Information Modelling and Knowledge Bases XXXII
M. Tropmann-Frick et al. (Eds.)

doi:10.3233/FAIA200828

Defects Recognition on Wafer Maps Using Multilayer Feed-Forward Neural Network

Radoslav ŠTRBA[a,1] and Daniela BORDENCEA[a]
[a] *ON Semiconductor, 1. máje 2230, 756 61 Rožnov pod Radhoštěm, Czech Republic*

Abstract. Wafer-defect maps can provide important information about manufacturing defects. The information can help to identify bottlenecks in the semiconductor manufacturing process. The main goal is to recognize random versus patterned defects. A patterned defect shows that a step in the process is not performed correctly. If same defect occurs multiple times, then the yield can rapidly decrease. This article proposes a method for yield improvement and defect recognition by using a feed-forward neural network. The neural network classifies wafer-defect maps into classes. Each class represents certain defect on the map. The neural network was trained, tested and validated using a wafer-defect maps dataset containing real defects inspired from manufacturing process.

Keywords: wafer, semiconductor, yield, defects, neural network

1. Introduction

As semiconductor devices are being incorporated into more and more devices, the need of producing high-performance cost-effectively integrated circuits' (IC) increases. The semiconductor IC fabrication process is complex, containing various processing steps, among the most important being crystal growth, photolithography, etching, thermal oxidation, implant, deposition. The sand used for wafers growth has to be very clean. It is heated above its melting point and then a pure silicon seed crystal is placed into the molten sand bath. While being rotated, the seed is pulled out resulting an ingot. The ingot is sliced into very thin wafers. Wafers are polished till are very smooth and then they are going through other steps where new layers of material are added. During the flow, the wafers can be affected by adhesion of dust or particles, cracks, scratches, contaminants, process variations, operators or equipment errors that can cause defects.

Probing has an important role in the management of manufacturing process being also used for finding defects. Rapid defect learning and reduction is important in order to quickly find the processes where the failure appears. Defect reduction can be achieved by:

- defects detection
- defect classification
- source defects identification
- process correction in order to eliminate or reduce the defects
- monitoring the process for yield excursions.

[1] Corresponding Author

Over the last period, the semiconductor companies are keenly seeking ways for elimination of the defects in order to improve the yield. Defect pattern recognition, clustering and classification is one of the well-investigated ways. Clustering defects refers to grouping the defects with closer relationship. Clustering may indicate an external surface damage, such as a scratch or also the wrong manipulation with the wafer.

Many studies were done by the researchers in this area. Young-Seon Jeong et al., in their paper [1], were showing a new methodology for classifying the defect patterns and also detecting the special autocorrelations using spatial correlogram. Their method was robust despite of the location and size of the defect.

Late in 2017, Kouta Nakata et al. [2] are presenting a way for monitoring the failure pattern, identifying the cause and also monitoring the failure recurrence by using machine learning and data mining algorithms. They are using:

- K-Means++ for pattern monitoring [3]
- FPGrowth for identification [4]
- standard supervised learning approach for recurrence.

The recent paper of Takeshi Nakazawa [5] is focused on defect pattern classification and image retrieval using Convolutional Neural Networks (CNN). The image retrieval is using a binary code generated for each wafer map.

Another paper focused on same subject is [6]. It describes the classification of missed-type defect patterns using CNN. Using both simulated and also real examples, CNN performed significantly better even if mixed pattern defects are on a wafer with random defects. Moreover, the paper presents a detailed way of generating WBM data that mimics a real WBM dataset.

