuation of JRT. However, once the recessive disturbances accumulate by time, it will turn into dominant disturbance and lead to inevitable rescheduling. Here, the JRT prediction should consider a dynamic and intricate setting where: (i) the orders are continuously released over time; (ii) the job shop processes multi-orders at the same time and production status change in real time and (iii) the diverse disturbances randomly emerge. In this regard, it is difficult for the existing approaches to approximate today's manufacturing system and make the accurate and efficient JRT prediction.

In this study, based on RFID driven BD collection in the job shop, the 'BD + DL' method is designed to aid operations managers to utilise advanced predictive model to take a deep dive into shop floor data, reveal the intricate relations and influences between different inputs (data from orders, job shop status and machine conditions) and output (JRT changes), and then achieve better accuracy and efficiency of JRT prediction to further promote the precise control of discrete manufacturing system.

The rest of this paper is organised as follows. Section 2 reviews the related studies relevant to our research. A BD analysis approach for JRT prediction is proposed in Section 3. The experiments and statistical results are presented in Section 4. Finally, the conclusions and suggestions for future study are outlined.

2. Literature review

2.1. Analytical and simulation methods

From methodology perspective, the existing methods in this stream could be categorised into three classes: analytical, simulation and artificial intelligence (AI) methods.

The analytical method mainly depends on constructing mathematical models such as queuing, Markov chain. Most of them derive the flow time distributions under some restrictive assumptions of Markovian property by queuing theory (Altendorfer and Jodlbauer 2011), and the lead time or due time is optimised to meet some objectives (Li et al. 2015). It has to be admitted that the analytical model is adaptable to some not complicated and ideal conditions. However, only several processes are included in these models and also the real manufacturing system involves much non-Markovian processes such as machine breakdowns, rush orders that should be considered.

On the other hand, the simulation method is also widely used in this stream with the development of discrete events simulation techniques. Vinod and Sridharan (2011) constructed a simulation modelling to estimate the due date in a produce-to-order environment. Hsieh, Chang, and Chien (2014) proposed a progressive simulation metamodelling method to create an optimal schedule policy and in reverse, the due date could be set equal to the scheduled completion date. However, since the products' process become more complex and the manufacturing system is getting larger, it is difficult to run a sophisticated simulation model.

2.2. Features design in AI method

The AI method, which treats the complex manufacturing system as a black box and designs the input features and regression models to make predictions, has gained much attention from researchers. For the JRT (or due date, flow time, cycle time, etc.) prediction tasks, choosing the informative and discriminating feature set is no doubt a crucial process for the efficiency and accuracy of results. Tirkel (2013) selects 19 features (wafer lot serial no., load time, processing time, etc.) as key measures in the operations to forecast flow time in semiconductor manufacturing. Wang and Zhang (2016) designed the candidate feature set (such as waiting queue length, total number of WIP, and so on) and proposed a feature selection method based on entropy to predict the cycle time. Mirshekarian and Sormaz (2016) designed the statistical features (e.g. mean, median, min, max, etc.) on the simulated job shop data to establish the correlation of these features with optimal make-span. The features in existing studies are usually manually designed and selected, which are difficult to describe the characteristics in complex production line. What's more, although some studies have already considered the load of production line, they neglected the real-time status of the job shop.

2.3. Neural networks for JRT prediction

The JRT prediction is a typical regression task in machine learning, and the neural network (NN) as well as its variants have been used recently (Yao et al. 2015; Silva et al. 2016; Ribeiro 2016). A fuzzy back propagation network is proposed by Chen and Wang (2013) to forecast the cycle time in a wafer fabrication factory. A back propagation NN is employed in Wang and Zhang (2016) for the cycle time prediction after feature selection. However, due to multi-collinearity among manufacturing data measurements, high dimension feature space, and highly stochastic and non-linear nature of JRT prediction task, the traditional shallow NN models are lack of generalisation and fitting capabilities to deal with today's big manufacturing data. In addition, most of past studies require domain expertise for feature extraction to reduce the input dimensionality, which leads to the final prediction results heavily depend on the engineered features.

As a breakthrough in AI, DL has achieved outstanding performance that is far beyond that of classical machine learning methods in many domains including speech, natural language, vision and playing games (Liu et al. 2017). DL enables the learning of complex features through automatic processing of high dimensional data. It represents complicated feature via a cascade of multiple layers, rather than manual feature 127 representation of data with domain expertise. In the manufacturing domain, there are some tentative studies utilising DL in fault diagnosis (Lu, Wang, and Zhou 2017; DiBiano and Mukhopadhyay 2017), remaining useful life prediction (Ren et al. 2017). Wang and Jiang (2017) deployed the RFID for tracking the WIP conditions (waiting,machining) to get more specific production data and developed the deep brief NN in prediction.

In addition, prior works and state-of-the-art do not count to today's manufacturing system and JRT prediction in production:

(1) It is necessary to utilise very fast techniques for JRT prediction since it requires the capability of real-time response, which means the JRT should be computed in a few seconds. As a result, both of analytical and simulation methods may lose their applicability in a large-scale dynamic system where disturbances and changes randomly emerge, and require real-time decision-making.
(2) The global features, which are selected from historical or simulation data, were usually used as input for different regression models in past studies. However, the global features are the overall measuring properties (such as total WIPs number, average machines utilisation) and lack of many other detailed information such as loading and quality problems in the job shop. As a result, these global features do not comprehensively reflect the real-time operating conditions of job shop.
(3) It could be observed from the past studies that different NNs have been introduced into estimation of due time/cycle time/lead time. However, with the accumulation of manufacturing BD, there are still limit models designed for learning high-level representative features and better approximation of the prediction value. In addition, few studies have considered the production disturbances when formalising the input for the NN model, which also have great influence on prediction accuracy.

With the help of IOT, BD and DL techniques, this study presents a BD driven method for real-time JRT prediction in discrete manufacturing system. It is designed to utilise advanced predictive model to take a deep dive into shop floor real-time data and make instant production predictions. The method considers the dynamicity of job shop and requires the operating status when the job shop under normal and disturbing conditions by collection of various production data. Furthermore, the Deep Stacked Sparse Autoencoder (S-SAE) model is designed, which enables the model to comprehensively learn highly varying status of manufacturing system for JRT prediction.

3. The BD driven JRT prediction

3.1. Raw data collection

The main goal of manufacturing system is to fabricate the ordered parts and to deliver the finished products. The job shop is a kind of manufacturing system, is defined as below:

Definition 1 The job shop is a type of manufacturing process in which small batches of a variety of custom products are made. In the job shop, similar equipment or functions are grouped together, when the order arrives in the shop floor, the part being worked on travels throughout the various areas according to a sequence of operations.

By applying IOT-related technologies in manufacturing job shop, the manufacturing objects are able to be identified and monitored, and production data could be collected in real time. Figure 1 shows a brief IOT configuration solutions in the job shop, on one hand, the important resources, such as devices, tools, vehicles, materials, WIP and finished products are labelled with RFID tags for their traceability. On the other hand, since the similar functional machines are grouped together

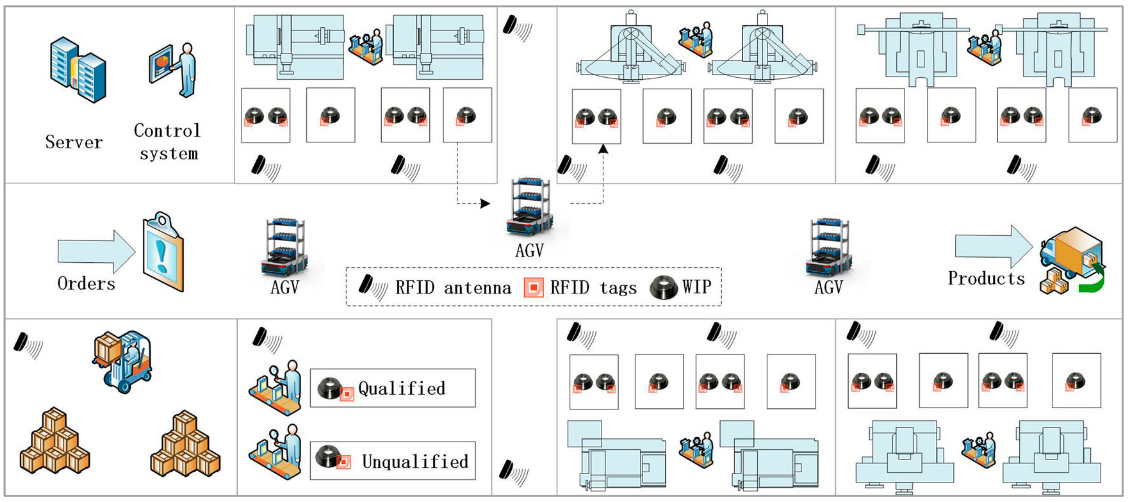

Figure 1. IOT deployment and job shop layout.

