Proceedings of the 2024 Winter Simulation Conference
H. Lam, E. Azar, D. Batur, S. Gao, W. Xie, S. R. Hunter, and M. D. Rossetti, eds.

ENHANCING MACHINE LEARNING FOR SITUATION AWARE DISPATCHING THROUGH GENERATIVE ADVERSARIAL NETWORK BASED SYNTHETIC DATA GENERATION

Chew Wye Chan[1,2], Boon Ping Gan[1], and Wentong Cai[2]

[1]D-SIMLAB Technologies Pte Ltd, SINGAPORE
[2]College of Computing and Data Science, Nanyang Technological University, SINGAPORE

ABSTRACT

Adapting dispatch rules via machine learning in a complex manufacturing environment has shown overall factory performance in various studies. However, the performance of the machine learning model depends on the training data. Limited data could reduce the prediction accuracy of the machine learning model, thereby negatively influencing the overall factory performance. Addressing this, we generate synthetic data for the lot attributes, simulate it through a discrete event simulator, and use the resulting data to improve the prediction accuracy for the machine learning model. We evaluate three synthetic data generation approaches: Latin Hypercube, Synthetic Minority Oversampling Technique, and Generative Adversarial Networks (GAN), demonstrating GAN suitability for synthetic data generation. To validate our approach, we apply two evaluation processes: Train on Real, Test on Real, and Train on Synthetic, Test on Real, showing the improved predictive accuracy of the machine learning model when trained with synthetic data.

1 INTRODUCTION

In a complex manufacturing environment such as the semiconductor industry, scheduling problems are NP-complete (Garey and Johnson 1979) and are usually done using dispatching rules (Metan and Sabuncuoglu 2005). However, it is concluded that no single dispatch rule consistently outperforms another in improving factory performance (Sabuncuoglu 1998). On the other hand, it is shown that adapting dispatch rules for lot dispatching can improve factory performance (Wu and Wysk 1989; Shiue et al. 2020). Machine learning has been used for lot dispatching to show improvement in factory performance by adapting dispatching rules to different manufacturing situations (Shiue and Su 2003; Shiue et al. 2011; Waschneck et al. 2018).

The performance of the machine learning model depends on the training data. However, historical manufacturing data might not exist or be missing due to introducing a new product (Libes et al. 2015). Due to the dynamic nature of semiconductor manufacturing, the manufacturing situations might differ greatly from historical ones. For example, the historical product mix consists of the majority of long cycle time products, whereas the current product mix consists of the majority of short cycle time products. If the training data of the manufacturing situations is limited, this will impact the prediction accuracy of the machine learning model. Hence, there is a need to have an approach to generate synthetic data to increase the training data on various manufacturing situations. On the other hand, a discrete event simulation considers various events in a manufacturing situation with high fidelity (Jahangirian et al. 2010; Seidel et al. 2017). By carefully designing factors that impact product mixes, it becomes feasible to simulate a wide range of manufacturing situations (Sanchez and Wan 2015).

On the other hand, most manufacturing situations may have the same dispatch rule label based on simulation evaluation (Shiue et al. 2011). This might give rise to imbalanced class distributions. The imbalanced distribution of classes in the training data happens when the proportion of one class is higher than the other classes (Sun et al. 2009). The majority class has the most training data, and the minority class has the least training data. The imbalanced class distribution would result in the machine learning algorithm better classifying the majority class (He et al. 2008). In comparison, the minority class will be considered a

979-8-3315-3420-2/24/$31.00 ©2024

noise or rare event by the machine learning algorithm, leading to misclassification instances. Manufacturing situations that belong to the minority class should not be regarded as noise and discarded. Hence, there is a need to generate more synthetic manufacturing situations for the minority class manufacturing situations.

This work evaluates three synthetic data generation approaches: Latin Hypercube (LH), Synthetic Minority Oversampling Technique (SMOTE), and Generative Adversarial Networks (GAN), to train a supervised learning model for dispatch rule prediction. SMOTE and GAN are chosen because they are machine learning techniques used for generating synthetic data (Chawla et al. 2002; Goodfellow et al. 2014). On the other hand, LH is used in the design of experiments for simulation studies due to its space-filling properties, allowing for comprehensive exploration of identified factors (Sanchez and Wan 2015). It is included in our comparison because, similar to SMOTE and GAN, it serves as a method to generate synthetic data (Sanchez and Sánchez 2017). We showed that GAN, originally used in synthetic image generation, could be adapted for synthetic data generation for semiconductor manufacturing. We apply two evaluation processes: Train on Real, Test on Real, and Train on Synthetic, Test on Real to validate our approaches.

2 LITERATURE REVIEW

2.1 Online-SMSP Problem

The complexity of semiconductor manufacturing, characterized by its reentrant process flow and multiple equipment (Mönch et al. 2011), presents complications in creating synthetic data. For example, to create synthetic data of lots queuing for the target equipment, the reentrant flow must be analyzed for all possible equipment that can feed lot to the target equipment. To effectively generate synthetic scenarios for such a complex system, the reentrant process flow and every piece of equipment in the model would need to be accounted for, leading to a complicated process for synthetic data generation.