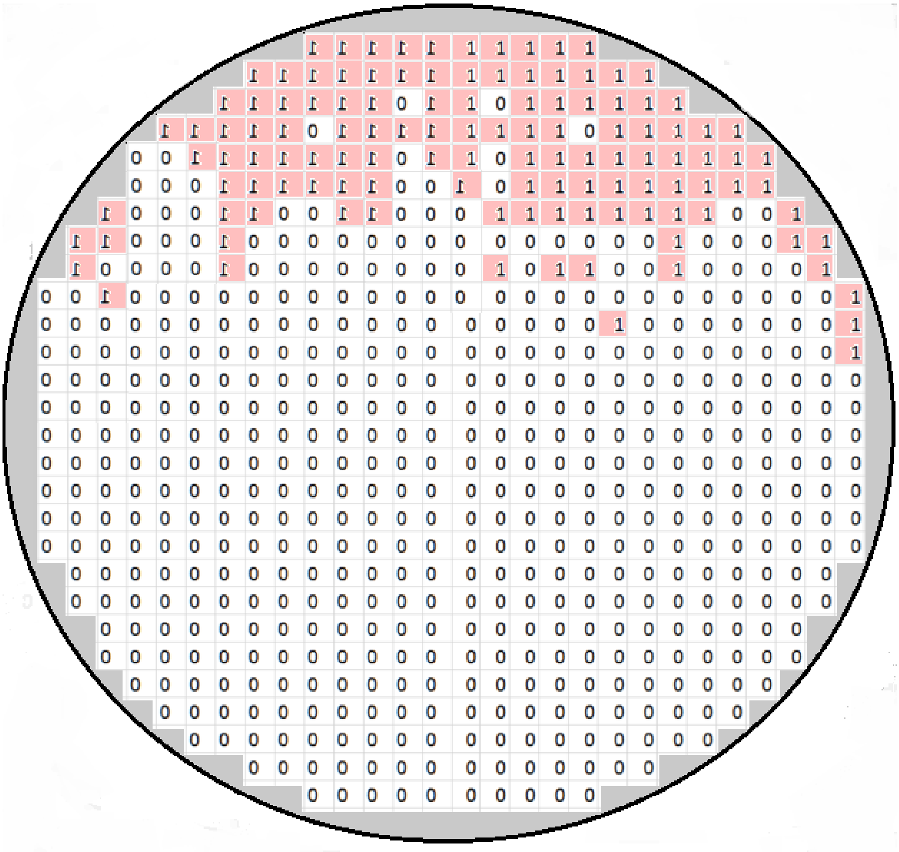
Single and mixed defects patterns using deep machine learning based approach were studied by [7]. At first, the random noise is filtered out with a spatial filter followed by the defect separation into single and mixed patterns. The single patterns are identified using randomized general regression network and the missed patterns are identified using deep structural convolutional network.

Clustering the defective patterns on wafers was also presented in [8]. The method used was based on the defect spatial dependence across all the wafers. The system was called DDPfinder and it uses the dominant defect as the base for clustering the patterns.

1.1 Defect Patterns and Recognition

The result of probing process is showed in wafer maps. Probing means testing of potential future chips while they are still part of the wafer. In Figure 1 can be seen the wafer map where the potential future chips represented as squares (also known as dice). A die can be pass (good die) or fail (bad die). In the figure, to a good die is associated the binary value 0 and to a bad die is associated the binary value 1.

The detailed problem is described in Chapter 2. The main goal of the paper is to automatically recognize patterns in wafer maps after probing. For better understanding of defects and patters were created few examples. The first example is shown in Figure 1 and illustrates a wafer map. As can be seen, the bad dice are located and clustered in the top part of the wafer. It can be, for example, the incorrect manipulation with the wafer or the defect of the prober (probe tool).



									1	1	1	1	1	1	1	1	1	1									
							1	1	1	1	1	1	1	1	1	1	1	1	1	1	1							
						1	1	1	1	1	1	0	1	1	0	1	1	1	1	1	1						
				1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1				
			0	0	1	1	1	1	1	1	1	0	1	1	0	1	1	1	1	1	1	1	1	1			
			0	0	0	1	1	1	1	1	1	0	0	1	0	1	1	1	1	1	1	1	1	1			
		1	0	0	0	1	1	0	0	1	1	0	0	0	1	1	1	1	1	1	1	1	0	0	1		
	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	
	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	1	1	0	0	1	0	0	0	0	1	
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					
							0	0	0	0	0	0	0	0	0	0	0	0	0	0	0							
									0	0	0	0	0	0	0	0	0	0									

Figure 1. Example of the wafer map

Defect types:

- **Random defects** (Figure 2) – bad dice are located in random positions. If no pattern is visible then wafer map is not so helpful regarding bottlenecks identification in manufacturing process.

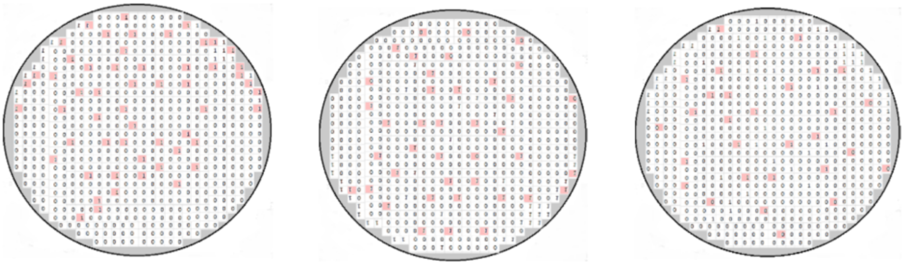


Figure 2. Example of random defects

- **Patterns** in following Figure 3

  a) RING_CENTER – bad dice are located around the center of the wafer.
  b) RING_CENTER_FILLED – bad dice are located in the center.
  c) TOP_CLUSTER – bad dice are clustered in the top half-part.
  d) RIGHT_TOP_EDGE – bad dice are located in the right top edge.
  e) SCRATCH – bad dice are showed as a line (straight or skewed).

