International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:4, No:9, 2010

Application of machine learning methods to online test error detection in semiconductor test

Matthias Kirmse
Dresden University of Technology
D-01187 Dresden

Uwe Petersohn
Dresden University of Technology
D-01187 Dresden

Elief Paffrath
Formerly - Dresden University of Technology
D-01187 Dresden

***Abstract*—As in today's semiconductor industries test costs can make up to 50 percent of the total production costs, an efficient test error detection becomes more and more important. In this paper, we present a new machine learning approach to test error detection that should provide a faster recognition of test system faults as well as an improved test error recall. The key idea is to learn a classifier ensemble, detecting typical test error patterns in wafer test results immediately after finishing these tests. Since test error detection has not yet been discussed in the machine learning community, we define central problem-relevant terms and provide an analysis of important domain properties. Finally, we present comparative studies reflecting the failure detection performance of three individual classifiers and three ensemble methods based upon them. As base classifiers we chose a decision tree learner, a support vector machine and a Bayesian network, while the compared ensemble methods were simple and weighted majority vote as well as stacking. For the evaluation, we used cross validation and a specially designed practical simulation. By implementing our approach in a semiconductor test department for the observation of two products, we proofed its practical applicability.**

***Index Terms*—Ensemble Methods, Fault Detection, Machine Learning, Semiconductor Test**

I. Introduction

High investments and short life-times related to semiconductor factories forces chip manufacturer to optimize almost every step in their production processes to gain maximum yield and minimal costs. Beside design and fabrication, the test of integrated circuits plays a more and more important role, as the corresponding costs can make up to 50% of the total costs [1]. Under this circumstances, it seems to be a valuable approach to improve the test-process using modern machine learning methods.

As todays microchips consist of hundreds of millions transistors, corresponding test procedures have to reflect this enormous complexity. Therefore, test systems, which apply thousands of individual electrical measurements per microchip, are equally prone to faults as the preceding production systems. In most cases, such test system faults cause test errors, such as functional devices being rated as non-functional. To prevent the resulting economic loss, several basic methods are already in use, but all of them suffer either from a poor detection accuracy or a high detection latency.

In our paper we present a new machine learning based approach for test error detection that should overcome the main drawbacks of the existing methods and therefore

- provide a faster detection of test system faults implying less necessary re-tests and
- improve the test error recall causing a higher yield.

The key idea is to learn an ensemble of classifiers to recognize typical test error patterns in wafer test results, thereby enabling their rating immediately after the wafer tests have been finished. As the high temporal variability of the production and test process would not allow to learn and apply a static model, we automatically derive examples for a continuously learning.

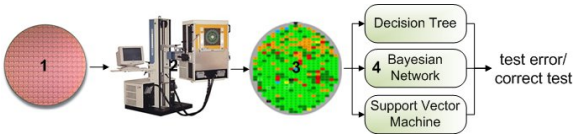

Fig. 1. Schematic representation of the online test error detection system. The classifier ensemble(4) rates the test results(3) immediately after the wafer(1) has been tested on the test machine(2).

To prove the applicability of our approach, we implemented TED[1] for a semiconductor test department. Thereby, TED was completely integrated into the test process, supervising wafer tests of two products.

II. Background

A. Semiconductor Test

Semiconductor test has two main objectives. Primarily, it is concerned with determining the functionality of the produced devices, guaranteeing the delivered chips to be faultless and to meet the required performance targets. On the other hand, test results are used to monitor and control the fabrication process. To obtain these goals, tests are undertaken at different points in the fabrication process [1]. Among those, the "Wafer Sort Test" plays a special role, as it is the first complete functional test of the produced devices, which are still located on the wafer.

During the "Wafer Sort Test" automated test equipment is used to apply a complex sequence of thousands of electrical tests. Basic test types, referred to as test sets, are for example, shorts tests, leakage tests or ring oscillator tests. For all of them, special limits are defined and checked. The results are

[1]**T**est **E**rror **D**etection System

finally aggregated into a so called bin class, whereby pass and fail bin classes are distinguished. While pass bin classes are assigned to functional devices, each fail bin class represents a special set of failed single tests, which caused the rating as non-functional. For instance, there is a special bin class for devices, which failed the shorts tests. Regarding a complete wafer test, we refer to the absolute frequency of bin classes as bin distribution.

