



## Article

# Traffic Flow Speed Prediction in Overhead Transport Systems for Semiconductor Fabrication Using Dense-UNet

Young Ha Joo <sup>1</sup>, Hoonseok Park <sup>1</sup>, Haejoong Kim <sup>2</sup>, Ri Choe <sup>3</sup>, Younkook Kang <sup>3</sup> and Jae-Yoon Jung <sup>1,\*</sup><sup>1</sup> Department of Big Data Analytics, Kyung Hee University, Yongin-si 17104, Korea<sup>2</sup> Department of Industrial and Management Engineering, Korea National University of Transportation, Chungju 27469, Korea<sup>3</sup> Material Handling Automation Group, Samsung Electronics, Hwaseong-si 18448, Korea\* Correspondence: [jjjung@khu.ac.kr](mailto:jjjung@khu.ac.kr)

**Abstract:** To improve semiconductor productivity, efficient operation of the overhead hoist transport (OHT) system, which is an automatic wafer transfer device in a semiconductor fabrication plant (“fab”), is very important. A large amount of data is being generated in real time on the production line through the recent production plan of a smart factory. This data can be used to increase productivity, which in turn enables companies to increase their production efficiency. In this study, for the efficient operation of the OHT, the problem of OHT congestion prediction in the fab is addressed. In particular, the prediction of the OHT transport time was performed by training the deep convolutional neural network (CNN) using the layout image. The data obtained from the simulation of the fab and the actual logistics schedule data of a Korean semiconductor factory were used. The data obtained for each time unit included statistics on volume and speed. In the experiment, a layout image was created and used based on the statistics. The experiment was conducted using only the layout image without any other feature extraction, and it was shown that congestion prediction in the fab is effective.



**Citation:** Joo, Y.H.; Park, H.; Kim, H.; Choe, R.; Kang, Y.; Jung, J.-Y. Traffic Flow Speed Prediction in Overhead Transport Systems for Semiconductor Fabrication Using Dense-UNet.

*Processes* **2022**, *10*, 1580. <https://doi.org/10.3390/pr10081580>

Academic Editors: Ming-Jong Tsai and Ricky Min-Fan Lee

Received: 10 July 2022

Accepted: 9 August 2022

Published: 11 August 2022

**Publisher’s Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

**Keywords:** deep convolutional network; UNet; semiconductor; overhead hoist transport; semiconductor fabrication plant

## 1. Introduction

Efficient operation of overhead hoist transport (OHT) systems is important for the productivity of semiconductor processes [1]. In particular, it is important to predict traffic flow and congestion over time because OHT operations, such as dispatching [2–4] and routing [5–8], are highly dependent on traffic conditions. In this study, the p OHT congestion prediction issue is addressed based on volume data. In the past, abnormal flow was detected through an agent-based system; however, this approach requires a schema that considers several factors for accurate prediction. Additionally, there is the possibility of a requirement for a new schema if the condition of the target factory or line under consideration changes.

The semiconductor process is complex and forms an environment in which hundreds of processes overlap. Efficient handling of the process in such a complex environment is a direct productivity issue. The scheduling method has traditionally been used for efficient deployment of OHT [1,9,10]. Recently, data-driven methods using machine learning and deep learning have been used to improve the efficiency of semiconductor processes [11,12]. Production planning and scheduling issues caused by complex environmental factors are solved by applying machine learning to past production data. Wang dealt with cycle time forecasting (CTF), which is an important issue in production planning [12]. In this study, a method for dealing with big data with parallel computing is introduced, and a deep neural network methodology for CTF is presented.

A convolutional neural network (CNN) was trained using a layout image, in which traffic information was input. A study was conducted to predict the OHT congestion in

each section in the near future, and an image was created and used, in which the volume and speed of each section were input at 10 s intervals. In particular, the model was trained by combining six volume images and speed images from the past 60 s. The trained model predicted the average speed for each section for the next 30 s at 10 s intervals. To develop an image-based congestion prediction model, a UNet-based model [13], which can be used for image segmentation, was used. The volume and speed of each section were input to the image as pixel values corresponding to the spatial location of the section, and the average speed for each section was extracted as a prediction result from the predicted future image.

The experiment was performed with the simulation data based on a semiconductor fabrication plant (fab) environment and production schedule data of a semiconductor factory in Korea. The CNN was trained on data for 24 h, the next 6 h were used for validation, and an additional 6 h of data were used for testing. Compared with the baselines used, it showed significant congestion predictability. The contributions of this study can be summarized as follows.

