Research Article

Classification of Silicon (Si) Wafer Material Defects in Semiconductor Choosers using a Deep Learning ShuffleNet-v2-CNN Model

Rajesh Doss,1 Jayabrabu Ramakrishnan,2 S. Kavitha,3 S. Ramkumar,4 G. Charlyn Pushpa Latha,5 and Kiran Ramaswamy6

1Department of Computer Science, National Defence Academy, Khadakwasla, Pune, Maharashtra, India
2Department of Information Technology and Security, College of Computer Science and Information Technology, Jazan University, Jizan, Saudi Arabia
3Department of Mechanical Engineering, Kalasalingam Academy of Research and Education (Deemed to be University), Sriviliputhur, Tamil Nadu, India
4Department of Business Analytics, School of Business and Management, Christ (Deemed to be University), Bengaluru, Karnataka, India
5Department of Information Technology, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences (Deemed to be University), Chennai, Tamil Nadu, India
6Electrical and Computer Engineering, Dambi Dollo University, Dambi Dollo, Ethiopia

Correspondence should be addressed to Kiran Ramaswamy; kiran.ramaswamy@ambou.edu.et

Received 9 June 2022; Revised 10 July 2022; Accepted 26 July 2022; Published 15 September 2022

1. Introduction

A semiconductor is a combination of a conductor and an insulator. Semiconductors are frequently utilized in the production of electronic chips, computer components, and other devices. In most cases, silicon, germanium, and other elemental purities are utilized in its production. However, the production of semiconductors is a process that is notably difficult, expensive, and time-consuming. It involves a variety of chemical, electrical, and mechanical processes, including deposition, photolithography, etching, ion implantation, chemical planarization, and diffusions. After all of these procedures have been applied, integrated circuits (ICs) are created by building circuit structures on different
layers of the same wafer and then connecting the structures together using wires [1]. In order to manufacture ICs of a high quality, the surface of a wafer needs to be clean, and every layer of the circuits need to be aligned perfectly. However, even highly skilled semiconductor engineers who operate highly automatic and accurate machinery in a highly sterile environments are unable to manufacture wafer dies that are free of errors. Following the completion of the fabrication, every wafer is passed over the circuit probe test. During this test, defect-free and defective wafer dies are distinguished from one another. The outputs of the tests are depicted on a wafer map, which is a two-dimensional (2D) representation of the wafers. In VLSI technology, the process of wafer manufacture includes defect detection as an important part of the process. It makes defect identification and classification possible throughout the process, which in turn increases the output of the fabrication. Every single defect that is found is treated as an indication that there is a problem with the process. This indicates that the defect cannot be fixed on the wafer where it occurred. To prevent these kinds of defects, the process engineers should make the necessary adjustments to the process [2]. The periodic table of elements used for manufacturing semiconductor is represented in Figure 1.

As shown in Figure 1, the materials that are more commonly used as semiconductors are indicated in blue. A semiconductor may be composed of a single element, like germanium (Ge) or silicon (Si), or it may be composed of a compound, like gallium arsenide (GaAs), indium phosphide (InP), or cadmium telluride (CdTe).

Process engineers with years of experience identified the wafer defect patterns and assign unique names to each one, such as “Center,” “Donut,” “Local,” “Edge-Loc,” “Edge-Ring,” “Scratch,” “Random,” “Near-full,” and “None,” as illustrated in Figure 2. In addition, the circuit’s rising integration density and the complexity of wafer designs both contribute to the rise in the frequency with which these defects occur. The precise unusual behaviour of a particular fabrication process is the cause of each and every wafer defect. For instance, center defects can develop as a result of problems with uniformity in mechanical and chemical planarization, edge-loc defect can take place as a result of thin film depositions, and edge-ring defect takes place as a result of etching issues. Therefore, wafer maps defect analysis gives significant information that may be used to uncover abnormal processes that occur during semiconductor manufacture and to take steps to rectify these issues. A precise classification of wafer map structures plays a significant role in the detection of wafer defects, which would ultimately improve the production and quality of semiconductors by leading to a more efficient wafer fabrication process [3].

