Proceedings of the 2023 Winter Simulation Conference
C. G. Corlu, S. R. Hunter, H. Lam, B. S. Onggo, J. Shortle, and B. Biller, eds.

COMBINING TIME SERIES DATA AND SNAPSHOT DATA FOR SITUATION AWARE DISPATCHING IN SEMICONDUCTOR MANUFACTURING

Chew Wye Chan
Boon Ping Gan

D-SIMLAB Technologies Pte Ltd
8 Jurong Town Hall Road, #23-05
Singapore, 609434, SINGAPORE

Wentong Cai

School of Computer Science and Engineering
Nanyang Technological University
Singapore, 639798, SINGAPORE

ABSTRACT

Dispatch rules are commonly used to schedule lots in the semiconductor industry. Previous studies have indicated that adapting dispatch rules can improve overall factory performance. Machine learning has proven useful in learning the relationship between manufacturing situations and dispatch rules. However, using only snapshot data at a given point in time to generate features for these models does not account for trends in the manufacturing situation, which can be represented as time series data. To address this issue, the proposed method generates features from time series data and combines them with features from snapshot data to train machine learning models for dispatch rule prediction. The results demonstrate the effectiveness of this methodology, as the combination of features from both types of data achieves the highest prediction accuracy. Simulation results show that this approach can adapt the dispatch rule according to the manufacturing situation and achieve a comparable factory performance.

1 INTRODUCTION

In a complex manufacturing environment such as the semiconductor industry, scheduling is usually done using dispatching rules (Metan and Sabuncuoglu 2005). Dispatching rules are used to assign a priority ranking to the lot in the queue of a machine. However, it is concluded that no single dispatch rules consistently generate better factory performance under different configurations of manufacturing environment and performance criteria (Sabuncuoglu 1998). On the other hand, it is shown that adapting dispatch rules can improve factory performance (Wu and Wysk 1989; Shiue et al. 2020). Simulation is commonly used to evaluate the performance of dispatch rules in different manufacturing situations. However, the decision to change dispatch rules is required in near real-time because manufacturing situations can change rapidly due to equipment breakdowns or product mix changes. Recent advancements in machine learning have enabled the development of models that adapt dispatch rules in different manufacturing situations.

Numerous studies have applied machine learning models such as supervised, unsupervised, and reinforcement learning for lot dispatching in semiconductor manufacturing. Shiue and Su (2003) utilized a supervised learning model to predict the next dispatch rule. Shiue et al. (2011) used an unsupervised learning approach to cluster different manufacturing situations and associate these clusters to predict the next dispatch rule. Approaches to apply reinforcement learning for lot dispatching have also been proposed (Shiue et al. 2020; Waschneck et al. 2018; Stöckermann 2022). These studies show positive improvement in factory performance by adapting dispatching rules to different manufacturing situations. Although these studies proposed numerous features to represent the manufacturing situation, they only represent the snapshot of the manufacturing situation at a given moment in time. We refer to data used to generate these features as snapshot data.

Time series data is a type of data where measurements are collected over time intervals and can be used to analyze trends and patterns (Tseng et al. 2015). Previous studies have utilized statistical methods and control charts on time series data to identify various manufacturing situations such as WIP congestion at a certain segment of the product line and abnormal cycle times (Hassoun and Rabinowitz 2010; Metan and Sabuncuoglu 2005). However, these methods rely on expert knowledge and identifying specific manufacturing situations with limited features on the time series data. With the recent advancements in machine learning models, such as LSTM, there has been increasing interest in the application of these models for fault detection and predictive maintenance in the manufacturing domain by collecting a sequence of equipment sensor data (Fan et al. 2020). Nevertheless, to the best of our knowledge, there is limited research on the application of machine learning models to time series data for lot dispatching.

This work proposed a methodology to combine features generated from time series data and snapshot data to train a supervised learning model for dispatch rule prediction. LSTM is applied to extract features from time series data. We showed that by generating features from snapshot data and time series data, we could improve the prediction accuracy of the next dispatch rule.

2 LITERATURE REVIEW

Recent advancements in machine learning have enabled the training of models to help decide dispatch rules in near real-time. Earlier studies have shown the usage of machine learning to adapt dispatch rules in different manufacturing situations. Shiue and Su (2003) used a decision tree to learn the relationship between the manufacturing situation and the dispatch rule to be applied. Priore et al. (2018) proposes ensemble methods for the selection of the most appropriate dispatching rule. Shiue et al. (2020) and Waschneck et al. (2018) developed approaches to apply reinforcement learning for lot dispatching. These studies have shown improved factory performance by adapting lot dispatching strategies in different manufacturing situations.

Feature engineering plays a critical role in constructing machine learning models (Guyon and Elisseeff 2003; Li and Olafsson 2005). In complex manufacturing situations such as the semiconductor manufacturing system, snapshot data of the work-in-progress (WIP) can be represented with a conceptual data model (Laipple et al. 2018). The snapshot data that contains WIP attributes is further processed to become features for subsequent machine learning model (Jun and Lee 2021; Schulz et al. 2022). Statistics summary such as minimum, maximum, summation, mean, and standard deviation of the WIP attributes can then be calculated from the snapshot data at a given moment of time (Chan et al. 2020; Shiue et al. 2020).

