Machine Learning-based Prediction of Test Power

Harshad Dhotre* Stephan Eggersglüß‡ Krishnendu Chakrabarty§ Rolf Drechsler*†
*Institute of Computer Science, University of Bremen, 28359 Bremen, Germany
†Cyber-Physical Systems, DFKI GmbH, 28359 Bremen, Germany
‡Mentor, A Siemens Business, Hamburg, Germany
§Duke University, Durham, NC 27708, USA
dhotre@uni-bremen.de, drechsler@uni-bremen.de

***Abstract*—With the increase in circuit complexity, the gap between circuit development time and analysis time has widened. A large database is required in order to perform essential analysis tasks such as power, thermal, and IR-drop analysis, which, in turn, leads to long run times. This work focuses on test power analysis. Due to the large number of test patterns for modern designs and the excessive power analysis run time for each test, it is not feasible to obtain complete power profiles for all the tests. However, test power-safety is essential to produce reliable manufacturing test results and prevent yield loss and chip damage. Accurate power profiling can typically be done for a small subset of pre-selected tests only. An essential task is therefore to determine those tests, which potentially provide the worst-case scenarios with respect to test power. We propose machine learning-based power prediction for test selection. The prediction is applied in two different ways. First, we predict the activity of a test to identify tests with high power consumption. Second, the switching activity and the power information are related to the layout of the chip to identify local hot spots. Various machine learning-based algorithms are used to evaluate this approach. Additionally, the algorithms are compared against each other. The results indicate high prediction accuracy and effectiveness. This makes these algorithms well suited for worst-case test selection.**

I. Introduction

The manufacturing test is an essential part in the chip development and production process. Its goal is to find and sort out defective devices. A test set is pre-generated and applied to each manufactured chip. In order to ensure timely and cost-effective testing, on-chip scan structures are heavily used, which create non-functional operating conditions. This, in turn, creates power issues [12], [25]. The chip's power consumption is tightly regulated by control mechanisms to meet the stringent power specification. However, non-functional operating conditions as used during test application exceed the power specification. This can lead to unreliable test results and even chip damage. Therefore, tests have to be checked in advance to ensure that they meet the power requirements to guarantee test power-safety. Another related issue is IR-drop, which can cause timing violations. Accurate simulation methods have to be applied to obtain reliable results in the sign-off stage before tapeout. But the corrective measures may take several iterations to reduce the IR-drop of each and every instance separately.

Unfortunately, accurate power and timing simulations need considerable resources and excessive runtime. Furthermore, the analyses can only be performed in a very late stage of the development process close to tapeout [19], [21]. The complete simulation of all tests is therefore infeasible. A small subset of tests is typically chosen to be simulated accurately. Ideally, this subset contains the potential worst-case scenarios. A critical issue is the selection of tests covering the scenarios, which can lead to power issues during test application. After the accurate analysis of some tests covering worst-case scenarios, it might be still possible that other tests exist, which will cause power-related issues. Also the corrective iteration for some patterns may not be applicable to other test patterns. This paper targets the identification of worst-case test power-related issues by means of prediction. We propose the use of a *Machine Learning* (ML)-based prediction mechanism to characterize the test power behavior of all tests, thereby avoiding the detailed resource-consuming analysis of all test patterns.

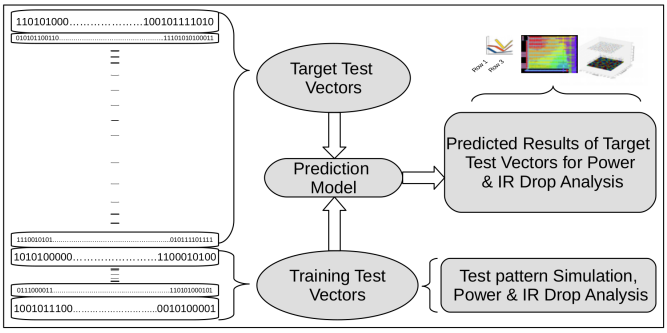

Fig. 1: Illustration of the proposed method.

Figure 1 illustrates the general idea. First, a few pre-selected tests are accurately simulated as in the regular flow. These tests are referred to as *training test vectors* (T_t). The training test vectors as well as the corresponding analysis results, i.e. the simulation data, are then used to train an ML model f. Then, this trained model is used to predict the power analysis result of the set of *prediction target test vectors* (T_p) without an explicit simulation of T_p. This methodology allows us to predict the overall power profile of a test. The application is done in two different ways:

- The overall (global) power consumption of a test t is targeted and predicted to identify critical tests.
- Since a low global power consumption does not guarantee the absence of hot spots, the power consumption is related to the layout of the chip. In this way, local hot spots can also be predicted.

