Artificial Neural Network Based Detection and Diagnosis of Plasma-Etch Faults

Shumeet Baluja & Roy A. Maxion
baluja@cs.cmu.edu, maxion@cs.cmu.edu
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA

(412) 268-2339
FAX (412) 268-5571

Abstract

The plasma-etch process is one of many steps in the fabrication of semiconductor wafers. Currently, fault-detection/diagnosis for this process is done primarily by visual inspection of graphically displayed process data. By observing these data, experienced technicians can detect and classify many types of faults. The tediousness and intrinsic human unreliability of this method, as well as the high cost of mistakes, makes automation attractive. In this paper, five artificial neural network approaches for detecting and diagnosing four common plasma-etch fault conditions are examined. The data used for training and testing the networks were collected during a 162 day period, in which over 46,000 wafers were etched. The best accuracy achieved in this study is approximately 98.7% correct fault-detection for the four fault types, 100% correct fault classification, and a 2.3% false alarm rate. The five neural-based approaches are described in detail, and results are given for each approach.

Keywords

Plasma-Etch, Fault Detection, Fault Diagnosis, Artificial Neural Networks, Prediction, Selective Attention

1. INTRODUCTION

Plasma etch is one of many steps in the fabrication of semiconductor wafers [Russ, 1985]. Currently, fault-detection/diagnosis for this process is done primarily by visual inspection of graphically displayed process data. The plasma chamber is fitted with sensors. Wafer by wafer, time-series waveforms depicting the etch process are displayed on a computer screen. By observing these waveforms, experienced technicians can learn to detect and classify many types of faults. Because the detection and diagnosis of faults by this method is an intrinsically unreliable human endeavor [Maxion, 1996][Pope, 1986], and because of the high cost of a mistake,[1] automation of this process would make an important contribution toward increasing the accuracy, reliability, and cost-effectiveness of this procedure.

In this paper, artificial neural networks (ANNs) are used for detecting and diagnosing fault conditions in the plasma-etch step by analyzing data gathered from etch-tool sensors. The automation of this process is complicated because of the nonstationary nature of the etcher. As the etcher produces more and more product, contaminant buildup and electrode wear alter the characteristics of the etch process, resulting in constantly changing waveforms. Over time, the waveforms can change significantly, thereby making exact template matches impossible.

In this paper, several methods for the detection and classification of faults using ANNs are presented. Two common applications of artificial neural networks are: (1) classification of inputs into a set of classes [Fogelman-Soulie, 1995]; and (2) forecasting future values of observances [Ungar, 1995]. The first approach follows the standard classification techniques often used with ANNs. The second approach combines prediction techniques with classification techniques; this is possible because of the sequential/temporal nature of the plasma-etch process. Both approaches will be described in detail, and experiments using each approach will be presented.

In the next section, the specifics of the experiment are given and an introduction to the artificial neural network (ANN) methods is provided. In Section 3, five ANN-based approaches to fault detection and classification are discussed. Section 4 explores in detail the most successful approaches, which are based on prediction and classification, and explains why these approaches work better than the others. Finally, Section 5 concludes the report, and provides suggestions for future research.

2. EXPERIMENT SPECIFICS

In this section, the specifics of the experiment are given. Details about the plasma etcher, the types of faults to be detected, and information about the gathering and usage of the data are provided.

2.1 The Plasma Etcher

The etcher used in these experiments (a Lam 490 from Lam Research, Fremont, CA) is in active use in a production wafer-fabrication plant. This etcher is a single-wafer, plasma-etch tool system designed for use in mass production of commercial semiconductor devices. There are over 1500 of these tools installed in integrated-circuit manufacturing facilities worldwide. The LAM-490 etcher used in the present work is employed primarily for etching four-inch polysilicon wafers using a chlorine/helium chemistry. Wafers are loaded into the front of the etch tool, passing singly through the process chamber where reactive gases, ignited by an RF plasma, etch the film on the front of the wafer to define circuit elements.

Two important attributes about the plasma-etch process are of concern in this study. First, as each wafer is etched, a small amount of contaminant is deposited onto the sidewalls of the etch chamber. Second, as the plasma is repeatedly ignited, the electrodes in the chamber erode slightly. The effects of contamination/ erosion become significant over the etching of many wafers. The etcher is cleaned regularly (termed the clean-cycle), and the electrodes are replaced regularly (termed the electrode cycle). The relevance of the contamination and degradation for this experiment is that the waveforms for the wafers, even those that are fault free, will change over time. Therefore, any successful fault-detection method must be able to account for these drifting effects.

2.2 Data

The data-collection phase of this experiment began on October 25, 1994, and continued until April 4, 1995. During this period, eight clean cycles were completed - an average of about 20 days/cycle. The total number of wafers etched during this time was 46,638 - with an average of 5830 wafers per clean cycle. The wafers were loaded in batches to be etched, termed “runs”, which contained from 1-25 wafers. Wafers are etched individually; data were gathered from multiple sensors for each etch. For the experiments conducted here, only a single sensor was used, which measured the intensity of light emitted from the plasma at the 520nm wavelength. Each etch was sampled once a second, providing approximately 140 samples per wafer.

In order to gather fault-wafer data, faulted wafers were injected periodically into the production stream. Each group of fault wafers contained one wafer of each of the four types of faults to be detected. These injections occurred approximately every 300 wafers etched. Because of the nonstationary aspects of this process, the fault wafers injected early in the clean cycle will have different waveforms than those injected later. For this study, only one clean cycle was examined, in which there were 4052 wafers etched without faults, and 56 fault wafers injected. The problem of detecting and classifying faults is exacerbated by two factors. First, there is a very small sample of fault-wafers. This problem is partially addressed by methods which use the differences between the current wafers and previous wafers to make classifications of the current wafer. For example, if a fault-injection run was made at *time (a)*, realistic training examples can be created by pairing each wafer in *a* with each of the wafers in the run made at *time (a-1).* In fact, these training examples are perhaps the most important, since faults at *time (a)* are hard to detect when they appear close to other wafers etched around the same time. For methods which are not based on comparisons, the issue of limited training data is harder to overcome. A second difficulty is that some of the wafers labeled "no-fault" may contain faults. In the preliminary analysis of the data, it was found that these unlabeled faults are mostly of different types than the four which are to be detected in this study. Complete information about the number of these other faults is not currently available. However, once the faults are labeled, the methods described in this paper can be applied to the detection of these faults in addition to the four faults currently examined.

2.3 Faults to Be Detected

In the wafer etch process, some of the essential steps include adding a photoresist coating, exposing the wafer, and developing the wafer. The four faults to be detected are described below. They were used because of their occurrence in manufacturing practice. All four fault types typically result from human error. Examples of the waveforms for each of the four fault types are given in Figure 1:

1. *Unexposed or undeveloped*: a wafer which was either unexposed or undeveloped during the photo process.

2. *Missing resist*: a wafer which was skipped in the photo-spin step.
3. *Previously etched*: a wafer which was etched a second time.
4. *Wrong recipe*: a wafer which was etched using the wrong process recipe.

[[[NOTE TO EDITOR: INSERT FIGURE 1 HERE]]]

2.4 Artificial Neural Network Training & Testing Specifics

All of the artificial neural networks presented in this paper were trained with the standard error backpropagation gradient-descent learning algorithm. Five experimental methods and their associated network architectures are described. The learning rate and momentum parameters were held constant across all tests. Standard sum-of-squares error (SSE) was used as the training error criterion. All performances on the testing set were measured in terms of classification accuracy, not SSE.

The total number of samples provided for this experiment was 4052 no-fault wafers and 56 fault wafers - 14 wafers of each type of fault. The training set is very heavily biased towards the no-fault wafers. To compensate for this, pattern presentations to the ANNs were weighted towards the fault wafers. This helps ensure that the network does not develop a bias towards the no-fault wafers due to the number of times the no-fault wafers were presented.