On the other hand, time series data is also important in semiconductor manufacturing, such as fault detection (Fan et al. 2020). Different approaches can be used to generate features from time series data for machine learning tasks. Time spatialization technique could be used to splitting time series data with a fixed number of time steps, with each time step representing a feature (Leontjeva and Kuzovkin 2016). Data represented in such a form could be used to train a decision tree model. However, it would suffer from the curse of dimensionality, making them difficult to use for large datasets (Leontjeva and Kuzovkin 2016). Another approach is using statistics calculation to extract features from the time series data (Guo et al. 2020). By reducing the number of features to a fixed size, this approach can help to alleviate the curse of dimensionality. Still, this technique has limited ability to learn the temporal relationship within the time series data.

Long Short-Term Memory (LSTM) is a type of recurrent neural network to learn temporal relationships within time series data (Hochreiter and Schmidhuber 1997). LSTM models are made up of memory blocks connected through gates. These gates can be opened or closed, allowing the model to remember or forget information selectively. The model can better capture long-term dependencies in time series data by selectively remembering or forgetting information. LSTM has shown promising results in classification tasks in the medical and financial domain (Borovkova and Tsiamas 2019; Saadatnejad et al. 2020). As LSTM can learn the temporal relationship of time series data, the learned network could also be used as feature generation for medical and video images for other classification algorithms (Leontjeva and Kuzovkin 2016).

Snapshot data and time series data are shown to be useful for training machine learning models. Thus, in this paper, we proposed a method of combining features from snapshot data and time series data to train a machine learning model and compare its performance with a machine learning model trained only with snapshot data.

3 METHODOLOGY

The proposed evaluation methodology comprises three stages: data collection and separation, feature generation for the prediction model, and feeding these features as input to a classification model. The flow chart of the methodology is shown in Figure 1 to provide an overview of the process.

The first stage is data collection and separation. This stage involves using a training data generation mechanism described in Section 3.1 to generate data for the machine learning model. The generated data is then separated into time series data and snapshot data. Time series data is generated from continuous measurements over time, while snapshot data is generated at a fixed point in time. The separation of these data types is necessary as they require different feature generation methods.

The second stage is feature generation for the prediction model. Different feature generation modules are used in this stage for time series and snapshot data. For time series data, the proposed methodology extracts feature from the activation layers of a Long Short-Term Memory (LSTM) neural network, which is described in Section 3.2. For snapshot data, the proposed methodology extracts feature that captures work-in-progress attribute about the manufacturing process, such as the number of lots currently queuing at the equipment when the dispatch decision is required.

The third stage is to feed these features as input to a classification algorithm. The proposed methodology uses supervised learning to train a machine learning model for the features from snapshot data and features from a combination of time series data and snapshot data. The machine learning model is trained using the training data generated in the first stage, and its performance is evaluated using the test data.

Finally, a comparison is made towards the performance of the classification accuracy by using features generated from snapshot data, and the combination of time series data and snapshot data.

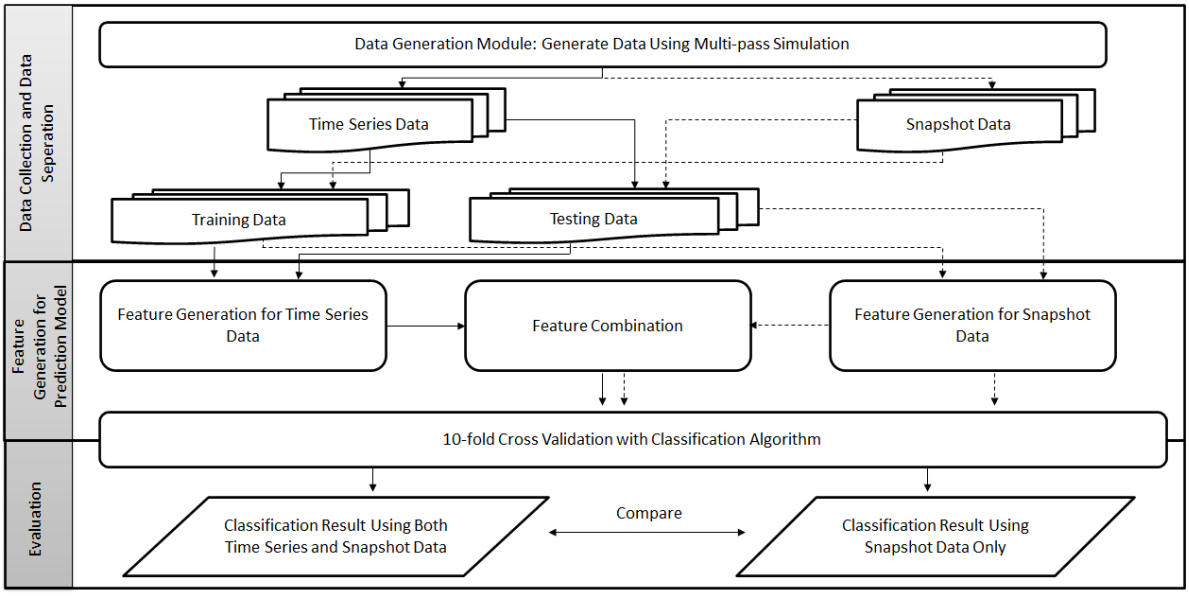


Figure 1: Flow chart of performance evaluation of combining features from time series and snapshot data.

3.1 Data Collection via Multi-Pass Simulation

The multi-pass simulation technique simulates and compares candidate dispatch rules within the same scheduling period (Wu and Wysk 1989). In Figure 2, an illustration of the multi-pass simulation is depicted, featuring two candidate dispatch rules (d_1, d_2) and two multi-pass periods with a constant scheduling period δt. Each pass of the simulation begins at the start of a scheduling period with the same initial manufacturing situation and a candidate dispatch rule is applied for lot dispatching. Once the multi-pass simulation is completed, the target factory performance is evaluated to determine the most optimal simulation path. The dispatching rule utilized for each scheduling period within the chosen simulation path is then extracted and assigned as the corresponding label.