B. Test system faults, measurement errors and test errors

Definition 2.1 (Test system fault): A test system fault is a abnormal condition that causes a reduction of the test systems[2] functionality.

There exists a causal connection between the three problem-relevant terms, as *test system faults* cause *measurement errors*, which, for their part, can lead to *test errors*. For instance, the most frequent test system faults in practice are electrical contact-issues[3], increasing the electrical resistance. Thereby, electrical current measurements deliver wrong values, causing a test error, if these values fall beneath a defined limit.

While each of the involved measurements for a device exhibits its individual uncertainties and errors related to test system faults, not all measurement errors are economically relevant. Therefore, we defined the test error of a device solely based on the correctness of its assigned bin class, as this is eventually used to sort them out. Since this bin class is an aggregation of several single measurements, the defined test error is likewise a complex superposition of their single uncertainties and errors.

Definition 2.2 (Device test error): A device test is defined as error, if the resulting bin class differs from the correct one[4].

Formula 1 describes the general relation between measurement and test errors. Thereby, a measurement error, distinguished into systematic and random parts e_s and e_r, becomes a test error if and only if the correct measurement value x_c would lie between the accepted upper and lower limits L_u and L_l , while the measured value does not.

$$\begin{aligned} x_c &\in \{L_u, L_o\} \wedge \\ x_c + e_s + e_r &\notin \{L_u, L_l\} \end{aligned} \tag{1}$$

This means that the occurrence of a test error for a concrete device depends equally on the defined limits, its correct measurement values and the measurement errors of the testing procedure. For instance, the same measurement error may lead to a test error for one device, while having no influence on the bin class of another one, depending on the distance of their correct measurement values to the corresponding limits. Thereby, even small random measurement errors may cause test errors, if the corresponding correct value is already near the limit.

As we want to classify wafer-tests, we have to provide a wafer based test error definition fulfilling two important properties. First, it has to reflect the trade off between the re-test costs and its economical yield due to the retrieved devices. And second, it has to ensure that random measurement errors, implying random test error patterns, do not impurify the learning population by introducing compromising noise. Therefore, we introduce a limit X for single device test errors of a wafer test, which reflects both properties.

Definition 2.3 (Wafer test error): A wafer test is defined as error, if it contains at least a specified percentage X of device test errors.

From a statistical point of view, the overall test signal of an entire product population is a superposition of the production signal and the measurement error signal. Thereby, the product variance is usually significantly higher than the test variance [3], which makes it in general difficult to identify reliably a conspicuous signal deviation as measurement error, and where appropriate, as test error.

C. Current methods

Today, different standard methods are established to prevent and detect faults in the test process, whereby we distinguish between off line and on line approaches.

Off line approaches, like regular calibration and maintenance or reference wafer [3], suffer from a common disadvantage. All of them need the test process to be interrupted. With respect to limited test resources, they can therefore only be applied infrequently.

In contrast, on line approaches allow a parallel supervision of the test process. A simple method is to set up static limits for a defined subset of test results, for instance the occurrence of certain bin classes. While this is a very fast detection method, it covers only a small subset of obvious test errors and requires continues personal effort to update the limits.

Another common approach are regularly applied re-tests. Certain wafers are tested twice or more, whereby significant differences between test runs of the same wafer indicate test errors. Although this method is much more reliable, it can not be applied to all wafers tests due to time and cost reasons. Until a test system fault is detected by this method, it may have affected several wafer tests, which all have to be re-tested.

TED should fill this gap between the fast but inaccurate static limit method and the reliable but slow re-test approach.

D. Related work

A statistical approach for on line failure detection can be found in [3]. It describes the application of statistical process control methods to the test-process. Based on batches of regular done re-tests, novel control charts are proposed, which represent the average difference between test and retests of selected parameters, i.e., the random measurement error or repeatability. The authors assumed from experience, that a

[2]A test system generally includes the complete process to obtain the test result, i.e. equipment, software, methods, environment, operations and personnel. Nevertheless, we use the term primarily as reference to the test equipment.

[3]They can be related to the wearing of probe-pins connecting the device with the testing machine. A more detailed listing of test system faults can be found in [2]

[4]The correct bin class could be defined as the one produced by an ideal test process.