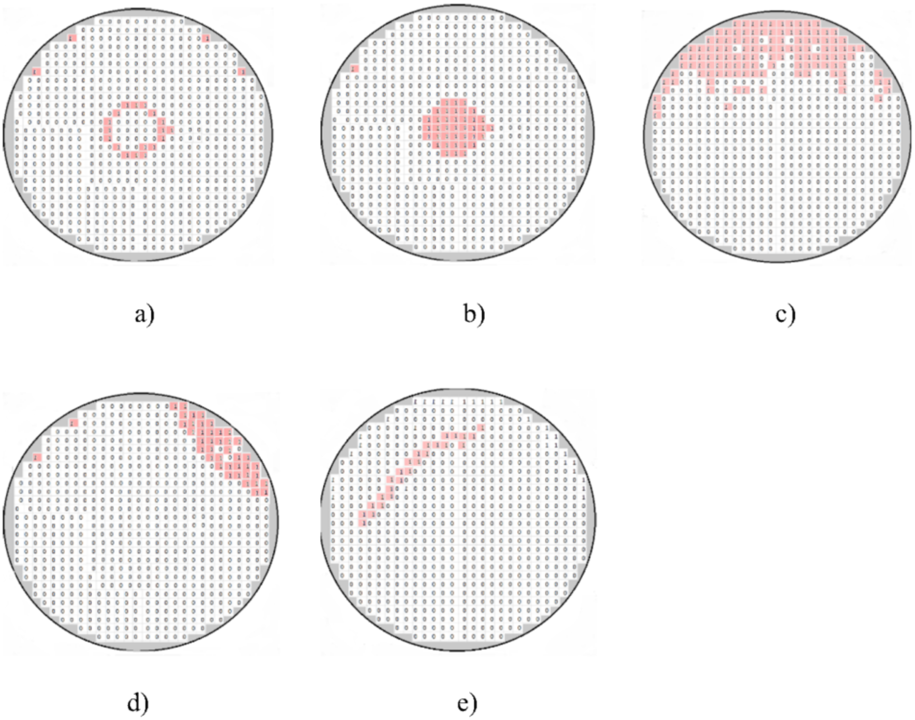


Figure 3. Patterns a) Ring Center; b) Ring Center Filled; c) Top Cluster; d) Right top edge; e) Scratch

This paper is focusing on providing a method for improving the yield by finding the defect die using a feed-forward neural network. The article is structured as follows. In the first chapter, a brief overview of the manufacturing process flow is given, followed by the state of the art. The chapter continues with the description of defects and defects patterns. The second chapter describes the problem. The solution proposed and the experiments are presented in chapter 3, respectively Chapter 4. The paper ends with Chapter 5 - Conclusions.

2. Problem description

The main problem during manufacturing process relates to high number of bad dice on the wafer that cause low yield. Let's say, on a wafer are 30 dice. If 3 of them are failed during the tests and 27 are passed, then it results in a 90% yield. Therefore, only 90% of the chips will be ready for shipping. This is called die yield, and it decreasing based on the number of total defective dice.

As already mentioned at the beginning of this chapter, the number of bad dice decrease yield. The paper main goal is to present a method for increasing the yield. Based on various reasons (for example: human error, machine error, combination of both), in most of the cases the yield will not reach 100%. The die yield loss reasons can be random and non-random.

When a certain type of scratch appears regularly in a similar place on the wafer, it shows that it is probably caused by a non-random event. The scratches could be due to machine error and the defects on the wafer can caused by the operator. A machine can get damaged and scratch the wafers but also a person can manipulate the wafers incorrectly. The operators are using tools and equipment that can also damage the chips.

The yield, respectively productivity can be increased by identifying the bottlenecks. In manufacturing, from time to time you can meet a slow process. This is so called bottleneck and it refers to a process that accumulates the longest queue. A bottleneck can be automatically identified by:

- using mathematical computations (analytically)
- running a digital model for a period of time
- analyzing the process data.

By analyzing the data can be found diverse defect patterns on the wafer that can reveal important information about the abnormalities. One of the way of recognizing the defect patterns is by using machine learning techniques, such as:

- clustering algorithms using Bayesian interface
- using correlograms
- multi class support vector machine (SVM)
- CNN
- fuzzy rules.

3. Proposed method

The method proposed in this article is focused on identification of process bottlenecks using pattern recognition. The method's steps are:

1. data preprocessing and transformation
2. configuration and training of a neural network
3. classification of new items (wafer defect maps)
4. identification of bottlenecks in a process.