Assuming the set of completed lots within the time period t is $CompleteLot_t$, the tardiness of a lot a is $Tardiness_a$ using equation (1), average tardiness is calculated using equation (2). Table 1 shows the list of candidate dispatch rules. Snapshot data of lot attributes will be collected as shown in Table 2. Features for snapshot data are generated using summary statistics (e.g., $OprT^{St}$ in Table 3). For each statistics feature, five summary statistics (St) are calculated. They are minimum (Mi), maximum (Ma), summation (Sum), average (Me), and standard deviation (Sd) (that is, $St \in \{Mi, Ma, Sum, Me, Sd\}$). Table 3 shows the statistics features generated from the snapshot data of lot attributes.

Time series data are generated by observing a sequence of a particular feature (Tseng et al. 2015). In our approach, we use the sequence of statistics features generated from the snapshot data collected at regular intervals to generate time series data. To focus on the historical trend of manufacturing situations, the generation of time series data excludes the snapshot data at the start of the next scheduling period. The decision on the length of the interval time for snapshot data would affect the number of time series data generated. If the interval time is the same as the scheduling period, there would be only one data point within a scheduling period. In order to capture the temporal patterns present in the data, it is necessary to collect sufficient time series data using an interval time that is shorter than the scheduling period. However, the specific length of this interval depends upon the factory's characteristics and the scheduling period.

Data segmentation involves dividing a continuous data stream into discrete segments, each with a fixed duration. In our approach, we divide a scheduling period from the multi-pass simulation as one segment of time series data. This is achieved by defining the start and end time of the scheduling period in the multi-pass simulation. The resulting time series data can then be used as input for machine learning models. By segmenting the data into fixed time windows in accordance with the multi-pass simulation scheduling period, the temporal relationship from the time series data is associated with the best dispatch rule for the next scheduling period used in the multi-pass simulation.

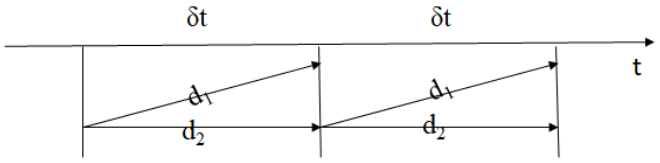


Figure 2: Multi-pass simulation.

$$Tardiness_a = Max(0, CompletiontTime_a - DueDate_a) \tag{1}$$

$$AvgTardiness = \frac{\sum_{a \in CompleteLot_t} Tardiness_a}{|CompleteLot_t|} \tag{2}$$

Table 1: Candidate dispatch rules in the study.

Dispatch Rule	**Description**
SRPT	Select the lot with the shortest remaining process time.
EDD	Select the lot with the earliest due date.
FIFO	Select the lot with the earliest arrival time.

Table 2: Lot attribute.

Lot Attribute	**Description**
$OprT_i$	Current operation processing time of lot i.
$ProT_i$	Total processing time of lot i.
$RemT_i$	Remaining processing time of lot i.
$SlkT_i$	Slack time of lot i.
$LtnA_i$	Lateness of lot i.
$QueT_i$	Queue time of lot i.
$RmdD_i$	Remaining due time of lot i.

Table 3: Feature generated by using statistic summary on lot attributes.

Feature	**Description**
$OprT^{St}$	Statistic summary of current operation processing time of all lots in the factory.
$ProT^{St}$	Statistic summary of total processing time of all lots in the factory.
$RemT^{St}$	Statistic summary of remaining processing time of all lots in the factory.
$SlkT^{St}$	Statistic summary of slack time of all lots in the factory.
$LtnA^{St}$	Statistic summary lateness of all lots in the factory.
$QueT^{St}$	Statistic summary of queue time of all lots in the factory.
$RmdD^{St}$	Statistic summary of remaining due time of all lots in the factory.

3.2 Feature Generation for Time Series Data

In LSTM, the input sequence is fed into the model one-time step at a time. The LSTM unit then processes the input and updates its hidden state. This hidden state can then be used to make a prediction for the next time step in the sequence. The process of generating features from time series data with LSTM involves training the model on a labeled dataset, where each input sequence is paired with a corresponding label. The model adjusts its parameters during training to minimize the error between its predicted and true labels in the training data. Once trained, the model's parameters can be used to generate features by making predictions on new input sequences.

Once the time series data is created as described in Section 3.1, it can be partitioned into training and testing datasets. The best dispatch rule is assigned as the label for each segment of time series data. The input to the LSTM layer is time series data, represented in three dimensions: the number of data, the number of time steps, and the number of features for each time step. The LSTM layer processes this input and passes the output to the Output layer. The Output layer produces an output representing the probability of the time series data being associated with a label (Farzad et al. 2019). We use the Output layer's output as a feature representation of the time series data. This feature representation results from associating temporal relationships in the time series data with the dispatch rule.

3.3 Classification Model

We use Random Forest (RF) as the classification algorithm for our machine learning model. RF is an ensemble algorithm combining multiple decision tree predictions to improve accuracy and reduce overfitting (Denil et al. 2014). Furthermore, it could also handle large datasets with high-dimensional feature space. The Scikit-learn implementation of RF was used in our experiments (Pedregosa et al. 2012). The prediction output of the random forest is the label representing the dispatch rule with the highest prediction probability (Boström 2008).

