A Database for Counterfeit Electronics and Automatic Defect Detection Based on Image Processing and Machine Learning

Navid Asadizanjani,[1] Nathan Dunn,[2] Sachin Gattigowda,[1] Mark Tehranipoor[1], and Domenic Forte[1]
[1]Electrical and Computer Engineering Department, University of Florida
[2]Electrical and Computer Engineering Department, University of Massachusetts Amherst

Abstract: *Counterfeiting is an increasing concern for businesses and governments as greater numbers of counterfeit integrated circuits (IC) infiltrate the global market. There is an ongoing effort in experimental and national labs inside the United States to detect and prevent such counterfeits in the most efficient time period. However, there is still a missing piece to automatically detect and properly keep record of detected counterfeit ICs. Here, we introduce a web application database that allows users to share previous examples of counterfeits through an online database and to obtain statistics regarding the prevalence of known defects. We also investigate automated techniques based on image processing and machine learning to detect different physical defects and to determine whether or not an IC is counterfeit.*

Keywords: Integrated Chips, Artificial Neural Network, Image Processing, Machine Learning.

1. Introduction

Counterfeit ICs are a growing issue in the global market, with some counterfeit ICs infiltrating high-risk applications such as those in the military or medical sectors [1], [2]. Counterfeit ICs may feature a reduced lifespan, fail under conditions mandated by the manufacturer's specification, leak secure data from the system to a remote location (i.e., hardware Trojans [3]), and more. In some cases, the adversary copies the design and labels counterfeit ICs with his own brand name or with the original equipment manufacturer's name using reverse-engineering techniques [10], [11], [14], [15]. Since the adversary does not incur the engineering research and development costs, his products are priced lower, thereby resulting in a loss for authentic products.

The taxonomy of counterfeit component types and defects in electronic components have been discussed in [4]–[5], where the two main classes of defects are classified as physical and electrical defects.

Here we will study the physical defects and try to expedite and facilitate the detection and recordkeeping of such physical defects. With the help of high-precision microscopes, most of the physical defects can be identified by a subject matter expert (SME). The SME has to carefully test the entire IC for each defect. This is a labor-intensive and timely process. For a modern electronics manufacturer, it is not feasible for an SME to analyze all the chips and components manually because a single US company can import tens of millions of chips annually.

The US Chamber of Commerce has identified 11 issues in the current state of the global electronic supply chain in regard to counterfeiting [13]. In this paper we begin to address two of those findings, namely: 1) the very limited recordkeeping on counterfeit incidents; and 2) the need for stricter testing protocols and quality control practices.

We will introduce an online web interface used to facilitate the sharing of counterfeit data between researchers, industry, and government. This database will utilize the defects taxonomy from [4] to track and report the frequency of these defects across various product types. This will be a graphical user interface for gathering and accessing collected data. We will also develop preliminary algorithms that analyze optical images and automatically extract defects using artificial neural networks and image processing algorithms.

2. Counterfeit Database

The establishment of a counterfeit defect database would be an important step in improving the recordkeeping of detected counterfeit electronics [12]. A central repository of defect images and statistics would serve both to educate industry and government professionals on the prevalence of counterfeiting and to provide an important source of data for future research of ensuring component authenticity.

In our database (as seen in Figure 1), we seek the results of SME inspections. Most physical examinations of electronic component shipments do not involve individually examining every component [1]. Rather, SMEs examine a small sample of the total shipments and perform the necessary inspections on each chip in the sample before passing a judgment on the legitimacy of the shipment as a whole. In our database, we store the results of that sample inspection—whether or not a particular chip has passed, and the particular flaws found in that chip.

2.1. Design

We wished for the results of an SME inspection to be publicly available, in order to showcase visual examples of counterfeit defects. In order to guarantee the integrity of collected data, however, it is desirable for only verified SMEs to have writing/editing access to the database. To ensure this, we have implemented an admin-based verification system. Any data or edits published to the website must first be verified by an administrator before they take effect.

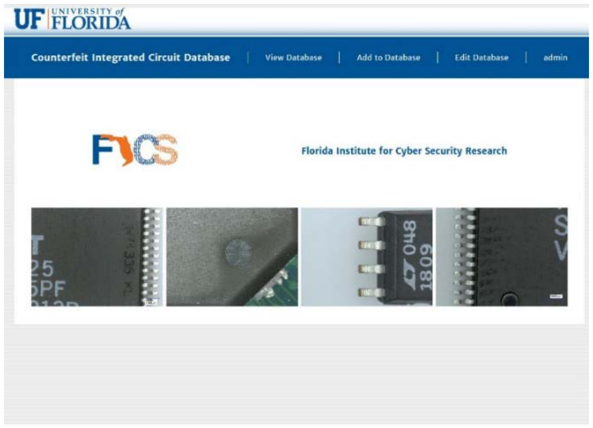

UF UNIVERSITY of FLORIDA
Counterfeit Integrated Circuit Database
View Database
Add to Database
Edit Database
admin
FICS
Florida Institute for Cyber Security Research

Figure 1. Homepage of the Counterfeit IC Database[1].