3.1. Process Description

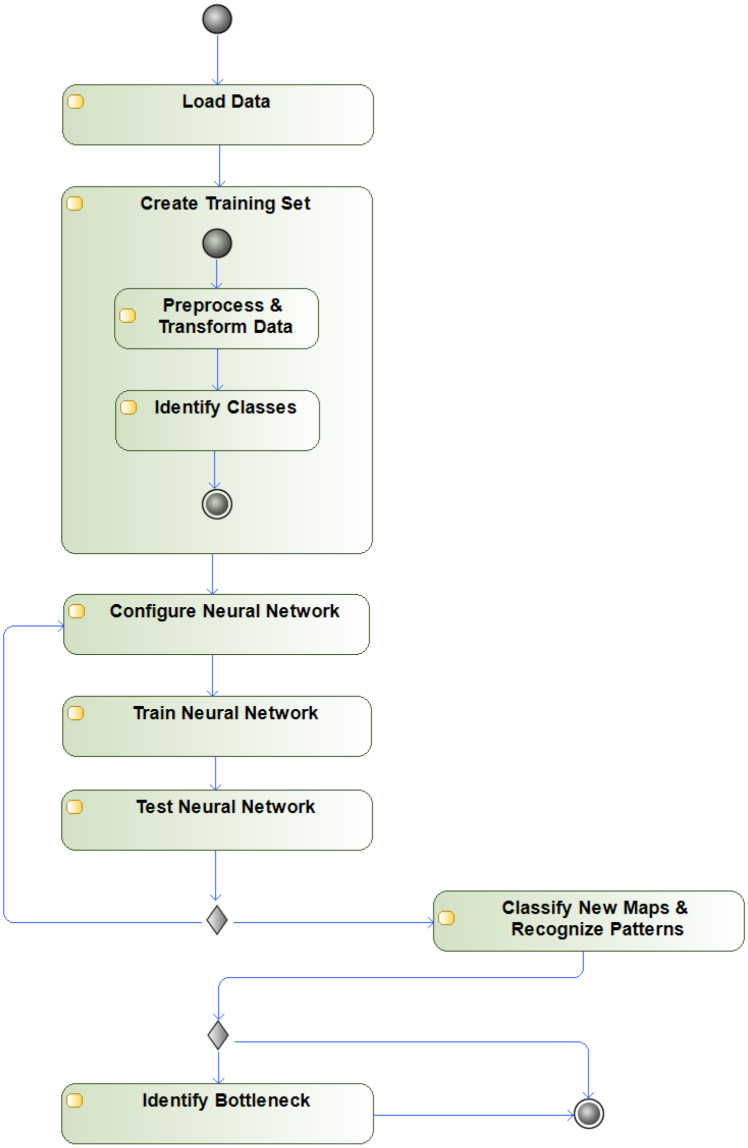


Figure 4. Activity Diagram.

3.2. Load Data, Preprocessing and Transformation

The classification accuracy mainly depends on the data quality. Low-quality data may lead to the training of over-fitting or not accurate classifiers. Thus, data preprocessing techniques are essential for data mining. Good preprocessing can improve the quality of the data, thereby helping to increase the accuracy and efficiency of the classifier. A series of data preprocessing techniques can be used, for example data cleaning or reduction. Cleaning means removal of the noisy data which can make the classifier mistaken. Reduction means subtraction of the data-size by aggregating and eliminating of the redundant or very similar (correlated) features. [9]

Each type of data requires different approach for preprocessing. Probe data from semiconductor manufacturing are stored in files generated by probers (probe tools) or retrieved from SQL databases. Therefore, all the available data from different sources should be considered for preprocessing. In this paper, preprocessing of data described is done by filtering the data, cleaning data and also performing transformations on those data (Figure 5).

The basic problem with creation of the training dataset is unequal sample-size. Therefore, the appropriate dataset for training of the classifier contains equal number of training data for each class. The dataset presented in this paper was generated from observation and replicating some real defects and patterns. This was done for having high-quality data and avoiding the imbalanced sample-sizes.

The transformation steps described below are shown to understand how the defect wafer maps represented as matrixes can be transformed to vectors. The top part Figure 5 is actually the top part of the wafer (first 3 rows) represented in the Figure 1 - Example of the wafer map. The maps are available for each wafer as binary matrixes, however the classifier presented in this article requires vectors instead of matrixes.