For training the networks, the first 36 fault wafers were used. This left the last 20 fault wafers for performing tests. As mentioned earlier, to test the detection algorithms accurately, the test faults must be found within the surrounding batch of no-fault wafers. It should be noted that the discussion of “surrounding wafers” only makes sense if *state of the machine* is considered. If, however, detection/classification can be done without considering the state of the machine (for example, by just using the waveform of the current wafer as input, and outputting the classification), the notion of “surrounding wafers” is not applicable. Examples of both types of classifiers will be described in the next section. In the cases in which machine state is considered, the 36 fault wafers used for training corresponded to the first 2641 no-fault wafers. The remaining 1411 no-fault wafers were used for testing. A more detailed description of the testing procedure will be provided in the next section.

3. ARTIFICIAL NEURAL NETWORK BASED METHODS FOR FAULT DETECTION & CLASSIFICATION

In this section, five artificial neural network (ANN) based methods for detecting the four faults enumerated in Section 2.3 are presented. Numerous network architectures were attempted for all of the methods. The best architecture found for each method is used for comparison.

In these experiments, there are two criteria which must be minimized: the number of missed faults and the number of false alarms (no-fault wafers which are erroneously identified as faults). Either of these criteria can be trivially optimized. The performance of each method is reported when the sum of the percentage of missed faults and the percentage of false alarms is minimized.

3.1 Method A: Base-Line

To provide a base-line for comparison to the other methods, a simple fully-connected single hidden layer network was used. This network had 140 input units (the complete waveform signature of a single wafer) and 5 outputs, representing no-fault & faults 1-4. The wafer's waveform is used as input, and the output with the highest value is considered to be the network's classification of the wafer. Although this network was able to classify correctly all of the 20 test-set faults, it often misclassified no-fault wafers into one of the fault categories. The most frequent misclassification occurred in fault-class-2. As will be seen throughout this paper, the no-fault wafers and the fault-class-2 wafers were often confused. The complete confusion matrix is given in Table 1.

[[[NOTE TO EDITOR: INSERT TABLE 1 HERE]]]

Due to the very high number of false alarms (53%), a system which automatically monitored the etcher based on this simple method would not be usable. The number of false alarms can be improved if we are willing to allow more missed faults. For example, if we allow a single missed fault, the false-alarm rate decreases to 47%. As the number of allowed missed faults increases, the number of false-alarm errors decreases.

For comparison, simpler methods were also attempted. Instead of using an ANN with a hidden layer, a simple perceptron was also tried. This resulted in a 20% miss rate and approximately a 40% false-alarm rate. A simple fault detection method (not classification) is to measure the difference between the average "no-fault" vector and the input vector to be classified. By varying the threshold, the system can be made more or less conservative, with respect to the number of faults signalled. Two hundred uniformly spread threshold settings, between 0 and the maximum difference between the average no-fault vector and the fault vectors, were used. The smallest error (missed rate + false-alarm rate) was a 14% miss rate and 41% false-alarm rate.

3.2 Method B: Using the Previous Wafer for Comparison - Incorporating Machine-State Information

As mentioned earlier, one of the known problems with plasma-etch tools is process degradation from chamber contamination and electrode erosion. Because these data were collected over many days, in which many etches were made, the changing conditions of the etcher must be considered. One method for incorporating state information is presented in this section.

The current state of the etcher can be considered by using the signature of the last known wafer which did not contain any of the four faults. This signature can be used as a basis with which to compare the current wafer's signature. Using the differences between the current wafer's signature and the previous wafer's signature provides a simple way to incorporate state into the decision process. An implementation of this idea is as follows: the classification neural network is given both signatures as input. Further, in order to accentuate the differences between the two input wafers, the point-by-point differences are also input. Although using the differences is strictly not necessary, since the ANN should be able to compute these differences if required, it was found to improve learning speed and accuracy. The drawback of this method is that it triples the number of inputs.

The first network architecture used was a standard, fully-connected network, similar to the one described in Method A, with 3x140 input units. However, this network's performance was poorer than that of many others; the best architecture found is described here. The architecture used was similar to those often used in neural network vision applications [Rowley *et al*., 1996]. This network architecture used a separate hidden unit for each set of 3x5 input units, as shown in Figure 2. The motivation for this architecture was to allow each hidden unit to specialize in analyzing only small portions of the input. The hope was that there would be recognizable features in these small portions. Similar architectures have been used in speech recognition and visual zip-code recognition. Although not needed here, these architectures are sometimes used in conjunction with weight sharing to promote translation invariance of feature detectors [Waibel *et. al,* 1989][Le Cun *et. al,* 1989].

[[[NOTE TO EDITOR: INSERT FIGURE 2 HERE]]]

As described previously, in Section 2.2, a fault could have been injected into any of the 25 positions in the run. Since training this network requires pairs of wafers as input, examples were created by pairing all members in the run made previous to the fault run with those in the fault run. Therefore, unlike Method A, in which only 20 fault wafers were tested (since no state information was used), many more pairs can be tested using Method B. The total test set size is 396 fault-wafer pairs[2], as well as the 1411 no-fault wafer pairs. This method achieved a miss rate of 18.4% and a false-alarm rate of 5.6%. A more detailed breakdown of the results is given in Table 2.

[[[NOTE TO EDITOR: INSERT TABLE 2 HERE]]]

Tests similar to the ones described in the previous section were also conducted here. Instead of using an ANN with a hidden layer, a perceptron was tried. This resulted in degraded performance. Also, instead of using an ANN-based method, detection of faults based on the magnitude of the summed point-by-point differences between the previous no-fault wafer and the wafer to be classified was attempted. Again, 200 threshold values were used, uniformly spanning the smallest distance between input pairs and the largest difference between input pairs. For the same percentage of missed faults (18.4%), the false-alarm rate was approximately 10%.

3.3 Method C: Using State & Individual Networks

In this set of experiments, rather than using a single network to predict the fault type of a particular wafer, as described in the previous section, five networks were used: four to detect each of the four fault types, and one to detect wafers which did not contain these faults. Each network was trained to produce an output of +1.0 only if the input wafer was of its assigned fault type. If the wafer was of a different fault-type (including no-fault), the network was trained to output a -1.0. The inputs were the same as used in Method B. There was only 1 output per network. Each network's output was interpreted as classifying the input wafer as belonging to its class if the output was a value greater than 0.0. If the value was less than 0.0, the output was interpreted as a classification outside the network's trained class. If more than one network classified the wafer to be in its class, a method for arbitration would be required. However, in the tests performed, this situation never arose.

The four networks trained to detect fault types 1-4 performed well. However, the network trained to recognize no-fault wafers often did not recognize no-fault wafers (20% error), and often responded positively to wafers which had faults (39% error). Nonetheless, because only 5 classes need to be discriminated, using 5 networks is redundant. Because of the accuracy of the networks 1-4, the system can still work without the no-fault recognition network. If networks 1-4 all respond negatively to the input wafer, then the wafer is classified as a no-fault wafer. This method reduces the percentage of missed faults to 12.1%, and reduces the percentage of false alarms to 1.5%. More details on the performance of each network are given in Table 3. Note that a confusion matrix cannot be given in this case as each network only indicates whether it believes each example is or is not a member of the class it was trained to recognize. The key to the table is given below:

- *Correctly Classified Faults*: These wafers have faults of the type the network was trained to recognize, and was successfully able to recognize. (Example: Network 1 successfully recognizes Fault 1.)
- *Misses:* These wafers have faults of the type the network was trained to recognize and was *not* able to recognize. In practice, this type of error may be the most damaging. (Example: Network 1 does not recognize Fault 1.)
- *Correct Other Faults:* These are wafers manifesting faults other than the ones the network was trained to recognize. This is the number of “other-faults” to which the network correctly responded negatively. (Example: Network 1 successfully rejects Fault 2.)
- *Misclassified Faults:* These are the same types of wafers as above, fault wafers of a different kind than the network was trained to detect. This is the number of wafers the network mistakenly took to belong to the class on which it was trained. (Example: Network 1 erroneously accepts Fault 2 as Fault 1.)
- *Correct No-Faults:* This is the number of no-fault wafers to which the network responded correctly. (Example: Network 1 successfully rejects no-fault.)
- *False Alarms:* This is the number of no-fault wafers that were classified as faulty. (Example: Network 1 erroneously accepts no-fault as Fault 1.)

