Failures in Embedded Systems using Long Short-Term Inference.

Snehasis Dey
College of Engineering Bhubaneswar

Abstract— **Customers of cyber-physical and embedded systems anticipate dependable performance in a wider range of settings and applications.Reactive self-diagnosis methods either don't stop catastrophic failures or employ unduly strict guardbands. In this letter, we describe how we designed a prediction engine using machine-learning approaches to anticipate on-device failures in embedded systems. We assess the performance of our prediction engine in forecasting temperature behavior on a mobile system-on-a-chip and suggest a workable hardware solution for the use-case.**

I. Introduction

THE COMPLEXITY of embedded system platforms and theapplicationstheysupportarecontinuouslyincreas- ing:theyrunlargeandevolvingapplicationsonheterogeneous multi- or many-core processing platforms. Examples include automated and autonomous driving, smart buildings, industry 4.0, and personal medical devices. Such systems are required to provide dependable operation for the user while dealing with a large number of internal and external variabilities, threats, and uncertainties in their lifetimes.

To provide such dependable operation, self-diagnosis tech- niques are developed for early detection of degradation and imminent failures, in order to maximize system life-cycle. Thesetechniquescanbecombinedwithunsupervisedplatform self-adaptation to meet performance and safety targets. Self- diagnosistechniquesthatarereactivemay:1)notbesufficient to address catastrophic failures; or 2) take overly conservative approaches that hinder performance.

Forexample,considerthermalmanagementofanembedded system-on-chip (SoC). One technique is to define a temper- ature threshold and throttle performance [e.g., via dynamic voltage-frequency scaling (DVFS)] when the threshold is exceeded. This approach is reactive and must act conser- vatively to prevent overheating. The conservative frequency throttling may degrade performance potentially unnecessarily. If the temperature behavior could be predicted, a proactive approach could manage the temperature without sacrificing performance excessively. However, system dynamics, such as temperature, can behave nonlinearly, and are hard to predict without workload knowledge.

Machine learning techniques, such as neural networks, are useful for identifying complex system dynamics. However, neuralnetworksarecomplexanddifficulttodeployonpower- constrainedembeddedsystems.Inthisletter,weproposea failure prediction technique for embedded systems using longshort- termmemory(LSTM),atypeofrecurrentneu- ral network (RNN). We demonstrate the effectiveness of our predictorforpredictingtemperaturebehaviorwithrespecttoa threshold on an ODROID-XU3 [9] platform, making it a can- didate for mitigating overheating failures and implementing efficient control policies. We specify an implementation thatisrealizibleinhardwareonlow-powerembeddedsystems.The specific contributions are as follows.

1) We propose a method for hardware hazard prediction called long short-term prediction model.
2) We propose an architecture and hardware implementa- tion of nonintrusive prediction engine based on long short-term prediction model to predict temperature behavior in the embedded systems.
3) We evaluate the predictor using measured temperature data from an ODROID XU-3.

II. BackgroundandRelatedWork

When modern SoCs operate near peak performance for extended periods, power dissipation can increase the tempera- ture to the point that adversely impacts the chip reliability. If we can provide proactive thermal management, we can avoid potentially dangerous execution scenarios. Proaction requires prediction. Anumber of strategies have been proposed for on- chipthermalprediction,andthemethodscanbeclassifiedinto two categories.

The first prediction method builds models based on mea- sured temperature and power consumption [14], [15],[17],[19], [21]. The second method builds the prediction model indirectly using equations, without thermal measurements [4],[5],[7]. However, there have been many successful appli- cations of machine learning techniques employed in failure detection or prediction of large-scale systems. With suf-ficient sensor input, machine learning models can extract complex or subtle dynamics, potentially resulting in accurate predictions when applied to new execution scenarios. Failure prediction has been proposed using support vector machines (SVMs)[3],[10],convolutionalneuralnetworks(CNNs)[16], and a combination of techniques [8].

RNNsarenaturallysuitedforlearningtemporal sequencesandmodelingtimeseriesbehaviors.RNNs have been applied to predict various behavior in large-scale systems [6],[13],[20]. Lima *et al.* [6] compared an RNN solution with an LSTM solution and observed that LSTMs significantly outperform RNNs in terms of accuracy.