After the defect has been located, it must be classified to make the necessary adjustments to the fabrication procedure. The qualities of the defect, such as its precise patterns, geometries, and other characteristics, are used to determine its classification. Certain defects are manifestations of broader problems, such as wafer scratches. In this instance, a large defect is formed by a collection of smaller faults that are found in specific locations. In order to assure precise defect reporting, pre-classification grouping is an absolutely necessary step. Macro-flaws are defects that manifest themselves on the wafer as a collection of smaller, localized defects all at once. When compared to the local defects that are predicated on a single occurrence, these global defects have bigger sizes (magnitudes of difference) than the local defects [5].

Wafer defect scanners are available today, created by top-line manufacturers that specialize in producing high-end products. These kinds of scanners are multi-purpose equipment that are quite accurate (electro-optical-computerized). Wafer defects, which can be found in some of the today’s most advanced technology, are found using a variety of microscopy and lighting approaches (as small as 10 nM). Comparison of pixels on neighboring dyes or cells forms the basis of the detection process. These high-resolution scanners are very expensive, and because of the high
resolution they use, the scanning process of an entire wafer takes an inexcusably long amount of time [6].

Over the last few years, new varieties of defect detecting scanners have emerged. These scanners are developed for macro-defects. They were developed specifically to deal with macro, or large-scale, problems. In most cases, the scanner is only capable of scanning the wafer in a single field of view (FOV). They are easier to operate, less complicated (in comparison to high-end scanners), less expensive, and have a quicker operation cycle than comparable alternatives. They are able to identify major defects, but not local ones of a smaller scale.

Macro-defect scanning serves its own purpose and has its own set of benefits. Every wafer can be inspected since the scanning happens so quickly. In most cases, the process is less complicated, and simple. Due to the fact that the fundamental structure consists of nothing more than a microscope, a camera, and some sort of handling mechanism. Defects in macro-wafers also require automatic defect identification and classification in order to be fixed. In this scenario, conventional methods of detection can be utilized. The deep learning technology is one of the advanced techniques that can be used for defect classification and also be used for the detection process itself. This is because, as opposed to high-resolution scanning, the deep learning classifier can be trained to detect macro-flaws, based on their visual signature, when the whole field of view is available. This is a significant advantage over high-resolution scanning. Deep learning makes the entire process more efficient while simultaneously improving accuracy [7].

The CNN classifier is trained using a substantial amount of data pertaining to each type of defect. They should have a significant number of instances of the macro-defect across the local wafer elements. For instance, a wafer scratch, which is a kind of macro-wafer defect, is passing over the various elements of a wafer, such as metal tracks, silicon spacers, and others. It is essential to provide samples for each of the intermediate processes involved in the production of the wafer. In addition, retraining is necessary to ensure that the classifier will be able to function properly with new designs derived from new applications. The elimination of human error from defect classification is made possible by the application of deep learning. Images with a resolution of 300 × 300 pixels, for instance, can be processed by a CNN with a simple architecture since it is designed to do so in order to maintain a level of performance. The CNN has four linear activation convolutional layers, subsequent to a fully connected (FC) layer that features sigmoid activation and further FC layers can be added to the structure. The very final layer is a SoftMax, which is responsible for classifying the data [8].

In this research, a deep-learning-based defect detection and classification model is proposed to detect and classify the defects on semiconductor chipsets. The proposed model analyzes and classifies the wafer surface defects of the input semiconductor wafer images, which is collected from WM-811K dataset. This work proposed ShuffleNet-v2 with CNN architecture for silicon wafer defect classification. The ShuffleNet-v2 performs as a feature extraction model and CNN performs as a classification model. This research proposes a deep-transfer-learning-based classification model for classifying the silicon wafer defects. The rest of the work is presented in the following sections. Section two presents the literature review for the research that covers the analysis of the related works. Section three presents the proposed research model, section four presents the results and analysis of the proposed model, and finally, section five presents the conclusion and future works.

2. Related Works

A transfer learning approach based on CNN was proposed in [9] that used the inception model for automatic defect classification. This work primarily focused on a defect analysis task that determined the reasons for a decrease in yield based on the results of defect classification. As a result of the fact that deep learning required a substantial quantity of labelled training data, classification performance can suffer in situations where insufficient or unreliable labelled data is available. Therefore, the transfer learning method utilized data that was either faulty or was tagged according to a variety of tasks. The results of this model in automatically classifying defects were satisfactory, despite the fact that the model had problems with overfitting.