Structurally, the organization of the data follows the logic of an SME inspection (as seen in Figure 2). The various products made by an IC manufacturer are at the top of this hierarchy. Under a given product, an SME may upload a sample or a collection of ICs with a common trait, such as date of inspection. Within this sample, the SME associates all inspected components, selecting the identified defects and uploading example images of the flaws.

2.2. Implementation

We built our application using PHP for both server-side responses and templating of client pages. The application was built around the Klein.php routing framework. The framework is only used to serve web pages for the application but in the future could be used to serve a RESTful API for external applications. This would allow external researchers and industry professionals to utilize our data and images for their own work.

[1] Note that all the images of the database interface shown in this paper are subject to change as it is in a constant state of improvement.

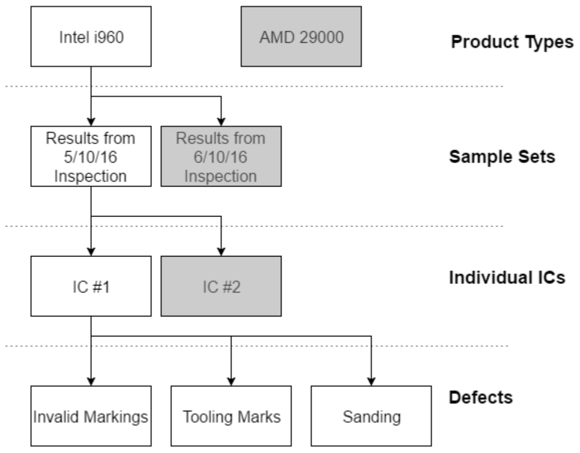


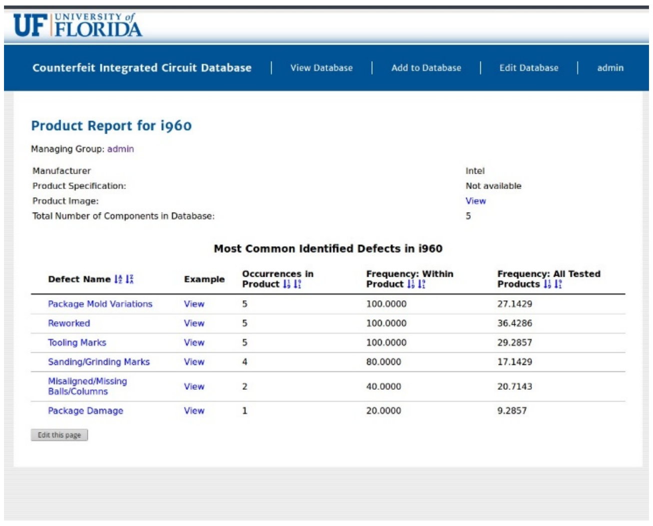
Figure 2. Access hierarchy of the Counterfeit IC Database, displaying the hierarchy of one IC within a sample. Each layer has one-to-many relationship with layer below it.

In our current implementation of the website, navigation is handled by a central navigation bar with drop-down options, offering users three methods of interacting with the database. Users may: 1) view reports within the database; 2) add new objects to the database; or 3) edit existing objects.

Under the Add to Database tab and the Edit Database tab, user access is restricted to verified users. The options under these menus allow users to contribute by uploading images and specifications for new product types, samples, and components. To prevent the tedious action of uploading components one at a time, large data entries (such as sample uploads) are typically accomplished by downloading a Microsoft Excel form from the website, filling in the results of the sample inspection, and then uploading the completed form. The server then parses the .xlsx file, records the associated defect data, and stores any images on the server.

The actual data in the database is queried through the View Database tab. These actions are available to all who visit the website. This tab contains options to generate three types of reports.

1**)** *Product Report:* In a product report, the user may view product profiles of confirmed, legitimate chips and find official product specifications. All counterfeit defects that have been observed within a given product in the database are listed on the report, as is the frequency at which the defect occurs. The user may further filter the output by selecting samples within the product.



Product Report for i960

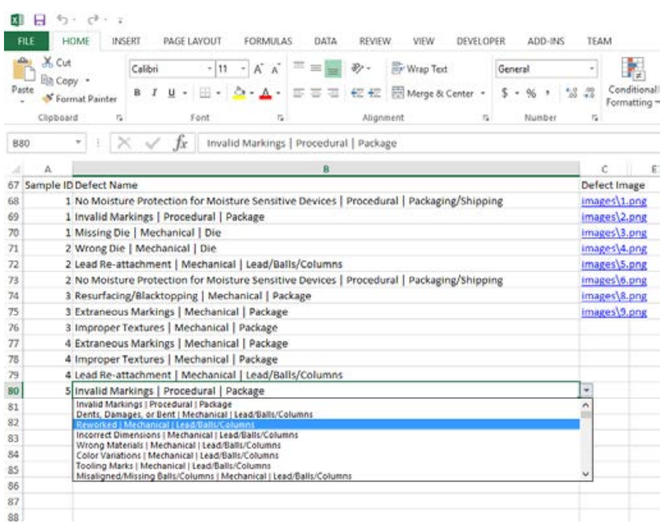
Managing Group: admin

Manufacturer	Intel
Product Specification:	Not available
Product Image:	View
Total Number of Components in Database:	5