1. In the previous studies, various features have been used to predict the travel time of OHT. However, in this study, without feature extraction, only basic information, such as the average volume and average speed data, was used. The simplicity and robustness of the model were secured by creating an image using only basic information and by identifying the relationship of the layout network.
2. This is the first study to use the UNet model to understand the layout network in the fab. For application to UNet, information is converted into an image and used.

The remainder of this paper is organized as follows: In Section 2, we introduce related studies. In Section 3, the proposed system is introduced. In addition, the overall framework, encoding to create images, CNN-based model, and decoding to read images are introduced. Section 4 introduces the actual data environment and the evaluation results, to which the proposed system is applied. Section 5 presents the conclusions and related future endeavors.

## 2. Background

### 2.1. Convolution Layer

We selected the UNet-based CNN [13] as a model for training to obtain future prediction images in the experiment. A CNN refers to a network, in which a convolutional layer, a pooling layer, and a fully connected layer are stacked to produce an output. The convolutional layer operates on the input data through Equation (1). A filter of a specific size continuously moves the input data using a set stride to create the next feature map. The output from the convolutional layer is typically used for the next operation via a non-linear activation function. We used linear activation, as in Equation (2), ELU [14], as in Equation (3), and sigmoid, as in Equation (4).

$$O_{i,j}^l = \delta \left( \sum_{k=0}^{F-1} \sum_{n=0}^{F-1} w_{k,m} I_{i+k, j+n}^{l-1} \right) \quad (1)$$

$$f(x) = x \quad \text{if } x > 0 \quad (2)$$

$$f(x) = \alpha(e^x - 1) \quad \text{if } x \leq 0 \quad (3)$$

$$s(x) = \frac{1}{1 + e^{-x}} \quad (4)$$

Figure 1 shows the output when the convolution operation is performed using a  $3 \times 3$  filter on the  $4 \times 4$  input data. The filters used shared weights and were continuously updated to determine the optimal value while training the model.



### 3. Proposed Method

#### 3.1. Framework

We obtained logistic data at 10 s intervals from the simulator. These logistic data were imaged one by one. When creating an image, the input volume and speed information are obtained using the coordinate information for each section in a 2D array. By concatenating the image created in this manner into channels by a certain window size, we obtain the input of the learning model. Additionally, we needed a label to be used when training the model. These labels used future images. As with input data, it is possible to consider future information for a specific time by concatenating it into channels of a specific window size. In our experiment, we addressed the problem of predicting 30 s into the future using data from the past 60 s. The model was trained using input and label data collected in this manner. When the model is trained, it outputs an image containing the prediction value of the future 30 s for the input. Finally, to compare our logistic data with the predicted value, we converted the predicted image back to the logistic data. Through this process, the learning result can be obtained with the volume and speed information in the fab, and a visualization of all these processes is shown in Figure 3.

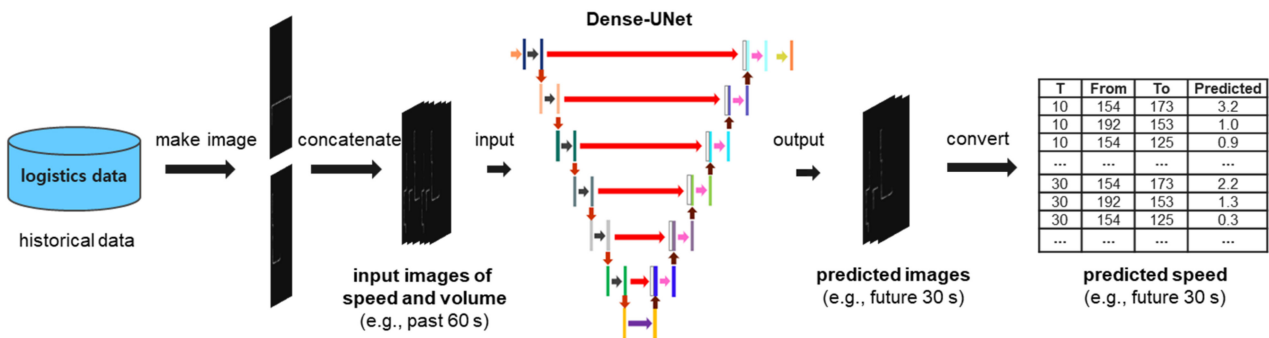


Figure 3. The procedure of the proposed method.