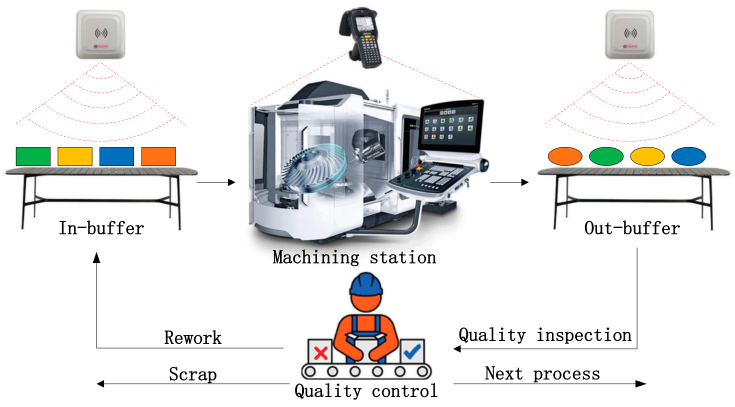

Figure 2. WIPs' steps in each machining station.

in one area and the process route is fixed, RFID readers are installed in the main fixed manufacturing resources, such as CNC centre and key equipment in the production line. More specifically, the fixed RFID reader and antenna are deployed to the in-buffer and out-buffer respectively, to identify the 'enter', 'stay' and 'exit' status of WIPs. Besides, the hand-held RFID reader is deployed to the workstation to record the in-time and out-time of the part machining and the quality inspection area. The specific RFID configuration in each machining station could also be seen in Figure 2.

After the deployment of IOT in Job shop, the manufacturing objects could be traced and monitored in real time. In this step, the goal is to cover as much information as possible to describe all the factors affecting the order completion time. The raw data set are defined as follows:

Definition 2 Each WIP status in its production process is defined as $WIP = < part\_ID, operation\_NUM, in\_Time, start\_Time, finish\_Time, out\_Time >$, where $part\_ID$ is a code uniquely identifying a WIP; $operation\_NUM$ represents the operation step in process routes; in_Time denotes the moment a WIP enters in-buffer; $start\_Time$ denotes the moment a WIP starts machining; $finish\_Time$ denotes The moment a WIP finishes machining; out_Time denotes the moment a WIP enters out-buffer.

Definition 3 Each working station in the job shop is defined as $WS = < machine\_ID, operation\_Type, team\_ID, operation\_ID, machine\_Status >$, where $machine\_ID$ denotes the unique identification for the machine, $operation\_Type$ represents the process functionalities like cut, drill; $team\_ID$ means the maintenance team for the machine; $operator\_ID$ denotes the operator in charge of the machine; $machine\_Status = < 1, 2, 3, 4 >$ is used to represent the real-time situation of the machine, means available, in-working, maintenance, breakdown.

Definition 4 Each released order in the job shop is defined as $Order = < order\_ID, part\_Type, part\_Quantity, release\_Date, lead\_Date, priority, process\_Route >$, where $order\_ID$ represents the unique code of the order; $part\_Type$ denotes the categories of parts in the order; *part_Quantity* denotes the amount of each part category; *release_Date* and *lead_Date* indicate the order start time and products delivered time respectively; *process_Route* denotes the detail operation sequences of job. Generally, the raw data set are collected and stored in distributed database, including both real-time collected data and historical data, and covering the information about orders composition, job shop real-time status and machine tools condition.

3.2. *Candidate dataset design*

3.2.1. Candidate dataset formalisation

The candidate dataset in this paper is extracted from the raw data set and covered the three general aspects which affect the JRT (Table 1). The explicit production disturbance such as machine breakdown and unqualified workpieces is included in the feature set. Further, the implicit disturbances or production uncertainties are implied in the collected dataset.

The job shop production process has the following three assumptions: (i) the operator selects the WIP to process following the first-in-first-out scheduling rule; (ii) transportation time is included in the waiting time and set-up time is included in the processing 214 time. In this section, the features for JRT prediction are formalised as follows:

(1) Real-time production task (TC^t)

In practice, multiple production orders are regularly processed simultaneously, and in this research, the production task is used to represent total jobs released from orders in the job shop. Normally, most production lines have the fixed types of products, and the difference between orders is the amount of different part types. Assuming the manufacturing system is capable of processing N types of parts, and there are k orders released in the job shop, which means the production task set could be denoted as $T = \{O_1, O_2, O_3, \ldots, O_k\}$, and in an order, the number of each part type to be processed on the time t could be represented as $O_k = \{P_{k,1}, P_{k,2}, \ldots, P_{k,N}\}$. Hence, the composition of undertaking task on the time t could be formalised as follows:

$$TC^t = \left(\sum_1^K P^t_{k,1}, \sum_1^K P^t_{k,2}, \ldots, \sum_1^K P^t_{k,N}\right). \tag{1}$$

(2) Real-time production status (PS^t)

Each WIP in production line has fixed process routing and go through multiple operations, and in each operation, a WIP in a machining station would experience four steps, which is shown in figure, including: (i) get to in-buffer to wait processing; (ii) be processed by machine tool, (iii) get to out-buffer to wait quality inspection; (iv) be inspected and decided to rework, scrap or get to next process. The information is captured from the RFID devices deployed in every machining station, which could record the waiting sequences in-buffer and out-buffer (Wang and Jiang 2017), and also the WIP under processing and inspecting.

- `Total WIP level` (N_WIP^t) – The total WIP level at the time t could be measured by counting the WIP number where the first process has been started and the final process has not been completed. The N_WIP^t denotes the total number of all types of WIPs in production line.

Table 1. Candidate dataset and data examples.

Categories	Candidate dataset	Data examples
Real-rime production task	Task composition	22,7,17,5,21, . . . ,23,24,15,6
	Total WIP level	422
Real-time production status	Waiting sequence in buffer	13,8,1,1,1,3,1,1,5
	Waiting sequence our buffer	13,1,8,8,5,1,13,3
	WIP in processing	5,3,1, . . . ,4,7,11
	Operation planned time	5,15,12, . . . ,9,10,20
	WIP in quality inspection	1,1,3, . . . ,6,5,12
	Quality ratings	1,1,1,1,3, . . . ,2,4,1,1,3,1
Machine tools condition	Machine utilisation rate	81,55,0,15,70,90.40,15
	Working time	1420,370,50, . . . ,2156,30
	Real-time machine status	1,1,1,1,1,1,2,1,1,3,1,1,4
Output	Jobs remaining time	742

- `WIP in waiting (`WS^t_{in}`)` – Supposing there are x WIP at in-buffer area of machining station m waiting to be processed at time t, so the WIP waiting sequence at time t WS^t_{in} could be formalised as follows:

$$WS^t_{in} = (WIP^t_{m-in,1}, WIP^t_{m-in,2}, \ldots, WIP^t_{m-in,x}), \tag{2}$$

  where $WIP^t_{m,x}$ denotes the x the WIP waiting at in-buffer of machining station m at time t. Similarly, in the meantime, WS^t_{out} and $WIP^t_{m-out,y}$ are used to respectively represent the WIP waiting sequence of out-buffer and y the wip at out-buffer area of machining station m waiting to be quality inspection.
- `WIP in processing (`WP^t_m`)` – The WIP in processing means the WIP has already tracked in machine tools to start an operation, so the status of WIP in processing WP^t_m could be denoted as

$$WP^t_m = (WIP^t_m, PT_m), \tag{3}$$

  where WIP^t_m is the WIP processing at machining station m at time t and the planned time of WIP^t_m in this operation is PT_m.
- `WIP in quality inspection (`WQ^t_m`)` – The processing quality inspection is arranged after each operation, and the status of WIP in quality inspection WQ^t_m could be represented as

$$WQ^t_m = (QI^t_{wip,m}, QR_{wip,m}), \tag{4}$$

  where $QI^t_{wip,m}$ is the WIP in quality inspection after processing in m at the time t and $QR =< 1, 2, 3, 4 >$ represents the quality ratings (e.g. qualified, rework, qualified after re-work, scrap respectively) of the WIP.