Most Common Identified Defects in i960

Defect Name	Example	Occurrences in Product	Frequency: Within Product	Frequency: All Tested Products
Package Mold Variations	View	5	100.0000	27.1429
Reworked	View	5	100.0000	36.4286
Tooling Marks	View	5	100.0000	29.2857
Sanding/Grinding Marks	View	4	80.0000	17.1429
Misaligned/Missing Balls/Columns	View	2	40.0000	20.7143
Package Damage	View	1	20.0000	9.2857

(a)

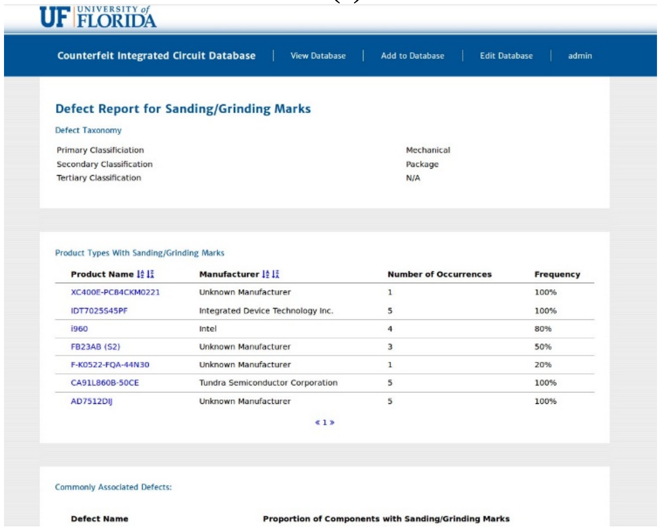


Invalid Markings | Procedural | Package

	A	B	C
67	Sample ID	Defect Name	Defect Image
68	1	No Moisture Protection for Moisture Sensitive Devices | Procedural | Packaging/Shipping	images\1.png
69	1	Invalid Markings | Procedural | Package	images\2.png
70	1	Missing Die | Mechanical | Die	images\3.png
71	2	Wrong Die | Mechanical | Die	images\4.png
72	2	Lead Re-attachment | Mechanical | Lead/Balls/Columns	images\5.png
73	2	No Moisture Protection for Moisture Sensitive Devices | Procedural | Packaging/Shipping	images\6.png
74	3	Resurfacing/Blacktopping | Mechanical | Package	images\8.png
75	3	Extraneous Markings | Mechanical | Package	images\9.png
76	3	Improper Textures | Mechanical | Package	
77	4	Extraneous Markings | Mechanical | Package	
78	4	Improper Textures | Mechanical | Package	
79	4	Lead Re-attachment | Mechanical | Lead/Balls/Columns	
80	5	Invalid Markings | Procedural | Package	

Invalid Markings | Procedural | Package
Dents, Damages, or Bent | Mechanical | Lead/Balls/Columns
Reworked | Mechanical | Lead/Balls/Columns
Incorrect Dimensions | Mechanical | Lead/Balls/Columns
Wrong Materials | Mechanical | Lead/Balls/Columns
Color Variations | Mechanical | Lead/Balls/Columns
Tooling Marks | Mechanical | Lead/Balls/Columns
Misaligned/Missing Balls/Columns | Mechanical | Lead/Balls/Columns

(b)



Defect Report for Sanding/Grinding Marks

Defect Taxonomy

Primary Classificiation	Mechanical
Secondary Classification	Package
Tertiary Classification	N/A

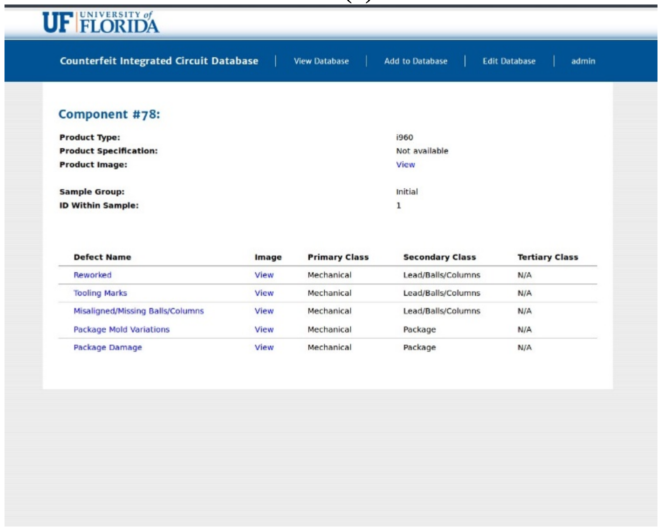
Product Types With Sanding/Grinding Marks