On the other hand, an Online Single Machine Scheduling Problem (online-SMSP) consists of a single machine with a different lot's release time to the machine (Jun et al. 2019). It is a simpler model and, hence, is preferred for synthetic data generation in this study. In an online-SMSP, the dispatcher can only act on lots post lot release, and the lot attribute values are not available for any decision making beforehand. Using online-SMSP simplifies the semiconductor manufacturing model (Gupta and Sivakumar 2005) and enables the study of the synthetic data generation approach in semiconductor manufacturing.

In online-SMSP, each lot contains four lot attributes: w_a is the weight of lot a, r_a is the release time of lot a, p_a is the processing time of lot a, and d_a is the due time of lot a. The attributes of a lot become known to the system immediately upon the lot's release (Jun et al. 2019). The system then must decide at each time which lot to process (Pruhs et al. 2004). Total weighted tardiness is used as the performance measure and is defined as equation (1) where c_a is the completion time of lot a and n is the total number of complete lots.

$$SumWTardiness = \sum_{a=1}^{n} w_a \cdot \max(0, c_a - d_a) \tag{1}$$

Lot attributes in semiconductor manufacturing refer to the information of individual lots at a time instance, by capturing each lot's status and characteristics within the manufacturing process (Laipple et al. 2018). These attributes, which include processing time, due date, current operation in the production, and queue time, collectively provide a comprehensive and dynamic view of each lot. Lot attributes have also been used as features in various machine learning studies to generate knowledge for lot dispatching (Olafsson and Li 2010; Waschneck et al. 2018; Jun et al. 2019).

2.2 Knowledge Generation

The lot dispatching knowledge is to predict the appropriate dispatch rule for lot dispatching for a given manufacturing situation (Priore et al. 2014). A general overview of the knowledge generation approach is shown in Figure 1 as proposed by Priore et al. (2014). The Manufacturing System provides the manufacturing situation, and the simulation model in the Example Generator is used to evaluate the best dispatch rule for the manufacturing situation. The output from the Example Generator is used as the input training data for the machine learning algorithm and generates knowledge of the relationship between the manufacturing situation and dispatch rule. The knowledge generated is then used to control the lot dispatching in the manufacturing system. Knowledge-based dispatching approaches offer greater flexibility and adaptability in dynamic manufacturing environments compared to traditional scheduling procedures (Priore et al. 2014). Once the knowledge is acquired, this approach enables real-time adjustments and optimizations that can more effectively handle the complexities and variability of semiconductor manufacturing.

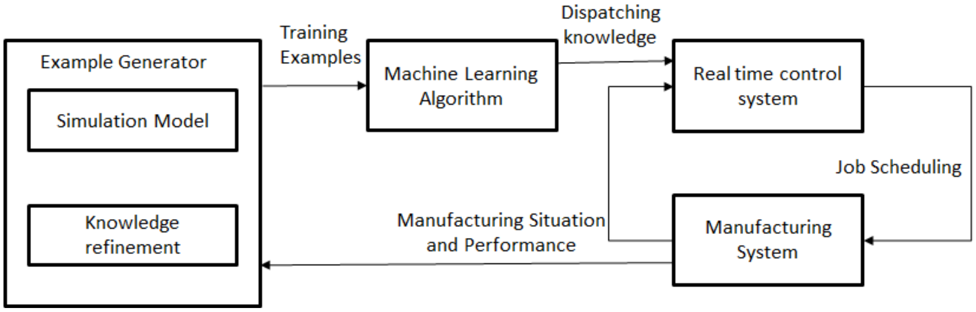

Figure 1: General overview of knowledge generation approach (Priore et al. 2014).

Decision trees (DT) have been used as the machine learning algorithm to generate lot dispatching knowledge in semiconductor manufacturing, demonstrating improvements in factory performance by adapting dispatch rules according to the manufacturing situation (Shiue and Su 2003; Priore et al. 2010; Jun et al. 2019; Chan et al. 2020). However, the performance of the DT depends on the training data (Shiue and Su 2003). Therefore, more training data is required to enhance the predictive accuracy of the decision tree model. Generating synthetic data to supplement the limited historical data can provide the diverse dataset needed for training, ensuring the DT can accurately predict the appropriate dispatching rule across a wide range of manufacturing situations.

2.3 Latin Hypercube

The design of experiments is traditionally used to analyze the impact of different experiment factors and their interactions (Sanchez and Wan 2015). However, in complex semiconductor manufacturing, many factors could impact factory performance. A brute force approach in analyzing these factors is not practical. Suppose there are 100 factors, each factor has two levels (binary), a brute force approach will result in 2^{100} possible experiments, and it is impossible to simulate them in a reasonable time (Sanchez and Wan 2015).

Latin Hypercube (LH) designs provide an efficient way of constructing the experiment designs (Cioppa and Lucas 2007). This approach has been used to generate experiments to evaluate various dispatch rules in semiconductor manufacturing (Feldkamp et al. 2015). LH partitions each factor to an equally spaced factor level, and it offers space-filling properties of experiment design by randomly sampling input factors from each partition. LH design requires the definition of boundaries and intervals of each factor that could impact the experiments. The limitation of LH is that it might create an invalid manufacturing situation in reality. For example, it is possible to create a product mix that will result in low utilization of bottleneck equipment (Feldkamp et al. 2015), but in reality, the factory would ensure enough lot to utilize the factory capacity.