[[[NOTE TO EDITOR: INSERT TABLE 3 HERE]]]

3.4 Method D: Computing Expectations

When humans monitor the etch tool visually, or when they perform many other types of visual scene analysis, one method of eliminating extraneous, or irrelevant, information is through filtering their input based upon expectation. In [Baluja, 1996] [Baluja & Pomerleau, 1994 & 1995], expectation was used in an artificial neural network-based system to filter noise and irrelevant features from an input visual scene. This was accomplished by suppressing the portions of the scene which did not match a computed expectation of the next scene. In the domain explored here, however, the portions of the input which are anomalous are important. Since the task being performed requires anomaly detection, expectation can be used to *accentuate the differences* between the expected inputs and the actual inputs. More details are given below; these ideas are returned to in Section 4. Using expectations for fault detection has also been explored in [Maxion, 1990].

Expectation is computed as follows: Given the waveform signature of wafer(T-1), an expectation of wafer(T) can be formed. This expectation is formed using a second neural network. The input to this second neural network is the waveform signature of wafer(T-1); the output is the expectation of the signature of wafer(T). This network is trained in a supervised manner by using the wafer's waveforms in sequential order. The target output for each example is the next wafer in sequence. The target output is the length of the input (140 units). This process is related to auto-encoding networks [Cottrell, 1990][Kramer, 1991]; however, in the present method, the *next* time-step is predicted. Somewhat similar prediction methods have been explored in [Tebelskis, 1995]. The network is trained only to predict the waveform of the next no-fault wafer; the network is not trained to predict fault wafers.

The training is augmented as follows: rather than only predicting the waveform for the next wafer, some training examples with input wafer(T-1) are given the waveform of wafer(T+1) (rather than wafer(T)) as the desired output. This helps to ensure that the network does not memorize specific transitions. The prediction network architecture is shown in Figure 3.

[[[NOTE TO EDITOR: INSERT FIGURE 3 HERE]]]

Based upon the expectation of the waveform of wafer(T) (termed expectation(T)), wafer(T), and the differences between expectation(T) and wafer(T), a classification of the wafer is made. The classification network is the same architecture network as the one used in Method B. However, in Method B, wafer(T-1) was used as input, while in this method expectation(T) is used as input. This method slightly reduces the percentage of missed faults to 9.3%, but there is an increase in the percentage of false alarms to 10.1%. The results for this method are shown in Table 4.

[[[NOTE TO EDITOR: INSERT TABLE 4 HERE]]]

3.5 Method E: Computing Expectations and Individual Networks

In Method E, five networks were used in a technique similar to Method C. Each network was assigned a particular fault type. Each network was trained to produce an output of +1.0 only if the input wafer was of its assigned fault type. If the wafer was of a different fault-type (including no-fault) the network was trained to output a -1.0. The inputs were the same as Method D (wafer (T), expectation (T), difference between expectation (T) and wafer (T)), and there was only 1 output. The strengths and weaknesses of this method were the same as for Method C. All of the fault detection networks worked well. However, the network that was trained to detect no-fault wafers often did not recognize no-fault wafers (26% error), and often responded positively to wafers which had faults (33% error). As was done in Method C, since four of the five networks worked well, accurate class discriminations could be made. The individual network architecture is shown in Figure 4. This system had the best performance overall; the percentage of missed faults was lowered to 1.3%, while the percentage of false alarms remained low at 2.3%. The results of the system without using the no-fault detection network are shown in Table 5.

[[[NOTE TO EDITOR: INSERT FIGURE 4 HERE]]]

[[[NOTE TO EDITOR: INSERT TABLE 5 HERE]]]

In Figure 5, the network outputs for each class of wafers are shown. These diagrams show how each network responds to each wafer. For example, in the graph "Network Simulation of Fault Type 1", networks 2,3 & 4 all responded with low values (less than 0.0). However, network-1 responded with values all greater than 0.0. In the graph "Network Simulation of Fault Type 2", network-2 responded with the highest values. In three of the 99 testing examples shown, network-2 responded with values less than 0.0, indicating missed detects. Also, on the last example, network-4 almost responded with a value greater than 0.0. In the graph "Network Simulation of No-Fault", the responses of the 4 networks to the 1411 no-fault testing wafers are shown. Ideally, each network should respond with a negative value, since these wafers do not belong in any of the classes the networks are trained to detect. Network-2 misclassified no-fault wafers as belonging to its class 27 times; network-4 made similar mistakes 5 times.

[[[NOTE TO EDITOR: INSERT FIGURE 5 HERE]]]

3.6 Summary of Empirical Results

The performance of five ANN-based methods for detection and classification of four faults was evaluated. A schematic diagram of these approaches is shown in Figure 6. The last four of these incorporate state information. The first method, the base-line, has a different scale on which to measure results because it does not take state information into account. Because of its very high false-alarm rate, it cannot be used in practice. In Table 6, the last four methods are compared in terms of misses and false alarms.

[[[NOTE TO EDITOR: INSERT TABLE 6 HERE]]]

[[[NOTE TO EDITOR: INSERT FIGURE 6 HERE]]]

In this study, a single large network did not provide results as good as those provided by separate networks that recognize each fault type. The expectation-based methods provide better results, in terms of the total number of missed faults, than their counterparts which used state information in the form of the previous no-fault wafer. There is a small degradation in the number of false alarms; nonetheless, the trade-off still heavily favors the individual-network expectation-based methods because of the fewer missed faults. Although the number of missed faults can also be reduced to this level for the other methods, the number of false-alarms increases beyond the level in Method E. It should also be noted that the differences between the number of false alarms in Method C and Method E could be due to the noise in the testing set (i.e., errors in labeling the no-fault wafers), as described in section 2.

4. WHY USING EXPECTATION IMPROVES PERFORMANCE

The use of ANN-based expectation for filtering has been studied in depth in [Baluja, 1996] [Baluja & Pomerleau, 1994 & 1995]. In those studies, expectation-based focus-of-attention was explored in the context of ANN vision-based systems for vehicle-road-following and vehicle-roadway-departure warning. The system was designed to track visually the road and the lane markings. In the expectation-based system, in each step, the expectation of the next-input scene was computed. These expected inputs were used to filter the actual inputs seen in the next time step. The filtering was done in two manners. If unexpected or noise features were to be removed, anything which did not match expectations was removed from the input image. This filtering removed extraneous features like old lane-markings, guardrails, or distractions in the periphery. This type of filtering was used to pre-process the inputs to the road-following module, which is responsible for finding the road's location in the image. For an obstacle-avoidance module, input filtered in the above manner is useless, since the filtering can also remove cars or other unexpected, but potentially important, features from the visual input. The second use of expectation is *to emphasize the unexpected by de-emphasizing the portions of the input which match closely* that which was expected. In this mode, the unexpected obstacles in the scene are emphasized. Input filtered in this manner can be used as input into a separate network or driving module which performs object detection/recognition and determines the appropriate action for avoiding the obstacle. The expectation-based system used in this study is constructed from the second form of filtering.

For the task of wafer anomaly detection, the important portions of the input are those that *do not match expectation, since the expectation was designed to predict only no-fault wafers*. One method of emphasizing these portions is by using the differences between the expected and actual waveforms as inputs into the network. This difference vector emphasizes more heavily the portions in which the difference between the expected and actual input is large.

The use of expectation provides a means to incorporate machine state information into the prediction. However, Methods B and C also provided state by explicitly using the previous no-fault wafer's waveform. A reason that expectation is able to outperform using the previous wafer's signature is that expectation-based methods are less sensitive to the previous wafer's waveform than methods which explicitly use the previous wafer's waveform. Although the prediction of the next wafer's waveform is based upon the input wafer's signature, using the prediction often mitigates the effects of small variations in the original input wafer, thus providing a cleaner training signal. Methods which explicitly use the previous wafer's waveform for classification are subject to these variations in their inputs. See Figure 7.