Product Name	Manufacturer	Number of Occurrences	Frequency
XC400E-PC84CKM0221	Unknown Manufacturer	1	100%
IDT7025S45PF	Integrated Device Technology Inc.	5	100%
i960	Intel	4	80%
FB23AB (S2)	Unknown Manufacturer	3	50%
F-K0522-FQA-44N30	Unknown Manufacturer	1	20%
CA91L860B-50CE	Tundra Semiconductor Corporation	5	100%
AD7512DIJ	Unknown Manufacturer	5	100%

Commonly Associated Defects:

Defect Name	Proportion of Components with Sanding/Grinding Marks

(c)



Component #78:

Product Type:	i960
Product Specification:	Not available
Product Image:	View
Sample Group:	Initial
ID Within Sample:	1

Defect Name	Image	Primary Class	Secondary Class	Tertiary Class
Reworked	View	Mechanical	Lead/Balls/Columns	N/A
Tooling Marks	View	Mechanical	Lead/Balls/Columns	N/A
Misaligned/Missing Balls/Columns	View	Mechanical	Lead/Balls/Columns	N/A
Package Mold Variations	View	Mechanical	Package	N/A
Package Damage	View	Mechanical	Package	N/A

(d)

Figure 3. (a) Product report; (b) uploading a sample through an Excel form; (c) defect report; (d) viewing an individual chip.

2**)** *Defect Report:* In a defect report, the user is able to trace the classification of a particular defect, according to the taxonomy provided by Guin and Tehranipoor [4], [5]. For a given defect, the user may also see a list of the product types that most commonly experience that defect, as well as the defects that are most likely to occur in tandem.

3**)** *Component View:* In a component report, the user may view all defects that have been registered to that image, and the associated images. Examples of these three reports are available in Figure 3.

This is valuable information for both manufacturers and consumers. Knowing prevalent defects, a manufacturer may develop additional mechanisms to guarantee the legitimacy of components, while consumers can target detection methods toward the most common indicators of counterfeiting.

To further assist in educating consumers on counterfeit defects, we plan on developing a mobile version of the application. For SMEs during inspection, a mobile application could be a useful tool by providing example images of particular defects. Creating a mobile app would likely involve creating an API for accessing the database, which could then be made publicly available to other researchers.

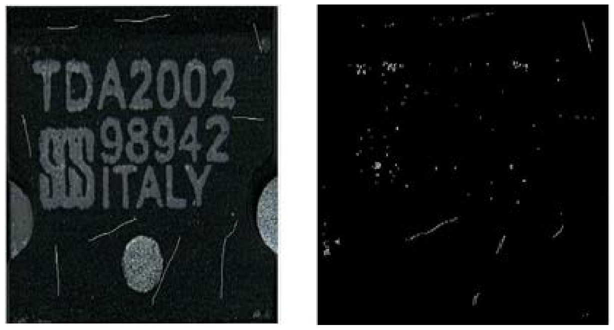
The database could also be further expanded to collaborate with future automated methods of counterfeit detection. Systems that automatically detect particular defects could subsequently upload samples to the database, thus increasing the overall amount of information available for a particular component. With consistent images and sampling, the database could also serve as a repository of training material for machine-learning algorithms. A program implementing a binary classifier could access the database, train the classifier using examples of legitimate and counterfeit components, and then examine new components of the same product type.

3. Automated Defect Detection

In this section two techniques for detecting defects on ICs are presented: 1) image processing and 2) artificial neural network (ANN)–based algorithms.

3.1. Defect Detection Using Image Processing

There are many image processing algorithms used to detect features on different types of images. These algorithms are designed in a way to highlight one specific property of a feature based on its histogram. These features can be defects on ICs such as scratches, color variations, ghost marking, etc. It is very important to select the right algorithm and right threshold in order to better highlight the defect.



(a) (b)

Figure 4. The counterfeit IC with scratch defect a) original image b) after image processing.

Figure 4 shows an IC with a scratch defect. We apply image processing techniques on this image to detect these scratches. The computer vision algorithms we used include:

1) Image filtering, with a modified Hough transform to detect circles. This will help to remove these circles from the image and reduce the complexity of the image for the next step, where edges will be detected.

2) Sobel filter and canny edge detection algorithm to detect the scratches. This is based on the image gradient, where the sudden change in the gradient will represent an edge or scratch.

The white pixel count is used for identifying whether the feature is a scratch or not. If it is above the user-defined threshold, then it is a scratch. Figure 4 shows the output image of the technique. It has clearly highlighted the areas with scratches on the IC surface.

However, many defects cannot be identified through this method alone. For example, many counterfeits display variations in color that can be identified based on the intensity of grayscale pixels rather than on the placement of binary pixels within the image. To determine whether color variations exist on a given IC, an entirely different image processing method would have to be run in addition to the scratch detection algorithm.