2.4 Synthetic Minority Oversampling Technique

Synthetic Minority Oversampling Technique (SMOTE) is an oversampling approach designed by (Chawla et al. 2002) specifically for class imbalanced problems. In this approach, predefined data points from minority classes are sampled. A line segment is created between the samples, and new synthetic data is generated by randomly sampling a point along that line. The synthetic data that are generated with SMOTE only consider the line segments drawn from the minority class samples. This approach will overgeneralize the minority class space without considering majority class data and increasing overlapping classes (He et al. 2008). Furthermore, the oversampling approach generates synthetic data along the boundaries of existing data.

SMOTE has been used for synthetic data generation to improve the performance machine learning model performance on imbalanced datasets for fault detection prediction models in semiconductor manufacturing processes (Kim et al. 2017). This study highlights the effectiveness of SMOTE in handling class imbalance and improving fault detection rates in highly imbalanced datasets common in semiconductor manufacturing. Additionally, Cho et al. (2022) explored various data preprocessing combinations, including SMOTE, to improve classification performance in manufacturing processes. Seol et al. (2023) investigated the use of SMOTE and other oversampling techniques to address class imbalance in semiconductor equipment anomaly detection and improve classification performance for fault detection. These applications highlight the adaptability of SMOTE in the semiconductor manufacturing domain, making it a valuable tool for generating synthetic data to improve the prediction accuracy of the machine learning model.

2.5 Generative Adversarial Network

Generative Adversarial Network (GAN) is used in image processing to discover and learn patterns in the training data (Goodfellow et al. 2014). GAN is unique as the number of output vectors equals the input vector count. This is useful for synthetic image generation, where a GAN model is trained on a set of photographs to generate new images that look realistic to the human observer (Salimans et al. 2016). GAN consists of two neural networks: a generator and a discriminator. The discriminator learns to differentiate the real samples from the synthetic sample, while the generator learns to generate a synthetic sample that could fool the discriminator. The two neural networks try to defeat each other by learning from each other's loss functions.

GAN has been explored to generate synthetic data for energy data (Fekri et al. 2019) and imbalance class problems (Ali-Gombe and Elyan 2019). Recent studies have further demonstrated the effectiveness of GAN in enhancing data generation accuracy and anomaly detection in semiconductor manufacturing. GAN is applied to generate synthetic data for fault detection in semiconductor manufacturing processes (Choi et al. 2023). By generating additional data for the minority class, the study demonstrated improved modeling performance in fault detection systems for plasma etching processes. Similarly, Lee et al. (2022) proposed applying GAN as missing data imputation method in the semiconductor industry, improving prediction performance when data is missing.

3 METHODOLOGY

An online-SMSP scenario consists of lot attributes for *n* number of lots. Figure 2 shows the synthetic data generation process. The *Factory Situation Generator* component receives lot attribute data and uses three different approaches to generate synthetic lot attributes. The subsections below discuss the data representation for the *Factory Situation Generator* and the three different approaches used to generate synthetic data.

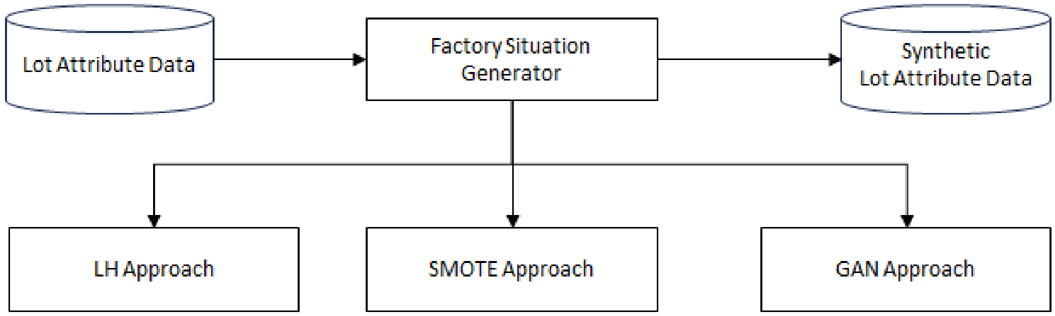

Figure 2: Synthetic data generation process for *Factory Situation Generation*.

3.1 Data Representation

The lot attributes $\langle w, r, p, d \rangle$ of n number of lots are passed to *Factory Situation Generator* to generate synthetic data. The synthetic data would comprise of n number of lots, each with $\langle w, r, p, d \rangle$ lot attributes. Assuming $n = 5$, meaning a scenario with 5 lots, each containing 4 attributes, then the vector is denoted as $X_{5\times4}$. The sample matrix is shown in Table 1.

Table 1: Matrix representation of $X_{5\times4}$ for manufacturing situation with 5 lots.