[[[NOTE TO EDITOR: INSERT FIGURE 7 HERE]]]

As an example of the phenomenon described above, see Figure 8. In Figure 8A, note that wafer #121 has an uncharacteristic hump starting around input 70. When wafer #120 is used as input to the prediction network, it predicts a wafer without the hump. Further, when wafer #121 is used as input to the prediction network, the predicted wafer is ***not*** similar to wafer #121; instead, the effects of the hump in #121 are mitigated. The expectation of wafer #122, which was derived from wafer #121, was made fairly accurately. The same types of effects are shown in Figure 8B: wafer #3654 has a small "notch" missing in its left top (inputs 40-55); the predicted wafer does not, and the next prediction (of wafer #3655) is only slightly affected by this notch. In summary, using expectation de-emphasizes the small variations in the waveforms; therefore, the classification network is given cleaner training data. *It should be noted that in the training phase, expectations are never formed from wafers which contain faults; expectations are only formed from wafers which do not contain any of the four faults. In testing, expectations are never formed from wafers which are classified as faulty.* Therefore, only anomalies in the wafers which have been classified as "not containing any of the four faults to be detected" are mitigated through the prediction process.

[[[NOTE TO EDITOR: INSERT FIGURE 8 HERE]]]

5. CONCLUSIONS & FUTURE DIRECTIONS

Due to the difficulty of controlling the plasma-etch step, and because the etch step is typically repeated several times for each wafer, attention has been devoted to automatically monitoring the plasma-etch chamber for detection and diagnosis of faults. Currently, monitoring is done by humans visually inspecting the etch waveforms as they appear on computer screens. Since plasma etching is a commonly used process, and because of the unreliability of human detection and diagnosis, automating this step is an important issue.

This paper has presented five artificial neural network based methods for detecting and classifying faults. Because of the nonstationary conditions of the plasma etcher, which cause waveforms to change over time, the state of the etch machine must be considered when making detections/classifications. In this study, two methods of providing the ANN with the machine state were explored. The first method used the last previous no-fault wafer, in addition to the current wafer, as input into the ANN. The second used an expectation of the current input. It was found that the expectation-based model improved the detection and classification score. In this study, a second ANN was used for computing the expectation of the next wafer's signature. However, other neural architectures, such as recurrent networks, can be used to maintain internal state in conjunction with output classifications [Jordan, 1989][Elman, 1990].

There are four immediate directions for future work with ANN-based fault detection. First, a measure of the network's reliability is needed. [Pomerleau, 1994] has proposed a method which is based upon determining how closely the network's internal units, which have been trained to perform the desired task, represent the inputs. If the network has represented the inputs well, the reliability is assumed to be high. This method can be used to determine how certain the networks are of their decisions.

A second direction for future work is in building redundancy into the fault-detection system. For example, in the systems which use four individual networks for detecting faults, if an additional network can be trained to detect no-fault wafers, it could provide an additional test to ensure that a fault does not escape the system.

A third direction for future work is using a moving average of the previous several no-fault wafer waveforms for comparison with each new waveform. This may yield insights and extensions to the expectation-based methods employed here.

Finally, perhaps the most important direction for future work is extending this work to the detection of novel fault conditions. Currently, only four labeled fault conditions are detected. However, in practice, the system should be able to detect anomalies on which the system has not been explicitly trained. This may necessitate the decoupling of the fault detection and diagnosis procedures. Decoupling the two procedures provides the benefit of detecting faults that were neither explicitly labeled nor available in the training set.

ANNs, as well as other machine learning methods, have been applied to similar problems of novelty detection [Japkowicz *et al*., 1995][Maxion, 1993]. One such system currently under investigation in this domain is the Harbinger system [Maxion, 1996]; this system uses rules derived from decision-tree methods. In the preliminary experiments applying Harbinger to this domain, the results appear promising. A way to use the ANN-based methods presented here in conjunction with the Harbinger system is to use Harbinger for the detection of faults and ANNs for the classification of detected faults.

6. ACKNOWLEDGMENTS

It is a pleasure to acknowledge the contributions of several colleagues and friends: Carl Almgren, consultant engineer at Symbios Logic, fabricated the fault-injection wafers and engineered the fabrication process so that it could accommodate live faults during routine production processes; Philip Syme, research programmer at Carnegie Mellon University, calibrated and assembled the test waveform data sets with meticulous care, and was enormously helpful with data analysis; Kaari Flagstad and Dean Pomerleau graciously commented on several drafts of the paper. Any remaining faults are solely those of the authors.

This work was started while the first author was supported by a National Science Foundation Graduate Fellowship. He is currently supported by a graduate student fellowship from the National Aeronautics and Space Administration (NASA), administered by the Lyndon B. Johnson Space Center. The second author was partially supported by National Science Foundation grant # IRI-9224544 under the IRIS Program for Research on Scientific Databases. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing official policies, either expressed or implied, of NSF, NASA, or the U.S. Government.

7. REFERENCES

Baluja, S. 1996. *Expectation-Based Selective Attention*, Ph.D. Thesis. Department of Computer Science, Carnegie Mellon University. Expected July, 1996.

Baluja, S. and Pomerleau, D.A. 1995. Using the Representation in a Neural Network's Hidden Layer for Task-Specific Focus of Attention, in *The International Joint Conference on Artificial Intelligence 1995* (IJCAI-95), edited by Mellish, C., Montreal, Canada. IJCAII & Morgan Kaufmann. San Mateo, CA., 133-139.

Baluja, S. and Pomerleau, D.A. 1994. Using a Saliency Map for Active Spatial Selective Attention: Implementation & Initial Results, in: *Advances in Neural Information Processing Systems (NIPS) 7*, edited by Tesauro G., Touretzky, D. and Leen, T.K., MIT Press, Cambridge MA, 1995. 451-458.

Cottrell, G. 1990. Extracting Features from Faces Using Compression Networks, in: *Connectionist Models: Proceedings of the 1990 Summer School*, edited by Touretzky, D., Elman, J., Sejnowski, T., and Hinton, G.,. Morgan Kaufmann Publishers, San Mateo, CA. 328-337.

Elman, J.L. 1990. Finding Structure in Time, *Cognitive Science* **14**, 179-211.

Fogelman-Soulie, F. 1995. Applications of Neural Networks, in: *The Handbook of Brain Theory and Neural Networks,* edited by Arbib, M., MIT Press. Cambridge, MA, 94-98.

Hertz, J. Krogh, A., Palmer R. 1989. *Introduction to the Theory of Neural Computation.* Addison-Wesley. Reading, MA.

Japkowicz, N., Myers, C., Gluck, M. 1995. A Novelty Approach to Classification, in *The International Joint Conference on Artificial Intelligence 1995* (IJCAI-95), edited by Mellish, C., Montreal, Canada. IJCAII & Morgan Kaufmann. San Mateo, CA., 518-523.

Jordan, M.I. 1989. Serial Order: A Parallel, Distributed Processing Approach, in *Advances in Connectionist Theory: Speech*, edited by Elman, J.L. and Rumelhart, D.E. Hillsdale: Erlbaum.

Kramer, M. 1991. Nonlinear Principal Component Analysis using Autoassociative Neural Networks, *AIChE Journal* **37:2**, 233-242.

Le Cun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W. and Jackel, L.D. 1989. Backpropagation Applied to Handwritten Zip Code Recognition, *Neural Computation* **1**, 541-551.

Maxion, R. 1990. Towards Diagnosis as an Emergent Behavior in a Network Ecosystem, *Physica D* **42:2-3**, 66-84.

Maxion, R. and Olszewski, R. 1993. Detection and Discrimination of Injected Network Faults, in: *23rd Annual International Symposium on Fault-Tolerant Computing*, IEEE Computer Society Press, 198-207.

Maxion, Roy, *et al.* 1996. Real-World Detection and Diagnosis of Plasma-Etch Anomalies. *In Preparation.*

Pomerleau, D.A. 1992. *Neural Network Perception for Mobile Robot Guidance*, Kluwer Academic Publishers, Boston, MA.