#### 3.2. Encoding Logistic Data to Traffic Image

Here, the process of converting logistic data into image data is described in detail. Because we know the coordinate values of all the sections, we can map the volume and velocity information of each section to specific coordinate values in a 2D array, as shown in Figure 4. However, if the size of the image becomes too large, it is difficult to use it as an input to the model; therefore, there are cases where pure coordinate information cannot be used as it is. Consequently, the coordinate information is mapped to a 2D array using a value divided according to the image size. At this time, the volume and speed values are converted to values between 0 and 255 for mapping, and the maximum volume and speed used according to the data are set and used. Because one piece of logistic data has two pieces of information, volume and speed, two images can be obtained from one piece of logistic data: a volume image and a speed image. As shown in Figure 5, by concatenating the created images one by one on a channel basis, it creates input data, which considers past information up to a specific time and creates label data, which considers future information up to a specific time.

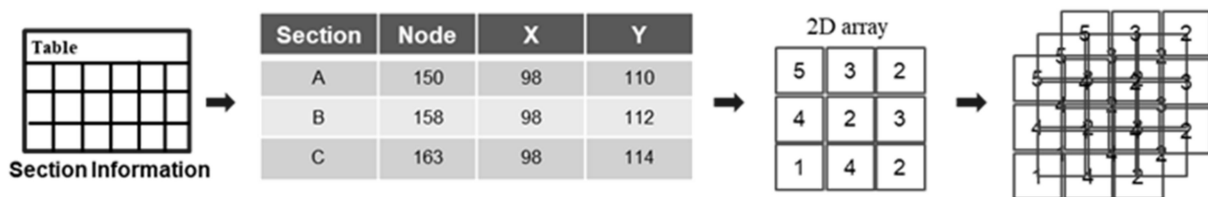


Figure 4. Conversion of logistic data into images.



**Figure 5.** Configuration of input and label data through window size setting.

### 3.3. Decoding Traffic Image to Logistic Data

From the trained model, we obtained a predicted image for the future. As the output image is grouped by the channel standard, it is used to obtain the predicted value by dividing it by the channel standard. In the encoding process, two images were obtained with one logistic dataset; however, in the decoding process, one logistic dataset was created from two output images. The volume and speed values were mapped to a 2D array using specific coordinate values during the encoding process. Inverse transformation was performed again using the maximum volume and speed used in the mapping process. First, the average value of the pixels between the two is calculated based on the start and end nodes of one section. The average pixel value obtained in this manner is inversely transformed into volume and speed and is used as a prediction value.

## 4. Experiments

### 4.1. OHT Traffic Data

The data used in the experiments were simulated data based on the actual semiconductor factory data. The semiconductor factory consists of several floors and buildings; however, we analyzed data from only one floor. It is composed of thousands of nodes, and the connection of nodes is called “link,” and the connection of several “links” is called a “section”. We created one image for thousands of sections and proceeded with the learning. In the 10 s interval data, 24 h were used as training data, the other 6 h were used as validation data, and another 6 h were used as test data.

### 4.2. Encoding Traffic Data to Traffic Image

We trained the model by creating images of the past 60 s of data as the input data. The volume or speed information was mapped onto one image channel. When mapping information to an image, using the actual coordinate value creates an unusable image size; therefore, the actual coordinate value is divided by 1000 and then used. Two pieces of information, volume and speed, can be found in the data at 10 s intervals. In other words, the input data had  $2 \times 6 = 12$  channels, and the image size was (512, 256) considering the coordinates of the sections. For training data,  $24 \text{ h} \times 60 \text{ min} \times 6 - 3$  (last window of data) = 8637 images were created, and  $6 \times 60 \times 6 - 3$  (last window of data) = 2157 images for validation and test data were created.

The output shape of each layer is listed in Table 1.  $B$  in the output shape indicates the batch size. The size of the first input image was (512, 256). In the contracting path, the channel size increases and the image size decreases. The image size increases again through the expanding path, and the size of the final output image becomes the same as that of the input image.