Lot Attribute a	w	r	p	d
1	$x_{1,w}$	$x_{1,r}$	$x_{1,p}$	$x_{1,d}$
2	$x_{2,w}$	$x_{2,r}$	$x_{2,p}$	$x_{2,d}$
3	$x_{3,w}$	$x_{3,r}$	$x_{3,p}$	$x_{3,d}$
4	$x_{4,w}$	$x_{4,r}$	$x_{4,p}$	$x_{4,d}$
5	$x_{5,w}$	$x_{5,r}$	$x_{5,p}$	$x_{5,d}$

There is a limitation with this representation. This representation of $X_{5\times4}$ can only generate one scenario for online-SMSP. One approach in generating many scenarios, with each scenario containing 5 lots, is to set n to a large value, effectively generating $X_{n\times4}$ of n synthetic lots. Then apply a search algorithm to select 5 synthetic lots to generate $X_{5\times4}$. However, this approach would result in high computing resources, as both the input and output data are comprised of many lots, and there is a need to ensure that the search algorithm selects unique lots. To avoid the issue, the matrix representation is changed to as follows: to generate Z number of scenarios, with each scenario containing n number of lots, the representation is $X_{Z\times(n\times4)}$. This is the data representation for the synthetic lot generation for *Factory Situation Generator*. Table 2 shows the data representation for the *Factory Situation Generator*, with n lot's and Z scenarios.

Table 2: Data representation of $X_{Z\times(n\times4)}$.

Scenario	lot 1				…	lot n			
	w	r	p	d	…	w	r	p	d
1	$x_{1,1,w}$	$x_{1,1,r}$	$x_{1,1,p}$	$x_{1,1,d}$	…	$x_{1,n,w}$	$x_{1,n,r}$	$x_{1,n,p}$	$x_{1,n,d}$
…	…	…	…	…	…	…	…	…	…
Z	$x_{Z,1,w}$	$x_{Z,1,r}$	$x_{Z,1,p}$	$x_{Z,1,d}$	…	$x_{Z,n,w}$	$x_{Z,n,r}$	$x_{Z,n,p}$	$x_{Z,n,d}$

3.2 Synthetic Data Generation

3.2.1 LH Approach

LH is a stratified random procedure by obtaining sample data points from their multivariate distributions (Cioppa and Lucas 2007; Li et al. 2017). Section 3.1 shows the data representation of the synthetic data as $X_{Z\times(n\times4)}$ where Z is the number of scenarios to generate and $n\times4$ is the number of variables to generate for the scenario. We denote $n\times4$ as d in the following description. A LH with Z scenarios and d random variables is denoted as $X_{Z\times d}=[x_1^T,x_2^T,\ldots,x_Z^T]^T$, where each row $x_i=[x_{i1},x_{i2},\ldots,x_{id}]$ represents a sample scenario. The procedure of $X_{Z\times d}$ is generated as follows (Mckay et al. 2000):

1. Assuming that each variable $X_j\sim U(0,1)$, $j=1,2,\ldots,d$. Divide the range of X_j into Z equal intervals $u_{ij}=\left[\frac{(i-1)}{Z},\frac{i}{Z}\right],i=1,2,\ldots,Z.$
2. For the ith interval of variable X_j, the sampled probability is determined by, $\text{Prob}_{ij}=\frac{(i-1)}{Z}+\frac{v_{ij}}{Z}$ where v_{ij} are either 0.5 or independent random variables in $[0,1]$.
3. For each variable X_j, the values $x_{ij}^{\#}$ are obtained by a selective random permutation of Prob_{ij}.
4. Transform $x_{ij}^{\#}$ into sampled value x_{ij} using the inverse of the cumulative distributions $x_{ij}=F_j^{-1}(x_{ij}^{\#})$.

In our approach, we adapt LH to generate synthetic data using the structure of our training data, with scenarios with $n\times4$ variables, representing n lots, and 4 attributes. In step 4, we utilize the inverse cumulative distributions derived from the training data, assuming uniformity across each variable to determine the actual values for lot attributes. This requires defining lower and upper bounds for each variable based on the training dataset, ensuring that all synthetic data falls within these ranges. While this method guarantees that the synthetic data remains within limits, the appropriate setting of these bounds is a challenge. If defined incorrectly, there's a risk of limiting the diversity of the synthetic scenarios.

3.2.2 SMOTE Approach

SMOTE generates synthetic data for the minority class, and the procedure is described in Section 2.4. In our approach, we adapt the SMOTE algorithm to generate synthetic data across all classes, not just the minority class. Each scenario in our dataset comprises n lots, and our objective is to generate Z such scenarios. The representation of our synthetic data assumes the form $X_{Z\times(n\times4)}$, where each data point is characterized by a feature vector of size $n\times4$. Applying SMOTE, we engage training data of this specified dimensionality. The process involves selecting data points from each class, identifying their nearest neighbors within the same class, and then generating new synthetic data points by interpolating between these selected points and their neighbors. This interpolation is achieved by adding a random value of the difference between the feature vectors to the original data point, thereby producing a new data point that maintains the $n\times4$ structure. This application of SMOTE extends beyond its traditional role in addressing class imbalances to enhance the overall dataset, ensuring that our generated synthetic data represents each class.

3.2.3 GAN Approach

GAN discovers and learns patterns from training data to generate synthetic data with similar characteristics (Goodfellow et al. 2014). GAN consists of two networks: a generator and a discriminator. The objective of the generator is to generate synthetic data. The objective of the discriminator is to learn to differentiate whether the data is real or synthetic. The generator training process is complete when the discriminator cannot differentiate the synthetic data from the real data.