"'significant change in the systematic error always correlates with a change in the random error"'. Therefore, an out of control situation of these control charts should give an early hint at underlying systematic measurement errors.

In [4] a similar statistical approach is used to handle parallel measurement systems. The authors developed a linear statistical model for this kind of systems, including as well the process-variability, the tester-variability and a random error for each tester. Based on batches of re-tests, similar to [3], they use common analysis of variance techniques to get both the variance contribution of each single measurement instruments to the overall variance and their variation over time. Control charts based on these criteria are used to decide if variations in the resulting signals are due to process or test variations. Therefore, they are able to therefore identify faulty measurement instruments.

As both proposed methods are of statistical nature, they suffer from the same disadvantage, namely that a potential test system fault is found at the earliest when the affected batch is completed. Besides, both methods use only univariate control charts, which are less appropriate to find complex test error patterns than machine learning methods, naturally handling multivariate feature spaces.

Extensive studies regarding the machine learning approach to test error detection are presented in [5]. We examined both, classifiers for single device test errors and such for wafer test errors. Thereby, we studied seven test sets in combination with representatives of established supervised learning methods, namely support vector machines, decision trees, rule learners, Bayesian networks, instance based learners and artificial neural networks. Although, the device test error classifiers showed reasonable results for cross validation, we could not confirm them by practical application due to implementation issues. In the wafer test error experiments, only the bin deviation achieved sufficient results and is therefore used primarily for our current studies. Besides the experiments, we developed ProSuLE[5] , a software framework for hierarchical classification in productive environments. It provides the application of classifier ensembles in combination with different feature groups on hierarchical ordered levels, such as device, wafer and lot level in the semiconductor domain.

III. Machine learning approach

The proposed test error detection systems can analytically be described as a classifier, assigning one of the both classes "'correct test"' and "'test error"' to each wafer test based on a subset of its test results.

A. Important domain properties

In this section, we want to sketch characteristic domain properties, which are important for the application of machine learning methods.

Regarding all wafer tests, there is a high variation due to different products, mask revisions, process flows, test platforms and test programs. All of them have a significant impact on test patterns, especially on test error patterns. Therefore, we have to carefully define *model contexts*. Choosing them too large introduces noise and inconsistencies in the sample populations. On the other hand, too few examples can compromise the generalization ability and lead to overfitting. For the current version of TED, we decided to set up one model for each product, whereas future versions will have a finer granularity.

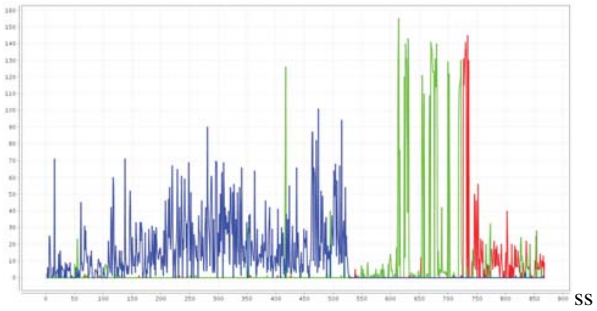

Fig. 2. Frequency of wrongly assigned fail bins (blue,red,green) with respect to chronologically ordered test error examples.

Concept drift is a related problem that occurs if the model-context is chosen too large. For example, new tests, changing test limits, varying process flow or adaption of concurrent failure detection methods result in significant changes of the target concept, thereby decreasing the performance of static models. These drifts can either be abrupt, if new limits are set, or gradually, caused, for instance, by slow wearing of probe-pins. Figure 1 shows a practical example for this phenomenon. While in the first interval the blue fail bin is the main test error source, it is later replaced by the green and finally by the red one. More detailed information about concept drift are presented in [6]. To confine this problem, we currently limit the training sets in TED to a defined fraction of the latest samples, trying to minimize concept drift related inconsistencies.

The fact that there are much less faulty tests then correct ones is known as *unbalanced data* in the machine learning community [7]. This significant difference in class prior probabilities can decrease the performance of standard learning methods, as they assume balanced data sets. Especially, the recognition quality of the minor class, in our case the test errors, can be affected.