### 4.3. CNN Based Prediction Model for Traffic Image

When training our model, the shape of the first input image was (12, 512, 256). In the contracting path, the number of channels increases and the image size decreases as it passes through the dense block and pooling layer. If it passes through six dense blocks and a pooling layer, the output shape becomes (128, 8, 4). The output of the contracting path passes through one convolution layer, and then through an expanding path. In the expanding path, the size of the image is increased again through the convolution transpose layer, which works in a manner opposite to the convolution layer. The output shape that has passed through the six convolution transpose layers becomes (128, 512, 256), and the final output of the shape (6, 512, 256) is the output through the last convolution layer and activation function. The output shapes passing through several layers are summarized in Table 1.

**Table 1.** Output shape of each layer in Dense-UNet.

Layer	Output Shape
Dense Block 1	( $B, 64, 512, 256$ )
Average Pooling	( $B, 64, 256, 128$ )
Dense Block 2	( $B, 96, 256, 128$ )
Average Pooling	( $B, 96, 128, 64$ )
...	...
Dense Block 6	( $B, 128, 16, 8$ )
Average Pooling	( $B, 128, 8, 4$ )
Convolution Layer	( $B, 128, 8, 4$ )
Convolution Transpose 6	( $B, 128, 16, 8$ )
Convolution Transpose 5	( $B, 128, 32, 16$ )
...	...
Convolution Transpose 2	( $B, 128, 256, 128$ )
Convolution Transpose 1	( $B, 128, 512, 256$ )
Convolution Layer	( $B, 6, 512, 256$ )

#### 4.4. Decoding Traffic Image to Traffic Data

Because we predict the future 30 s at 10 s intervals, our model outputs six channels with volume and speed as channels. Each channel represents a volume and speed of 10 s, 20 s, and 30 s in the future. Because we know the location information of each section used in the encoding, we can read the pixel value of the required section from the predicted image. Only the pixel values between the start and end positions of the section are averaged and used as the pixel values for the section. After obtaining the average pixel value of each section, the prediction speed value is obtained through an inverse transformation process of mapping to the image.

#### 4.5. Experimental Results

We compared the experimental results with the following four models:

1. Historical average (HA) model: The overall average speed of the train data is used as a predicted value;
2. Rct60s model: The average speed from the current time point  $t$  to the past 60 s is used as the predicted value;
3. Rct30s model: The average speed from the current time point  $t$  to the past 30 s is used as the predicted value;
4. Rct10s model: The average speed from the current time point  $t$  to the past 10 s is used as the predicted value.

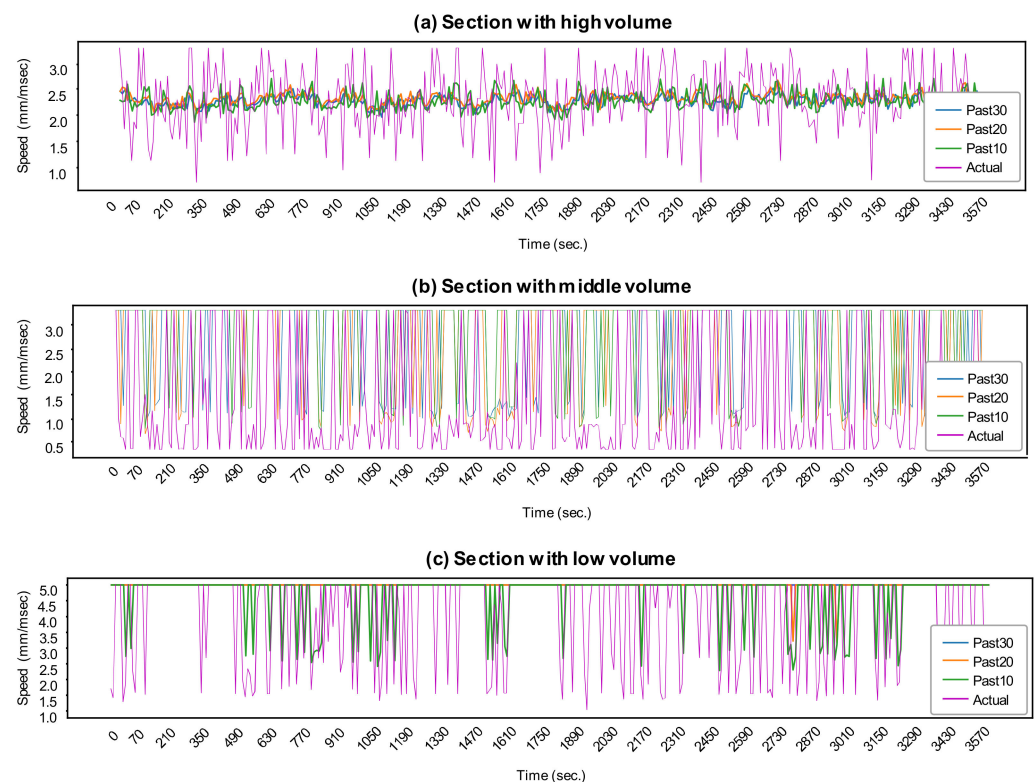
Root mean square error (RMSE), mean absolute error (MAE) and coefficient of determination ( $R^2$ ) are used as evaluation indicators. To predict the near future, we divide the time interval into 10 s, 20 s and 30 s to predict the future average speed. Table 2 shows the experimental results for each time interval  $T$ . The experiment was conducted in a PyTorch environment using Intel Xeon Silver 4210 CPU and Nvidia Titan RTX 24 GB.