In [10], a deep learning model CNN was implemented for automatically identifying wafer map defects. The majority of earlier analysis of wafer defect used machine-learning-based techniques for classification, which required process of feature extraction manually and a large number of hyperparameter setting. In contrast, CNN models were able to automatically extract effective features of a number of different defect classes. A data augmentation method was carried out so that the issue of the WM-811K dataset’s problematic data imbalance could be resolved. This model has attained an accuracy of 96.2 percent when identifying defects; however, the performance may have been improved by integrating a transfer learning model with the CNN architecture.

A deep-learning-based lightweight CNN model was implemented for the classification of automatic wafer defect detection in [11]. The model was used for automatic wafer defect detection. This model adopted multiple 1 × 1 standard convolution in order to improve the network depth. Depthwise separable convolutions and global average pooling were utilized in order to reduce the number of parameters and calculations. The problem of automated defect identification and pattern classifications for semiconductor silicon wafers can be effectively solved by using approaches that were based on CNN. These methods were quite effective. This work made use of a supervised learning CNN model, which required the data to be labelled by hand in order to learn anything from it. For this particular problem, a model of unsupervised learning could have been utilized.

In [12] a deep learning model was proposed to automatically classify various types of wafer surface defects. This system was based on the idea of automatic defect classification. This model integrated CNN and kNN, with CNN extracting useful information for kNN to use in classifying
wafer surface defects. Contrarily, the kNN approach clusters the feature vectors of CNN and performed membership testing for the new defects based on the total squared distances among the images and its k-nearest neighbours in the space of feature vector. This distance was calculated using the total squared distance among the images and its neighbours in the space of feature vector. This model might have been built on an unsupervised clustering model instead, which would have created a cluster every time a new class image was acquired for enhanced performance if that model had been used.

Using the attention mechanism and cosine normalization, a deep learning approach was described in [13] that has the capability of learning robust information from an imbalanced dataset. An enhanced convolutional attention module was introduced to the deep CNN in order to improve classification. This was done in order to compensate for the uneven distribution of features. To augment the positioning information of defect clusters, particularly, the feature-map-based direction mapping model was designed. The fine tuning of classifier was achieved by a minimal iterative training, which decreased the quantitative distribution sensitivity. This was done in response to quantity distribution imbalance, which was addressed by proposing the cosine normalization algorithm as a replacement for the fully connected layer. It is possible that resolving the issue of imbalanced datasets may facilitate the implementation of the algorithm in actual manufacturing.

In [14], a simplified representation of wafer maps was accomplished by using a collection of unique features that are invariant under rotation and scale. When analyzing large-scale datasets using wafer map failure pattern identification and wafer map similarity ranking, such features were absolutely essential. The support vector machine was utilized as the classifier in this model. It would have been helpful to uncover more robust features that are applicable to both the recognition of failure patterns in wafer maps and the ranking of wafer maps according to their similarity if an error analysis had been performed. In addition, a dimensionality reduction strategy might have been useful in determining whether or not a feature set with less dimensions might achieve equivalent performance.

A vision-based deep learning model was implemented in [15] to classify apparent surface defects on semiconductor wafers. This model identified and classified four distinct types of surface defects by utilizing deep CNN. These were center, local, random, and scraping defects. Following the training of the CNN with 5 convolution layers utilizing semiconductor wafers images with defects, a pretrained quicker RCNN was used and applied as a transfer learning model. Despite the fact that the model produced the best classification results, the fact that misclassifications occurred while it was being validated demonstrates that the proposed design has scope for further improvement.

A CNN approach was used for the pattern classification of wafer maps and the retrieval of wafer map images in [16]. The use of theoretically generated wafer maps for CNN training has the advantage of enabling classification tasks to be performed on imbalanced datasets derived from real wafers. Another CNN model based on rotation-based augmentation was developed in [17] for the purpose of improving wafer map structure classification models. The CNN that was trained with the enhanced data was able to provide consistent predictions for multiple rotational variations of a new wafer map. As a result, the classification performance was significantly improved. This model was very effective in acquiring higher classification performances, but it obtained high computational cost in the training process. The amount of computational cost extends with the number of training data.

For the purpose of wafer defect localization and classification, deep learning architectures such as YOLOv3 and YOLOv4 were proposed in [18]. When it comes to identifying and classifying wafer maps, YOLOv4 performed better than its predecessor, YOLOv3. These techniques of deep learning are able to cope with fault patterns that have complex structures. It might have been possible to get higher performances by making use of the usefulness of deep-learning-based models for effective defect localization and classification.