Most standard classifiers assume the training set to be independent identically distributed. But like for other real world problems such as computer aided diagnostic [8] this assumption is not fulfilled in our case. Wafers in a lot or generally those passing similar process steps are strongly correlated, introducing unwanted signals in the training set, which may be wrongly learned as test error patterns. For instance, a test error pattern derived by a sample of wafers originating from the same lot, may be strongly influenced by a underlying lot-specific production fault. This problem of *correlated samples* has to be focused in further studies.

[5]Production specific supervised learning environment.

B. Classification features

As we mentioned earlier, the "Wafer Sort Test" consists of a sequence of different electrical measurements, which all are potential features for the test error detection. Nevertheless, not all of them are appropriate, as most of them either are only infrequently available[6] or exhibit a natural high variance.

Finally, we have chosen the bin distribution as feature set for two reasons. It performed best in the mentioned previous studies [5] and required considerably less implementation effort than other test sets[7]. We can explain the comparatively better performance to a great extend by the fact, that we defined test errors solely based on bin classes, implying that each so defined error is reflected in the bin distribution. In contrast, the other test sets, such as shorts tests, are only affected by the corresponding subset of the so defined test errors.

C. Automated labeling procedure

In most classical learning settings there exists a human labeled example set, used to learn a static concept. Known examples are image recognition or fault detection for semiconductor tools [9]. Nevertheless, this approach is not suitable in our case as the labeling would be very time-consuming and have to be repeated in regular intervals due to the described concept drift.

Therefore, we utilize the existing test redundancy, namely regular done retests, together with our test error definition to automatically derive examples. Basically, the automated labeling is only possible for wafers, which have been tested at least twice, as the correctness of single wafer tests cannot be verified reliably. For those, the single test runs are compared, whereas we define the chronologically last one as correct, as we expect the responsible engineers to repeat re-tests until the correctness of the last re-test is assured. The previous test runs are then labeled according to the wafer test error definition, whereby the percentage of device test errors is determined by comparing their assigned bin classes to the corresponding assignments of the last re-test.

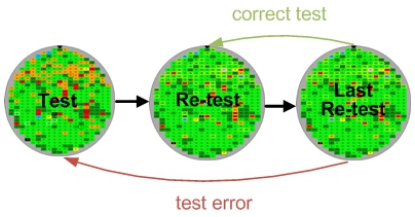

Fig. 3. Automatically deriving wafer test labels by comparison to the last re-test.

[6]For instance, only for 0.1 percent of all examples the shorts test were available in the used data source. For space reasons, they were only added, if at least one test failed.

[7]Bin frequencies are uniformly represented in the used data source, while the representation, such as identifiers, of other test sets differs between products or even test programs. This would cause an high configuration effort.

D. Models and learning algorithms

Based on our previous studies, we selected three algorithms, a support vector machine, a decision tree learner and a Bayesian network, for further experiments.

As kernel for the *support vector machine*, we used a radial basis function. Detailed informations about Support Vector Machines can be found in [10].

The *decision tree* has been learned with the C4.5 algorithm [11]. We used a minimal leaf size of four and a minimal split size of eight examples. Both parameters represent the trade-off between overfitting to noise and the ability to learn rare failure patterns.

To learn the network structure of the applied *Bayesian network* we used the K2 algorithm, described in [12]. The conditional probabilities of the random variables have been estimated by counting value combinations in the training set[8].

E. Ensemble methods

In many applications and scenarios classifier ensembles, i.e., combinations of individual classifiers, performed better than their corresponding single counterparts. Therefore, we decided to apply an ensemble instead of just selecting the best learning algorithm. Crucial for the success of such ensembles is diversity among the individual classifiers, meaning that they have to misclassify examples in different areas of the feature space. This can be achieved, for example, by learning the classifiers with different training sets. Instead, we used three conceptually completely different learning algorithms to obtain this property.

Nevertheless, it remains the question how to aggregate the results of the individual models into a total classification. Therefore, we compared three aggregation methods, simple majority vote, weighted majority vote and stacking. In the *weighted majority vote* each individual classifier has an assigned weight determining its influence on the total classification. Given a set of classifiers T, a set of classes C, the single classifications $d_{t,c}$, which are one if model t returns class c and zero otherwise, and the corresponding model weights w_t, the resulting class j is determined by

$$j = \underset{c \in C}{\operatorname{argmax}} \sum_{t \in T} w_t d_{t,c} \tag{2}$$

Thereby, the model weights should reflect the future performance of the individual models, which is usually estimated by their performance on a training or additional validation set. We studied the first variant, determining the weights based on the training set accuracy a_t through

$$w_t = \frac{a_t}{1 - a_t} \tag{3}$$

Stacking [13] uses a meta classifier to combine the single model results. This meta-classifier is learned in two steps.