The proposed methodology is based on the use of supervised learning algorithms. This approach has been implemented using various learning algorithms and the results have been evaluated. The prediction time is very small compared to the

actual accurate analysis time. At the same time, the predicted values are highly reliable and show only a small variance to the actual values obtained by the accurate simulation. This enables the processing of all tests and their corresponding power profile prediction in order to find potentially power-unsafe tests. The proposed approach therefore increases the overall possibility to cover critical tests during the sign-off stage.

The structure of the paper is as follows. Section II describes related prior work. Section III discusses the analysis, test data and features used for the learning techniques. The different learning algorithms are briefly described in Section IV. The application of the learning algorithms is discussed in Section V and the experimental results are reported in Section VI. Conclusions are drawn in Section VII.

II. Related Prior Work

Power-estimation techniques offer a trade-off between accuracy and run time. Simulation-based power analysis gives the most accurate results. However, these techniques are very time consuming. Usually, the accuracy varies within various stages of the design flow, i.e., from the specification across system level, algorithmic level, RTL, gate level to the circuit level. The analysis time increases significantly with the increase in accuracy. Sometimes, detailed analysis techniques, e.g., IR-drop analysis, are even not possible at a higher level because of the structure-dependent characteristics. The SystemC transaction level power-estimation presented in [20] for different applications is highly approximate due to its abstraction level. Functional-level power-estimation methodologies for predicting the power dissipation of embedded software are presented in [23]. Statistical methods for estimating the peak power dissipation are explained in [7], [26]. Other work [13] introduced the prediction of switching statistics at RTL with the help of analytical models. In contrast to these methods, our work deals with the prediction at the circuit level, which is used in the sign-off stage of the chip development. Monte-Carlo based approaches [10], [19], [24] and other statistical approaches [3], [4], [6], [8] estimate the average power in a less simulation-dependent way and need less time for the average functional power estimation.

However, these previous methods cannot be used to predict the test power, which is very high compared to the functional power [12]. A recently proposed ML-based method [11], [16] predicts the IR-drop after an engineering change order (ECO) by using simulation results generated before the ECO. However, this approach also suffers from the targeted problem of completeness and test selection since it relies on the accurate simulation results. In contrast, we use an ML-based method to predict the test power of tests based on the accurate analysis data from a small fraction of the tests. Additionally, the feature set used in the proposed approach includes the test patterns.

Neural network-based methods for VLSI power estimation [15] give acceptable results with a specific net structure. However, the accuracy of the results is low because no technology parameters, i.e., the liberty or spice (.lib or .sp), are used. Additionally, only the switching of the functional operation mode is targeted using I/O information and cell numbers. A similar approach is presented in [14]. This work targets

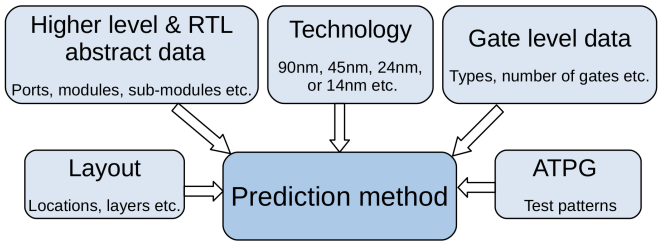

Fig. 2: Data preparation and feature extraction.

the specification level. In contrast, we consider parameters from the technology library, design and simulation stages in order to predict the switching activity in a more accurate way. Furthermore, the power activity with respect to the layout is considered.

The method in [17] identifies areas where excessive power consumption is likely to occur. This is done in a vectorless manner by considering the probability measures based on the circuit structure. In contrast, the proposed approach uses ML techniques to predict the power profile of explicit test vectors.

III. Power Analysis and Feature Extraction

Since it is not feasible to simulate all test patterns in an accurate way to obtain their power profile, the use of ML techniques to predict the activity of all test patterns globally and locally to save run time is proposed. In our work, global activity refers to the overall power of a test, while local activity refers to the test power with respect to the layout.