(3) Machine tools condition (MC^t)

Machine tools condition is one of the uncertain factors that affect the production schedule. In this research, the utilisation, real-time condition and working time of each machine tool are taken as influential factors for jobs completion prediction. Supposing there are mth machining tools (stations) in the production line, the machine tool condition MC^t_m at time t could be formalised as

$$MC^t_m = (WS^t_m, UR_m, WT^t_m), \tag{5}$$

where $WS^t_m =< 1, 2, 3, 4 >$ is the real-time condition of the machine tool, which conditions could be divided into 'in-processing', 'set-up', 'vacant' and 'in-maintenance'. The UR_m means the average utilisation rate of the machine tool during time *t-1* to *t*. WT^t_m denotes the continuous working time since last maintenance.

As a consequence, the candidate feature set FS for JRT prediction could be formalised as follows:

$$\begin{aligned} FS = (TC^t, PS^t, MC^t) = \Bigg(& \sum_1^K P^t_{k,1}, \sum_1^K P^t_{k,2}, \ldots, \sum_1^K P^t_{k,n}, N\_WIP^t \\ & WIP^t_{m-in,1}, WIP^t_{m-in,2}, \ldots, WIP^t_{m-in,x} \\ & WIP^t_{m-out,1}, WIP^t_{m-out,2}, \ldots, WIP^t_{m-out,y} \\ & WIP^t_m, PT_m, QI^t_{wip,m}, QR_{wip,m} \\ & WS^t_m, UR_m, WT^t_m \Bigg) \end{aligned} \tag{6}$$

3.3. The S-SAE model for JRT prediction

The Deep Stacked Sparse Autoencoder (S-SAE) model, which is one kind novel DL model, is designed in this research to extract high-level representative features from candidate shop floor data, and furthermore, make accurate and robust JRT predictions by concatenating regression NN. The S-SAE model employs the hidden layer from each SAE as the building block to construct a deep NN.

3.3.1. The basic SAE model

An autoencoder is a symmetrical NN, which reproduces the input dataset as the target output, and aims at learning latent feature representations from shopfloor candidate dataset in this paper. The architecture of basic AE is shown in Figure 3.

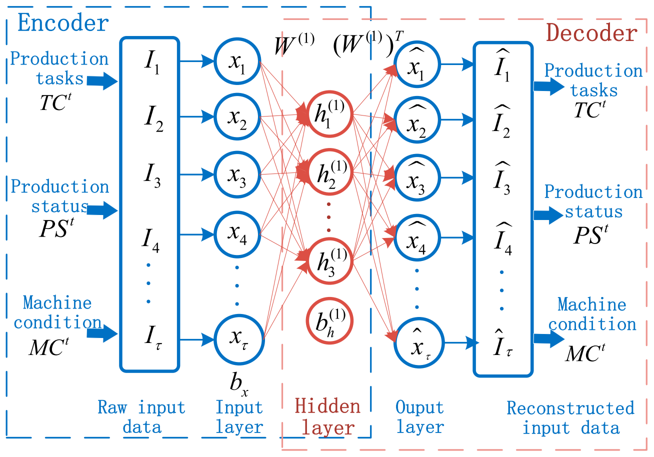

Figure 3. Basic autoencoder.

Overall, the AE model includes a input layer, a hidden layer and a output layer. The encoder part transforms the candidate shopfloor data x_τ into latent representation $h(x)$ based on (7), and it could be regarded as the new features representation for input data. Reversely, in (8), the output layer is a decoder for re-constructing the hidden representation $h(x)$ as an approximation $\widehat{x}$. The typical form of encoder and decoder is shown as follows:

$$Encoder: h(x) = \sigma(W^{(1)}x + b_x), \tag{7}$$

$$Decoder: \widehat{x}(h) = \sigma'(W^{(2)}h + b_h^{(1)}), \tag{8}$$

where $W^{(1)}$, $W^{(2)}$ and b_x, $b_h^{(1)}$ are the weight matrix and bias for encoder and decoder respectively. σ and σ' represent the activation function, where the rectified linear unit (ReLU) (Glorot, Bordes, and Bengio 2011) is employed.

However, the AE merely maps the input (x) to hidden layer $(h(x))$ and recovered the input data to $(\widehat{x})$ by its output, leading to the ineffective to extract latent features. The SAE, which is an extension of the basic AE, is able to extract more meaningful features by employing a sparse penalty term into AE (Olshausen and Field 1996). Specifically, the sparse penalty works at the hidden layer $h(x)$ to control the amount of 'inactive' and 'active' neural units, so that the learned features in the hidden layer are of the constraint rather than merely copying inputs.

In order to achieve the sparse representation and minimise the discrepancy between x and $\widehat{x}$, the cost function of an SAE which comprises two items is written as follows:

$$Cost(W,b) = \left[\frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}\parallel x_i - \widehat{x_i}\parallel^2\right] + \gamma\sum_{j=1}^{n} KL(\rho \parallel \widehat{\rho_j}). \tag{9}$$

The first term is a mean-squared error (MSE) term which represents the discrepancy between x_i and $\widehat{x_i}$ over the whole dataset N. According to the second term, n denotes the number of neurons in the hidden layer and j is the summing over the neurons. $KL(\rho \parallel \widehat{\rho_j})$ represents the Kullback–Leibler divergence between $\widehat{\rho_j}$ (average activation of neurons) and desired activation neurons ρ. A forward pass over entire training data is utilised to compute the $\widehat{\rho_j}$ to get sparse error and MSE, then the back-propagation (BP) algorithm is introduced to update parameters W and b. After this, the latent sparse feature could be learned by SAE.

3.3.2. S-SAE + regression predictor

As shown in Figure 4, the S-SAE model is generated by stacking multiple AEs to create a deep NN. Specifically, supposing there are two stacked layers as the hidden layers in the designed S-SAE, the first stacked layer is trained as an SAE by taking the shop floor data as input, then the output of the first stacked layer is employed as the input of the second SAE. In this way, the multiple SAEs could be hierarchically stacked to create an S-SAE.

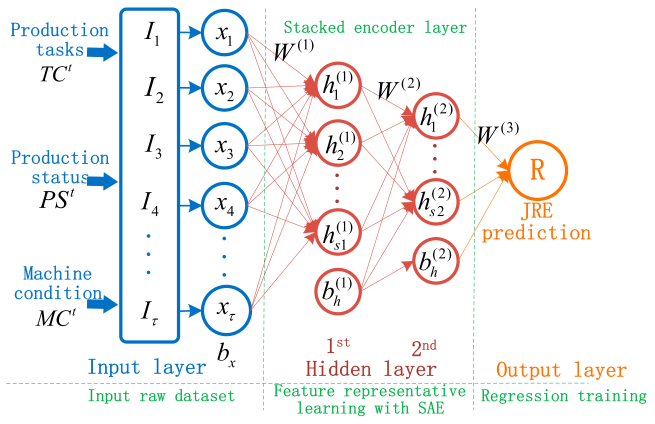

Figure 4. SAE + regression layer for feature representative learning and JRT prediction.