Deep convolutional GAN is applied to generate synthetic data (Radford et al. 2015). A scenario with n number of lots is denoted as $X_{n\times4}$. As the definition of input vector in deep convolution GAN needs to be a square matrix, value (-1) is included for the unused dimensions. When the training completes, synthetic lot attributes can be generated randomly from the latent variable space (Goodfellow et al. 2014). If a scenario contains n number of lots and needs to generate Z number of scenarios, the representation of the synthetic data generated is $X_{Z\times(n\times4)}$. Unlike LH and SMOTE, this approach does not need to predefine the boundary of each dimension or the number of clusters to generate synthetic data. However, the parameters for the generator and discriminator neural networks, like the number of hidden layers, must be set.

3.3 Data Collection via Multi-Pass Simulation

The multi-pass simulation technique simulates and compares candidate dispatch rules for the same decision point (Wu and Wysk 1989). In online-SMSP problem, the objective is to dispatch the n number of lots for a scenario. Hence, in Figure 3, an illustration of the multi-pass simulation is depicted, featuring two candidate dispatch rules (d_1, d_2) and n multi-pass decision points. 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 decision period within the chosen simulation path is then extracted and assigned as the corresponding label.

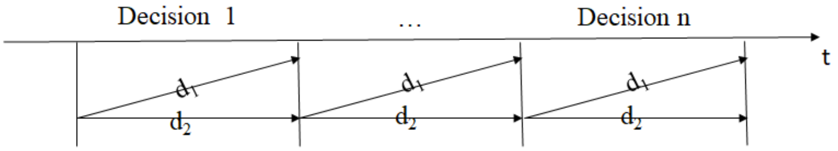

Figure 3: Multi-pass simulation.

Table 3 shows the list of candidate dispatch rules. Lot attributes at each decision point will be collected as shown in Table 4. Features for snapshot data are generated using summary statistics (e.g., $ProT^{St}$ in Table 5). 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 5 shows the statistics features generated from the snapshot data of lot attributes (Chan et al. 2020; Chan et al. 2023).

Table 3: Candidate dispatch rules in the study.

Dispatch Rule	**Description**
SPT	Select the lot with the shortest process time.
EDD	Select the lot with the earliest due date.

Table 4: Lot attribute.

Lot Attribute	**Description**
$ProT_a$	Total processing time of lot a.
$SlkT_a$	Slack time of lot a.
$LtnA_a$	Lateness of lot a.
$QueT_a$	Queue time of lot a.
$RmdD_a$	Remaining due time of lot a.

4 EVALUATION OF SYNTHETIC DATA GENERATION APPROACHES

4.1 Evaluation Process

The main objective of the *Factory Situation Generator* is to generate a large variation of manufacturing situations to refine the knowledge for DT to predict the dispatch rule for a manufacturing situation. The

Table 5: Feature generated by using statistic summary on lot attributes.

Feature	**Description**
$ProT^{St}$	Statistic summary of total 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.

Scikit-learn implementation of DT was used in our experiments (Pedregosa et al. 2012). DT uses synthetic manufacturing situations to improve prediction accuracy for unseen manufacturing situations. Hence, the *Factory Situation Generator* is evaluated by assessing the accuracy of DT. To evaluate the *Factory Situation Generator*, two methods are used: Train on Real Test on Real (TRTR) and Train on Synthetic and Test on Real (TSTR) (Fekri et al. 2019).

TRTR uses cross-validation to evaluate DT. However, in the evaluation, real manufacturing situations are randomly sampled into datasets with increasing numbers of manufacturing situations. This is to evaluate the performance of the DT in the event of different numbers of manufacturing situations. For example, if the real manufacturing situation has 500 scenarios, with each scenario containing n lots, cross-validation is performed on randomly sampled manufacturing situations, each containing 100, 200, 300, 400, and 500 scenarios.

In TSTR, the DT is trained with synthetic manufacturing situations and tested with a real manufacturing situation. The same sampling strategy of TRTR is used for testing.

4.2 Online-SMSP Training Data Generation

A total of n lots are generated for each online-SMSP manufacturing scenario. In the experiments, n is set to 10. A uniform distribution between 1 and 10 is used to decide the p_a and w_a (Chu 1992). The total process time is defined as $\sum_{a=1}^{n} p_a$. The release time of lot a, r_a, is generated from a uniform distribution ranging from 0 to $\alpha \sum_{a=1}^{n} p_a$, where four different α values $[0, 0.5, 1.0, 1.5]$ are used as a scaling factor for the lot's process time (Chu 1992). For due date d_a, slack time is generated from the uniform distribution ranging from 0 to $\beta \sum_{a=1}^{n} p_a$, where 3 different β values $[0.05, 0.25, 0.5]$ are used as a scaling factor for the lot's slack time (Chu 1992). There are a total of twelve combinations of α and β as shown in Table 6. Each $\alpha\beta$ combination generates 100 manufacturing scenarios, and a total of 1200 manufacturing scenarios are generated. The manufacturing scenarios generated using this approach are considered the hypothetical "real manufacturing scenarios" in this experiment.

Table 6: Alpha and beta combinations.

Combination	0	1	2	3	4	5	6	7	8	9	10	11
α	0	0	0	0.5	0.5	0.5	1	1	1	1.5	1.5	1.5
β	0.05	0.25	0.5	0.05	0.25	0.5	0.05	0.25	0.5	0.05	0.25	0.5

Applying the multi-pass simulation technique described in Section 3.3, the *SumWTardiness* can be calculated for every scenario and training data is generated accordingly. Section 3.3 describes the lot attributes used to generate features to represent manufacturing situations. The manufacturing situation features (**S**) and the corresponding best-performing dispatch rule (**D**) to achieve the minimum *SumWTardiness* (**P**) are used as the training data $\{\mathbf{P}, \mathbf{S}, \mathbf{D}\}$ for the DT to generate dispatching knowledge.