[8]Each conditional probability got an additional prior to avoid zero probabilities. Without the additional prior, this method equals the maximum likelihood solution.

First, the single classifiers are learned and applied on the training set and second, the meta model is learned with the same examples extended by the single model results. For our studies we used a decision tree learner as meta model. Further information about the examined ensemble methods can be found in [14].

IV. Experiments and practical results

A. Experimental and practical setting

For our experiments we used 5709 wafer tests from a highly frequented product A, 4841 of them labeled as correct and 868 as test error. All have been tested at least twice and labeled according to the automated procedure.

Based on our framework ProSuLE, we developed TED, which has been applied in a semiconductor test department to supervise the tests of two products for a period of six months. Thereby, we used earlier wafer tests to learn initial models. As intended, our system observed each wafer test and alarmed the engineers in case of detected test errors.

B. Evaluation methods

To evaluate the performance of the single classifiers and the ensemble methods, we used a five-fold cross validation and a practical simulation. Thereby, we additionally implemented the practical simulation, because we assumed the ability of cross validation to yield a good estimation for the practical model performance to be compromised by the special domain properties. More precisely, the concept drift is not represented by the standard cross validation, as the random split subsets contain examples from all temporal segments. And furthermore, overfitting is rewarded by correlates samples and leads to a overestimation of the classification accuracy. An example: if the example set contains several test error examples of one lot, it is probable that the learning algorithm extracts some lot characteristics, such as the abnormal high count of a certain bin-class due to a special production fault, as general test error property. This is rewarded, as a fraction of the affected wafer tests are as well in the test set, and therefore correctly classified by the overfitted model.

Our second evaluation method simulates the real test sequence of the given example test runs. All sample wafer tests are therefore grouped into two kinds of chronologically ordered batches, each representing a date. While classification batches contain all wafer tests that were executed on the corresponding date, learning batches include those that have become available for learning. Sequentially for each date, first a model is learned based on all learning batches up to this date and then applied to the current classification batch. As this method is an exactly simulation of the real process, it is a valuable practical performance estimator.

Due to the imbalance problem, usual measures like model accuracy may be misleading. Given the hypothetical situation of one percent test errors, even a classifier labeling each wafer test as correct would achieve an accuracy of 99 percent. Therefore, we took additionally the f-measure F_β as performance criterion, which is an aggregation of the test error recall r and precision p, whereby β reflects their weighting[9]. Based on our purpose to minimize false-alarms, considering user acceptance, we set β to 0.5, implying that precision is weighted twice as much as recall.

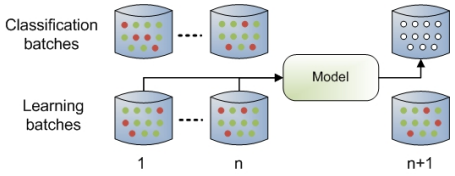

Fig. 4. Nth iteration of the practical simulation, whereby the first n learning batches are used to train a model that is applied to the (n+1)th classification batch.

$$F_\beta = (1+\beta^2)\frac{p\,r}{\beta^2 p + r} \tag{4}$$

C. Results

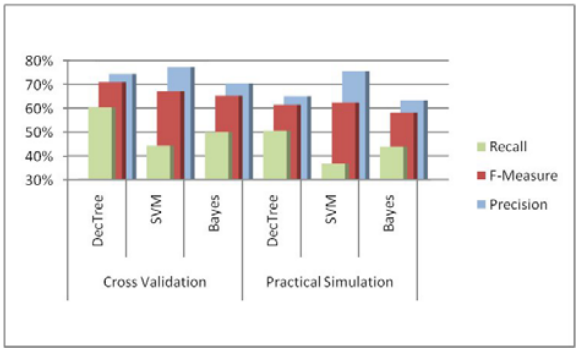

Fig. 5. Compares recall, precision, and f-measure of single classifiers for cross-validation and practical simulation.