Further, image processing requires a high degree of user interaction. In these algorithms, the user must define the correct threshold that the algorithm uses to classify images. The correct threshold may vary according to the background, lighting, and orientation of the IC within the image. The user would have to define the threshold for a given set of conditions, and then those conditions would have to be maintained throughout the sample [7]-[9].

These restrictions increase the difficulty in developing a system to determine IC authenticity, by both increasing the number of implemented algorithms and reducing the variety of images that may be used as input. Ideally, a system should take a more general approach, such as a single algorithm that can identify a wide variety of defects under various image conditions. An artificial neural network is an example of a system with some of these properties.

3.2. Defect Detection Using ANN

3.2.1. Artificial Neural Networks (ANN)

An artificial neural network (ANN) is a machine-learning technique that is modeled similarly to the structure of a human brain. A typical ANN consists of neurons in different layers, such as input, hidden, and output layers. The information is stored in the interconnections between neurons across the adjacent layers [6]. The number of hidden layers should be at least one, and the number can be increased depending on the application. The output of a neuron is calculated by using an appropriate activation function for the input value. Step function, Tanh, and ArcTan are some of the examples of activation functions.

Figure 5 shows the structure of a typical ANN. Each neuron from the input layer is connected to each neuron in the first hidden layer. The input value at a node of a hidden layer is the weighted summation of all the input nodes. The factor by which each node contributes to the value of the input of the next node will decide the weight matrix between the two layers. The weight matrix stores the crucial information on how each layers are connected. This connection is unique for each class of inputs. The output of the neuron is computed using the activation function. This weighted summation of adjacent layers continues until the output layer. During the training phase, the expected output is compared with the output obtained by the computations. The error is then sent back to the network layers to update the weights. The neural network is then trained for more iterations for the same image until the error level is less than the accepted tolerance level.

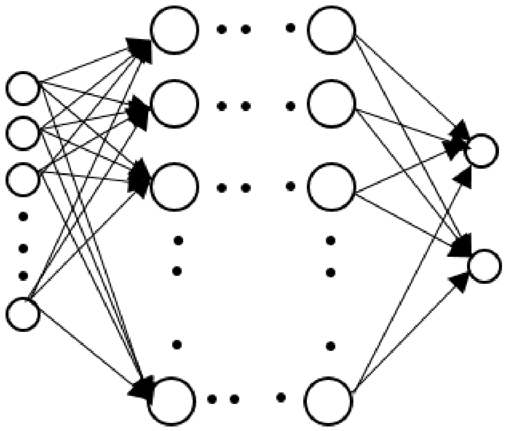


Figure 5. Artificial neural network showing input layer, hidden layers, and the output layer, from left to right.

The complexity of the operations in the ANN depends on factors such as the number of nodes in each layer, the learning rate, the number of iterations done for training, and the number of hidden layers. The artificial neural network is an extremely useful tool for pattern recognition applications.

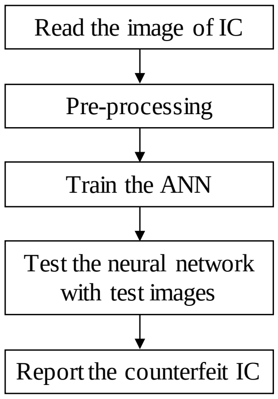

Read the image of IC
Pre-processing
Train the ANN
Test the neural network with test images
Report the counterfeit IC

Figure 6. Flowchart for the counterfeit IC detection using neural network.

The ANN has been studied extensively for applications such as character recognition, face recognition, etc. The flowchart in Figure 6 briefly describes the steps for how to use ANN for automatic IC defect detection.

3.2.2. Backpropagation Algorithm to Train ANN

The backpropagation algorithm (i.e., propagating the error in the backward direction of the network) is a typical artificial neural network algorithm used for training the network. The basic idea behind backpropagation algorithm is to reduce the error energy of a neuron in the neural network [6]. The error energy is represented as:

$$E_j(n) = \frac{1}{2} e_j{}^2(n) \tag{1}$$

where $e_j(n)$ represents the error at the node, i.e., the difference between expected value and the actual value. Differentiating $e_j(n)$ with respect to the weights gives us the condition for minimum error energy. The weight correction term obtained is:

$$\Delta w_{ji}(n) = \eta\delta_j(n)x_i(n) \tag{2}$$

where w is the weight matrix, η is the learning rate, $\delta_j(n)$ is the local gradient, and x is the input vector.

The input for a hidden neuron $z_j(n)$ is a linear combination of input vector x with weights v between the layers as presented in Equation 3. Output of the hidden layer will be the result of the activation function for the input.

$$z_j(n) = \sum_{i=1}^{N} v_{ji}\,(n)x_i(n) \tag{3}$$

Similarly an output node $y1_k(n)$ is updated based on the linear combination of the hidden nodes z and weights w between hidden and output layers. The output of the neuron will be:

$$y_j(n) \;=\; \phi_j(y1_k(n)) \tag{4}$$

where Φ_j is the activation function.