**Table 2.** Experimental results according to time interval  $T$ .

$T$	MAE			RMSE			$R^2$		
	10 s	20 s	30 s	10 s	20 s	30 s	10 s	20 s	30 s
HA	0.613	0.613	0.613	0.901	0.901	0.901	0.671	0.671	0.671
Rct60s	0.615	0.62	0.618	0.972	0.979	0.976	0.617	0.612	0.614
Rct30s	0.618	0.625	0.623	1.001	1.016	1.012	0.59	0.582	0.585
Rct10s	0.625	0.642	0.637	1.1	1.12	1.112	0.51	0.492	0.499
Dense-UNet	0.490	0.534	0.548	1.04	1.118	1.125	0.561	0.494	0.487

As shown in Table 2, the MAE of the average speed prediction of 10 s, 20 s, and 30 s in the future is the smallest value when using Dense-UNet. In contrast, RMSE and  $R^2$  yielded the best results for the HA model. In the case of a large error, the performance of the HA model is better; however, the absolute difference between the actual value and the predicted value shows that Dense-UNet has a better prediction performance. It can be observed that the error tends to increase as the time interval  $T$  increases. It has been confirmed that it predicts the near time more accurately; it becomes more difficult to predict further into the future.

The average volume of each section differed. The change in the average speed prediction according to the average volume can be expressed as a time series graph, as shown in Figure 6. Figure 6a shows a section with a high volume of traffic, Figure 6b shows a section with medium volume, and Figure 6c shows a section with low volume.



**Figure 6.** Time series graph of predicted and actual values for each time interval; (a) section with high volume, (b) section with medium volume, and (c) section with low volume.

In the case of high traffic volume, the change in average speed appears very rapidly in a short time. In this case, the predicted value follows the trend of the average speed change but does not show a large change in value. In the case of medium volume, the speed decreases momentarily and then returns to the standard speed. In this case, it can be confirmed that the predicted value follows the trend and exhibits a large change in value. It can be observed that changes in the average speed occur occasionally when the volume of traffic is small, and in this case, it was confirmed that the predicted value partially followed the instantaneous fall.

## 5. Conclusions and Future Work

In this study, we dealt with the time series problem through the segmentation technique Dense-UNet. Unlike previous studies that have generated and utilized many features for time series prediction, this study used only the most basic average volume and average speed information. In addition, to understand the flow in the fab, we present a method for training the model by converting the information into an image.

For the future 10 s, 20 s, and 30 s predictions, Dense-UNet showed the best results for MAE, and the HA model was good in terms of RMSE and  $R^2$ . Additionally, we examined the time series graph according to the volume, and it was confirmed that the predicted values changed effectively for the sections with a medium volume.

Although only the basic information of each section was used, it is expected that design information, such as section length, can be used in future studies. In addition, because it is not very efficient to use a complex model to predict the sections with a very low traffic volume, it is expected that it will be possible to train the model by selecting only the sections with a high traffic volume and creating a smaller image.

**Author Contributions:** Conceptualization, Y.H.J., H.K. and J.-Y.J.; methodology, Y.H.J. and J.-Y.J.; software, Y.H.J.; validation, H.P., H.K., R.C., Y.K. and J.-Y.J.; data curation, Y.K. and R.C.; writing—original draft preparation, Y.H.J.; writing—review and editing, H.P. and J.-Y.J.; visualization, Y.H.J. and J.-Y.J.; supervision, J.-Y.J.; project administration, J.-Y.J.; funding acquisition, J.-Y.J. All authors have read and agreed to the published version of the manuscript.