Furthermore, the novel dropout layer (Srivastava et al. 2014) and batch normalisation layer (Ioffe and Szegedy 2015) are introduced in the designed S-SAE model, which are described as follows:

- `The dropout layer`: Dropout is a novel technique in DL, which could effectively reduce the ‘overfitting’ phenomenon when training a deep NN. In general, the ‘overfitting’ means the poor performance on the test set compared to which on the train set. In the designed S-SAE model for JRT prediction, the dropout layer is added after the SAEs to avoid the extraction of similar features and also in the fully connected (dense) layer to refrain from ‘overfitting’ problem.
- `The batch normalisation layer`: Training a deep NN is complicated by the fact that the distribution of each layer’s inputs changes during training, as the parameters of the previous layers change (Ioffe and Szegedy 2015). The batch normalisation potentially helps to solve this problem by normalising the inputs of each layer and also accelerate the training process. In the designed S-SAE model, the batch normalisation layer is added behind the stacked layer to reduce the impacts of earlier stacked layers on the later ones in S-SAE, which to some extent makes the layers independent with each other.

After extracting the features by stacking AEs, we set the output layer of S-SAE as the linear regression, which is only one neuron with Linear activation function. As a consequence, a deep NN, which comprises multiple SAEs and regression predictor, is constructed for JRT prediction.

3.3.3. The training algorithm for S-SAE + regression predictor

For the past shallow layers NN, it is straightforward to train them by utilising the BP method with gradient decent optimisation. However, according to the deep architecture, the gradient-based optimisation which starting from random initialisation appears to often get stuck in poor solutions (Bengio et al. 2007).

The unlabelled shop floor data which include production tasks, production status and machine condition are first used to pre-train the SAE model by the following steps:

Step 1 Initialise the learning rate ι, sparse weight γ, dropout rate μ and randomly initialise the weight W and bias b in each *SAE*.
Step 2 Feed the input shopfloor data X into $SAE^{(1)}$ to get primary representations in the hidden layer $h^{(1)}(X)$ and get weights $W^{(1)}$ after trained.
Step 3 Feed the $h^{(1)}(X)$ as input into second $SAE^{(2)}$ to get the representations $h^{(2)}(X)$ and trained weights $W^{(2)}$.
Step 4 Iterate as Steps 2 and 3 to get the desired numbers of $h^{(n)}(X)$ in *SAE*.
Step 5 Stack the layers of $h^{(n)}(X)$, and treat the last one as the input to the regression predictor, then form the final architecture S-SAE with several $h^{(n)}(X)$ and a final regression predictor.
Step 6 Set the pre-trained $(W^{(1)}, W^{(2)}, \ldots, W^{(n)})$ to the final deep NN, and take the shopfloor data X as input and JRT data as output.
Step 7 Conduct the BP algorithm to update the weights of all layers and fine-tune the overall network in a supervised way.

4. Experiment design

4.1. The experiment environment

In order to verify the applicability of the BD driven JRT prediction, the proposed method is implemented in a large-scale job shop for machining parts of aeroengine in China. The job shop environment, the deployment of IOT devices and the different modules of production information records are shown in Figure 5. The chosen job shop consists of 44 working stations in total, which include (A) machining centre for 5, (B) milling machine for 4, (C) lathe machine for 8, (D) grinder for 7, (E) drilling machine for 4, (F) reaming machine for 4, (G) electric discharge machine for 2 and (H) fitter bench for 10. There are different sizes of in-buffer and out-buffer areas deployed at each working station. According to the policy of the enterprise, the processing quality is inspected after each operation. This production line mainly machines 13 kinds of parts, the parts type and processing route could be seen in Table 2, where the process route is represented by the working stations ID. The collected data comes from the recorded information of completed orders and shop floor status in history during the data collection (shown in Figure 5), supposing we extract one data sample at historical time point t, then the JRT is computed by the time difference between the actual completion date of the last job in the order and the time point t.

According to the configuration of this job shop, the candidate dataset for JRT prediction in this production line is in 1265 dimension based on Equation (6), where $TC^t = 13$, $N\_WIP^t = 1$, $WP^t_M = 2 \times 44$, $WQ^t_M = 2 \times 44$, $MC^t_M = 3 \times 44$, $WIP^t_{m-in,1} + WIP^t_{m-in,2} + \cdots + WIP^t_{m-in,x} = 579$, $WIP^t_{m-out,1} + WIP^t_{m-out,2} + \cdots + WIP^t_{m-out,x} = 364$.

On the other hand, 15,284 data samples are collected by over 4000 RFID tags from different historical moments in this job shop. Before training, we use the min–max normalisation method via scikit-learn (Pedregosa et al. 2011) to re-scale the data to [0, 1] in order to reduce the data redundancy and improve data integrity. The fivefold cross-validation method (Kohavi 1995) is utilised to divide the training set and test set as (4:1), which the number is 12,227 and 3057 respectively. Since it is a supervised regression task, the training set could be formalised as $Train\_X = d_1, d_2, \ldots, d_{12227}$ and $Train\_Y = jrt_1, jrt_2, \ldots, jrt_{12227}$. Also the test set could be formalised as $Test\_X = d'_1, d'_2, \ldots, d'_{3057}$ and $Test\_Y = jrt'_1, jrt'_2, \ldots, jrt'_{3057}$.

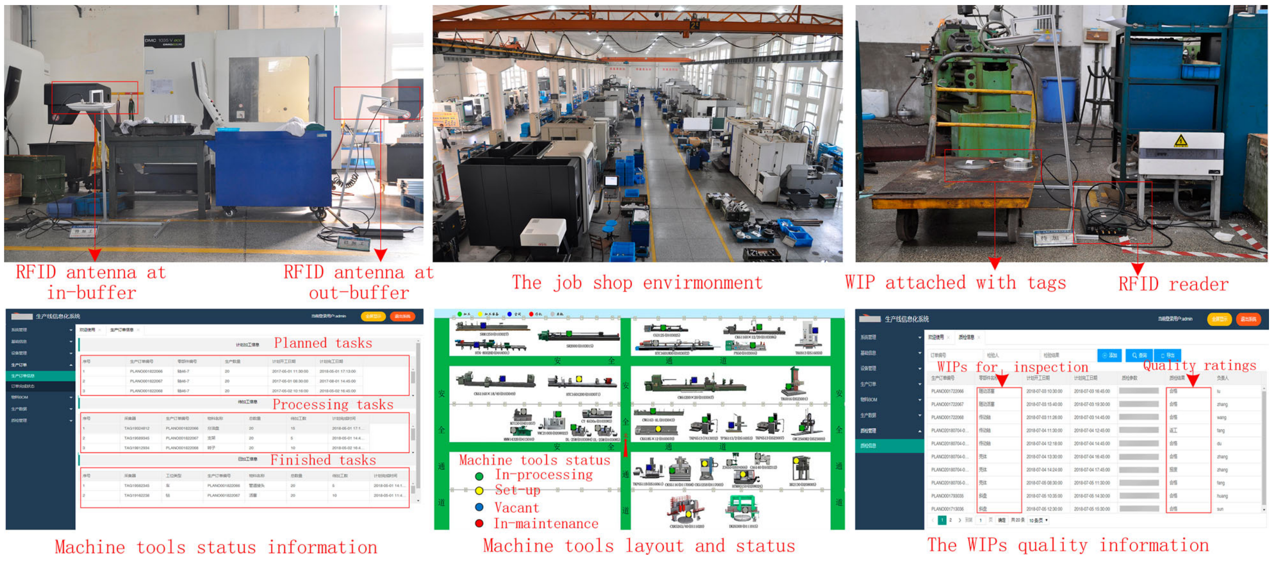

Figure 5. IOT deployment and data collection.

Table 2. The process route of each type of part.