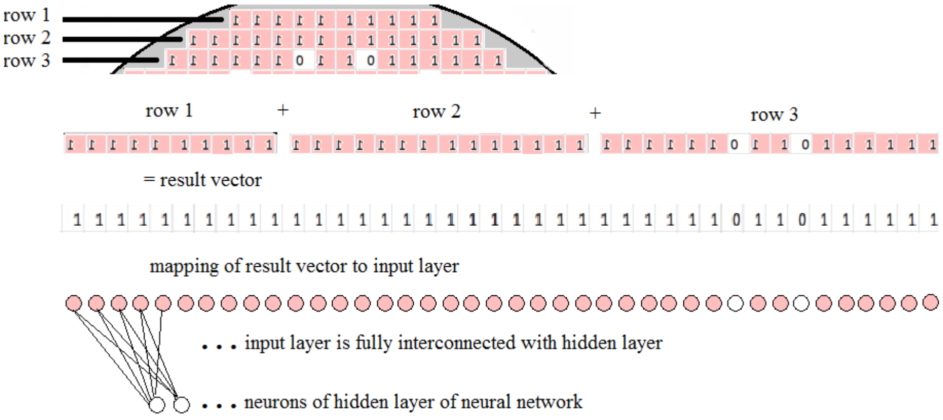

row 1
row 2
row 3
row 1
+
row 2
+
row 3
= result vector
mapping of result vector to input layer
. . . input layer is fully interconnected with hidden layer
. . . neurons of hidden layer of neural network

Figure 5. Example of the transformation step from a wafer map to a binary vector

Figure 5 shows the transformation steps from wafer map matrix to vector by joining all rows:

row 1 + row 2 + row 3 + ... = result vector.

The result vector is interpreted as an array of binary values (the term array comes from programming languages). Each cell of the vector (array) contains 0 or 1 value. The number of input vector cells equals to the number of input neurons. All the neurons of the input layer (input neurons) are fully interconnected with all neurons of the first hidden layer of the neural network. Always when new item is provided to a neural network, it is provided to the input layer of this network.

3.3. The classification using Multilayer Feed-forward Neural Network

The techniques for classification of the wafer maps can be also used as supportive techniques for defect recognition. The machine-learning methods help to classify the wafer maps and recognize the patterns, which are useful for identification of the bottlenecks in the manufacturing process. The classification is a process that is closely related to the pattern recognition. The neural network trained for classification is designed to take input samples and classify them into groups (classes) [10, 11]. The set of the features given is represented by the cells of the binary values in the vector. The decision to be made is to which class number the value belongs. The given features can be filled into an input vector x, also known as result vector in Figure 5. Being given a set of pattern samples, where each consists of a vector of the attribute values and the corresponding class label, the classifier will be trained to group the wafer maps. [12]

The Feed-forward is a common architecture for data classification. The Feed-forward with more hidden layers is commonly used with the supervised training algorithm called Back-Propagation. “Back-Propagation training of the neural network searches for a set of weights and biases that most accurately predicts value from input samples. Once we have these weight and bias values, we could apply them to an upcoming dataset of items to classify them.” [13, 14]

The supervised training is deducted from the training dataset. [10] The labeled data transformed into vectors is necessary for training the neural network.

“The training of the neural network is the process of finding a set of weight and bias values. For a given set of inputs, the outputs produced by the neural network are very close to some target values.” [13]

3.4. Design and Configuration

Nowadays, the deep learning is a popular term in the research community. Today, many neural networks are still based upon manual design and configuration due to practical and performance reasons. A person having knowledge about the specific application specifies a network architecture, configuration and activation dynamics. This is perhaps not surprising, given the fact that the general space of possible neural networks is large and complex. The automatically searching for an optimal network architecture may be computationally intractable or impractical for complex applications [15, 16]. Feed-forward network begins with input layer. The input layer is connected to hidden layers, which are connected to the output layer.

In this case the neural network is used to solve a classification problem. Items are classified into separate groups. Each output neuron represent specific pattern. [10] More information about the selection of the optimal neural network architecture and activation functions has been published in the paper named “Finding an Optimal Configuration of the Feed-forward Neural Network”. [17]

The four-layered feed-forward network with hyperbolic-tangent activation function in hidden layer and soft-max activation function in the output layer neurons is trained using the scaled conjugate gradient back-propagation training algorithm. The training of the neural network automatically stops when generalization stops improving, as indicated by an increase in the cross-entropy error of the validation items. The example of designed neural network is showed in Figure 6.