The features and the training data are the most important constituents for the use of learning algorithms. For test power prediction and its distribution over the layout, features are to be extracted from different stages of the design flow. The design-related data is available in different formats, used by different tools, and it needs to be processed accordingly. The information in these files is used for different purposes. The required information has to be extracted and formulated for the learning part. Figure 2 illustrates the data-set preparation and the extraction of features for the learning algorithm. The necessary data is as follows:

- *Design information* – In order to read, store and annotate the extracted and learned data, design information is necessary, e.g., hierarchy and port information.
- *ATPG generated test patterns* – An essential part of the test is made up by the test patterns or test vectors, which are generated by ATPG tools. Typically, scan structures enable a direct assignment of the flip-flops of the circuit without using functional operations. In order to provide a high test coverage and short test application time, scan structures are heavily used. A scan test is first scanned in using shift cycles and than applied during the capture cycle(s). As a disadvantage in the context of test power, scan tests induce non-functional behavior, since the scanned state of the flip-flops is not guaranteed to be reachable in the functional mode. In our proposed methodology, the required data, i.e., the content of the scan cells during shift and capture operation for each scan chain and test pattern, is extracted from the test-pattern files and stored for further processing.

- *Simulation and analysis data* – Shift and capture operations of the applied tests cause activity in the logic parts of the design. The activity can be obtained by simulating the test patterns. After simulation, this information is extracted and can be stored, e.g., in a VCD file (or in another format), which describe the change in logic values over discrete time. Note that logic simulation usually does not process technology information and, therefore, it does not provide information regarding power consumption. Figure 3 illustrates the simulation and analysis results (after accessing the switching activity information from the simulation) for a test pattern. In order to obtain a power profile of a simulation run, the simulation data has to be simulated again with the necessary cell and technology data. This is the resource- and time-consuming part. After power analysis, this data is stored for training purposes.
- *Technology and physical layout information* – For the case that only the global power consumption is targeted, the physical location of the gates and other layout data is not required. However, this data has to be processed and stored in order to identify local hot spots. Therefore, the location and sizes of the cells are extracted from the layout information, e.g. from the .def and .lef files. This data can be fed into commercial EDA tools or other approaches in order to identify power-critical areas, e.g., by using heatmaps.

IV. Machine-Learning Algorithms

We use supervised ML algorithms to predict the power profile of a test. The extracted data and features are used to generate and train the underlying model. Figure 4 shows the generalized structure of the prediction method. Here, features are extracted from the design and simulation data to form the training data, which is passed to the ML model. Based on the trained model, results for other data can be predicted. However, the accuracy and performance of the overall approach depends on the underlying ML algorithm. In order to find the best suited approach for the targeted application, different approaches were evaluated. These are briefly described below [9]:

A. Linear Least-Square Regression

Linear Least-Square (LLS) regression is a basic and widely used modeling method in machine learning [9]. The LLS method generates a model by the linear approximation technique for the estimation function. The error (residual sum of squares) between the predicted values and actual values is minimized to generate the coefficients $w = (w_1, \ldots, w_m)$ for the model. During the training period, the error of the predicted value is minimized. i.e., the goal is given by:

$$\min_w ||Xw - y||^2$$

The model generated is sensitive to random errors if the matrix X becomes singular. The model terms are independent of the estimated coefficients. If the columns of the design matrix X have linear dependencies due to correlated terms then there is the problem of multi-collinearity [5], [22]. Due to this, the error can be large while predicting multiple values.

B. Ridge Regression

In order to minimize the error of LLS based methods, the *Ridge Regression* algorithm has been introduced [18]. Here, an additional parameter to compensate the error is used. The additional parameter, known as ridge coefficient, reduces the error by imposing a penalty on the size of the coefficients. The cost function to minimize the error becomes:

$$\min_w ||Xw - y||^2 + \alpha||w||^2,$$

where α is the tuning parameter that controls the strength of the penalty.

C. Nearest Neighbors Regression

The parametric-based regression methods described above are difficult to balance. Hence, we also used a non-parametric approach, i.e., *K-Nearest Neighbors* (KNN) regression, which identifies the training observation T_o, that is nearest to the prediction point P_o [2]. This method further estimates the final value, i.e., $V(P_o)$, by averaging the responses around T_o. The formal representation is given by:

$$V(P_o) = \frac{1}{K} \sum_{x_i \in T_o} y_i,$$

where the optimum value of K depends on a bias-variance tradeoff. The value of K can be specified during the execution or can be estimated. Usually, a small value of K provides a low bias but a high variance, and vice-versa.