Part no.	Part type	Process route	Part no.	Part type	Process route
1	Pipe fitting	C–E–F–C–H	8	Bracket	B–E–C–B
2	Transmission shaft	C–E–H	9	Monoblock	A–G–H
3	Piston	C–E–F–H	10	Swashplate	C–B–E–D–H
4	Dispenser	C–B–D–H	11	Plunger	C–E–G–D–H
5	Screw plug	C–B–E–C–H	12	Rotator	C–B–F–D–H
6	Backing plate	C–B–D	13	End cover	C–A–H
7	Bracket adapter plate	B–E–F–B			

All in all, the shape of *Train_X* is (12227 × 1265), *Train_Y* is (3057 × 1) and *Test_X* is (3057 × 1265), *Test_Y* is (3057 × 1). Furthermore, 150 new data samples are collected in real time for verifying the performance of the model in the real settings.

4.2. *The experiment of S-SAE + regression for JRT prediction*

4.2.1. The training process and feature extraction of S-SAE + regression model

With regard to the proposed architecture of S-SAE in Section 3.3, several hyper parameters should be pre-determined, including the number of AE layers, the number of nodes in AE layers, the number of fully connected layers, the number of nodes in fully connected layers, etc. The random search method (Bergstra and Bengio 2012) with fivefold cross-validation is employed to investigate the effect of each parameter while keeping the other parameters fixed. For example, the number of neurons in AE layer is set to an equivalent value chosen from a candidate set of $\{200, 400, 600, 800, 1000\}$. We choose the settings in Table 3 on the results of several comparative experiments, which shows that this combination yield the best performance. As for the input and output layers, the training data train X and train Y are fitted in the model. Dense 1 and Dense 2 are the stacked autoencoder layer for latent features extraction, the batch normalisation layer is added in the middle of dense layers for accelerating the training speed, and the dropout layer is added after the upper stacked layers to avoid the extraction of redundant features and refrain the over-fitting problem which always occurs in deep NN.

The mean squared error (MSE) and mean absolute error (MAE) are introduced in this research to measure the loss between actual value Y_i and predicted value $\widehat{Y_i}$, and show the convergence history of S-SAE + regression model.

Figure 6(a) and (b) shows the convergence history during the training process of the proposed model. It can be observed that the loss on the training set and the test set both keep dropping as training proceeds till epoch 20, while MAE converges till epoch 60. After training epoch 60, both the loss and MAE indicators on the test dataset converges to the similar level as those on the training dataset, with the values as 2516.9 and 31.6, respectively. Also the fluctuation on the convergence curves has been dramatically relieved, which indicates the proposed model is robust, well-trained and no overfitting phenomenon occurs.

Table 3. The architecture of S-SAE + regression model for JRT prediction.

Layer (type)	Output dimension	Parameters num
Input (input layer)	(None, 1265)	0
Dense_1 (full connect layer)	(None, 800)	1,012,800
Batch_normalisation_1	(None, 800)	3200
Dense_2 (full connect layer)	(None, 400)	320,400
Batch_normalisation_2	(None, 400)	1600
Dropout (dropout layer)	(None, 400)	0
Dense_3 (full connect layer)	(None, 10)	4010
Output (output layer)	(None, 1)	11

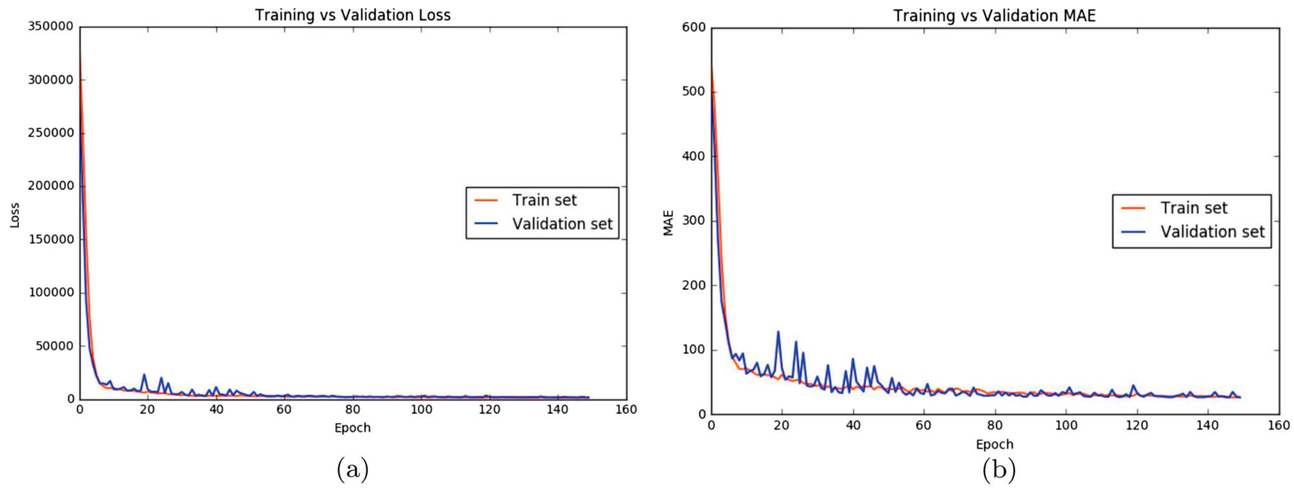

Figure 6. The training process of S-SAE + Regression: (a) the loss changes during training and (b) the MAE changes during training.

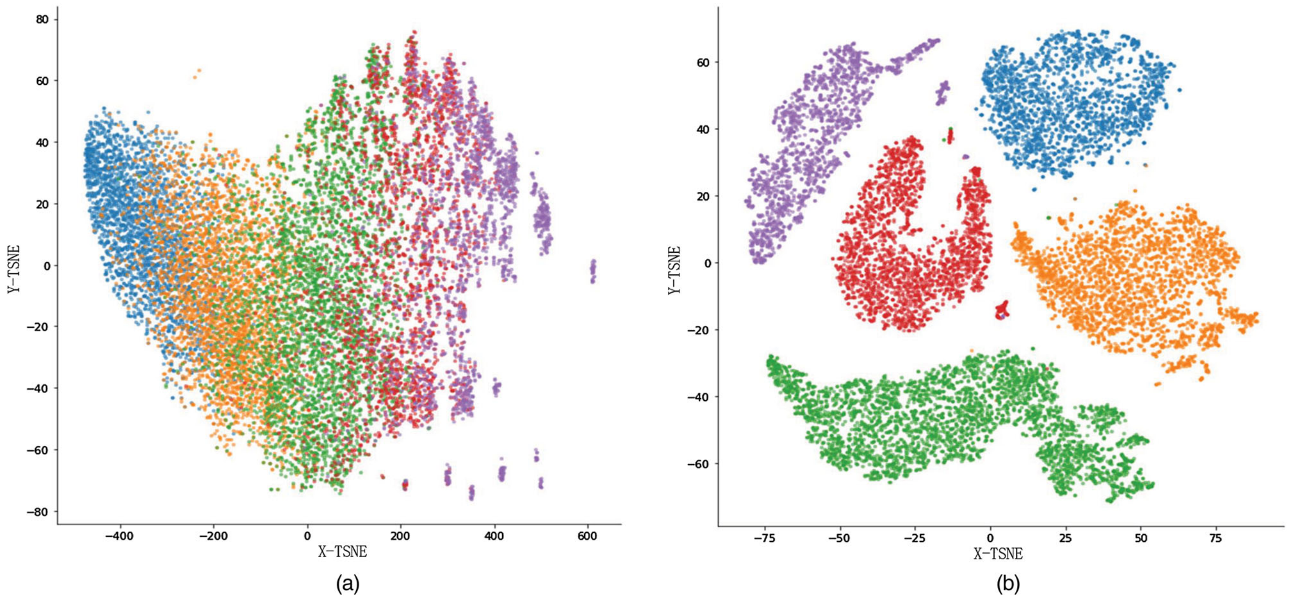

Figure 7. T-SNE for the distribution of JRT prediction dataset, each dot represents an individual data sample (15,284 totally) and different colours represent different JRT range, blue for [− 200,0], yellow for [0,200], green for [200,400], red for [400,600] and purple for [600,1000]. (a) T-SNE for two-dimensional projection of raw dataset (1265-dimensional). (b) T-SNE for two-dimensional projection of extracted features (400-dimensional), the representation is learned by SAE hidden layers.