3. Proposed Deep Learning Model

In this research, a silicon wafer defect classification model based on deep transfer learning model was proposed. The pretrained deep transfer learning model ShuffleNet-v2 with a CNN architecture was proposed as the classification model. In this proposed model, the ShuffleNet-v2 performs as the feature extractor and the CNN performs as the classifier. The proposed model performs in the following stages such as data preprocessing, data augmentation, feature extraction, and classification. The workflow of the ShuffleNet-CNN is shown in Figure 3.

3.1. WM-811K Dataset. The WM-811K wafer dataset [19], which is the largest wafer map dataset that is freely available to the public, is used for training and testing purposes in the evaluation process. The dataset was obtained from the actual production of wafer process and includes 811,457 images with 9 different patterns of defect. Some of these defect patterns include Donut, Center, Edge-Ring, Edge-Loc, Random, Local, Near-full, Scratch, and None. Due to the fact that wafer images only exist in a two-dimensional format and have varying pixel rates along the width and length of the images, wafer images can take on a variety of different forms and dimensions. Within this dataset, 78.7 percent of the wafers do not have any labels, 3.1 percent of the wafers have real defect patterns, and 18.2 percent of the wafers are labelled with the None class. For the purposes of this analysis, the evaluation takes into account only wafers that have either been patterned or labelled. As a result, the classification procedure makes use of 25,519 wafers. The defining of nine distinct defect classes for wafer maps and the manual labelling of 172,950 wafer maps (21.3 percent of the total) were both the responsibility of domain experts. Unfortunately, the labelled dataset has a significant imbalance, and the None defect class was found to occupy 147,431
3.2. Data Preprocessing. The real-world dataset WM-811K was used to sample wafer defect images. This dataset contains 811,457 real wafer maps, but just 21 percent of these wafers have labelled classes. Due to the extreme imbalance that exists within this dataset, the proposed model’s training process will result in an overtraining of the data sample classes that constitute the majority. The effectiveness of deep learning can frequently be enhanced by increasing the amount of available data. Therefore, it is necessary to manually augment the training data that already exists using some methodologies. These methodologies were called data augmentation. It is possible to generate new training samples by applying techniques specific to the domain to the data samples taken from training samples. In addition, the data augmentation technique was utilized in order to address the issue of class imbalance. For the purpose of data augmentation, the convolutional autoencoder approach was utilized.

3.3. Data Augmentation. The proposed dataset does not have a uniform distribution, and while some of the wafer defect classes have an abundance of data, others have very minimal data. This problem, which affects the majority of image databases, is sometimes referred to as the class-imbalance problem. Because of the uneven wafer defect class distributions, the classification system may be forced to achieve a greater level of accuracy for the sample classes available majorly while it is in the training phase. Increasing the proportion of minority data sample classes in the dataset manually is the quickest and the easiest way to address the class-imbalance and overfitting issues. Data augmentation is a technique that is frequently utilized for the purpose of model regularization (Table 1) [20].

Following the data augmentation, two thousand samples of new wafers were obtained for each individual case. During the process of augmentation, noise has also been eliminated. Hence, produced new samples for the dataset, the data that has been augmented are concatenated with the data that was originally collected. At this time, the database contains 30707 images. The images labelled None have been removed because it does not play a significant role in this context. Therefore, there are 19707 images in the new dataset. The dataset was partitioned into training and testing data, with the training data containing 14780 images and the testing data containing 4927 images, respectively.

3.4. ShuffleNet-v2-CNN Model. The network design known as ShuffleNet is considered to be very efficient. The architecture of ShuffleNet was influenced by the effective CNN architectures like GoogleNet, ResNet, and Xception. GoogleNet-based operation for the group convolutions, ResNet-based function for the skip connection, and Xception-based operation for the depth-wise separable convolution. The group convolutions of point-wise convolution along with the structure of residual shortcut path is the ShuffleNet proposed as an approach [21].