To train RF using snapshot data only, the snapshot data is partitioned into the training and testing dataset. Features for the training dataset are calculated using summary statistics as described in Section 3.1. A triplet of $\{\mathbf{P},\mathbf{S},\mathbf{D}\}$ is used to represent the training data (Shiue and Su 2003). P represents the user-defined performance; S represents the features; D represents the best dispatch rules under the current system attribute and user-defined performance. *AvgTardiness* described in Section 3.1 is used as the factory performance.

RF is also used to train with the combination of time series and snapshot data. This enables fair comparison with RF trained with snapshot data only, as both use the same classification algorithm. Time series data and snapshot data are generated from each scheduling period. These data are separated into the training dataset and the testing dataset. In the first stage of training, the time series data from the training dataset is used to train the LSTM to generate LSTM encoded features. In the second stage of training, the snapshot data from the same scheduling period as the time series data is combined with the LSTM encoded features to train the RF. The two-stage training for the combination of time series and snapshot data is shown in Figure 3.

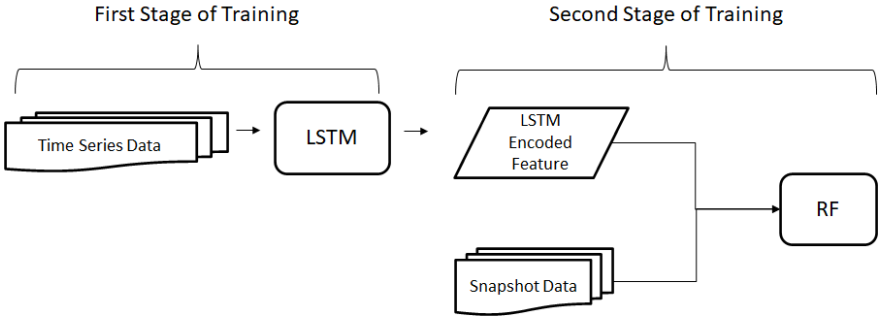


Figure 3: Training RF with time series data and snapshot data.

4 EXPERIMENT AND RESULTS

4.1 Mini-Fab Model

The Mini-Fab model is a collaborative effort between Arizona State University and Inter Research (Spier and Kempf 1995; El-Khouly et al. 2009). It consists of one route that is shared between the three products. The route has six re-entrant steps, three work centers, and five pieces of equipment, as shown in Figure 4. On average, 84 lots are started each week, with 51 lots for Product_1, 30 lots for Product_2, and 3 lots for Product_3. The due date of a lot is uniformly distributed at between 2.1 to 2.5 times the product's total process time. The "Diff" work center serves steps 1 and 5, and equipment from this work center batches three lots at a time, with the constraint of not batching lots from different steps together. The work center has 75 minutes of preventive maintenance every 24 hours. The "Imp" work center has two pieces of equipment, and preventive maintenance is carried out for 120 minutes every 12 hours. Emergency maintenance happens randomly every 24 and 76 hours; repair takes 6 to 8 hours. The "Lith" work center has only one piece of equipment, and preventive maintenance takes 30 minutes every 12 hours. This work

center has setup time for processing the next lot from a different product or operation. Ten minutes of setup time for the next lot from the same product but a different step, five minutes for the same step but a different product, and twelve minutes for different products and different steps. Table 4 summarized the parameters for the Mini-Fab model.

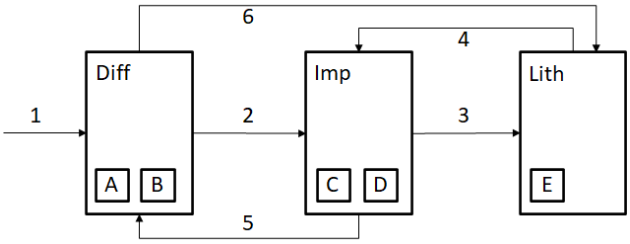


Figure 4: The Mini-Fab Model.

Table 4: Summary descriptions of the model.

Work Centre	**Equipment**	**Step**	**Process Time (min)**	**Characteristic**
Diff	A / B	1	285	Batching
		5	315	
Imp	C / D	2	60	Emergency Maintenance
		4	80	
Lith	E	3	75	Setup Changes
		6	30	

4.2 Data Generation

The study uses the D-SIMCON simulator to model Mini-Fab (D-SIMLAB 2023). The data is generated through multi-pass simulation as described in Section 3.1. EDD and SRPT dispatching rules are considered in the multi-pass simulation. 35 features as shown in Table 3 are generated from the snapshot data. Although more features will improve the discriminating power of the classifier, it also increases the risk of overfitting (Lee and Landgrebe 1993; Shiue and Su 2003). As the selected dispatching rules only consider lot attributes $RemT_i$ and $RemD_i$, we selected a total of 10 summary statistics features ($RemT^{St}$ and $RemD^{St}$) that are generated from these lot attributes. In the future, more features should be included when other dispatch rules are considered.

To generate enough data, 500 random seeds are used to generate different product arrival patterns, lot due dates, and equipment maintenance. The simulation runs for a total of 42 days, including a 30-day warm-up period followed by 12 multi-pass scheduling periods, where each scheduling period lasts for 24 hours. Time series data is collected from the start of the previous scheduling period to the time step before the start of the current scheduling period. A regular time step interval of 1 hour is used, resulting in 24 data within each scheduling period. This provides one sequence of time series data per scheduling period, and each random seed run generates 12 sequences of time series data. On the other hand, snapshot data is collected at the start of each scheduling period, and each random seed generates 12 snapshot data.