**Funding:** This study was conducted with the support of Samsung Electronics (IO200423-07296-01). In addition, this work was partially supported by the Korea Institute for Advancement of Technology (KIAT) grant funded by the Korea Government (MOTIE, Korea) (Advanced Training Program for Smart Factory, No. N0002429) and the BK21 FOUR (Fostering Outstanding Universities for Research) funded by the Ministry of Education (MOE, Korea) and National Research Foundation of Korea (NRF).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

- Zhou, Q.; Zhou, B.H. An impending deadlock-free scheduling method in the case of unified automated material handling systems in 300 mm wafer fabrications. *J. Intell. Manuf.* **2018**, *29*, 155–164. [[CrossRef](#)]
- Liao, D.Y.; Fu, H.S. Speedy delivery-dynamic OHT allocation and dispatching in large-scale, 300-mm AMHS management. *IEEE Robot. Autom. Mag.* **2004**, *11*, 22–32.
- Kim, B.I.; Shin, J.; Jeong, S.; Koo, J. Effective overhead hoist transport dispatching based on the Hungarian algorithm for a large semiconductor FAB. *Int. J. Prod. Res.* **2009**, *47*, 2823–2834. [[CrossRef](#)]
- Sakr, A.H.; Aboelhassan, A.; Yacout, S.; Bassetto, S. Simulation and deep reinforcement learning for adaptive dispatching in semiconductor manufacturing systems. *J. Intell. Manuf.* **2021**, 1–14. [[CrossRef](#)]
- Bartlett, K.; Lee, J.; Ahmed, S.; Nemhauser, G.; Sokol, J.; Na, B. Congestion-aware dynamic routing in automated material handling systems. *Comput. Ind. Eng.* **2014**, *70*, 176–182. [[CrossRef](#)]
- Nakamura, R.; Sawada, K.; Shin, S.; Kumagai, K.; Yoneda, H. Model reformulation for conflict-free routing problems using Petri net and deterministic finite automaton. *Artif. Life Robot.* **2015**, *20*, 262–269. [[CrossRef](#)]
- Hwang, I.; Jang, Y.J. Q( $\lambda$ ) learning-based dynamic route guidance algorithm for overhead hoist transport systems in semiconductor fabs. *Int. J. Prod. Res.* **2020**, *58*, 1199–1221. [[CrossRef](#)]
- Ahn, K.; Lee, K.; Yeon, J.; Park, J. Congestion-aware dynamic routing for an overhead hoist transporter system using a graph convolutional gated recurrent unit. *IIEE Trans.* **2022**, *54*, 803–816.
- Kim, H.J.; Lee, J.H.; Baik, S.; Lee, T.E. Scheduling in-line multiple cluster tools. *IEEE Trans. Semicond. Manuf.* **2015**, *28*, 171–179.
- Wan, J.; Shin, H. Predictive vehicle dispatching method for overhead hoist transport systems in semiconductor fabs. *Int. J. Prod. Res.* **2022**, *60*, 3063–3077. [[CrossRef](#)]
- Lingitz, L.; Gallina, V.; Ansari, F.; Gyulai, D.; Pfeiffer, A.; Sihm, W.; Monostori, L. Lead time prediction using machine learning algorithms: A case study by a semiconductor manufacturer. *Procedia CIRP* **2018**, *72*, 1051–1056. [[CrossRef](#)]
- Wang, J.; Yang, J.; Zhang, J.; Wang, X.; Zhang, W. Big data driven cycle time parallel prediction for production planning in wafer manufacturing. *Enterp. Inf. Syst.* **2018**, *12*, 714–732. [[CrossRef](#)]
- Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-assisted Intervention, Munich, Germany, 5–9 October 2015.
- Clevert, D.A.; Unterthiner, T.; Hochreiter, S. Fast and accurate deep network learning by exponential linear units (elus). *arXiv* **2015**, arXiv:1511.07289.
- Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 June 2015.
- Ba, J.L.; Kiros, J.R.; Hinton, G.E. Layer normalization. *arXiv* **2016**, arXiv:1607.06450.



17. Ulyanov, D.; Vedaldi, A.; Lempitsky, V. Instance normalization: The missing ingredient for fast stylization. *arXiv* **2016**, arXiv:1607.08022.
18. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
19. Wu, Y.; He, K. Group normalization. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018.