1) Single classifier: Among the single classifiers, the decision tree achieved due to its high recall the best f-measure in the cross-validion experiments, followed by the support vector machine with a slightly better precision and the Bayesian network. Totally, the single classifiers achieved an accuracy average of 89.8 percent and a f-measure average of 67.7 percent.

As expected, the simulation experiments showed worse results. While the accuracy average stayed nearly equal with 88.3 percent, the f-measure average decreased by about 7.1 percent. As the decision tree reached nearly 10 percent less recall and precision, the support vector machine became the best single classifier. Besides, the considerable differences in precision and recall, especially between the support vector machine and the decision tree, indicate a basic diversity between these models.

2) Ensemble methods: In cross validation, none of the studied ensemble methods achieved a better f-measure than the single decision tree model, although they performed only slightly worse. Compared to each other, while showing nearly

[9]A β of one represents the harmonic mean between precision and recall.

the same f-measure, they exhibited strong differences regarding their precision-recall distribution. While the simple majority vote gained the highest test error precision, the other methods showed a significantly higher test error recall.

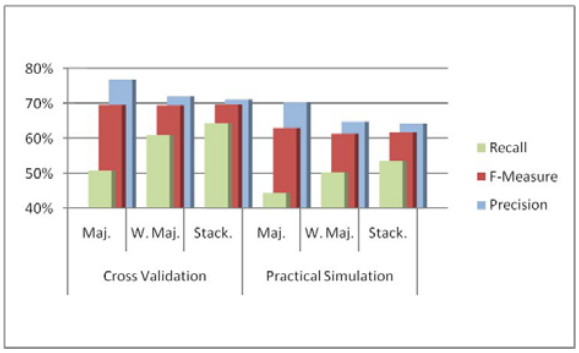

Fig. 6. Compares recall, precision, and f-measure of ensemble methods for cross-validation and practical simulation.

We found a similar situation for the simulation results with the difference that the simple majority vote achieved the best overall f-measure performance. Compared to the best single algorithm, the support vector machine, it had 5.4 percent less precision, but 7.7 percent more recall, resulting in a slightly better f-measure. The other studied ensemble methods achieved about 6 percent less precision and 6 to 9 percent more recall than the simple majority vote. Crucial for this outcome has been the implicit bias of simple majority vote for a high precision, which is rewarded by our adjusted f-measure. Based equally on these results, our goal to minimize the false rate and the least implementation complexity, we decided to use simple majority vote in the first version of TED.

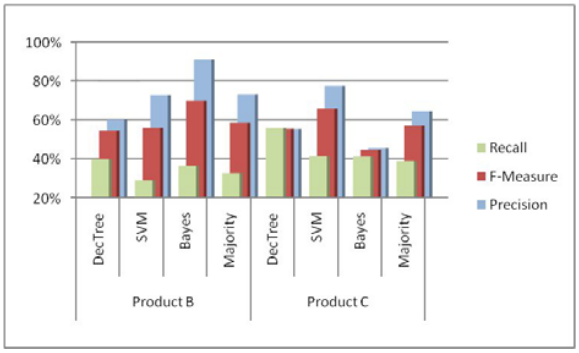

Fig. 7. Compares recall, precision, and f-measure of the single classifiers and the simple majority vote for the practical results of the products B and C.

3) Practical system: In practice, the simple majority vote achieved on average better results than the single decision tree and the Bayesian network. Only the support vector machine alone reached a slightly better average f-measure, although for two of the three products the simple majority vote performed best. Nevertheless, it is most valuable to combine the support vector machine in an ensemble instead of using it alone, as it is in contrast to the decision tree not human interpretable.

Additionally, we found that the performance of the individual models exhibits a strong variation between the products, indicating significant differences in their test error patterns. For instance, the precision of the Bayesian network ranges from 45.4 to 90.9 percent. The majority vote compensated this variation best as it showed the least f-measure standard deviation, regarding all products.

Averaged over both supervised products, TED detected 35.6 percent of all known test errors, whereby 68.7 of its test error alarms were valid. Moreover, 36 percent of all permanent test system faults, affecting more than one wafer test in a row, have been detected by TED at an early stage. A consequent reaction on the test error messages could have saved thereby 38 percent of the related necessary re-tests.