Figure 5 shows time series data and snapshot data for two features for one scheduling period. The generated data train the machine learning model to predict the dispatching rule for the next scheduling period. The aim of this approach is to consider both time series and snapshot data to generate a combination of features that can effectively predict the dispatching rule with high accuracy. Table 5 summarizes the total amount of data generated per random seed for snapshot data and time series data.

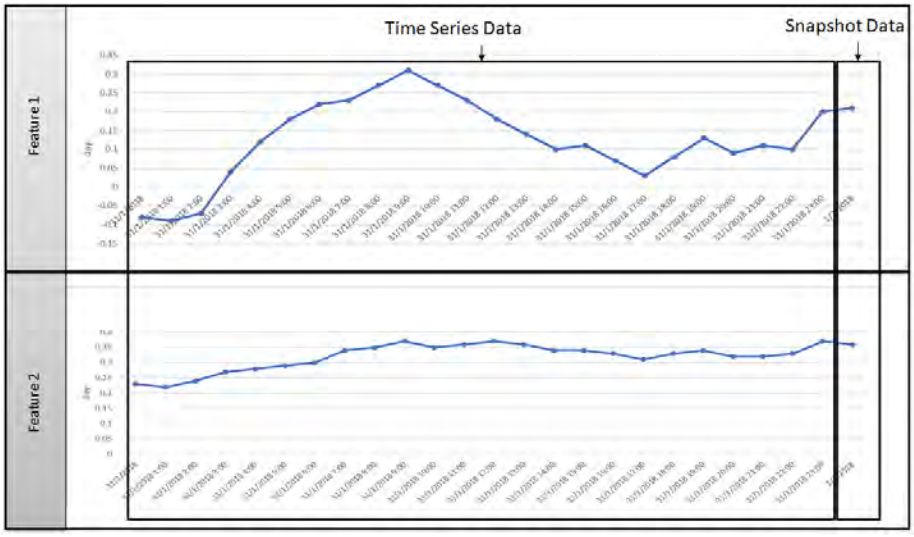


Figure 5: Generation of snapshot data and time series data at the start of the next scheduling period.

Table 5: Data generation summary.

	Snapshot Data	**Time Series Data**
Number of Features	10	10
Total Number of Scheduling Period per Multi-Pass Simulation	12	12
Total Amount of Data per feature per Scheduling Period	1	24
Total Amount of Data per Random Seed	120	2880

Table 6: Average accuracy of 10-fold cross-validation across different features from snapshot data and combination of time series and snapshot data.

Data Type	**10-fold Prediction Accuracy**
Snapshot Data Only	65%
Snapshot Data and Time Series Data	78%

4.3 Validation of Machine Learning Model

In this study, our aim is to enhance the class separability between the features generated from snapshot data and the combination of snapshot and time series data. We first visualize the features obtained from different data types using t-Distributed Stochastic Neighbor Embedding (t-SNE), a technique that reduces high-dimensional data points into low-dimensional space while maintaining the inter-point relationships (van der Maaten and Hinton 2008). Figure 6 depicts the resulting two-dimensional projection, where the axes do not directly correspond to a specific feature and different colors represent dispatch rule labels. It shows that combining snapshot and time series data can lead to better class separation with fewer overlapping data points than using snapshot data alone.

To validate the RF model, we employ 10-fold cross-validation, where the data is randomly divided into 10 partitions, and 10 iterations of training and testing are performed. During each validation iteration, 9 partitions of data are retained for training, and 1 partition is used for testing. The number of exact matches between the predictions label and the testing label measures the accuracy of the evaluation. Table 6 shows that the machine learning model with a combination of snapshot data and time series data achieves the highest prediction accuracy.

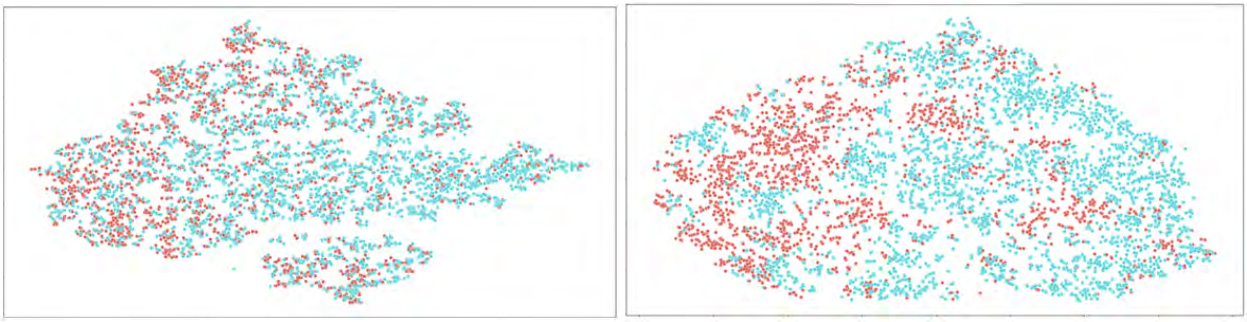


Figure 6: Data visualization of snapshot data (left) and combination of snapshot data and time series data (right).