4.3 Results

Scenarios are generated using the approach described in Section 4.2 and represent hypothetical "real manufacturing scenarios". They are randomly sampled into 10 sets of data with an incremental number of manufacturing situations [100, 200, 300, 400, 500, 600, 700, 800, 900, and 1000], where 100 means the dataset has 100 manufacturing scenarios. The cross-validation accuracy of TRTR using the above 10 sets of data is shown in Figure 4. In TRTR for this dataset, cross-validation accuracy achieves around 70% for these 10 sets of data. The result shows increased accuracy with a higher number of manufacturing scenarios. The difference between the accuracy of the DT is around 5%, as the scenarios are generated with a uniform distribution of α and β values (Section 4.2), and k-fold validation randomly splits into training and testing data. In practice, less data should result in much lower accuracy in TRTR, because data from semiconductor manufacturing is not uniformly distributed.

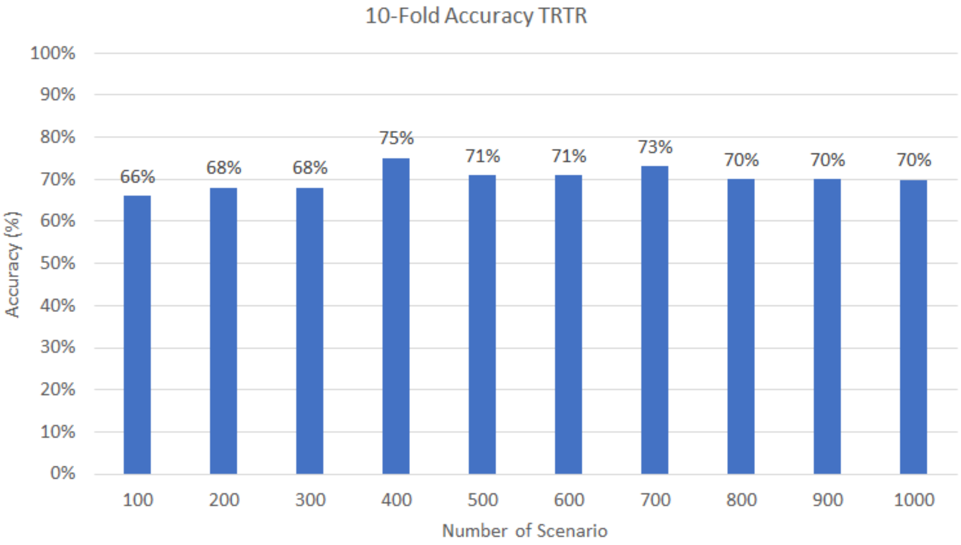

Figure 4: TRTR accuracy.

Using the minimum set of real manufacturing situations (100), 1000 synthetic data are generated using different approaches (LH, SMOTE, GAN) and is used to train the DT. The trained model is then used to test the data representing real manufacturing situations. Figure 5 shows the prediction accuracy of TSTR across ten sets of data for testing. The SMOTE and GAN approach consistently achieves around 70% accuracy (orange and grey bar chart).

LH achieves a low prediction accuracy of around 50% in TSTR. LH treats each variable independently and generates synthetic data based on the boundaries of these individual features. Hence, it overlooks inter-feature relationships, thereby limiting accuracy improvement. In contrast, SMOTE achieves the highest accuracy in TSTR because it is used to generate synthetic data across all classes, not just the minority class. The approach considers the inter-feature relationships through the linear interpolation process of feature vectors. The underlying assumption of uniform data distribution in training data (Section 4.2) is the reason that the SMOTE approach achieves the highest prediction accuracy in TSTR among these approaches. SMOTE applies a linear interpolation process of feature vectors from the training data, achieving higher accuracy.

However, data distributions are varied and non-uniform in semiconductor manufacturing. While GAN achieves slightly less accuracy (less than 5%) in TSTR as compared to SMOTE, its ability to generate diverse synthetic scenarios makes it useful for capturing complex manufacturing situations. GAN learns complex data distributions directly from training data (Goodfellow et al. 2014). While LH and SMOTE are relatively fast to execute, training the GAN requires a longer time due to the complexity of training neural networks. This difference in computational demand illustrates the trade-off between the simplicity of LH and SMOTE and the advanced capabilities of GAN in generating synthetic data.