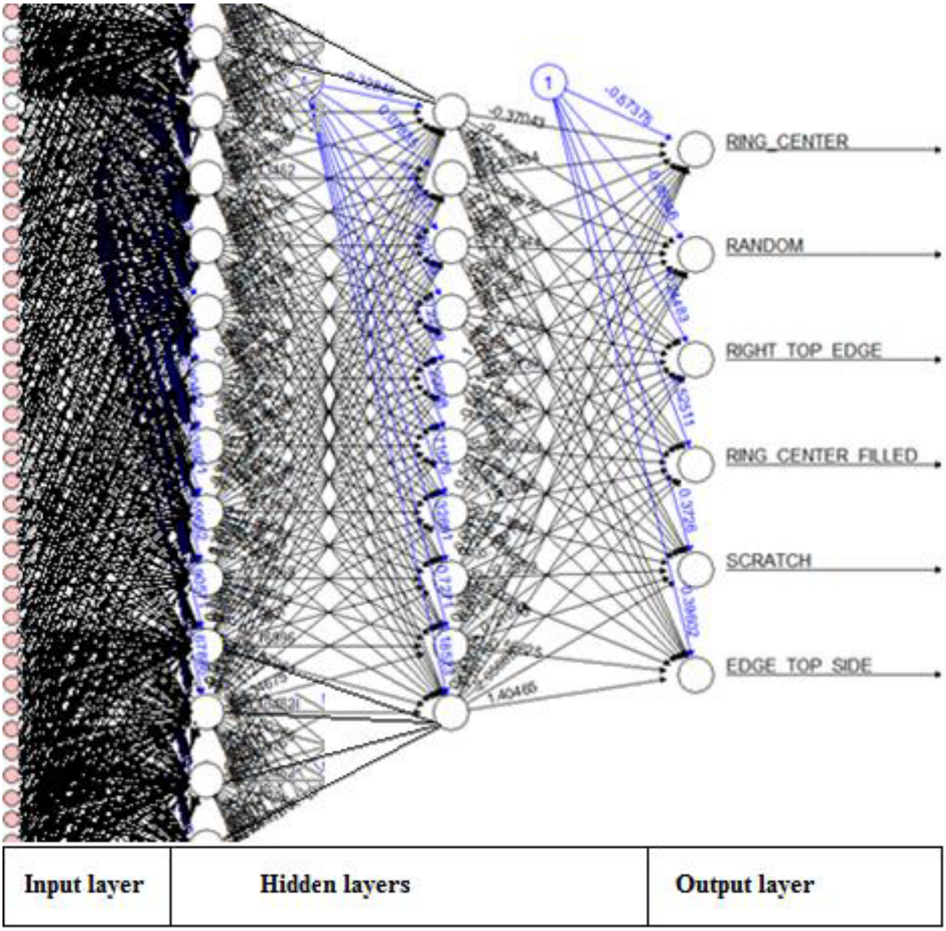


Figure 6. The designed Feed-forward multilayer neural network

The Table 1 below shows more detailed description of four layers included in the designed neural network from Figure 6.

Table 1. Description of layers.

Layer	Number of Neurons	Activation Function
Input	- dependent on wafer size -	
Hidden 1	35	Hyperbolic tangent
Hidden 2	10	Hyperbolic tangent
Output	6	Softmax

The hyperbolic tangent activation function (tanh) is used for neurons of the hidden layers. This activation function has been chosen for hidden layer because it returns both positive and negative values. When graphed, the hyperbolic tangent function looks quite similar to the log-sigmoid function.

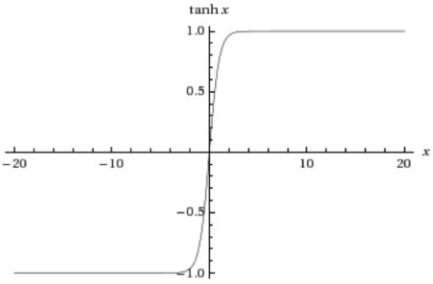


Figure 7. Hyperbolic tangent activation function.

The important difference between log-sigmoid and tanh is that tanh function returns a value between -1 and +1 instead of a value between 0 and 1. The algebraic expression of the hyperbolic tangent activation function from the family of hyperbolic functions is illustrated bellow (1).

$$\tanh x = \frac{\sinh \mathrm{x}}{\cosh x} = \frac{\mathrm{e}^{\mathrm{x}} - \mathrm{e}^{-\mathrm{x}}}{\mathrm{e}^{\mathrm{x}} + \mathrm{e}^{-\mathrm{x}}} = \frac{\mathrm{e}^{2\mathrm{x}} - 1}{\mathrm{e}^{2\mathrm{x}} + 1} = \frac{1 - \mathrm{e}^{-2\mathrm{x}}}{1 + \mathrm{e}^{-2\mathrm{x}}} \tag{1}$$

The softmax activation function is popular as activation function for the output neurons of the feed-forward neural network. A softmax converts an arbitrary real-valued vector into a multinomial probability vector. It is used in the classification problems. This is the version of winner-take-all nonlinearity, in which maximum output is transformed to 1.0. The winner-neuron is activated and exactly the one neuron always wins.

$$h(z) = \frac{e^{z}}{\sum_{i=1}^{n} e^{z_i}} \tag{2}$$

4. Experiment

A neural network was designed and trained for classification of the wafer maps. Back-propagation as a supervised training has been used for the training of the neural network. [18], [19], [20] The experiment was performed in R environment by using R studio and MxNet library.