GoogleNet was the first system to use group convolution, which seeks to achieve efficient convolution operation by separating the filter banks and the related feature maps into groups to process. This makes it possible to perform convolution on many GPUs in parallel, helping with each group’s processing. This not only makes the training process quickly, but it also helps the model to develop a more accurate representation of the data it was fed. By separating the
spatial correlations from the cross-channel correlations, the Xception is utilized for depth-wise separable convolutions. This resulted in a significant reduction in the number of trainable parameters. In conjugation, this was accomplished by conducting two functions known as depth-wise and point-wise convolutions in sequence. Convolution in the depth direction requires spatial convolution operations to be carried out by filters for every channel individually. This is the output at the intermediate level. Following this, a point-wise convolution was carried out, which involves convolving a filter with dimensions of $1 \times 1 \times C$, where $C$ represents the total count of channels, on the interceded output. This results in a significant decrease in the number of adjustable parameters and, as a result, saves both speed and computational power. The outputs of particular channels are exclusively obtained from those channels when numerous group convolutions are layered one on top of the other. This is one of the most important observations that can be made regarding the grouped point-wise convolution configuration.

In the CNN model, the operation known as “channel shuffle” is used to facilitate the flow of information across feature channels. It was incorporated into the architecture of the ShuffleNet. If a group convolution was permitted to collect input data from many groups, then, the relationship between the input channels and the output channels would be complete. Particularly, in order to build the feature map from the group layer that came before, first, the channels in every group are subdivided into many subgroups, and then, the different subgroups are fed into every group in the layer that comes after that.

A channel shuffle operation can be used to efficiently implement the CNN model by considering the following scenario: a convolutional layer has $g$ groups, and its output has $g \times n$ channels. Initially, the output channel dimension was reshaped into $(g, n)$, and then, it was transposed and flattened back before being used as the input of the next layer. Since channel shuffling is also differentiable, it can be incorporated into network architectures in order to facilitate end-to-end training (Figure 4) [22].

The fundamental components of ShuffleNet-v1 and ShuffleNet-v2 are illustrated in Figure 5. First, the figure shows the typical basic units of ShuffleNet-v1 in a skip connection network. This ShuffleNet-v1 includes point-wise group convolution as well as channel shuffle operation. The illustration on the left represents the network with the stride parameter set to one, while the representation on the right uses stride set as two and the “Concat” function to help the representations of output features dimensionality. It is composed of two point-wise functions, the second of which was carried out so that the dimension can be matched with the skip connections, because performing the shuffle operations on the second point-wise convolution yields results that are comparable to those obtained when the operation is not performed.

The ShuffleNet-v2 model proposes an improved version of the ShuffleNet-v1 model that was based on the following strategies, which are postulated as being necessary for an effective network architecture. These strategies are as follows:

(i) memory access cost (MAC) can be reduced to a minimum by maintaining a constant channel width. It is shown that the MAC was directly equivalent to the differences in channel widths for the similar number of FLOPs.

(ii) An excessive amount of group convolution will raise the MAC. Experiments have shown that the size of the group was directly equivalent to the MAC. Therefore, increasing the count of groups results in increased computations, which ultimately slows down the process.

(iii) The degree of parallelism that may be achieved in a network is decreased when it is fragmented. Fragmentation was indirectly related to parallel computations.

(iv) Operations based on element-wise are non-negligible: The decomposition charts of Run-time demonstrate that doing simple operations element-wise results in a performance overhead.

The basic units of ShuffleNet-v2 are also represented in Figure 5. At the start of all individual units, the channels of input feature were split into create two separate branches. This was a general channel split procedure. In accordance with the third strategy described earlier, one of the branches serves as the identity. In order to fulfill the requirements of...
the first strategy, the other branch features with 3 convolutions share the similar output and input channels. Both of the $1 \times 1$ convolutions were not group-wise. Since the split function already generates two groups, this is done in part to follow the second strategy. After the convolution is complete, the two different branches are concatenated. As a result, there is no change to the total number of channels. Then, the same channel shuffle process that was utilized in ShuffleNet-v1 was performed in order to make it possible for transmission among the two different branches.

Following the shuffling, the next unit was initiated. It is important to mention that the Add operation is no longer available in ShuffleNet-v1. Only one of the branches provides access to element-wise techniques such as depth-wise convolutions and ReLU. Additionally, the three element-wise operations that are performed in succession, namely Concat, Channel Shuffle, and Channel Splits, were combined into one element-wise function. According to the fourth strategy, these shifts are effective. One of the key distinctions between ShuffleNet-v1 and v2 is that the latter includes an additional $1 \times 1$ convolution layer right before the global averaged pooling step to shuffle features, whilst the former does not [23].