Pomerleau, D. A. 1994. Reliability Estimation for Neural Network Based Autonomous Driving, in: *Robotics and Autonomous Systems* **12**, 113-119.

Pope, R.H. 1986. Human Performance: Improvement from Human Reliability Assessment, in: *Reliability Data Collection and Use in Risk and Availability: Proceedings of the 5th EureData Conference*, edited by Wingender, H.J., Springer-Verlag, Berlin, 455-465.

Rowley, H., Baluja, S., Kanade, T. 1996. Neural Network Based Human Face Detection, to Appear in: *Computer Vision and Pattern Recognition (CVPR) '96.*

Russ, M. 1985. *Plasma Etching in Semiconductor Fabrication*. Elsevier: Amsterdam.

Tebelskis, J. 1995. *Speech Recognition Using Neural Networks*. Ph.D. Thesis. Department of Computer Science, Carnegie Mellon University.

Waibel, A., Hazanazawa, T., Hinton, G., Shikano, K., and Lang, J. 1989. Phoneme Recognition using Time-Delay Neural Networks, in *Readings in Speech Recognition*, edited by Waibel, A. and Lee, K.F., Morgan Kaufmann Publishers. CA 393-404.

Ungar, L. 1995. Forecasting, in: *The Handbook of Brain Theory and Neural Networks,* edited by Arbib, M., MIT Press. Cambridge, MA, 399-403.

NOTES:

1. As an example of the cost of a mistake, note that hundreds of chips, costing several hundred dollars each, may be placed on a single wafer. Therefore, an etch error, which destroys a single wafer, can cost tens of thousands of dollars.

2. This number is not greater because not all of the runs contained 25 wafers.

FIGURE CAPTIONS

Figure 1: **A-D:** Composite wafer signature of fault-wafer types 1-4. **E-F:** Wafer Signatures of No-Fault Wafers gathered at two different times in the clean cycle. 5 wafers are shown in each graph.

Figure 2: Network architecture used for Method B.

Figure 3: Network architecture for creating expectations of the next input waveform.

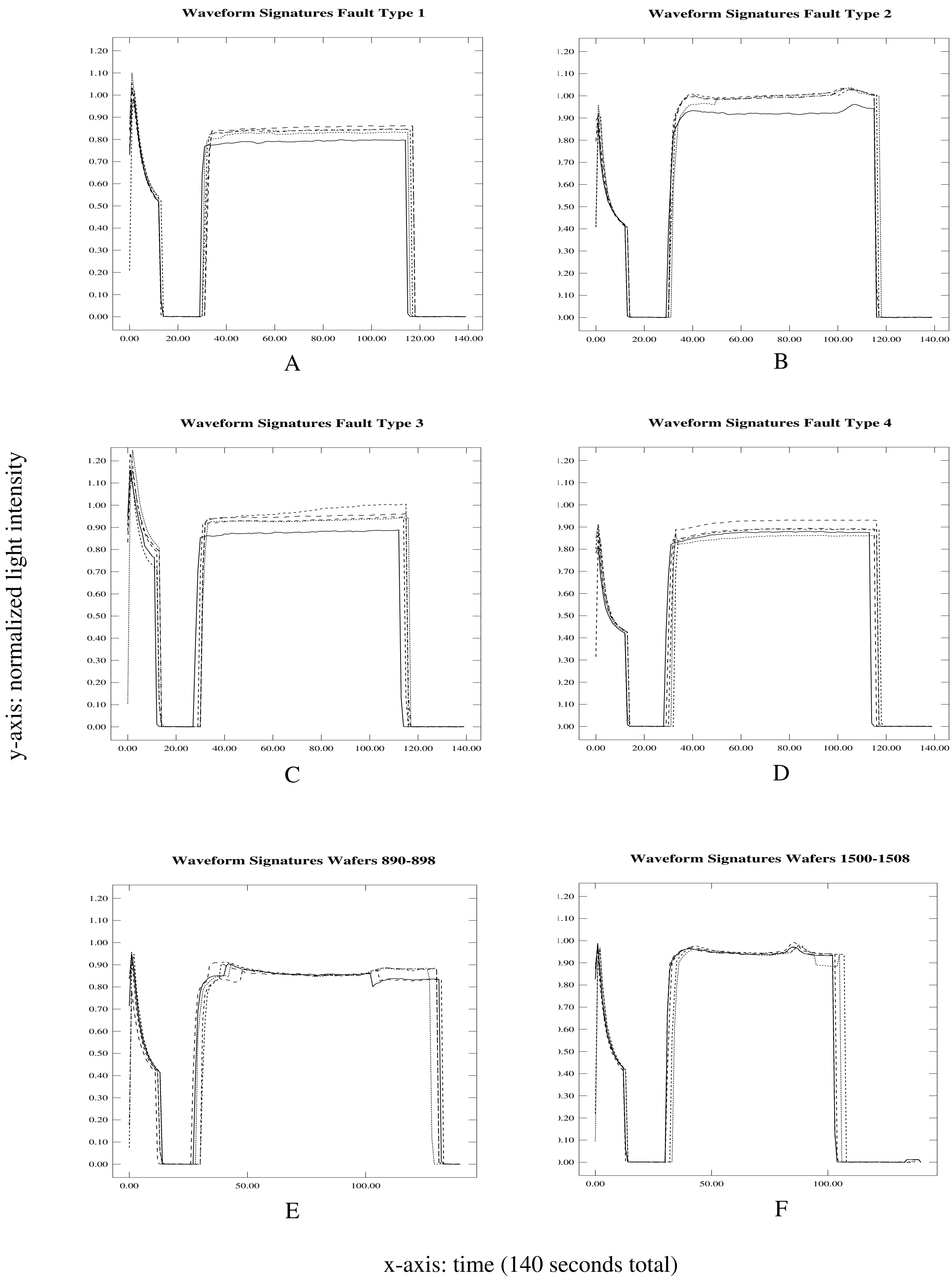
Figure 4: Network for Method E: 5 of these networks were trained; one for each of the four fault types, and one to indicate none of the four faults.

Figure 5: The individual network's outputs (Method E) to each fault type, and no-fault wafers. The X-Axis is the wafer number, and the Y-Axis is each network's response.

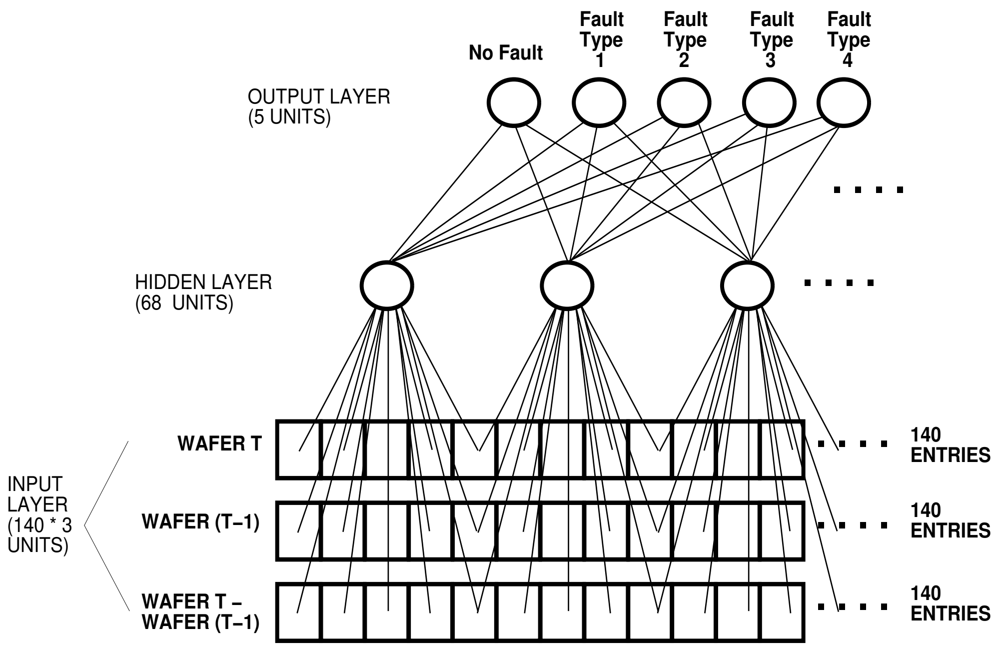
Figure 6: An overview of the five approaches to fault detection and classification.