To illustrate the effectiveness of our proposed method for feature extraction, the raw data and learned features by S-SAE have been visualised in Figure 7 by T-SNE algorithm (Maaten and Hinton 2008), which aims to project the high-dimensional representation into a two-dimensional space. In this paper, we use T-SNE to show how the S-SAE help with understanding features and it is observed that the hidden layers in S-SAE rearrange the data samples so that the distance between data samples better represents their difference in values. The sparsity and separability of Figure 7(b) demonstrates the capability of our S-SAE to extract discriminative and informative features from production data.

4.2.2. *Performance comparison with different models*

We compare the performance of the proposed S-SAE model with the performances of the linear regression, Back propagation NN (BPNN), Multi-layer NN and Deep brief NN (DBN) models. For fair comparison, these models were trained and tested using the same collected dataset via a fivefold cross-validation. However, the input features slightly differed in different networks. For deep structures, we directly use 1265 dimensional features as inputs to hierarchical extract features along the successive layers. In addition, we employ Pearson correlation (Pearson and Lipman 1988) as features filter method for selecting the input variables to the shallow structures (linear regression and BP). Features which the correlation with JRT greater than 0.7 are then selected as inputs.

Linear regression: As the baseline model, we directly deploy linear regression by scikit-learn (Pedregosa et al. 2011) for JRT prediction (the selected 257 dimensional features as inputs), so that we can derive the incremental value-add of the much more complex and computationally intensive deep structure in comparison to the basic regression model.

BPNN: We deploy the standard BP network with only one hidden layer to show the relative advantages of deep network's performance over the conventional NN. Specifically, the selected features with 257 dimension are fit into the network with structure $257 \times 60 \times 1$, and the activation functions are 'Relu' for hidden layer and 'linear' for output layer. The BP algorithm is applied for optimisation and learning rate is set to 0.01.

Multi-layer NN: We deploy the multi-layer NN with the same hierarchical layers as S-SAE to show the advantages of S-SAE in features representation over normal deep NN. Same with S-SAE, the network structure is $1265 \times 800 \times 400 \times 10 \times 1$, and the activation function for hidden layer is 'Relu' and for output layer is 'Linear'. The Dropout (with dropout rate $= 0.5$) and batch normalisation methods are applied to each hidden layer, the 'nadam' is employed for optimisation with learning rate setting to 0.01.

DBN: We deploy the RBM for stacking DBN to compare the performance of different feature representation methods. And same with S-SAE, the DBN adopts the layer-wise pre-training for RBM to optimise the inter-layer weighting matrix. Specifically, the contrastive divergence method is employed to pre-train the RBM (with learning rate 0.05), and then stacking the RBM to form a DBN with the structure 1265 × 800 × 400 × 10 × 1, and the BP is utilised to train the overall DBN with learning rate 0.01. Besides, the DBN shares the same settings of activation function, Dropout and batch normalisation with multi-layer NN and S-SAE.

The performance of different NN models is shown in Figure 8. In order to compare these models with S-SAE in various aspects, we employ four kinds of performance indicators for regression task, which include MAE, mean absolute percentage error (MAPE, Equation 10), root mean squared error (RMSE, Equation 11) and R-square (R_2, Equation 12).

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{Y_i - \widehat{Y}_i}{Y_i}\right|, \tag{10}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{Y}_i)^2}, \tag{11}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(Y_i - \widehat{Y}_i)^2}{\sum_{i=1}^{n}(\widehat{Y}_i - \overline{Y}_i)^2}, \tag{12}$$

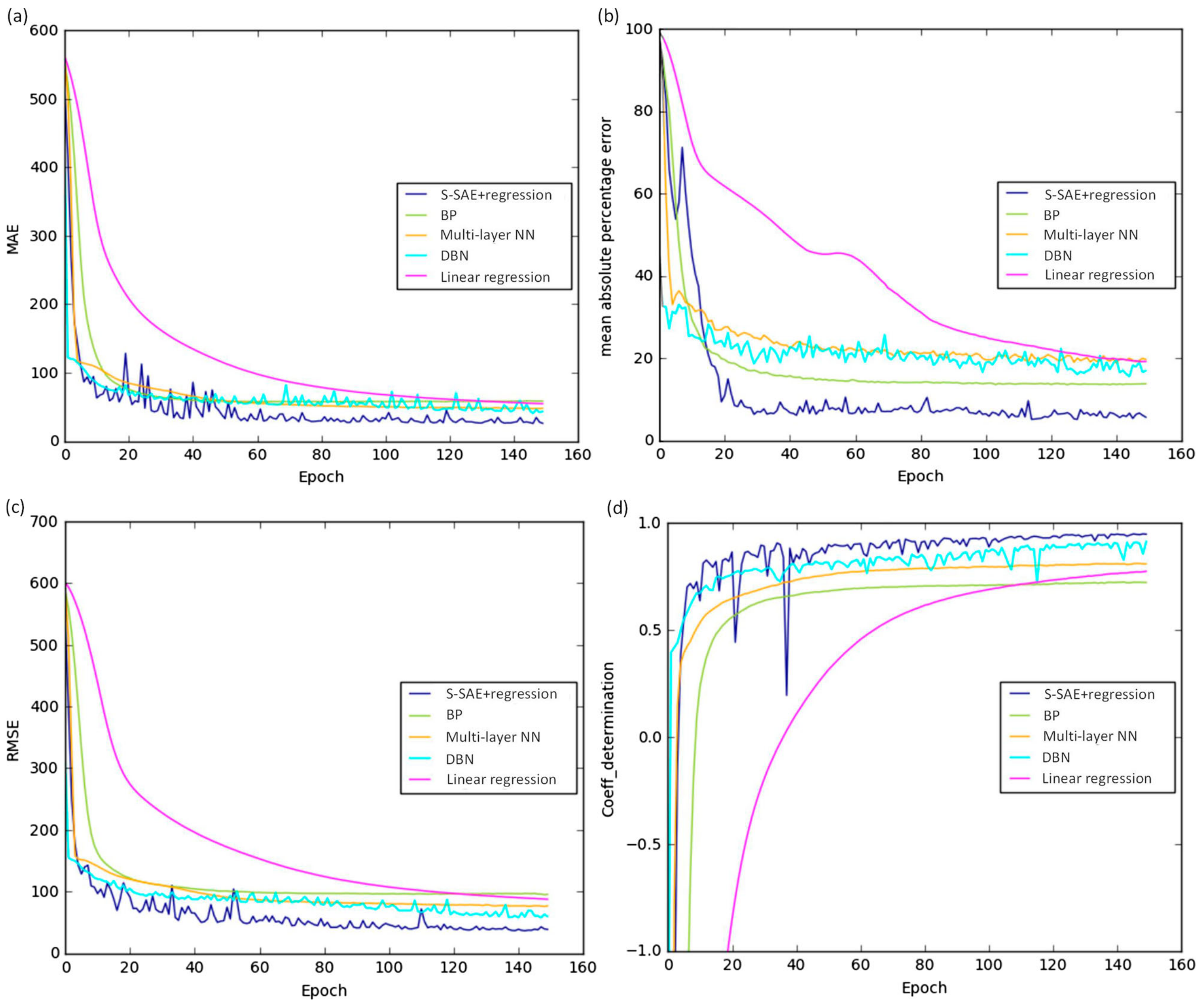

Figure 8. Comparing the proposed S-SAE + regression with BP, multi-layer NN, DBN and linear regression on test set by four model performance indicators: (a) MAE, (b) MSE, (c) RMSE and (d) R^2.

Table 4. The statistical results of different models.

Models	S-SAE + regression	BP	Multi-layer NN	DBN	Linear regression
MAE	25.97	58.38	47.95	43.41	54.62
MAPE	5.63%	13.71%	19.73%	16.92%	19.24%
RMSE	44.26	104.28	79.29	63.71	93.19
R_2	0.96	0.76	0.87	0.93	0.78

Table 5. Robustness evaluation with Gaussian noise and Impulse noise.