Table 2 presents an overview of the network’s architectural composition. Figure 5 cells are arranged in a manner that stacks Stage 2, Stage 3, and Stage 4 on top of one another. The number of stacks is represented in the repeat column.

The learning ability of the CNN architecture is dependent on the sequential or spatial properties of the data. Whenever the input information to the network was sparse highly, the learning capacity of the networks suffered a significant reduction. When two input matrices are multiplied together, the result is a convolution. At last, the input matrix was traversed in its entirety, and the result matrix was calculated. Given that every kernel was smaller in height and width than the inputs, all neurons in the activation map were just linked with the small local zone of the input images, implying that the size of every neuron’s receptive field was limited, about equal to the size of each kernel.

As represented in equation (1), convolution was a mathematical operator, which generates the function by combining the values of two other functions, where $r(x)$ indicates the overlap length integral of the overlapping functions value of the functions $f$ and $g$, which is generated by shifting and flipping.

$$r(x) = \int_{-\infty}^{\infty} f(\tau)g(x - \tau)d\tau.$$  

(1)

Convolution is defined as the weighted superpositions of one function over another in terms of its physical meaning. The output of the system is the consequence of several inputs being superimposed on one another. In the analysis of image, the
actual pixel point was represented by \((x_i, y_i, z_i)\), and the action point was represented by \((x, y, z)\). All of the action points were integrated into a single convolution kernel for further processing. It is possible to acquire the result of the final convolution once all of the action point on the convolutions kernel have been utilized to each of the actual pixel points in order.

By applying an element-wise operation equation on the nonnegative values acquired from the convolutional layer, the ReLU reduces the nonnegative values to zero (2). The elements of the input vector are represented by the term “I.”

\[
\sigma(I) = \max(0, I).
\]  

(2)

The pooling layer, also known as the down-sampling layer, has the objective of reducing the size of the matrix created by a convolutional layer by a factor of two. The use of pooling minimizes the number of parameters and features, hence reducing the difficulty of calculating convolutional network. In addition, pooling helps to prevent overfitting to a particular level, creating optimization more straightforward. It is possible to boost the network invariance against translations by the use of pooling, which is extremely important for the improvement of the network’s ability to generalize. Because of pooling, the model is more concerned with the presence of particular features rather than the precise position of those features. There are two basic pooling functions: maximum pooling and average pooling. Average pooling is the most common. Average pooling, to put it simply, is the process of considering the average of a minimum range of values. Max pooling refers to taking the most extreme value from a small range.

It is possible to think of the completely connected layer as a typical neural network that is capable of making logical inferences. Fully linked layer in the proposed system transforms the 3D matrix acquired from the preceding layer to the 1D vector by performing the full convolution operations on the 3D matrix. The following was the mathematical expression for the fully linked layer’s mathematical operation.

\[
Z_{V_x \times 1} = \text{Weight}_V \times X \cdot V_{x \times 1} \cdot \text{Bias}_V \times 1.
\]  

(3)

In this equation, \(V_o\) and \(V_i\) denote the output and input vector sizes, respectively, and \(Z\) was the output of the FC layer. Aside from that, the terms \(\text{Weight}\) and \(\text{Bias}\) were used to denote the weight and bias matrix, respectively.

When a CNN architecture is finished, it has a final layer called the SoftMax layer, which was utilized for calculating probabilities of normalized class \(P(y^{(i)} = n^{(i)} | x^{(i)}; W)\) for every class \(n^{(i)}\), in \(n\) classes, by equation (4).