Figure 7: Benefit of using a prediction to compare with the input wafer instead of the previous wafer. Small differences which may not be true anomalies can be compounded when using two noisy waveforms. The resulting system is less sensitive to small anomalies in the previous wafer.

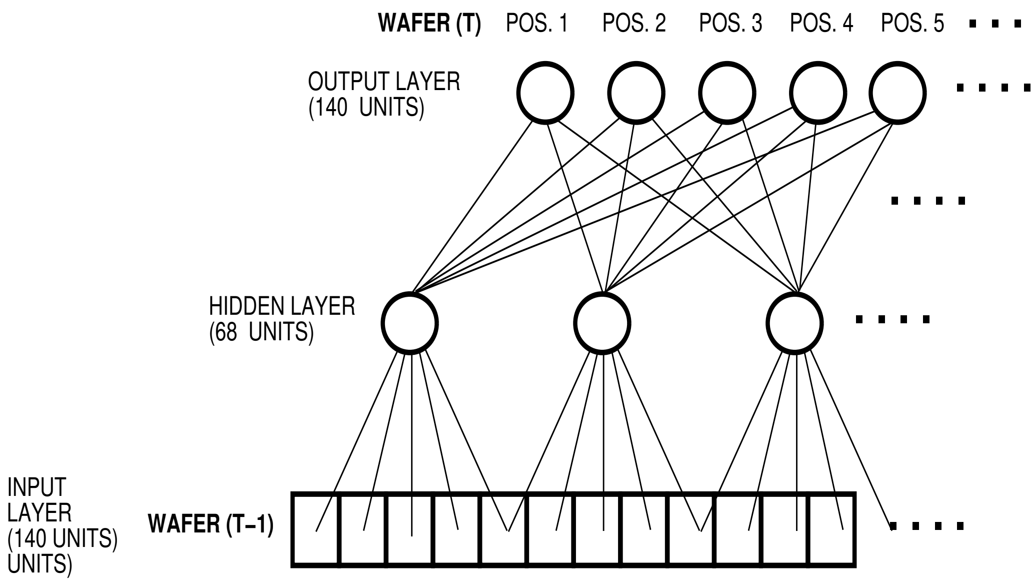
Figure 8: Sample Input Wafers & Next Wafer Predictions. Note the hump in wafer #121, which is mitigated in the prediction of wafer #122. The same effect is seen for wafers #3654 and #3655.



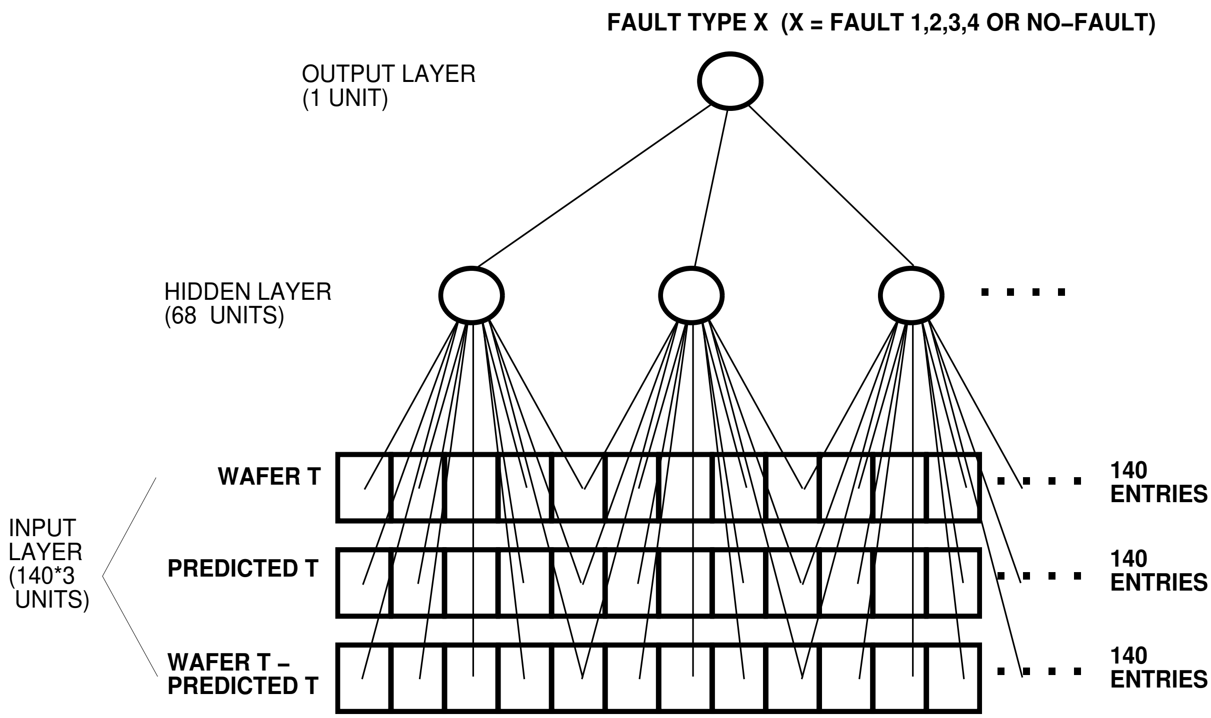
Baluja/Figure 1:



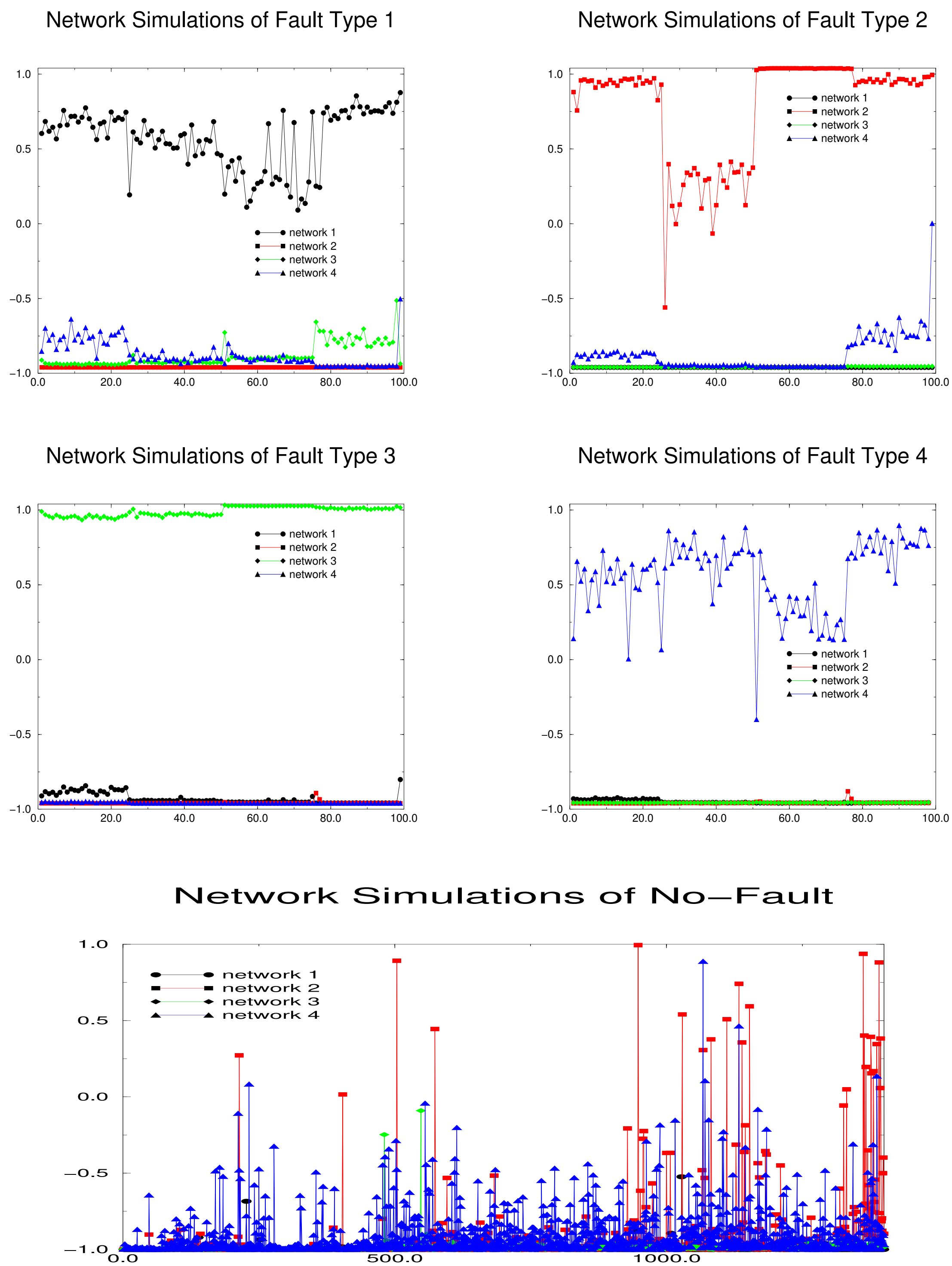
Baluja/Figure 2:



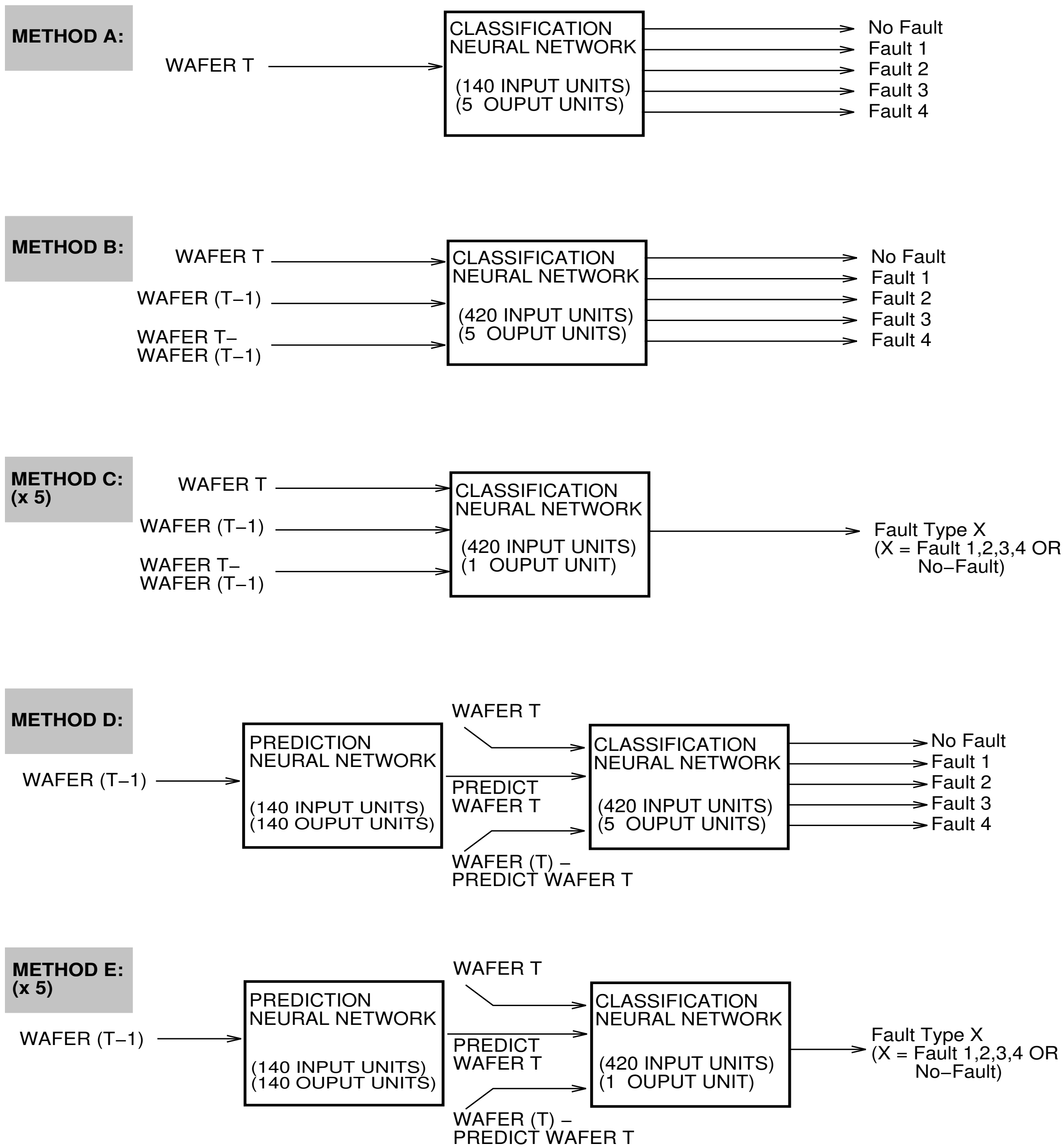
Baluja/Figure 3:



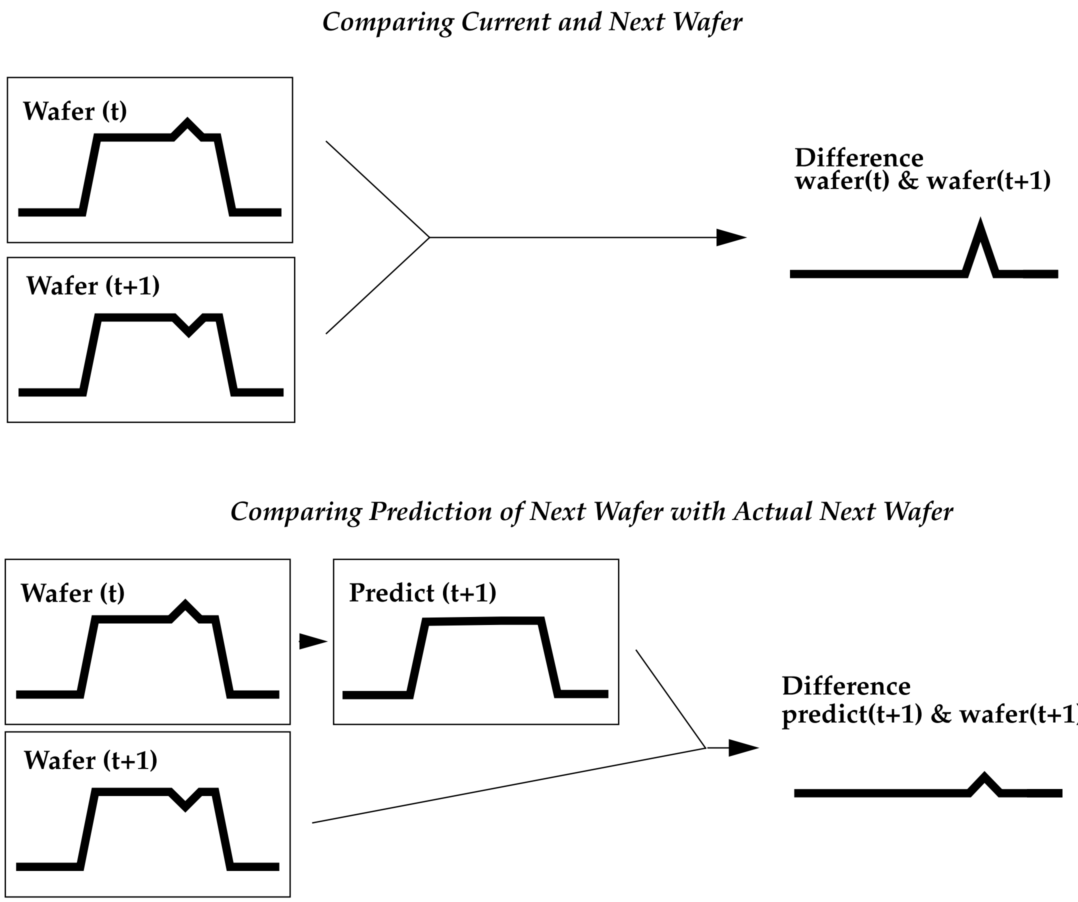
Baluja/Figure 4: N



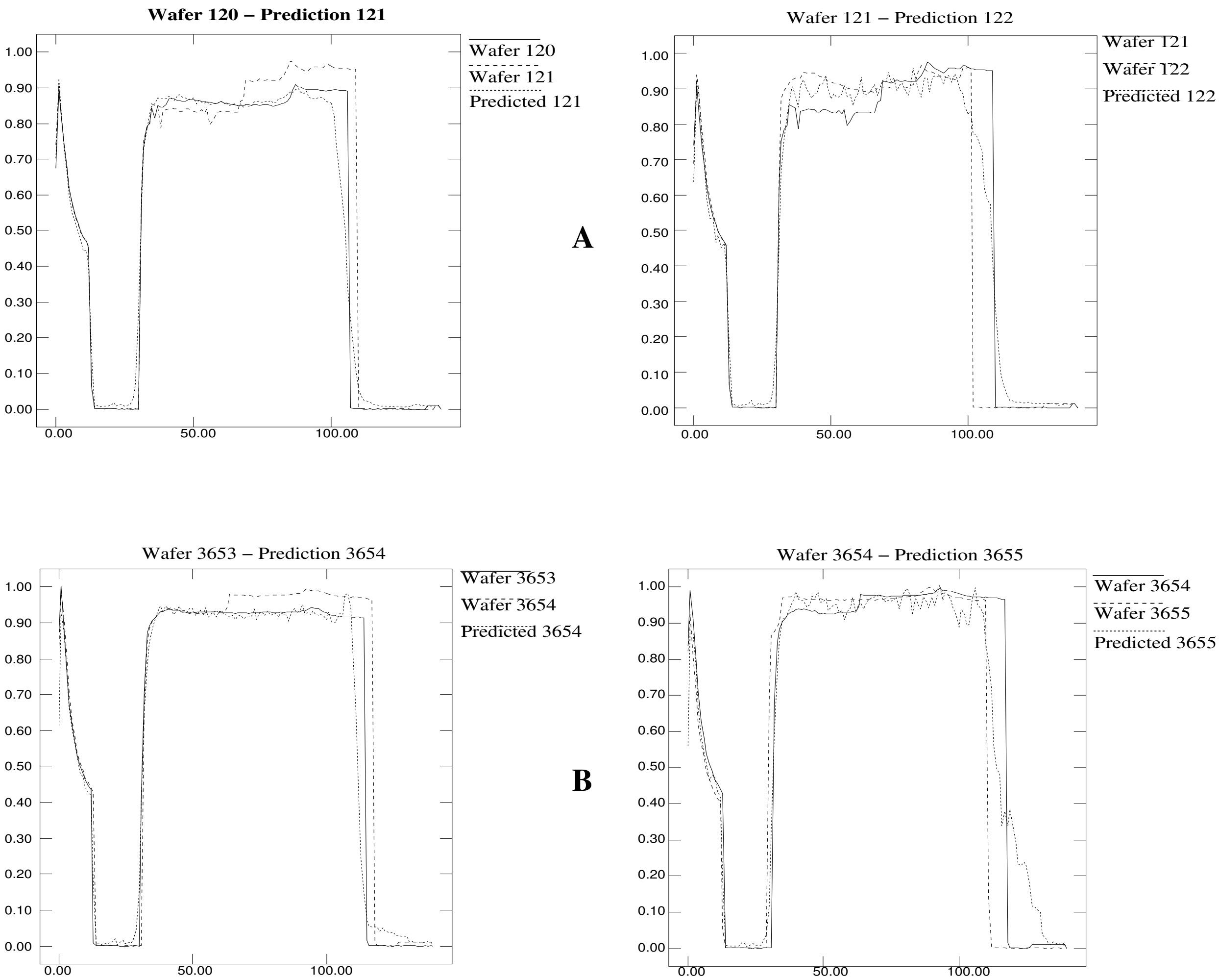
Baluja/Figure 5:



Baluja/Figure 6:



Baluja/Figure 7:



Baluja/Figure 8:

Table 1: Classification Results: Simple ANN Base-Line (Method A)

<table>
<tr><td colspan="2" rowspan="2"></td><td colspan="5">Actual Class</td></tr>
<tr><td>No-Fault</td><td>1</td><td>2</td><td>3</td><td>4</td></tr>
<tr><td rowspan="5">Predicted Class</td><td>No-Fault</td><td>663 (47.0%)</td><td>-</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td>1</td><td>10 (0.7%)</td><td>5 (100%)</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td>2</td><td>563 (39.9%)</td><td>-</td><td>5 (100%)</td><td>-</td><td>-</td></tr>
<tr><td>3</td><td>156 (11.1%)</td><td>-</td><td>-</td><td>5 (100%)</td><td>-</td></tr>
<tr><td>4</td><td>19 (1.3%)</td><td>-</td><td>-</td><td>-</td><td>5 (100%)</td></tr>
</table>

Table 2: Classification Results: Single Network with State Information (Method B)

		Actual Class				
		No-Fault	**1**	**2**	**3**	**4**
Predicted Class	**No-Fault**	1332 (94.4%)	36 (36.4%)	33 (33.3%)	-	4 (4.0%)
	1	1 (0.1%)	63 (63.6%)	-	-	-
	2	72 (5.1%)	-	66 (66.7%)	-	-
	3	4 (0.3%)	-	-	99 (100%)	-
	4	2 (0.1%)	-	-	-	95 (96.0%)

Table 3: Classification Results: Multiple Networks with State Information (Method C)

		Correctly Classified Faults	Misses	Correct Other Faults	Misclassified Faults	Correct No-Faults	False Alarms
Network	**1**	60 (60.6%)	39 (39.4%)	297 (100%)	0	1411 (100%)	0
	2	98 (99.0%)	1 (1.0%)	297 (100%)	0	1390 (98.5%)	21 (1.5%)
	3	99 (100%)	0	297 (100%)	0	1411 (100%)	0
	4	91 (91.9%)	8 (8.1%)	297 (100%)	0	1411 (100%)	0

Table 4: Classification Results: Computed Expectations - Single Network (Method D)

		Actual Class				
		No-Fault	**1**	**2**	**3**	**4**
Predicted Class	**No-Fault**	1268 (89.9%)	9 (0.9%)	28 (28.3%)	-	-
	1	5 (0.3%)	90 (91.0%)	-	-	-
	2	125 (8.9%)	-	71 (71.7%)	-	-
	3	4 (0.3%)	-	-	99 (100%)	-
	4	9 (0.6%)	-	-	-	99 (100%)

Table 5: Classification Results: Computed Expectations - Multiple Networks (Method E)

		Correctly Classified Faults	**Missed Detects**	**Correct Other Faults**	**Misclassified Faults**	**Correct No-Faults**	**False Alarms**
Network	**1**	99 (100%)	0	297 (100%)	0	1411 (100%)	0
	2	96 (97%)	3 (3%)	297 (100%)	0	1384 (98.1%)	27 (1.9%)
	3	99 (100%)	0	297 (100%)	0	1411 (100%)	0
	4	97 (98%)	2 (2%)	297 (100%)	0	1406 (99.6%)	5 (0.4%)

Table 6: Performance Summary

		# Classification Networks Used	Total Missed Faults (%)	False Alarms (%)
Method	**B: State Info.**	1	73/396 (18.4%)	79/1411 (5.6%)
	C: State Info.	4	48/396 (12.1%)	21/1411 (1.5%)
	D: Expectation	1	37/396 (9.3%)	143/1411 (10.1%)
	E: Expectation	4	5/396 (1.3%)	32/1411 (2.3%)