Names of models	10% Gaussian noise			10% Impulse noise		
(r)2-7	MAPE	RMSE	R_2	MAPE	RMSE	R_2
S-SAE + regression	10.51%	52.71	0.87	12.09%	57.93	0.89
BP	19.36%	108.28	0.69	22.07%	109.25	0.69
Multi-layer NN	23.82%	91.05	0.73	24.51%	94.07	0.76
DBN	16.18%	68.13	0.79	21.66%	79.53	0.83
Linear regression	23.16%	101.42	0.67	25.10%	107.72	0.69

where R^2 is used to reflect the prediction accuracy, MAE, MAPE and RMSE are used for measuring the error performed models and represent their precision. Y_i, $\widehat{Y_i}$ and $\overline{Y_i}$ are the actual, predicted and average values respectively. The statistical results of different models are shown in Table 4.

It can be observed from Figure 8 and Table 4 that the S-SAE + regression is the most accurate and precise model in our experiment. Meanwhile, the deep model DBN and multi-layer NN perform the second best according to the RMSE and R^2, while BP performs very unstably according to different indicators, and linear regression perform the worst in our case. In overall, the ‘deep’ structure performs better than ‘shallow’ structure, this is because the deeper networks indeed fit better to the training data and achieve smaller empirical errors. In Figure 8(a) and (d), the S-SAE + regression has much fluctuation in the first 50 epoches while the performance of other models has been flattened out at around 20–40 epoches. Comparing with S-SAE, the DBN converges more quickly during training and performs close to the S-SAE in terms of MAE, RMSE and R^2. However, according to the MAPE, the DBN performs fairly worse regarding to MAPE, and it illustrates that there are bigger deviations of the error distributions. The reason lies in the different feature representation methods between DBN and S-SAE. The RBM (which is the basic unit of DBN) minimises the Boltzmann cost function and tries to learn the features representation in a probabilistic manner. However, for our data samples is very discrete distributed (shown in Figure 8a) and it is more applicable to extract features based on multi-layer nonlinear transformations as S-SAE did. Furthermore, we add Kullback–Leibler divergence to cost function to enhance the capability of sparse representation in normal AE (which can be demonstrated in Figure 8b). Another reason may lie in the anomaly nature of production data (with many outliers), due to AE is able to reconstruct data from an input of corrupted data, the robust features could be learned from the corrupted data.

Furthermore, in order to further evaluate the robustness of the models, we add two types of common additive noise (Gaussian noise and Impulse noise) to JRT prediction data, and study how they effect the performance of different models. The performance results under noise are shown in Table 5. We can observe that the S-SAE + regression still performs better than other models in terms of MAPE, RMSE and R^2, this further demonstrate the denoising capability of the S-SAE. In the meantime, the DBN is sensitive to noise, which the performance drops dramatically. This is due to AE based on the Euclidean cost function, whereas there is no output layer in RBM and it is based on the contrastive divergence, which is not optimal for additive noise.

In conclusion, the proposed S-SAE model shows obvious superiority over the others under the measures of MAE, MAPE, RMSE and R^2.

4.2.3. The effect of proposed model in the real settings

After comparing different models on test set, we also make a trial on the test set to evaluate the performance of S-SAE + regression in real settings. Figure 9 displays the output of the S-SAE + regression on 150 test data samples (the red solid line) and the actual JRT changes (the green dashed line). It shows that the predicted JRT trends have very similar pattern with the actual changes. Furthermore, it matches well no matter the JRT in small or big values. However, there still exists few outliers that the values lie away from the predicted JRT changes. The reason for these outliers lie in the extreme cases in the job shop which is unavoidable and could not be predicted.

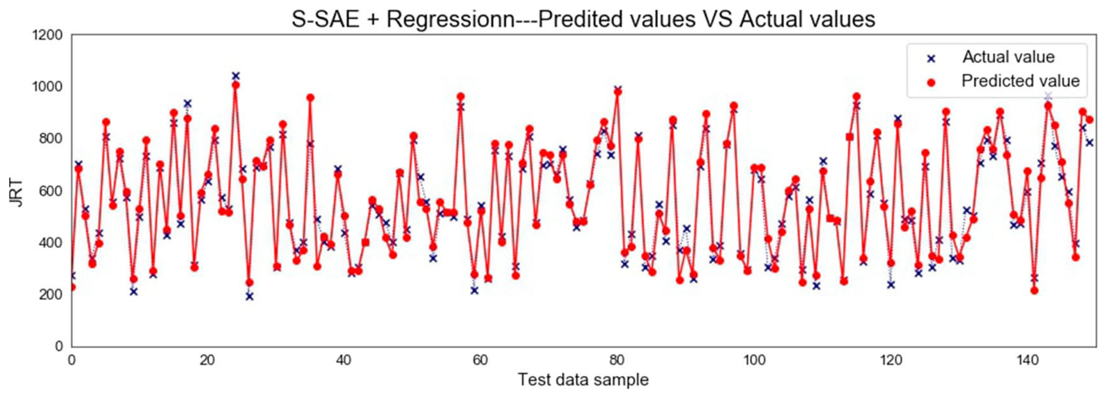

Figure 9. Comparing the predicted values by S-SAE + regression and the actual value in the real settings.

In practice, the S-SAE model is trained based on the offline historical shop floor monitored data and make predictions upon the real-time online data collected from the IOT in the shop floor.

According to the computational efficiency of the proposed model, we monitor the time consumed during model training and predicting. All of our experiments are conducted using two Nvidia GTX 1080Ti GPU with a 3.60 GHz Intel CPU on tensorflow and keras frameworks. The training time on historical data samples is 24.7 min and predicting time on each data sample is only 0.016 s, which is suitable for the real-time prediction. In general, the proposed model could meet the requirements of an overwhelming majority situation in the actual production line and make an accurate and efficient JRT prediction.

From the experiments, it is concluded that (i) the convergence curves show the S-SAE model is well trained and no over-fitting problem occurs; (ii) the T-SNE visualisation shows the features are effectively extracted and informatively learned by S-SAE, (iii) the proposed model has the higher accuracy rate than other models and yields better performance in models robustness test; (iv) the prediction values of JRT are very close to the actual values in the real settings, which show the applicability and reliability of the proposed method.

5. Conclusion

In this paper, we apply the stacked-sparse autoencoder to a large-scale manufacturing system prediction task based on the IOT deployment and BD collection. From global perspective, we find a joint point of different hotspot techniques including Internet of manufacturing things, manufacture BD and DL. Specifically, with our work, we make two key contributions to the literature.

First, we define the JRT to measure the real-time progress of dispatched order and formalise a JRT prediction task to trigger the real-time scheduling decisions in response to the ever-changing production status under disturbances. In practice, when we observe the large difference between the JRT and the scheduled remaining time, it is inevitable to implement the re-scheduling policy to meet the due date of the order. In addition, when we observe the small difference between the JRT and the scheduled remaining time, it is necessary to slightly modify the original schedule to prevent the accumulation of the gap.

Second, our findings through statistical experiments show that DL could have been an effective predictive modelling technique in the manufacturing domain. Furthermore, we find the S-SAE, which is one of the deep structures, is inherently suitable for features representative learning, to beat the other peer models and deep models by a very clear margin and also comparatively remain the robust in the anomaly production data. Compared to the other models, it is a method of choice with respect to prediction accuracy and with respect to model reliability.

In the future work, we are going to develop long–short-term memory (LSTM) model to analyse the time series problems in production and dynamically find the due-date bottlenecks to the production line. Moreover, we will explore the Reinforcement learning (RL) mechanism and introduce it to the production scenario to help the self-learning of policies in production control and optimisation.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work is supported in part by the National Natural Science Foundation of China under Grant 51575274, in part by the National Defense Basic Scientific Research of China under Grant JCKY2016605B006 and Grant JCKY2017203C105, and China Scholarship Council under Grant 201706830037.

ORCID

Weiguang Fang http://orcid.org/0000-0003-4072-273X
Shaohua Huang http://orcid.org/0000-0003-4446-1733

References

Altendorfer, Klaus, and Herbert Jodlbauer. 2011. "An Analytical Model for Service Level and Tardiness in a Single Machine MTO Production System." *International Journal of Production Research* 49 (7): 1827–1850.