The equation for updating weights between output layer and the hidden layer is:

$$W_{new} = W_{old} + \eta\delta_j(n)x_i(n) \tag{5}$$

where W_{new} is the updated weight matrix between output and hidden layer, W_{old} is the previous weight matrix, η is the learning rate, x is the input vector to the neural network, and $\delta_j(n)$ is as followed:

$$\delta_j(n) = \left(d_j(n) - y_j(n)\right) * (1 - tanh\left(y_j(n)\right) * (1 + tanh(y_j(n))) \tag{6}$$

where, $d_j(n)$ is the expected output value, $y_j(n)$ is calculated output value in the forward path, and *tanh* is the activation function.

Similarly, the equation for updating weight matrix between hidden and input layer is:

$$V_{new} = V_{old+}\eta\delta_j(n)x_i(n) \tag{7}$$

where V_{new} is the updated weight matrix between hidden and input layer. V_{old} is previous weight matrix and $\delta_j(n)$ is:

$$\delta_j(n) = (1 - tanh\left(z_j(n)\right)(1 + tanh\left(z_j(n)\right) * \sum_{k=1}^{M} \delta_k(n)\, w_{kj}(n) \tag{8}$$

In the next section, we will discuss how to use these equations to implement the algorithm for defect detection purpose.

3.2.3. ANN Implementation

The implementation of the counterfeit IC detection using artificial neural network can be divided into two parts: 1) the **training phase**; and 2) the **testing phase**. In our experiments, we divided the available set of images in the database for training and testing. For example, if the IC with a scratch defect has 10 samples, we can use 7 samples for training and 3 for testing. The proportion can be varied to check the efficiency of the system. During the training phase, we can train for multiple defects at once. This would help us determine the defect type on its own without a prior knowledge during the testing phase.

Training Phase

We utilize the backpropagation algorithm discussed earlier to train the artificial neural network, using *tanh*, the hyperbolic tangent function, as the activation function. With a range of (-1,1), *tanh* allows both positive and negative responses. It also ensures that the intermediate values are within range for the computations since the size of the images in our application is higher. Only one hidden layer was used to reduce the complexity of the ANN.

Figure 7 shows the flowchart for the counterfeit IC detection procedure. The input image is read using the MATLAB function imread, which gives the matrix consisting of the pixels of the image. The image is then converted to a column matrix by arranging the columns of the input image matrix one below the other. If the input image is of size M × N, the new column matrix will have a size of MN × 1. For a 1,600×1,200 image, the dimension will be 1,920,000. This can be reduced by resizing the image using various preprocessing techniques.

Other preprocessing techniques that we used were image filters to remove pixels over or below a threshold, edge detection, normalizing the image, and converting RGB images to gray scale. The input image is normalized by dividing pixel values by 255 for grayscale images. Until this step, the techniques applied for the processing of the image are the same for training and testing. During training, the output for a particular input image is known. This information is also sent to the ANN.

The weight matrices between input, hidden, and output layers are initialized with random values between −0.5 and +0.5. After the preprocessing of the image, the expected output layer is initialized with its corresponding class. The linear combination of the initial column matrix and the first weight matrix gives the input for the hidden layer. Using the activation function *tanh*, we get the output of hidden layer. Since the linear combination of input with weight matrix can be very high numerically, activation helps in keeping them in the range of −1 to 1. This output will then be multiplied with weight matrix between the hidden and output layer to get the input.

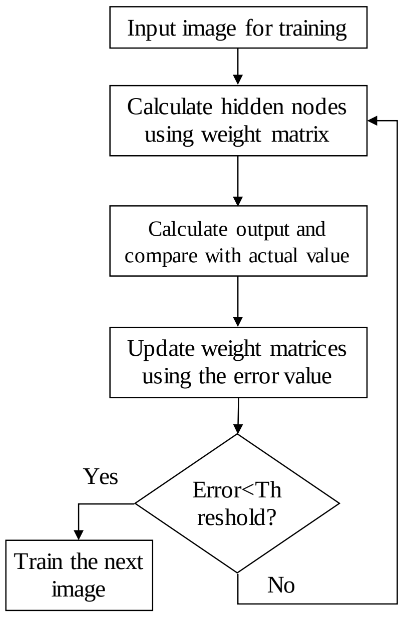

Input image for training
Calculate hidden nodes using weight matrix
Calculate output and compare with actual value
Update weight matrices using the error value
Error<Threshold?
Yes
No
Train the next image

Figure 7. Logical flowchart for artificial neural network training.

Similar to nodes in the hidden layer, output of the output layer is calculated using the activation function. The error at the output is calculated by comparing the activation function with the expected output given at the start of training. The error is propagated back to the network. This loop can be continued until we reach a point where the error is within the expected range.

Testing Phase

During testing, we do not use the backpropagation algorithm, as the ANN is already trained with error corrections for the weights. We do not know the output value prior to the computations in ANN. The ANN will then determine if the IC possesses a defect or not depending on the output of the ANN. We apply the same preprocessing methods that were employed during the training phase for the images. For the purpose of testing the techniques, we compare the output of the ANN during testing with the expected output and calculate the efficiency of the system. The flow of training and testing of the ANN is explained in Figures 7 and 8.