4.1. Dataset Description

The training dataset contains the row data (transformed wafer-defect maps) represented by binary parameters. The goal is to classify the items into 6 categories (patterns). These patterns have been created based on real wafer-defects observed during manufacturing process. The defects are divided into six groups. One group contains random defects, and next five groups contains patters, e.g. ring-center, ring-center filled, right top edge, edge top side, scratch. A whole dataset consists of 862 items. The dataset has been divided into two subsets. The first subset includes 518 items. It is used for computing the gradient and updating the network weights and biases. The second subset includes 344 test items. It is used for measurement of the neural network accuracy.

4.2. Training, Validation and Results

The four-layered feed-forward network with hyperbolic tangent hidden and soft-max output neurons is trained using scaled conjugate gradient back-propagation training algorithm with a learning rate of 0.02. The training of neural network automatically stops when generalization stops improving. It is indicated by an increase in the cross-entropy error of the validation items. The goal is to avoid the over-fitting of the classifier. Therefore, the neural network should be trained as accurate as possible but not over-trained.

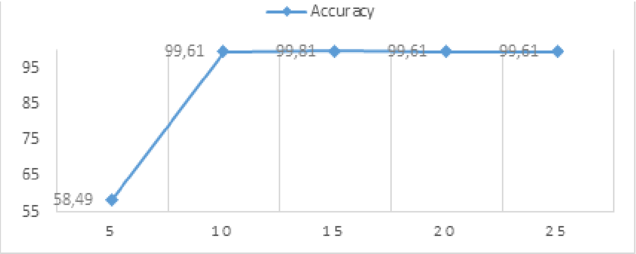
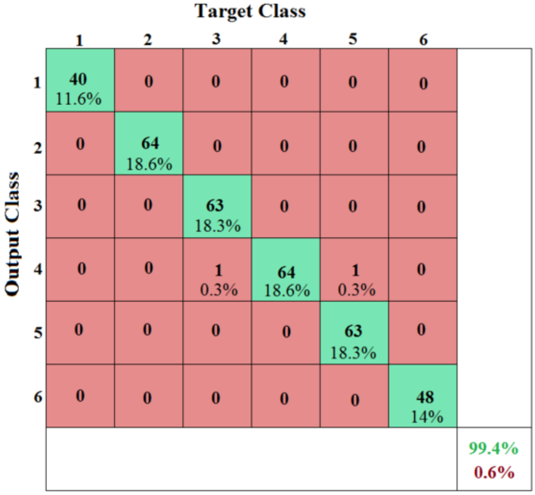


Figure 8. Part of the Training State Plot – Validation Checks

The plot with the validation performance (Figure 8) shows the results of the validation error during the training process. The X-axis represents the trainings rounds and the Y-axis represents the training accuracy or error value. The best results of the validation are indicated at epoch 15. The best validation performance at this epoch is when error value is 0.19 (validation accuracy is 99.81%).



Output Class \ Target Class	1	2	3	4	5	6	
1	40 11.6%	0	0	0	0	0	
2	0	64 18.6%	0	0	0	0	
3	0	0	63 18.3%	0	0	0	
4	0	0	1 0.3%	64 18.6%	1 0.3%	0	
5	0	0	0	0	63 18.3%	0	
6	0	0	0	0	0	48 14%	
							99.4% 0.6%

Figure 9. Test Confusion Matrix.

The results of the classification testing accuracy are visualized using confusion matrix. The confusion matrix (Figure 9) shows the items number and its percentage classified correctly or misclassified to each class. Overall accuracy of designed classifier (feed-forward neural network) is 99.4% using independent testing dataset.

5. Conclusion

This paper shows a way of improving the yield using the machine-learning supportive technique called classification. The wafer-defect maps are classified, using a multilayer feed-forward neural network, into six groups: random defects, ring-center pattern, ring-center filled pattern, right top edge pattern, edge top side pattern, scratch pattern. One group is non-pattern and five are pattern groups. The network is trained by back-propagation training algorithm.

The training dataset was composed by 862 items: 518 items are used for computing the gradient and updating the network weights and biases are 344 items are used for measuring the neural network accuracy. A four-layered feed-forward network with hyperbolic tangent activation function on hidden layer and with soft-max activation function on output layer was trained. The training algorithm used is scaled conjugate gradient back-propagation. It is shown that the lowest validation error value is 0.19 in the epoch 15 when the neural network stopped improving. Using the confusion matrix (Figure 8), it is shown that 342 items with percentage 99.4% were classified correctly to given groups and 2 items with percentage 0.6% were misclassified.

The overall accuracy of proposed method for improving of yield which is using custom designed classifier (feed-forward neural network) is 99.4%. The accuracy was measured using an independent testing dataset.