D. Neural Network Regression

A *Neural Network*-based learning technique, namely Multi-Layer Perceptron (MLP), has been proposed in order to improve the accuracy of the predicted result. In this prediction model, the non-linearities are hidden within layers between inputs and outputs. During training, the MLP-based learning algorithm predicts a function $f(\cdot) : R_i \rightarrow R_o$, where i is the number of dimensions for the inputs and o is the number of dimensions for the outputs. If the training data is labeled i.e. has both the feature $X = x_1, x_2, ..., x_m$ and a target y, MLP can predict a non-linear function approximator for the regression.

V. Application

This section describes the problem formulation as a learning-based application. A test vector t is a sequence of Boolean values $\{0, 1\}$. As described before, a set of training vectors T_t is used to train a model f. Simulation is conducted for this small subset in order get a power profile of each test. The proposed methodology is independent of the estimation metric, i.e., Weighted Switching Activity (WSA) or other more accurate power dissipation metrics can be used. The simulation data is used to train the model f.

The first application is the prediction of the total (global) power profile. The power profile P_t is defined as a function

$$P_t = w_1(n_1) + w_2(n_2) + \ldots + w_m(n_m),$$

where $w_1, \ldots, w_m$ are the coefficients representing the power metric value of their corresponding node $n_1, \ldots, n_m$. The data is obtained by the accurate (time-consuming) simulation of the training test vectors for the training purpose.

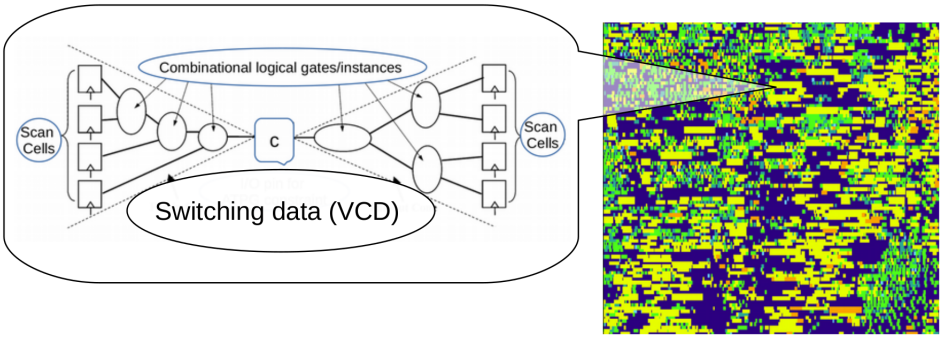

Fig. 3: Test simulation and analysis.

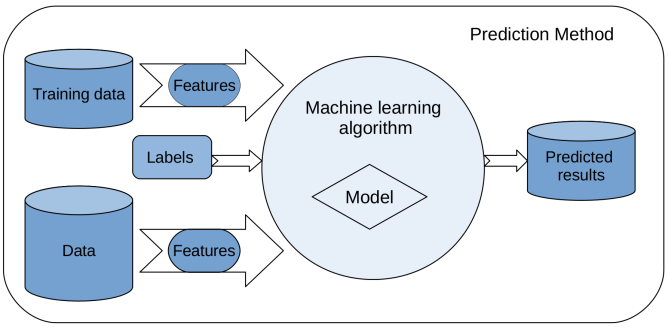

Fig. 4: Prediction method.

During training, the learning algorithms, which are used as a black box, learn a function f_g to predict the value of P_t depending on the logic values at the input nodes of the circuit, i.e., the scan cells:

$$f_g(P_t) = f(v_1, v_2, ..., v_k),$$

where $v_1, v_2, ..., v_k$ represent the value of the input nodes. This is done by continuous refinement of $f_g(P_t)$ during training. For the i-th training example and the j-th node, the error e is internally represented as $e_j(i) = t_j(i) - p_j(i)$, where t represents the actual training value and p is the predicted value of this node. Depending on the learning algorithm used, a mathematical optimization function is applied to $f_g(P_t)$ in order to minimize the error between the predicted value and the original value, e.g. by adjusting weights.

After the training phase, the function $f_g(P_t)$ is applied to the other test vectors T_p to predict their power profile based on the trained model $f_g(P_t)$.

The second application is the prediction of the power distribution over the layout (local power). In order to apply the learning algorithms, the coordinates of the nodes x, y in the layout have to be included in the learning process.

$$f_l(P_t(x, y)) = f(w_1(x_1, y_1), w_2(x_2, y_2), ..., w_n(x_n, y_n))$$

Therefore, a node n is replaced with its location (x, y). By doing this, the layout of the circuit is taken into account for the prediction of the activity of each node. The training and the prediction of the test vectors is done in a similar way as described above. The outcome of the prediction process is then the switching activity related to the location.