The output classes considered for this experiment were 0 and 1. The output value 0 represents no defect condition, and the output value 1 indicates that the IC under consideration is defected. The output of the neural network may not be exactly equal to 1 or 0. Due to this, we assign the class based on how close the output is to any of the values. For example, if the output is 0.9, it is assigned class 1. If the output is 0.1, it is assigned class 0.

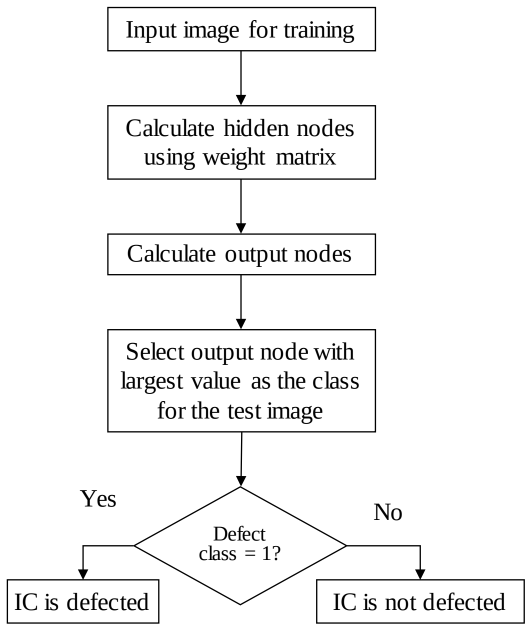

Input image for training
Calculate hidden nodes using weight matrix
Calculate output nodes
Select output node with largest value as the class for the test image
Yes
No
Defect class = 1?
IC is defected
IC is not defected

Figure 8. Logical flowchart for artificial neural network testing.

3.2.4. Defect Detection Results Using ANN

The backpropagation ANN was implemented using MATLAB platform. The training was done using two images with and without defects for 10 iterations each. Figure 9 shows the images used for training. The ANN was tested for detecting the counterfeit ICs.

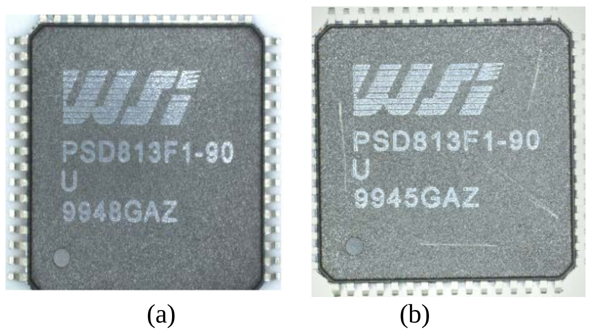


Figure 9. Images used for training a) IC without defect b) IC with scratches.

The original image had more than 4 million pixels. This means we need 4 million input nodes, where we have only two output nodes. By employing just one hidden layer, we would lose information during the training process. This was resolved by reducing the size of the images to a resolution of 512×512 during the preprocessing stage. The number of input nodes for the neural network was then reduced to 262,144.

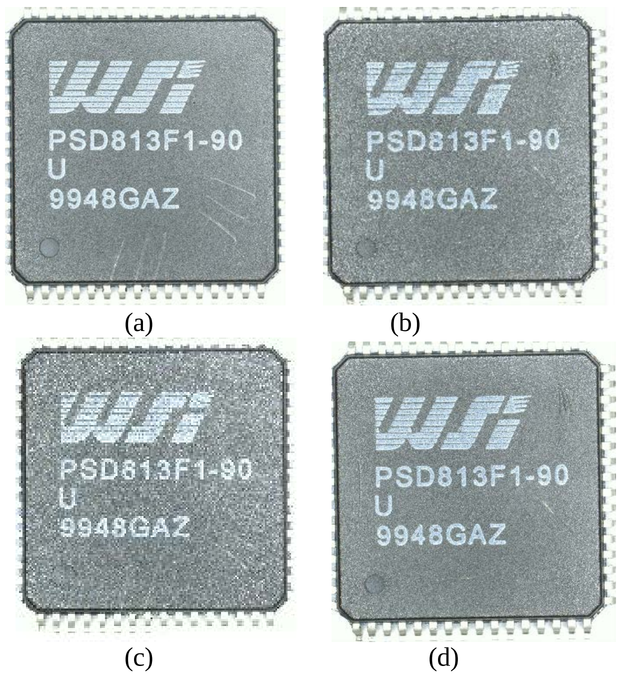


Figure 10. (a)–(d) IC images with scratch defect used for testing.

The number of hidden nodes were kept as a variable to be adjusted depending on the results. One hundred hidden nodes gave us good results. As we increase the number of nodes, the computational time rises. For our experiments, we set the number of hidden nodes as 100. The learning rate parameter was varied between 0.01 and 2. The learning rate of 0.1 gave us the minimum number of iterations required for training. The number of IC images used for testing were four, out of which four were counterfeit. The weight matrices were initialized using random functions available in MATLAB between −0.5 to 0.5.