References

[1] Young-Seon Jeong , Seong-Jun Kim , and Myong K. Jeong, Automatic Identification of Defect Patterns in Semiconductor Wafer Maps Using Spatial Correlogram and Dynamic Time Warping, IEEE Transactions on Semiconductor Manufacturing vol 21, no. 4, 625 – 637, December 2008

[2] Kouta Nakata, Ryohei Orihara, Yoshiaki Mizuoka, and Kentaro Takagi, A Comprehensive Big-Data-Based Monitoring System for Yield Enhancement in Semiconductor Manufacturing, IEEE Transactions on Semiconductor Manufacturing, vol. 30, no. 4, November 2017

[3] B. Bahmani, B. Moseley, A. Vattani, R. Kumar, and S. Vassilvitskii, Scalable k-means++, Proc. VLDB Endowment, vol. 5, no. 7, pp. 622–633, 2012

[4] C. Borgelt, An implementation of the FP-growth algorithm, in Proc. 1st Int. Workshop Open Source Data Min. Frequent Pattern Min. Implement., Chicago, IL, USA, 2005, pp. 1–5

[5] Takeshi Nakazawa and Deepak V. Kulkarni, Wafer Map Defect Pattern Classification and Image Retrieval Using Convolutional Neural Network, IEEE Transactions on Semiconductor Manufacturing, vol. 31, no. 2, May 2018

[6] Kiryong Kyeong and Heeyoung Kim, Classification of Mixed-Type Defect Patterns in Wafer Bin Maps Using Convolutional Neural Networks, IEEE Transactions on Semiconductor Manufacturing, vol. 31, no. 3, August 2018

[7] Ghalia Tello, Omar Y. Al-Jarrah, Paul D. Yoo, Yousof Al-Hammadi, Sami Muhaidat, and Uihyoung Lee, Deep-Structured Machine Learning Model for the Recognition of Mixed-Defect Patterns in Semiconductor Fabrication Processes, IEEE Transactions on Semiconductor Manufacturing, vol. 31, No. 2, May 2018

[8] Kamal Taha, Khaled Salah, and Paul D. Yoo, Clustering the Dominant Defective Patterns in Semiconductor Wafer Maps, IEEE Transactions on Semiconductor Manufacturing, vol. 31, issue 1, February 2018, pp. 156-165

[9] D. M. Farid, L. Zhang, C. M. Rahman, M. a. Hossain, and R. Strachan, Hybrid decision tree and naïve Bayes classifiers for multi-class classification tasks, Expert Syst. Appl. (2014), vol. 41, no. 4 PART 2, 1937–1946.

[10] J. Heaton, Introduction to Neural Networks for Java, 2nd ed. Heaton Research, Inc., 2008. ISBN 1604390085.

[11] G. P. Zhang, "Neural Networks in Business Forecasting," Rev. Econ. Sci. (2004), vol. 6, pp. 161–176. Doi:10.4018/978-1-59140-176-6

[12] A. Holst, The Use of a Bayesian Neural Network Model for Classification Tasks. 1997. 110p. Dissertation Thesis at Department of Numerical Analysis and Computing Science, Stockholm Universidad.

[13] J. Mccaffrey, Neural Network Back-Propagation Using C#. NEURAL NETWORK LAB [online]. (2013) [vid. 2015-12-02]. Available from: https://visualstudiomagazine.com/articles/2013/08/01/neural-network-back-propagation-using-c.aspx

[14] A. Azadeh, M. Saberi, and M. Anvari, An integrated artificial neural network algorithm for performance assessment and optimization of decision making units, Expert Syst. Appl. (2010), vol. 37, no. 8, 5688–5697. ISSN 09574174. doi:10.1016/j.eswa.2010.02.041

[15] S. Das. Elements of Artificial Neural Networks. IEEE Transactions on Neural Networks. (1998), Vol 9(1), 234–235.

[16] J. Jung, J. Reggia, The Automated Design of Artificial Neural Networks Using Evolutionary Computation. Success in Evolutionary Computation. (2008), vol 41(2), 19–41. Doi: 10.1007/978-3-540-76286-7_2

[17] R. Štrba, S. Štolfa, J. Štolfa, Finding an Optimal Configuration of the Feed-forward Neural Network. Information Modelling and Knowledge Bases XXVII, Frontiers in Artificial Intelligence and Applications. 25th International Conference on Information Modelling and Knowledge Bases (EJC) (2015), Vol 280, 199 – 206. doi: 10.3233/978-1-61499-611-8-199

[18] Zhang, G.P.: Neural Networks in Business Forecasting. Rev. Econ. Sci. 6, 161–176 (2004).

[19] Ghiassi, M., Nangoy, S.: A dynamic artificial neural network model for forecasting nonlinear processes. Comput. Ind. Eng. 57, 287–297 (2009).

[20] Sharma, P., Kaur, M.: Classification in Pattern Recognition: A Review. Int. J. Adv. Res. Comput. Sci. Softw. Eng. 3, 298–306 (2013).