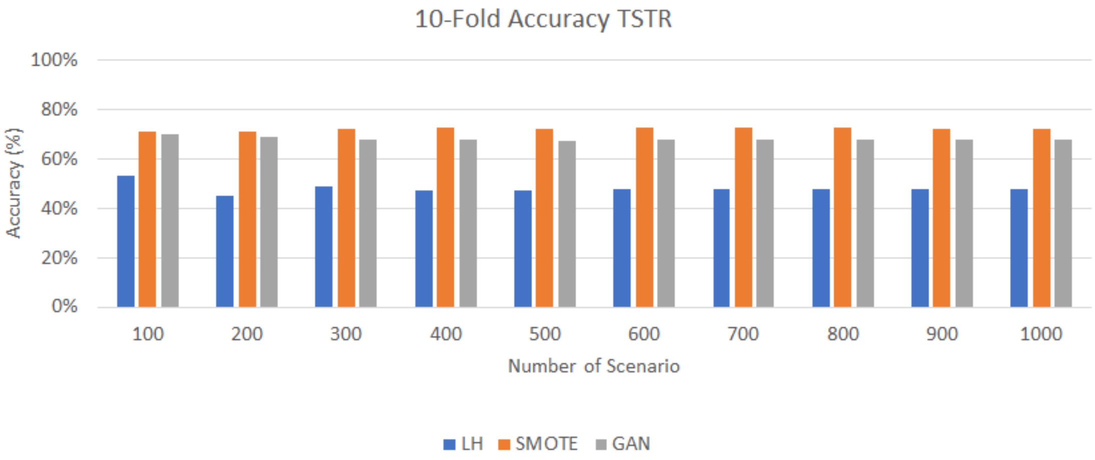

Figure 5: TSTR accuracy.

5 CONCLUSION AND FUTURE WORK

In conclusion, selecting the appropriate dispatch rule is crucial for improving factory performance in a situation-aware dispatching system. As machine learning's performance depends on the training data, this study has analyzed different approaches to generate synthetic data to improve the prediction accuracy of the machine learning model for dispatch rule adaptation.

The result showed that both the SMOTE and GAN approaches can generate synthetic data that achieve 70% prediction accuracy in TSTR. Despite GAN achieving lower prediction accuracy than SMOTE, its ability to learn inter-feature relationships and independence from the assumption of uniform data distributions on training data make it a better approach to generating synthetic data for semiconductor manufacturing.

One limitation of the current approach is the simplification of the semiconductor manufacturing to an online-SMSP problem with a fixed number of lots n. However, the number of lots n at each semiconductor manufacturing equipment depends on the scheduling period. A short scheduling period will result in fewer lots required but does not represent the overall manufacturing situation. Furthermore, there are more lot attributes in semiconductor manufacturing. This will impact the number of synthetic lot attributes that need to be generated. GAN can be extended to more complex semiconductor manufacturing environments, but the number of lots and their attributes will need to be expanded.

REFERENCES

Ali-Gombe, A. and E. Elyan. 2019. "MFC-GAN: Class-imbalanced Dataset Classification Using Multiple Fake Class Generative Adversarial Network". *Neurocomputing* 361:212–221.

Chan, C. W., B. P. Gan, and W. Cai. 2020. "Towards Situation Aware Dispatching in a Dynamic and Complex Manufacturing Environment". In *2020 Winter Simulation Conference(WSC)*, 528–539 https://doi.org/10.1109/WSC48552.2020.9383991.

Chan, C. W., B. P. Gan, and W. Cai. 2023. "Combining Time Series Data and Snapshot Data for Situation Aware Dispatching in Semiconductor Manufacturing". In *2023 Winter Simulation Conference (WSC)*, 2310–2312 https://doi.org/10.1109/WSC60868.2023.10407873.

Chawla, N. V., K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. "SMOTE: Synthetic Minority Over-sampling Technique". *Journal of Artificial Intelligence Research* 16:321–357.

Cho, E., T. W. Chang, and G. Hwang. 2022. "Data Preprocessing Combination to Improve the Performance of Quality Classification in the Manufacturing Process". *Electronics* 11(3):477.

Choi, J. E., D. H. Seol, C. Y. Kim, and S. J. Hong. 2023. "Generative Adversarial Network-Based Fault Detection in Semiconductor Equipment with Class-Imbalanced Data". *Sensors* 23(4):1889.

Chu, C. 1992. “A Branch-and-Bound Algorithm to Minimize Total Tardiness with Different Release Dates”. *Naval Research Logistics (NRL)* 39(2):265–283.

Cioppa, T. M. and T. W. Lucas. 2007. “Efficient Nearly Orthogonal and Space-filling Latin Hypercubes”. *Technometrics* 49(1):45–55.

Fekri, M. N., A. M. Ghosh, and K. Grolinger. 2019. “Generating Energy Data for Machine Learning with Recurrent Generative Adversarial Networks”. *Energies* 13(1):130.

Feldkamp, N., S. Bergmann, and S. Strassburger. 2015. “Knowledge Discovery in Manufacturing Simulations”. In *Proceedings of the 3rd ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, 3–12. New York, NY, USA: ACM.

Garey, M. R. and D. S. Johnson. 1979. *Computers and Intractability: A Guide to The Theory of NP-completeness*. New York: W. H. Freeman and Company.

Goodfellow, I. J., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, *et al*. 2014. “Generative Adversarial Networks”. *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*:2672–2680.

Gupta, A. K. and A. I. Sivakumar. 2005. “Single Machine Scheduling with Multiple Objectives in Semiconductor Manufacturing”. *International Journal of Advanced Manufacturing Technology* 26(9-10):950–958.

He, H., Y. Bai, E. A. Garcia, and S. Li. 2008. “ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning”. In *2008 IEEE International Joint Conference on Neural Networks*, 1322–1328: IEEE.