The images used for testing are shown in Figure 10. ICs (a) and (c) have scratch defects and are correctly detected by the ANN.

We could not test ANN on more samples in the first phase of this project, since a new round of imaging is required for acquiring standard images of ICs as seen in Figures 9 and 10. But this technique can be applied on a large number of images and samples as long as the quality of the image is enough to capture pixel contrast differences between normal and defected regions. In fact, the more images that are provided for training the algorithm, the more accurate the testing-phase results will be.

4. Conclusion and Future Work

In this paper, we described a counterfeit database that has been developed to improve recordkeeping of counterfeit incidents, support education and increase awareness about counterfeit ICs/defects, and provide resources for research in automated counterfeit IC and defect detection. As an exemplary instance, data was used to investigate image processing and neural network algorithms for scratch detection. Image processing can successfully detect ICs with scratches, but fails when the scratches are not highly visible or their color is not significantly different from that of the IC surface. Another current limitation is that the exact same algorithm will not apply for other defects. However, the artificial neural network should be more amenable to other types of defects in the future. In future work, we will extend the existing ANN technique to detect other physical defects and to data from additional sources (X-ray, SEM, etc.).

6. Acknowledgments

The authors would like to acknowledge support from National Science Foundation (grants CCF-1423282, CCF-1559772, and CNS-1558516) for this work.

7. References

[1]. U. Guin D. DiMase and M. Tehranipoor, "Counterfeit Integrated Circuits: Detection, Avoidance, and the Challenges Ahead," Journal of Electronic Testing: Theory and Applications (JETTA), vol. 30, no. 1, pp. 9-23, 2014.

[2]. U. Guin, K. Huang, D. DiMase, J. Carulli, M. Tehranipoor, and Y. Makris, "Counterfeit Integrated Circuits: A Rising Threat in the Global Semiconductor Supply Chain," Proceedings of the IEEE, vol. 102, no. 8, pp. 1207–1228, 2014.

[3]. U. Guin, D. Forte and M. Tehranipoor, "Anti-counterfeit techniques: From design to resign", Proc. 14th Int. Workshop Microprocessor Test Verification (MTV), pp. 89-96.

[4]. U. Guin D. Forte and M. Tehranipoor, "A Comprehensive Framework for Counterfeit Defect Coverage Analysis and Detection Assessment", Journal of Electronic Testing (JETTA), Volume 30, Issue 1, pp 25-40, February 2014.

[5]. R. Karri, J. Rajendran, K. Rosenfeld and M. Tehranipoor, "Trustworthy Hardware: Identifying and Classifying Hardware Trojans", Computer, vol. 43, no. 10, pp. 39-46, 2010.

[6]. S. Haykin, S. Simon, "Neural Networks and Learning Machines", Vol. 3. Upper Saddle River, NJ, USA: Pearson, 2009.

[7]. E. Ardizzone, H. Dindo, and G. Mazzola, "Multidirectional Scratch Detection and Restoration in Digitized Old Images," Eurasip Journal on Image and Video Processing, vol. 2010, no. June 2015, 2010.

[8]. K. M. Buddhiraju and I. A. Rizvi, "Comparison of CBF, ANN and SVM Classifiers for Object-based Classification of High Resolution Satellite Images", Proc. IGARSS, pp. 40-43, 2010.

[9]. X. Zhang, K. Xiao, and M. Tehranipoor, "Path-Delay Fingerprinting for Identification of Recovered ICs" in Proc. IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS), 2012.

[10].S. Quadir, J. Chen, D. Forte, N. Asadizanjani, S. Shahbazmohamadi, L. Wang, J. Chandy, and M. Tehranipoor. "A survey on chip to system reverse engineering." *ACM Journal on Emerging Technologies in Computing Systems (JETC)* 13, no. 1 (2016): 6.

[11].N. Asadizanjani, S. Shahbazmohamadi, M. Tehranipoor, and D. Forte. "Non-destructive PCB Reverse Engineering Using X-ray Micro Computed Tomography." In *41st International Symposium for Testing and Failure Analysis (November 1–5, 2015)*. Asm, 2015.

[12].www.counterfeit-ic.org

[13]. Measuring the Magnitude of Global Counterfeiting available online "http://www.theglobalipcenter.com/wp-content/themes/gipc/map-index/assets/pdf/2016/GlobalCounterfeiting_Report.pdf".

[14]. N. Asadizanjani, "3D Imaging and Investigation of Failure and Deformation in Thermal Barrier Coatings Using Computed X-ray Tomography." (doctoral dissertation) , 2014.

[15]. N. Asadizanjani, and E. H. Jordan. "Optimization and Development of X-ray Microscopy Technique for Investigation of Thermal Barrier Coating." Processing and Properties of Advanced Ceramics and Composites VII: Ceramic Transactions, Volume 252, pp. 425-440, 2015.