4.4 Evaluation of Learned Models for Situation Aware Dispatching

The performance evaluation of the trained machine learning model is conducted using the D-SIMCON Python Interface and D-SIMCON Simulator (D-SIMLAB 2023). The objective is to assess the model's ability to adapt dispatch rules to different manufacturing situations. To achieve this, we use 100 different random seeds to generate scenarios with different lot arrival patterns, due date patterns, and machine down patterns. We simulate the EDD, SRPT, and FIFO dispatch rules separately on these scenarios to provide a baseline for comparison.

We divide the 100 scenarios into two groups based on the dispatch rule's performance: (i) the group of scenarios where SRPT outperforms EDD; (ii) the group of scenarios where EDD outperforms SRPT. We train two different machine learning models to adapt dispatch rules: the TS+SN scheduling strategy using a model trained with the combination of time series and snapshot data, and the SN scheduling strategy using a model trained with snapshot data only. RANDOM scheduling strategy uses a uniformly distributed random variable to select a dispatch rule from the baseline dispatch rules. The MULTI_PASS simulation provides the lower bound for factory performance.

It is important to note that high prediction accuracy in machine learning models does not necessarily indicate high prediction probability (Boström 2008). A high prediction probability indicates the learned model is confident in the prediction. However, the baseline shows that EDD and SRPT can positively impact certain manufacturing situations. To avoid wrong predictions that result in good factory performance, a high prediction probability threshold of 0.8 was set in our evaluation. However, retaining the previous dispatch rule and not adapting the predicted dispatch rule if the threshold is not met would also have minimal impact on factory performance in certain manufacturing situations. Hence, during the evaluation process, we choose the FIFO rule if the prediction probability does not exceed the threshold to show the impact on factory performance. The decision to use FIFO is because it had the worst performance among the baseline of dispatch rules and would negatively impact the factory's performance. This strategy will also impact the factory performance of the right prediction with low confidence. Nonetheless, we could compare the trained models by measuring the number of high-confidence predictions.

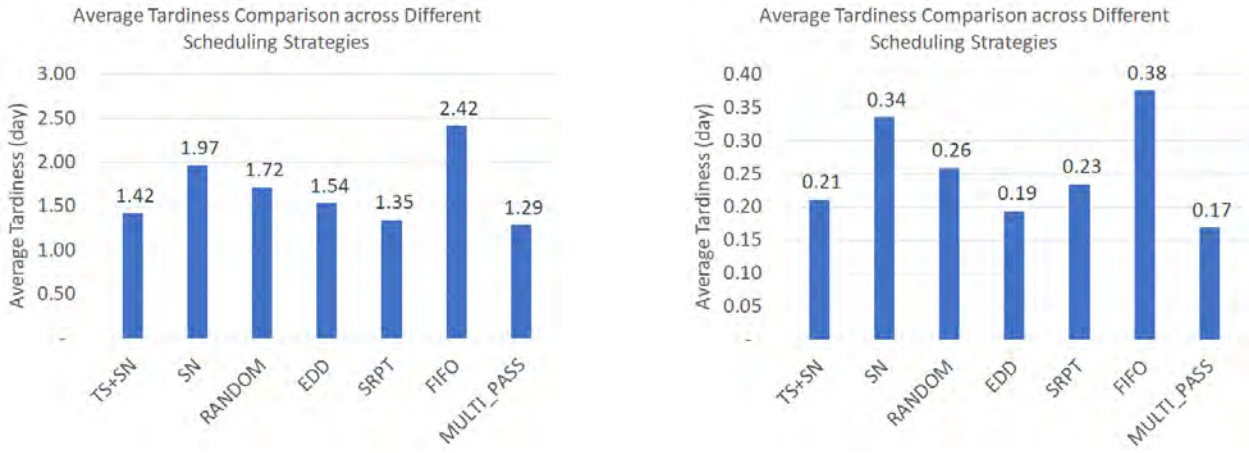
Figure 7a shows the average of *AvgTardiness* across the scenarios from the group (i). Figure 7b shows the average of *AvgTardiness* across the scenarios from the group (ii). In both groups of scenarios, it shows that TS+SN was able to adapt dispatch rules better than SN. Table 7 shows the percentage of a scheduling period instance where the probability threshold is met. SN performance was worse than RANDOM because of the high number of predictions that do not meet the threshold, and we chose the FIFO rule for these predictions. This also shows that TS+SN performs better than SN in making highly confident predictions. This shows that the machine learning model trained with the combination of time series and snapshot data achieves a higher prediction accuracy and higher prediction probability.

On the other hand, TS+SN is also shown to adapt dispatch rules in different groups of scenarios compared to the baseline dispatch rules. Figure 7a shows the result for group (i) scenarios and TS+SN

Table 7: Percentage of scheduling period where prediction probability is more than the threshold of 0.8.

Scheduling Strategy	Percentage
TS+SN	89%
SN	22%

is shown to perform better than the EDD rule. Figure 7b shows the result for group (ii) scenarios, and TS+SN is shown to perform better than the SRPT rule. However, in both groups of scenarios, TS+SN did not perform better than the best dispatch rule. This is the impact of setting a high prediction probability threshold that penalizes the right prediction with low confidence. Furthermore, there is still a performance gap between the best dispatch rule and MULTI_PASS, indicating potential in adapting dispatch rules. It is also possible that TS+SN did not outperform the best baseline dispatch rule due to the factory model used. The average performance difference between MULTI_PASS and the best baseline dispatch rule was less than 5% for group (i) and 11% for group (ii), and more analysis is required. In the future, it will be interesting to evaluate the approach of combining time series and snapshot data with other factory models with high-mix low volume, such as the SMT2020 (Hassoun et al. 2019).



(a) Group (i) scenarios.

(b) Group (ii) scenarios.