\[
y^{(i)} = n^{(i)} | x^{(i)}; W = \left\{ \begin{array}{l} y^{(i)} = 1 | x^{(i)}; W = \left[ \begin{array}{l} y^{(i)} = 1 | x^{(i)}; W = \left[ \begin{array}{l} y^{(i)} = 2 | x^{(i)}; W = \left[ \begin{array}{l} y^{(i)} = n | x^{(i)}; W = \left[ \begin{array}{l} e^{W_1 x^{(i)}} \\ e^{W_2 x^{(i)}} \\ \vdots \\ e^{W_n x^{(i)}} \\ y^{(i)} = 1 | x^{(i)}; W = \left[ \begin{array}{l} y^{(i)} = 2 | x^{(i)}; W = \left[ \begin{array}{l} y^{(i)} = n | x^{(i)}; W = \left[ \begin{array}{l} e^{W_1 x^{(i)}} \\ e^{W_2 x^{(i)}} \\ \vdots \\ e^{W_n x^{(i)}} \\ y^{(i)} = 1 | x^{(i)}; W = \left[ \begin{array}{l} y^{(i)} = 2 | x^{(i)}; W = \left[ \begin{array}{l} y^{(i)} = n | x^{(i)}; W = \left[ \begin{array}{l} e^{W_1 x^{(i)}} \\ e^{W_2 x^{(i)}} \\ \vdots \\ e^{W_n x^{(i)}} \end{array} \right] \\ \end{array} \right] \\ \end{array} \right] \\ \end{array} \right] \right] \right] \right] \right] \right] \right]
\]  

(4)

In this equation, \(n\) denotes the overall total of data samples, with \(i = 1, \ldots, n\). The weights are represented by \(W_i\), while the input to the classifier is marked by the notation \(e^{W_i x^{(i)}}\).

In order for the model to reach the local and global optimal points, the optimizer works constantly during the training phase of the model to continually optimize the values of the loss functions. This is accomplished by calculating and updating the model’s parameters of network. When applied in the real world, the speed of convergence and overall model effect are both determined by the loss function and optimizer that are chosen. The improper loss functions and optimizer could lead the classifier into local optimal point that was when the values of the loss functions linger about the local ideal points and it cannot reach the global optimal points, which will result finally in the output having poor accuracy. In the context of this analysis, the cost function of cross-entropy was utilized to perform the calculation necessary to determine the difference in probability distribution that exists among the conventional model’s distributions and the actual distributions in order to derive the values of the loss function. An expression for the cost function of cross-entropy was expressed in equation (5), where the value \(a\) represents the output values of the neuron activation functions and \(y\) represents the output values [24].

\[
C = -\frac{1}{n} \sum_i [y \ln a + (1 - y) \ln (1 - a)].
\]  

(5)

In order to prevent the model from settling into the position of the local optimal point, the Stochastic Gradient Descent (SGD) optimizer was utilized throughout this work to accomplish the task of optimizing the model. The initial learning rate (\(\mu\)) in the ShuffleNet model was set at 0.001.

3.5. Experimental Analysis. The proposed model was implemented and tested using the PYTHON 3.7.12 programming language tool. The experiments are performed on Google Colab Pro. The experiments were carried out using a PC with an Intel i7-processor CPU running at 2.7 GHz, 16 GB RAM, and a 64-bit Windows 10-OS. The classification of silicon wafer defects is the primary objective of this work. After the process of data augmentation, 19707 images were divided into testing and training sets. The training set comprised of 12630 images, and the testing set comprises of 705 images. Figure 6 represents the sample images from the WM-811K dataset.

3.6. Performance Metrics. This proposed research included a variety of assessment parameters like accuracy, recall, precision, and F1 score so that appropriate comparisons could be made between the results of the experiments. According to equation (6), accuracy might be defined as the proportion of correct identifications to the total number of input images.

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}
\]  

(6)

The proper identification of wafer defects is denoted by the terms TP (true positive), TN (true negative), while the
improper detection of wafer defects is denoted by the terms FP (false positive) and FN (false negative). According to equation (7), precision refers to the proportion of wafer defects that were correctly classified in comparison to the total number of defects that were found by the model. The following is an expression of precision:

\[
\text{Precision} = \frac{TP}{TP + FP} \tag{7}
\]

According to equation (8), sensitivity or recall can be defined as the proportion of the number of correctly detected defects to the total number of wafer defects, with the former being greater than the latter. This is the case when the number of wafer defects is greater than the total number of defects.

\[
\text{Recall} = \frac{TP}{TP + FN} \tag{8}
\]

The F1 score, which was defined as the harmonic mean of the model’s recall and precision, provides a definition that is quite similar to this one. Equation (9) is the mathematical expression that can be used to describe the F1-score.

\[
\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{9}
\]

Table 3 represents the performance evaluation of the WM-811K dataset based on the ShuffleNet-v2-CNN model. The performances of the proposed model were measured related to accuracy, recall, precision, and F1-scores. Although the dataset has nine classes, only eight classes were used for the evaluation as tabulated in the above table. The proposed ShuffleNet-v2-CNN model has achieved better performances related to the all eight classes and obtained 96.93% overall accuracy. The proposed model has achieved some better performances for the classes such as Scratch, Near-full, Edge-Loc and Random with and above 97% accuracy. Figure 7 represents the graphical plot of the accuracy performance obtained by the proposed model.