The predicted local data can then be processed in the same way as the original accurate data in the test validation flow. For example, it can be fed into EDA tools or clustering algorithms, where hot spots can be identified based on the calculated data.

TABLE I: Benchmark 'ac97_ctrl'

#Alg	#TS	#PS	Err(%)	#Var	#T(s)	#Tr(ms)	#Bt	#Avg	#Bs
NNwk	10	80	10.40	-1.30	35.83	50.16	42	16	22
	20	70	6.01	-0.39	33.88	82.92	35	22	13
	40	50	5.57	-0.54	31.89	40.00	30	14	6
	60	30	5.08	-0.17	31.38	46.72	26	2	2
	80	10	3.96	-0.58	29.87	37.81	8	0	2
LSR	10	80	18.03	-13.06	35.20	20.16	27	19	34
	20	70	10.19	-0.63	34.76	20.85	20	23	27
	40	50	5.71	-0.66	32.88	22.44	18	20	12
	60	30	5.54	-0.34	30.59	22.35	10	12	8
	80	10	5.25	-0.42	29.55	31.14	3	4	3
NNbr	10	80	10.19	-0.44	37.22	22.89	29	21	30
	20	70	6.02	-0.41	33.72	22.68	15	32	23
	40	50	5.08	-0.54	32.32	25.05	16	20	14
	60	30	4.16	-0.52	31.51	25.99	11	11	8
	80	10	4.08	-0.49	29.89	24.983	4	3	3
RR	10	80	10.142	-0.52	35.12	308.45	33	24	23
	20	70	5.72	-0.63	34.49	565.5	21	25	24
	40	50	5.30	-0.62	33.44	1354.4	20	21	9
	60	30	5.24	-0.33	32.29	1878.6	11	12	7
	80	10	4.52	-0.41	32.08	2375.3	6	2	2

VI. Experimental Results

The experiments were performed on IWLS benchmark circuits [1]. Commercial tools were used for test pattern generation as well as for the accurate simulation. The test patterns, the simulation database, the technology files and the design files were processed using an in-house tool to generate the data set and extract the features for the learning stage. The application of the ML algorithms was done in Python using publicly available algorithms [5], [22]. The capture cycle transitions were considered for the simulation of the data. The method can also be used for other simulated time cycles such as shift-in and shift-out. Note that this work does not provide new metrics to assess power-criticality, but it relies on the data generated by the underlying simulation method. The (power) simulation methods can be easily interchanged.

The predicted results of the global power of the test patterns are shown in Table I, II, III and IV for all the described learning algorithms. The concrete results for each pattern are not given due to space limitation. The mean error values and the variance of the algorithms are used to compare the results effectively. The '#Alg' columns provide the information about the algorithm, where 'NNwk', 'LSR','NNbr' and

TABLE II: Benchmark 'wb_conmax'

#Alg	#TS	#PS	Err(%)	#Var	#T(s)	#Trn(ms)	#Bt	#Avg	#Bs
NNwk	10	167	6.52	-8.92	52.96	68.40	89	46	32
	20	157	4.94	-5.50	49.11	104.06	86	43	28
	40	137	3.25	-0.49	44.95	88.14	79	40	18
	60	117	3.15	-0.33	43.68	91.00	68	34	15
	80	97	3.13	-0.13	40.51	66.14	57	28	12
	120	57	2.85	-0.28	47.27	55.66	30	22	5
LSR	10	167	9.73	-10.03	40.43	35.24	68	34	65
	20	157	5.03	-2.13	48.67	63.72	56	49	52
	40	137	4.25	-1.29	50.62	44.28	51	47	39
	60	117	3.83	-0.49	40.49	41.14	48	44	25
	80	97	3.69	-0.83	43.78	53.23	44	35	18
	120	57	3.23	-0.53	50.79	54.41	24	19	14
NNbr	10	167	6.69	-4.49	49.56	49.53	75	39	53
	20	157	4.03	-1.12	50.26	53.17	71	38	48
	40	137	3.78	-0.48	43.69	45.23	54	41	42
	60	117	3.64	-0.88	49.56	49.35	50	39	28
	80	97	3.51	-0.73	44.66	51.25	45	35	17
	120	57	3.22	-0.53	51.39	48.49	22	23	12
RR	10	167	3.23	-0.39	43.94	1898.72	79	37	51
	20	157	3.15	-0.14	41.54	3124.51	74	36	47
	40	137	3.11	-0.46	45.90	3286.66	58	39	40
	60	117	2.89	-0.33	49.30	2578.15	52	42	23
	80	97	2.85	-0.28	50.46	2766.22	47	34	16
	120	57	2.80	-0.27	48.12	1277.11	24	23	10