4) Evaluation methods: Compared to the practical simulation of product A, the corresponding cross validation results overestimated the f-measure about seven percent. This residual increases to ten percent, if the cross validation is compared to the averaged practical results of the three products, assuming that the simulation results for product A are a good estimation of its potential practical results. The overestimation is thereby equally distributed between precision and recall, whereas recall has a slight predominance. These results confirm the assumptions made in section IV-B and therefore the additional use of the simulation method to estimate the practical performance.

V. Conclusion

In this paper, we presented a completely new approach to online test error detection based on machine learning methods. As this problem has not yet been covered by the machine learning community, we defined and motivated basic terms, analysed their relationships and illustrated basic issues, such as the unfavourable ratio of the test and product variance. Furthermore, we pointed out important domain properties, which have to be handled to gain an optimal detection performance with machine learning methods.

To provide empirical evidence for our approach, we studied three established classifiers and ensemble methods. Because of our preferences for maximal precision, stable performance regarding different products and human readability, we have chosen the simple majority vote of a decision tree, a support vector machine and a Bayesian network as best combination. Nevertheless, alternative preferences can be satisfied by the other studied models, such as a significantly higher recall by stacking.

We implemented TED, a completely autonomous test error detection system, to prove the practical applicability of our approach. TED independently derives training examples based on the automated labelling procedure, which requires no additional test resources, but utilizes existing test redundancies. Its application proved at least on of the two objectives, as 36 percent of all permanent test system faults have been detected at an early stage.

Future studies have to focus on the described domain properties, for instance, determining an optimal model context or studying and adjusting existing approaches for concept drift. Furthermore, correlated samples are mostly ignored in the machine learning community. Nevertheless, appropriate algorithms as well as suitable evaluation methods have to be developed to improve and rate the test error classification ability. Besides, a reliable fault diagnosis is one of our next main goals to provide a more targeted reaction on test system faults.

While the studied application domain in this paper has been semiconductor test, our approach to online test error detection can in principle be transferred to each complex test scenario in high volume productions, such as automotive test. As primary precondition for its application, the target domain has to have regularly done re-tests to provide automatically derived training examples. Furthermore, we have to define an appropriate test error criterion and find suitable classification features. Finally, the presented case study in semiconductor test has shown the potential of our approach to improve the efficiency of increasingly complex test procedures with reasonable effort.

REFERENCES

[1] I. A. Grout, *Integrated Circuit Test Engineering: Modern Techniques*, 1st ed. Springer, 2005.
[2] Y. S. Chang, J. E. Chen, and Y. Y. Chen, "Error classification by wafer map analysis," Taiwan, 1994.
[3] J. van der Peet and G. van Boxem, "SPC on the IC-Production test process," *Test Conference, International*, vol. 0, p. 605, 1996.
[4] H. Shu-guang, Q. Er-shi, and L. Li, "Study on the model of analysis and control of parallel measurement systems," in *Management Science and Engineering, 2007. ICMSE 2007. International Conference on*, 2007, pp. 633–638.
[5] M. Kirmse, *Evaluation hierarchisch eingesetzter, maschineller Lernverfahren zur automatisierten, fruehzeitigen Erkennung von Testsystemfehlern beim Wafer-Test*. TU Dresden, 2008.
[6] A. Tsymbal, "The problem of concept drift: Definitions and related work," 2004.
[7] N. Japkowicz, "The class imbalance problem: Significance and strategies," in *In Proceedings of the 2000 International Conference on Artificial Intelligence (ICAI*, 2000, p. 111117.
[8] M. Dundar, B. Krishnapuram, J. Bi, and R. B. Rao, "Learning classifiers when the training data is not IID," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007.
[9] B. E. Goodlin, D. S. Boning, H. H. Sawin, and B. M. Wise, "Simultaneous fault detection and classification for semiconductor manufacturing tools," 2002. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.15.8602
[10] B. S. und Alexander J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, 2001.
[11] J. R. Quinlan, *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., 1993. [Online]. Available: http://portal.acm.org/citation.cfm?id=152181
[12] G. F. Cooper and E. Herskovits, "A bayesian method for the induction of probabilistic networks from data," *Machine Learning*, vol. 9, no. 4, pp. 309–347, 1992. [Online]. Available: http://dx.doi.org/10.1007/BF00994110
[13] D. H. Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, p. 241259, 1992.
[14] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits and Systems Magazine*, vol. 6, no. 3, p. 2145, 2006.