The ShuffleNet-v2-CNN model has achieved better precision scores regarding the eight classes of the dataset and obtained 95.40% overall precision. The proposed model has achieved some better performances for some classes with an average precision rate of 95% precision. Figure 8 represents the graphical plot of the precision performance obtained by the proposed model.

Figure 9 represents the graphical plot of the recall performance obtained by the proposed ShuffleNet-v2-CNN model.
The proposed ShuffleNet-v2-CNN model has achieved better performances related to the all eight classes and obtained 96.26% overall recall score. This recall evaluation is also referred as detection rate, which plays a significant part of evaluation in image processing analysis. The proposed model has achieved some better performances for some classes with a constant 96% precision.

The ShuffleNet-v2-CNN model has achieved better F1-scores regarding the wafer defect classes of the WM-811K dataset and obtained 95.75% overall F1-score. The proposed model has achieved some better performances for some classes with an average rate of 95% F-score. Figure 10 represents the graphical plot of the F1-score obtained by the proposed model.

Table 4 represents the comparison of performance analysis of the proposed ShuffleNet-v2-CNN model with the various models analyzed from the literature review. The compared models were DCNN, CNN-kNN, CNN-FRCNN, and CNN-Inception. These compared models have obtained some better performances as discussed in the related works, but comparing with the proposed model, the ShuffleNet-v2-CNN model has overcome all these models with better performances in each and every parameter of evaluation. Figure 11 represents the graphical plot of the performance analysis comparison of the proposed model with other analyzed models. As per the comparison, the proposed ShuffleNet-v2-CNN model has acquired 1.2% to 5.7% improved accuracy performance, 1.6% to 5.07% better precision performance, 1.08% to 5.37% improved recall performance, and 0.9% to 5.9% better F1-score performances. Overall, based on the comparison, the proposed model has acquired best performances compared with these models, where the CNN-FRCNN model has some close performances to the proposed model. The DCNN model is the least performed model following the CNN-kNN model.

4. Conclusion

In this research, a deep learning-based silicon wafer defect detection and classification model was proposed. The
pretrained deep transfer learning model ShuffleNet-v2 integrated with the CNN architecture was the developed model for this silicon wafer defect classification. In this proposed model, the ShuffleNet-v2 performs as the feature extractor and the CNN performs as the classifier. The proposed model performs in the following stages such as data preprocessing, data augmentation, feature extraction, and classification. For the evaluation, the WM-811K dataset was used in this work, which was a high imbalanced dataset. For balancing this dataset, the data preprocessing and data augmentation were performed. The convolutional autoencoder technique was used for the data augmentation that balances the dataset with newly generated samples with 30707 images. As “None” class was not considered in this work for classification, those images were excluded. Hence, the dataset has 19707 images. The dataset was split into training and testing, which the training data includes 14780 images and testing data includes 4927 images. For performance evaluation, the proposed ShuffleNet-v2-CNN was evaluated with parameters like accuracy, recall, precision, and F1-score. The proposed model has obtained an overall accuracy of 96.93%, 95.40% precision, 96.26% recall, and 95.75% F1-score. These performances were compared with the other models analyzed from the literature review such as DCNN, CNN-kNN, CNN-FRCNN, and CNN-inception. As per the comparison, the proposed ShuffleNet-v2-CNN model has acquired 1.2% to 5.7% improved accuracy performance, 1.6% to 5.07% better precision performance, 1.08% to 5.37% improved recall performance, and 0.9% to 5.9% better f1-score performances. The main limitation of this work was the highly imbalanced dataset, where the data augmentation process could have been better in balancing the dataset with large number of samples. In future, the proposed method can be upgraded with implementing the filtering and image enhancement techniques, and feature selection technique for improved performances.

Data Availability
The data used to support the findings of this study are included within the article.

Conflicts of Interest
The authors declare that they have no conflicts of interest.

References