'RR' represent 'Neural Network' , 'Least Square', 'Nearest Neighbor' and 'Ridge' regression techniques, respectively. The two columns '#TS' and '#PS' give the number of training tests T_t as well as the number of predicted tests T_p.

The size of T_t has an effect on the predicted results. Therefore, experiments with training test sets of different sizes are done to show the impact. The mean error percentage and variance are shown in the '#Err' and '#Var' columns, respectively. The variance need not always be the squared value of the distribution. In the regression methods, it is the deviation of the prediction model in distribution on predicted values [22]. The next two columns '#T' and '#Tr' give the total run time and the training time, respectively. The result of the test patterns are categorized in three groups. Each group represents a prediction accuracy interval:

- '#Bt': Prediction accuracy of 95% and above (Best)
- '#Avg': between 95% and 85% (Average)
- '#Bs': between 85% and 75% (Base)

The actual accurate values were only generated in order to assess the quality of the prediction. In practice, only the training vectors have to accurately analyzed.

The results show that with a small training set of 20-40 test vectors, a very low error rate can be achieved. Increasing the number of training vectors shows further improvements in the prediction accuracy.

It can be seen that the Neural Network-based approach and the Ridge Regression-based approach are the most accurate prediction approaches, i.e., they have the lowest error rates. Since the Neural Network-based approach is significantly faster and also has more tests in the 'Best' category, this approach is recommended for actual use.

In another experiment, we used the learning algorithm for local power prediction. The results are shown in Table V. Due to page-count limitation, we only show the results for the preferred Neural Network-based approach. It can be seen that the prediction is also effective for local power.

TABLE III: Benchmark 'ethernet'

Alg	TS	PS	Err(%)	Var	T(s)	Tr(ms)	Bt	Avg	Bs
NNwk	10	90	4.83	-0.35	22.19	61.24	44	19	27
	20	80	4.71	-0.09	21.96	48.69	37	18	25
	40	60	4.62	-0.13	23.90	57.20	32	14	14
	60	40	4.51	-0.03	24.07	39.15	20	12	8
	80	20	4.50	-0.03	23.63	43.35	12	5	3
LSR	10	90	18.03	-13.06	23.55	24.31	34	18	38
	20	80	4.99	-0.44	18.82	25.89	35	18	27
	40	60	4.73	-0.09	20.32	24.35	27	15	18
	60	40	4.64	-0.14	20.41	43.67	15	9	16
	80	20	4.51	-0.04	22.26	23.45	7	3	10
NNbr	10	90	7.44	-1.64	20.61	22.44	35	24	31
	20	80	7.29	-1.49	22.47	22.85	28	28	24
	40	60	5.83	-1.16	18.66	24.71	24	21	15
	60	40	4.53	-0.01	22.41	21.99	15	11	14
	80	20	3.91	-0.38	33.71	21.00	8	3	9
RR	10	90	4.83	-0.35	25.14	6689.76	36	22	32
	20	80	4.71	-0.09	27.13	5050.29	37	18	25
	40	60	4.62	-0.13	25.31	3340.07	30	13	17
	60	40	4.62	-0.13	25.31	3340.07	18	9	15
	80	20	4.50	-0.03	24.85	870.17	10	3	7

TABLE IV: Benchmark 'pci_bridge'