Bengio, Yoshua, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. 2007. "Greedy Layer-Wise Training of Deep Networks." In *Advances in Neural Information Processing Systems*, 153–160.

Bergstra, James, and Yoshua Bengio. 2012. "Random Search for Hyper-parameter Optimization." *Journal of Machine Learning Research* 13 (Feb): 281–305.

Brettel, Malte, Niklas Friederichsen, Michael Keller, and Marius Rosenberg. 2014. "How Virtualization, Decentralization and Network Building Change the Manufacturing Landscape: An Industry 4.0 Perspective." *International Journal of Mechanical, Industrial Science and Engineering* 8 (1): 37–44.

Chen, Toly, and Yi Chi Wang. 2013. "An Iterative Procedure for Optimizing the Performance of the Fuzzy-Neural Job Cycle Time Estimation Approach in a Wafer Fabrication Factory." *Mathematical Problems in Engineering* 2013 (2013): 61.

DiBiano, Robert, and Supratik Mukhopadhyay. 2017. "Automated Diagnostics for Manufacturing Machinery Based on Well-regularized Deep Neural Networks." *Integration, the VLSI Journal* 58: 303–310.

Ding, Kai, and Pingyu Jiang. 2018. "RFID-based Production Data Analysis in An IoT-enabled Smart Job-shop." *IEEE/CAA Journal of Automatica Sinica* 5 (1): 1–11.

Gansterer, Margaretha. 2015. "Aggregate Planning and Forecasting in Make-to-order Production Systems." *International Journal of Production Economics* 170: 521–528.

Glorot, Xavier, Antoine Bordes, and Yoshua Bengio. 2011. "Deep Sparse Rectifier Neural Networks." In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 315–323.

Hsieh, Liam Y, Kuo Hao Chang, and Chen Fu Chien. 2014. "Efficient Development of Cycle Time Response Surfaces Using Progressive Simulation Metamodeling." *International Journal of Production Research* 52 (10): 3097–3109.

Ioffe, Sergey, and Christian Szegedy. 2015. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." *arXiv preprint arXiv:1502.03167*.

Kohavi, Roy. 1995. "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection.' In *Ijcai*, Vol. 14, 1137–1145. Montreal, Canada.

Kusiak, A. 2017a. "Smart Manufacturing Must Embrace Big Data." *Nature* 544 (7648): 23–25.

Kusiak, A. 2017b. "Smart Manufacturing." *International Journal of Production Research* 7: 1–10.

Lee, Jay, Hung An Kao, and Shanhu Yang. 2014. "Service Innovation and Smart Analytics for Industry 4.0 and Big Data Environment." *Procedia Cirp* 16: 3–8.

Li, Minqi, Feng Yang, Hong Wan, and John W Fowler. 2015. "Simulation-based Experimental Design and Statistical Modeling for Lead Time Quotation." *Journal of Manufacturing Systems* 37: 362–374.

Liu, Weibo, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E Alsaadi. 2017. "A Survey of Deep Neural Network Architectures and Their Applications." *Neurocomputing* 234: 11–26.

Liu, M. Z., Xi Zhang, M. X. Zhang, M. G. Ge, and Jing Hu. 2014. "Rescheduling Decision Method of Manufacturing Shop Based on Profit-loss Cloud Model." *Control and Decision* 29 (8): 1458–1464.

Lu, Chen, Zhenya Wang, and Bo Zhou. 2017. "Intelligent Fault Diagnosis of Rolling Bearing Using Hierarchical Convolutional Network Based Health State Classification." *Advanced Engineering Informatics* 32: 139–151.

Maaten, Laurens van der, and Geoffrey Hinton. 2008. "Visualizing Data Using T-SNE." *Journal of Machine Learning Research* 9 (Nov): 2579–2605.

Mirshekarian, Sadegh, and Dušan N Šormaz. 2016. "Correlation of Job-shop Scheduling Problem Features with Scheduling Efficiency." *Expert Systems with Applications* 62: 131–147.

Nguyen, S, M Zhang, M Johnston, and K. C. Tan. 2014. "Genetic Programming for Evolving Due-date Assignment Models in Job Shop Environments." *Evolutionary Computation* 22 (1): 105–138.

Olshausen, Bruno A, and David J Field. 1996. "Emergence of Simple-cell Receptive Field Properties by Learning a Sparse Code for Natural Images." *Nature* 381 (6583): 607.

Pearson, William R, and David J Lipman. 1988. "Improved Tools for Biological Sequence Comparison." *Proceedings of the National Academy of Sciences* 85 (8): 2444–2448.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, and Mathieu Blondel, et al. 2011. “Scikit-learn: Machine Learning in Python.” *Journal of Machine Learning Research* 12 (Oct): 2825–2830.

Ren, Lei, Jin Cui, Yaqiang Sun, and Xuejun Cheng. 2017. “Multi-bearicng Remaining Useful Life Collaborative Prediction: A Deep Learning Approach.” *Journal of Manufacturing Systems* 43: 248–256.

Ribeiro, Vera Matilde Veloso da Costa. 2016. “Job Shop Flow Time Prediction Using Artificial Neural Networks.” Master’s Thesis.

Silva, Cristóvão, Vera Ribeiro, Pedro Coelho, Vanessa Magalhaes, and Pedro Neto. 2016. “Job Shop Flow Time Prediction Using Neural Networks.” *Procedia Manufacturing* 11: 1767–1773.

Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. “Dropout: a Simple Way to Prevent Neural Networks From Overfitting.” *The Journal of Machine Learning Research* 15 (1): 1929–1958.

Tirkel, Israel. 2013. “Forecasting Flow Time in Semiconductor Manufacturing Using Knowledge Discovery in Databases.” *International Journal of Production Research* 51 (18): 5536–5548.

Vinod, V., and R. Sridharan. 2011. “Simulation Modeling and Analysis of Due-date Assignment Methods and Scheduling Decision Rules in a Dynamic Job Shop Production System.” *International Journal of Production Economics* 129 (1): 127–146.

Wang, Chuang, and Pingyu Jiang. 2017. “Deep Neural Networks Based Order Completion Time Prediction by Using Real-time Job Shop RFID Data.” *Journal of Intelligent Manufacturing* 30 (4): 1–16.

Wang, Chuang, and Pingyu Jiang. 2018. “Manifold Learning Based Rescheduling Decision Mechanism for Recessive Disturbances in RFID-driven Job Shops.” *Journal of Intelligent Manufacturing* 29 (7): 1485–1500.

Wang, Jinjiang, Yulin Ma, Laibin Zhang, Robert X Gao, and Dazhong Wu. 2018. “Deep Learning for Smart Manufacturing: Methods and Applications.” *Journal of Manufacturing Systems* 48: 144–156.

Wang, Junliang, and Jie Zhang. 2016. “Big Data Analytics for Forecasting Cycle Time in Semiconductor Wafer Fabrication System.” *International Journal of Production Research* 54 (23): 7231–7244.

Yao, Li, Li Yao, Jin Yang, and Zheng Wang. 2015. “Due Date Assignment and Dynamic Scheduling of One-of-a-kind Assembly Production with Uncertain Processing Time.” *International Journal of Computer Integrated Manufacturing* 28 (6): 616–627.

Zengqiang, Liu Mingzhou Shan Hui Jiang, and Ge Maogen Hu Jing Zhang Mingxin. 2009. “Dynamic Rescheduling Optimization of Job-shop Under Uncertain Conditions.” *Journal of Mechanical Engineering* 45 (10): 031.

Zhang, Qingchen, Laurence T. Yang, Zhikui Chen, and Peng Li. 2018. “A Survey on Deep Learning for Big Data.” *Information Fusion* 42: 146–157.

Zhong, Ray Y, Chen Xu, Chao Chen, and George Q. Huang. 2015. “Big Data Analytics for Physical Internet-based Intelligent Manufacturing Shop Floors.” *International Journal of Production Research* 55 (9): 2610–2621.