Jahangirian, M., T. Eldabi, A. Naseer, L. K. Stergioulas and T. Young. 2010. “Simulation in Manufacturing and Business: A Review”. *European Journal of Operational Research* 203(1):1–13.

Jun, S., S. Lee, and H. Chun. 2019. “Learning Dispatching Rules Using Random Forest in Flexible Job Shop Scheduling Problems”. *International Journal of Production Research* 57(10):3290–3310.

Kim, J. K., Y. S. Han, and J. S. Lee. 2017. “Particle Swarm Optimization–deep Belief Network–based Rare Class Prediction Model for Highly Class Imbalance Problem”. *Concurrency and Computation: Practice and Experience* 29(11):e4128.

Laipple, G., S. Dauzere-Peres, T. Ponsignon, and P. Vialletelle. 2018. “Generic Data Model for Semiconductor Manufacturing Supply Chains”. In *2018 Winter Simulation Conference (WSC)*, 3615–3626 https://doi.org/10.1109/WSC.2018.8632349.

Lee, S. Y., T. P. Connerton, Y. W. Lee, D. Kim, D. Kim and J. H. Kim. 2022. “Semi-GAN: An Improved GAN-Based Missing Data Imputation Method for the Semiconductor Industry”. *IEEE Access* 10:72328–72338.

Li, W., L. Lu, X. Xie, and M. Yang. 2017. “A Novel Extension Algorithm for Optimized Latin Hypercube Sampling”. *Journal of Statistical Computation and Simulation* 87(13):2549–2559.

Libes, D., S. Shin, and J. Woo. 2015. “Considerations and Recommendations for Data Availability for Data Analytics for Manufacturing”. In *Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015*, 68–75: Institute of Electrical and Electronics Engineers Inc.

Mckay, M. D., R. J. Beckman, and W. J. Conover. 2000. “A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output From a Computer Code”. *Technometrics* 42(1):55–61.

Metan, G. and I. Sabuncuoglu. 2005. “A Simulation Based Learning Meachanism for Scheduling Systems”. In *2005 Winter Simulation Conference (WSC)*, 2148–2156 https://doi.org/10.1109/WSC.2005.1574500.

Mönch, L., J. W. Fowler, S. Dauzère-Pérès, S. J. Mason and O. Rose. 2011. “A Survey of Problems, Solution Techniques, and Future Challenges in Scheduling Semiconductor Manufacturing Operations”. *Journal of Scheduling* 14(6):583–599.

Olafsson, S. and X. Li. 2010. “Learning Effective New Single Machine Dispatching Rules from Optimal Scheduling Data”. *International Journal of Production Economics* 128(1):118–126.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel *et al*. 2012. “Scikit-learn: Machine Learning in Python”. *Journal of Machine Learning Research* 12:2825–2830.

Priore, P., A. Gómez, R. Pino, and R. Rosillo. 2014. “Dynamic Scheduling of Manufacturing Systems Using Machine Learning: An Updated Review”. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 28(1):83–97.

Priore, P., J. Parreño, R. Pino, A. Gómez and J. Puente. 2010. “Learning-Based Scheduling of Flexible Manufacturing Systems Using Support Vector Machines”. *Applied Artificial Intelligence* 24(3):194–209.

Pruhs, K., J. R. Sgall, and E. Torng. 2004. “Online Scheduling”. In *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, edited by J. Y.-T. Leung, 1–41. Boca Raton, FL: CRC press.

Radford, A., L. Metz, and S. Chintala. 2015. “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”. *CoRR* abs/1511.06434.

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

Salimans, T., I. Goodfellow, W. Zaremba, V. Cheung, A. Radford and X. Chen. 2016. “Improved Techniques for Training GANs”. In *Advances in Neural Information Processing Systems 29*, 2234.

Sanchez, S. M. and P. J. Sánchez. 2017. *Better Big Data via Data Farming Experiments*, 159–179. Cham: Springer International Publishing.

Sanchez, S. M. and H. Wan. 2015. “Work Smarter, Not Harder: A Tutorial on Designing and Conducting Simulation Experiments”. In *2015 Winter Simulation Conference (WSC)*, 1795–1809 https://doi.org/10.1109/WSC.2015.7408296.

Seidel, G., C. F. Lee, A. M. Kam, B. P. Gan, C. W. Chan, A. Naumann *et al*. 2017. “Harmonizing Operations Management of Key Stakeholders in Wafer Fab Using Discrete Event Simulation”. In *2017 Winter Simulation Conference (WSC)*, 3670–3678 https://doi.org/10.1109/WSC.2017.8248079.

Seol, D. H., J. E. Choi, C. Y. Kim, and S. J. Hong. 2023. “Alleviating Class-Imbalance Data of Semiconductor Equipment Anomaly Detection Study”. *Electronics* 12:585.

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.

Sun, Y., A. K. Wong, and M. S. Kamel. 2009. “Classification of Imbalanced Data: A Review”. *International Journal of Pattern Recognition and Artificial Intelligence* 23(4):687–719.

Waschneck, B., A. Reichstaller, L. Belzner, T. Altenmuller, T. Bauernhansl, A. Knapp *et al*. 2018. “Deep Reinforcement Learning for Semiconductor Production Scheduling”. In *2018 29th Annual SEMI Advanced Semiconductor Manufacturing Conference (ASMC)*, 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 College of Computing and Data Science 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 College of Computing and Data Science 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.