Figure 7: Average tardiness comparison across different scheduling strategies.

5 CONCLUSIONS AND FUTURE WORK

In conclusion, selecting the appropriate dispatch rule is crucial for improving factory performance in a situation aware dispatching system. This study has analyzed the impact of combining time series data and snapshot data in improving the prediction accuracy of the machine learning model for dispatch rule adaptation.

The results showed that the model with a combination of time series and snapshot data outperformed the model trained using snapshot data alone. Furthermore, the approach also shows the ability of the machine learning model to adapt dispatch rules under different manufacturing situations where the best baseline dispatch rule was different. However, a performance gap remains compared to the multi-pass simulation results, indicating room for further refinement in the machine learning model. More studies are needed to analyze features specific to time series and snapshot data to enhance the machine learning model that could further improve factory performance. Future work should consider a broader range of dispatch rules and a weighted combination of dispatch rules to make the model more realistic.

One limitation of the current approach that utilizes LSTM for encoding time series data is its reliance on labeled data, which results in a supervised learning approach. In the future, an unsupervised learning approach, such as the use of autoencoders with LSTM, could be explored to extract time series data features without the need for labeled data. Additionally, the current approach of combining time series and snapshot data could be extended to other machine learning models, including reinforcement learning. Given the goal of improving factory performance through situation aware dispatching, training a policy for adapting dispatch rules using reinforcement learning that relies on feedback from factory performance could be more effective than supervised learning, which primarily focuses on improving prediction accuracy.

REFERENCES

Borovkova, S., and I. Tsiamas. 2019. "An Ensemble of LSTM Neural Networks for High-Frequency Stock Market Classification". *Journal of Forecasting* 38:600–619.

Boström, H. 2008. "Calibrating Random Forests". In *2008 Seventh International Conference on Machine Learning and Applications*. Dec 11th-13th, San Diego, CA, 121-126.

Chan, C. W., B. P. Gan, and W. Cai. 2020. "Towards Situation Aware Dispatching in a Dynamic and Complex Manufacturing Environment". In *Proceedings of the 2020 Winter Simulation Conference*, edited by K.-H. G. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, 528–539. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

D-SIMLAB 2023. "Forecaster and Scenario Manager". http://www.d-simlab.com/category/d-simcon/products-d-simcon/forecaster-and-scenario-manager, accessed 15th April.

Denil, M., D. Matheson, and N. de Freitas. 2014. "Narrowing the Gap: Random Forests in Theory and in Practice". In *Proceedings of the 31st International Conference on Machine Learning*, Volume 32. Jun 21nd-26th, Bejing, China, 665-673.

El-Khouly, I. A., K. S. El-Kilany, and A. E. El-Sayed. 2009. "Modelling and Simulation of Re-entrant Flow Shop Scheduling: An Application in Semiconductor Manufacturing". In *2009 International Conference on Computers & Industrial Engineering*. July 6th-9th, Troyes, France, 211-216.

Fan, S. K. S., C. Y. Hsu, D. M. Tsai, F. He, and C. C. Cheng. 2020. "Data-Driven Approach for Fault Detection and Diagnostic in Semiconductor Manufacturing". *IEEE Transactions on Automation Science and Engineering* 17:1925–1936.

Farzad, A., H. Mashayekhi, and H. Hassanpour. 2019. "A Comparative Performance Analysis of Different Activation Functions in LSTM Networks for Classification". *Neural Computing and Applications* 31:2507–2521.

Guo, C., M. Lu, and J. Chen. 2020. "An Evaluation of Time Series Summary Statistics as Features for Clinical Prediction Tasks". *BMC Medical Informatics and Decision Making* 20(1):1–20.

Guyon, I., and A. Elisseeff. 2003. "An Introduction to Variable and Feature Selection". *Journal of Machine Learning Research* 3:1157–1182.

Hassoun, M., D. Kopp, L. Monch, and A. Kalir. 2019. "A New High-Volume/Low-Mix Simulation Testbed for Semiconductor Manufacturing". In *Proceedings of the 2019 Winter Simulation Conference*, edited by N. Mustafee, K.-H. G. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y.-J. Son, 2419–2428. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Hassoun, M., and G. Rabinowitz. 2010. "Hunting Down the Bubble Makers in Fabs". *IEEE Transactions on Semiconductor Manufacturing* 23:13–20.

Hochreiter, S., and J. Schmidhuber. 1997. "Long Short-Term Memory". *Neural Computation* 9:1735–1780.

Jun, S., and S. Lee. 2021. "Learning Dispatching Rules for Single Machine Scheduling with Dynamic Arrivals Based on Decision Trees and Feature Construction". *International Journal of Production Research* 59(9):2838–2856.

Laipple, G., S. Dauzere-Peres, T. Ponsignon, and P. Vialletelle. 2018. "Generic Data Model for Semiconductor Manufacturing Supply Chains". In *Proceedings of the 2018 Winter Simulation Conference*, edited by M. Rabe, A. A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, 3615–3626. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Lee, C., and D. Landgrebe. 1993. "Feature Extraction and Classification Algorithms for High Dimensional Data". Technical report, School of Electrical Engineering, Purdue University.

Leontjeva, A., and I. Kuzovkin. 2016. "Combining Static and Dynamic Features for Multivariate Sequence Classification". In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. Oct 17th-19th, Montreal, QC, 21-30.

Li, X., and S. Olafsson. 2005. "Discovering Dispatching Rules Using Data Mining". *Journal of Scheduling* 8:515–527.