Alg	TS	PS	Err(%)	Var	T(s)	Tr(ms)	Bt	Avg	Bs
NNwk	10	184	5.97	-0.91	38.19	78.13	97	52	38
	20	174	5.81	-1.00	36.06	82.608	95	47	32
	40	154	4.94	-0.73	47.77	104.773	82	48	24
	60	134	4.79	-0.66	47.24	72.481	74	41	19
	80	114	4.38	-0.45	48.67	57.620	65	32	17
	120	74	3.89	-0.22	47.24	56.690	43	18	13
	150	44	3.56	-0.09	44.34	57.516	28	8	8
LSR	10	184	5.96	-0.91	45.16	65.26	83	49	52
	20	174	5.80	-1.00	43.68	77.52	81	47	46
	40	154	4.94	-0.73	49.50	69.97	75	49	30
	60	134	4.86	-0.70	52.47	71.82	71	38	25
	80	114	4.47	-0.50	52.68	72.08	61	34	19
	120	74	3.89	-0.22	54.26	51.54	36	20	18
	150	44	3.57	-0.09	48.11	56.30	12	18	14
NNbr	10	184	4.45	-0.44	44.16	62.01	84	50	50
	20	174	4.20	-0.23	42.10	59.86	82	49	43
	40	154	3.54	-0.08	76.45	72.74	76	48	29
	60	134	3.54	-0.01	57.31	68.00	74	35	25
	80	114	3.45	-0.05	42.62	48.40	58	36	20
	120	74	3.45	0.00	55.38	47.88	35	22	17
	150	44	3.36	-0.01	48.12	60.95	12	20	12
RR	10	184	6.29	-0.96	46.43	8368.61	88	48	48
	20	174	6.17	-1.07	45.79	10293.27	86	47	41
	40	154	5.20	-0.75	45.95	5797.38	77	50	27
	60	134	5.05	-0.68	46.51	5538.73	75	35	24
	80	114	4.61	-0.47	46.13	3407.77	64	35	18
	120	74	4.10	-0.22	47.88	1439.64	43	20	16
	150	44	3.75	-0.09	44.76	738.67	14	19	11

The training data set is important for the tuning of the generated model. Typically, the larger the training test is, the smaller is the error between the predicted value and the actual value. Another experiment was carried out to evaluate the impact of the training data selection on the prediction results in terms of error and variance. It has been observed that the predicted values differ for different training sets because of the change in the model, i.e., a change in the coefficients of the learned equations. Still, the overall error was less than 4% for all training sets. The selection of an optimal training set is therefore left for future work.

In general, the results show that ML-based learning algo-

TABLE V: Prediction for local power activity of 'pci_bridge' benchmark

Algorithm	Training set	Prediction set	Error(%)	Variance	Total time(s)	Training time(s)	Best	Average	Base
Neural Network	10	184	3.1708	0.1167	38.186638	0.078128	128	27	29
	20	174	2.9552	0.3433	36.063363	0.082608	125	26	23
	40	154	2.9312	0.3217	44.342213	0.057516	118	20	16
	60	134	2.9243	0.2987	48.677432	0.057620	102	17	15
	80	114	2.9221	0.3851	47.772351	0.104773	85	15	14
	120	74	2.9144	0.4236	47.248791	0.072481	49	14	11
	150	44	2.9092	0.2805	47.249760	0.056690	31	7	6

rithms are well suited to predict the power profile of a test. The number of training tests influences the prediction quality. The Neural Network based learning technique was found to be the best choice compared to the other algorithms in terms of overall accuracy as well as run time.

VII. Conclusion and Future Work

It is difficult to carry out the power analysis for all tests due to practical time and resource constraints. The accurate simulation of the complete test set for complex circuits is not possible. We have proposed the use of machine learning-based techniques to predict the test power of test patterns without explicitly simulating them. The detailed analysis results of a few tests are used to train a model. The model is then applied for the prediction of the power profile of the remaining tests to identify potentially power-unsafe tests. The method is found to be effective in predicting the test power profile of most tests. The error is less than 5% when we rely on the accurate simulation data of a few training vectors. Experimental results show that hours of run time for the actual analysis is reduced to seconds with the help of the prediction techniques. Here, the Neural Network-based learning algorithm has been found to be the best in terms of run time and error rate. Overall, it has been shown that machine learning-based prediction is an effective alternative to the accurate simulation of all test patterns to identify power-critical tests when the latter is infeasible. In future work, more parameters such as time-dependent features will be considered to improve the accuracy further.

VIII. ACKNOWLEDGMENT

The work has been supported by the Deutsche Forschungsgemeinschaft (DFG) under contract number EG 290/5-1 and SFB 1232 in subproject P01 Predictive function under Project 276397488 by German Research Foundation. The work of Krishnendu Chakrabarty was supported in part by a Humboldt Research Award from the Alexander von Humboldt Foundation.

References

[1] C. Albrecht, "IWLS 2005 benchmarks," Tech. Rep., Jun. 2005.

[2] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[3] S. Bhanja and N. Ranganathan, "Switching activity estimation of vlsi circuits using bayesian networks," *IEEE Trans. on VLSI Systems*, vol. 11, no. 4, pp. 558–567, 2003.

[4] ——, "Cascaded bayesian inferencing for switching activity estimation with correlated inputs," *IEEE Trans. on VLSI Systems*, vol. 12, no. 12, pp. 1360–1370, 2004.

[5] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.

[6] R. Burch, F. N. Najm, P. Yang, and T. N. Trick, "A monte carlo approach for power estimation," *IEEE Trans. on VLSI Systems*, vol. 1, no. 1, pp. 63–71, 1993.

[7] M. Chadha, T. Ilsche, M. Bielert, and W. E. Nagel, "A statistical approach to power estimation for x86 processors," in *International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. IEEE, 2017, pp. 1012–1019.

[8] R. Chandramouli and V. K. Srikantam, "Multimode power modeling and maximum-likelihood estimation," *IEEE Trans. on VLSI Systems*, vol. 12, no. 11, pp. 1244–1248, 2004.

[9] M. B. Christopher, *PATTERN RECOGNITION AND MACHINE LEARNING*. Springer-Verlag New York, 2016.

[10] Y. A. Durrani and T. Riesgo, "Statistical power estimation for ip-based design," in *IEEE Industrial Electronics, IECON -32nd Annual Conference on*, 2006, pp. 4935–4939.

[11] Y.-C. Fang, H.-Y. Lin, M.-Y. Su, C.-M. Li, and E. J.-W. Fang, "Machine-learning-based dynamic ir drop prediction for eco," in *Int'l Conf. on CAD*. ACM, 2018, p. 17.

[12] P. Girard, N. Nicolici, and X. Wen(Eds.), *Power-Aware Testing and Test Strategies for Low Power Devices*. Springer, 2009.

[13] S. Gupta and F. N. Najm, "Analytical models for rtl power estimation of combinational and sequential circuits," *IEEE Trans. on CAD of Integrated Circ. and Systems*, vol. 19, no. 7, pp. 808–814, 2000.

[14] L. Hou, X. Wu, and W. Wu, "Neural network based power estimation on chip specification," in *International Conference on Information Sciences and Interaction Sciences (ICIS)*. IEEE, 2010, pp. 187–190.

[15] L. Hou, L. Zheng, and W. Wu, "Neural network based vlsi power estimation," in *International Conference on Solid-State and Integrated Circuit Technology, ICSICT*. IEEE, 2006, pp. 1919–1921.

[16] S.-Y. Lin, Y.-C. Fang, Y.-C. Li, Y.-C. Liu, T.-S. Yang, S.-C. Lin, C.-M. Li, and E. J.-W. Fang, "IR-drop prediction of ECOs-revised circuits using machine learning," in *VLSI Test Symp.*, 2018, pp. 1–6.

[17] K. Miyase, M. Sauer, B. Becker, X. Wen, and S. Kajihara, "Identification of high power consuming areas with gate type and logic level information," in *IEEE European Test Symp.*, 2015, pp. 1–6.

[18] K. P. Murphy and F. B. (Ed.), *Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning series)*. MIT Press, 2012.

[19] A. K. Murugavel, N. Ranganathan, R. Chandramouli, and S. Chavali, "Least-square estimation of average power in digital cmos circuits," *IEEE Trans. on VLSI Systems*, vol. 10, no. 1, pp. 55–58, 2002.

[20] V. Narayanan, N. Dhanwada *et al.*, "A power estimation methodology for systemc transaction level models," in *Third IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis,CODES+ ISSS*, 2005, pp. 142–147.

[21] N. Nicolici and B. Al-Hashimi, *Power-constrained testing of VLSI circuits*. Kluwer Academic Publishers, Boston, MA, 2003.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[23] G. Qu, N. Kawabe, K. Usami, and M. Potkonjak, "Function-level power estimation methodology for microprocessors," in *Design Automation Conf.*, 2000, pp. 810–813.

[24] V. Saxena, F. N. Najm, and I. Hajj, "Monte-carlo approach for power estimation in sequential circuits," in *Design, Automation and Test in Europe*, 1997, p. 416.

[25] L.-T. Wang, C.-W. Wu, and X. Wen, *VLSI Test Principles and Architectures*. Elsevier, 2006.

[26] Q. Wu, Q. Qiu, and M. Pedram, "Estimation of peak power dissipation in vlsi circuits using the limiting distributions of extreme order statistics," *IEEE Trans. on CAD of Integrated Circ. and Systems*, vol. 20, no. 8, pp. 942–956, 2001.