Metan, G., and I. Sabuncuoglu. 2005. "A Simulation Based Learning Meachanism for Scheduling Systems". In *Proceedings of the 2005 Winter Simulation Conference*, edited by M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, 2148–2156. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, A. Müller, J. Nothman, G. Louppe, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and Édouard Duchesnay. 2012. "Scikit-learn: Machine Learning in Python". *Journal of Machine Learning Research* 12:2825–2830.

Priore, P., B. Ponte, J. Puente, and A. Gómez. 2018. "Learning-based Scheduling of Flexible Manufacturing Systems Using Ensemble Methods". *Computers and Industrial Engineering* 126:282–291.

Saadatnejad, S., M. Oveisi, and M. Hashemi. 2020. "LSTM-Based ECG Classification for Continuous Monitoring on Personal Wearable Devices". *IEEE Journal of Biomedical and Health Informatics* 24(2):515–523.

Sabuncuoglu, I. 1998. "A Study of Scheduling Rules of Flexible Manufacturing Systems: A Simulation Approach". *International Journal of Production Research* 36:527–546.

Schulz, B., C. Jacobi, A. Gisbrecht, A. Evangelos, C. W. Chan, and B. P. Gan. 2022. "Graph Representation and Embedding for Semiconductor Manufacturing Fab States". In *Proceedings of the 2022 Winter Simulation Conference*, edited by B. Feng, G. Pedrielli, Y. Peng, S. Shashaani, E. Song, C. G. Corlu, L. H. Lee, E. P. Chew, T. Roeder, and P. Lendermann, 3382–3393. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Shiue, Y.-R., R.-S. Guh, and K.-C. Lee. 2011. "Study of SOM-based Intelligent Multi-controller for Real-time Scheduling". *Applied Soft Computing* 11:4569–4580.

Shiue, Y. R., K. C. Lee, and C. T. Su. 2020. "A Reinforcement Learning Approach to Dynamic Scheduling in a Product-Mix Flexibility Environment". *IEEE Access* 8:106542–106553.

Shiue, Y.-R., and C.-T. Su. 2003. "An Enhanced Knowledge Representation for Decision-Tree Based Learning Adaptive Scheduling". *International Journal of Computer Integrated Manufacturing* 16:48–60.

Spier, J., and K. Kempf. 1995. "Simulation of Emergent Behavior in Manufacturing Systems". In *Proceedings of SEMI Advanced Semiconductor Manufacturing Conference and Workshop*. Nov 13th-15th, Cambridge, MA, 90-94.

Stöckermann, P. 2022. "Dispatching in Real Frontend Fabs with Industrial Grade Discrete-Event Simulations by Use of Deep Reinforcement Learning". In *Proceedings of the 2022 Winter Simulation Conference*, edited by B. Feng, G. Pedrielli, Y. Peng, S. Shashaani, E. Song, C. G. Corlu, L. H. Lee, E. P. Chew, T. Roeder, and P. Lendermann. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Tseng, Y.-J., X.-O. Ping, J.-D. Liang, P.-M. Yang, G.-T. Huang, and F. Lai. 2015. "Multiple-Time-Series Clinical Data Processing for Classification With Merging Algorithm and Statistical Measures". *IEEE Journal of Biomedical and Health Informatics* 19(3):1036–1043.

van der Maaten, L., and G. Hinton. 2008. "Visualizing Data Using t-SNE". *Journal of Machine Learning Research* 9(86):2579–2605.

Waschneck, B., A. Reichstaller, L. Belzner, T. Altenmuller, T. Bauernhansl, A. Knapp, and A. Kyek. 2018. "Deep Reinforcement Learning for Semiconductor Production Scheduling". In *2018 29th Annual SEMI Advanced Semiconductor Manufacturing Conference (ASMC)*. April 30th-May 3rd, Saratoga Springs, NY, 301-306.

Wu, S.-Y. D., and R. A. Wysk. 1989. "An Application of Discrete-Event Simulation to On-line Control and Scheduling in Flexible Manufacturing". *International Journal of Production Research* 27:1603–1623.

AUTHOR BIOGRAPHIES

CHEW WYE CHAN is a Software Engineer of D-SIMLAB Technologies (Singapore). He holds a Master of Computing degree from the National University of Singapore. He is currently working as a doctoral student at the School of Computer Engineering at Nanyang Technological University, Singapore. His research interests include machine learning, data science, and simulation-based optimization. His email address is chew.wye@d-simlab.com.

BOON PING GAN is the CEO of D-SIMLAB Technologies (Singapore). He has been involved in simulation technology application and development since 1995, with a primary focus on developing parallel and distributed simulation technology for complex systems such as semiconductor manufacturing and aviation spare inventory management. He was also responsible for several operations improvement projects with wafer fabrication clients which concluded with multi-million dollar savings. He holds a Master of Applied Science degree, specializing in Computer Engineering. His email address is boonping@d-simlab.com.

WENTONG CAI is a Professor in the School of Computer Science and Engineering at Nanyang Technological University, Singapore. He received his Ph.D. in Computer Science from University of Exeter (UK) in 1991. His expertise is mainly in the areas of Modeling and Simulation and Parallel and Distributed Computing. He has published extensively in these areas and has received a number of best paper awards at the international conferences for his research in distributed simulation. He is an Associate Editor of the ACM Transactions on Modeling and Computer Simulation (TOMACS), an Editor of the Future Generation Computer Systems (FGCS), and in the Editorial Board of the Journal of Simulation. His email address is aswtcai@ntu.edu.sg.