In[2],[11],and[18],LSTMsareusedinotherdomains fortimeseriespredictions,suchaswaterqualityestima- tion, stock transaction prediction, mechanical states, etc. The authors compared the LSTM networks with alternatives, such asbackpropagationneuralnetworks,onlinesequentialextreme learning machines, and support vector regression machines (SVRMs), and demonstrated the superiority of LSTMs.

III. Contributions

We propose a method for predicting runtime behavior in hardware: the long short-term prediction engine. In this sec- tion, we describe how our predictor is composed by walking through our use-case: predicting runtime temperature behav- ior on an embedded SoC. Our goal is to predict temperature behavior such that critical thermal scenarios can be detectedin advance and avoided with a solution that can feasibly be integrated in an embedded SoC. Our SoC consists of four ARM A15 cores, with shared L2 cache connected via bus.We measure total power and temperature of the entire core cluster, as well as per-core utilization. To generate workloads, we use a synthetic microbenchmark [12] that is configurable. Themicrobenchmarkisabletostressthearchitectureinawide rangeandwegenerateda"general-purpose"workloadbyexe- cuting the microbenchmark in phases that exercised different behavior in these various dimensions. We execute different sequences on multiple cores to emulate different applications to train the model and test its performance. The prediction engine consists of two parts: 1) a short-term binary model;and 2) a long-term regression model. The short-term binary model makes precise predictions quickly, useful for subtle changes, i.e., anticipating violations of a temperature thresh- old. The long-term regression model can make a prediction further in advance, useful to predict general behavior in less- critical scenarios, i.e., predicting temperature trends in a safe state.

A. *Short-TermBinaryModel*

The short-term binary model is used to predict unwanted behavior, i.e., constraint violation. In our case, in which, we have a temperature threshold we do not want to violate, the short-term binary model is utilized when the measured tem- perature is nearing the threshold. In this scenario, a slight rise in temperature will cause a failure (violation of constraint), thereby, it is important to have a high recall rate. The recall ratemustbetunedcarefullytobalanceaccuracyandoverhead.

*1) Model Definition:*Our initialshort-termbinary model is defined as follows.

1) *Input:*Temperature,coreutilization,power.

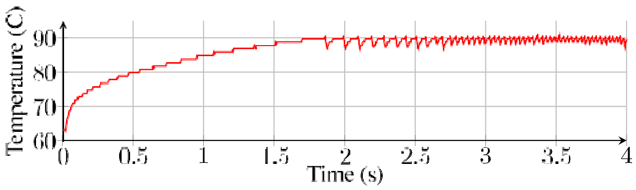


Fig. 1.Temperature data collected from the ODROID XU-3 executing acombination of synthetic microbenchmarks.

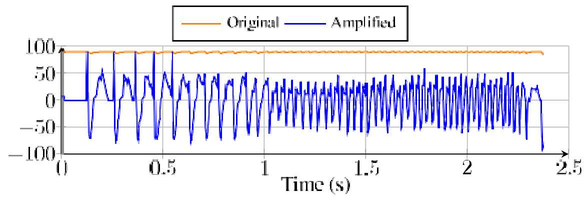


Fig. 2.Temperature data amplified using sliding average amplification. Wefocus on data above 85 °C (critical temperature).

2)*Output:*Probabilityoffailure(afterboundarylimitation, the model produces a binary result: "0" refers to normal and number "1" refers to failure).

*2) ModelTraining:*Fig.1showsthemeasuredtemperature data from the ODROID-XU3.[1] We first isolate the data above the critical point (85 °C) to use as the training data. Because the range of the data is reduced, we amplify the changes of data to increase its variation. When performing amplification at runtime, we must consider constraints such as the real-time hardware implementation and the short failure intervals. We create a method called sliding average amplification to efficiently preprocess data in order to increase variation and applied it on the four features. The method takes local data (five timesteps) and uses min–max normalization to amplify the values. The following equations show the calculation of sliding average amplification. *D(t)*refers to the feature value at *t* and *i* refers to the number of timesteps defined as localdata

$$\text{average}(t) = \frac{1}{n}\sum_{i=0}^{n} D(t-i) \tag{1}$$

$$\max(t)=\text{MAX}\{D(t-i),D(t-i+1),\ldots,D(t)\} \tag{2}$$

$$\min(t)=\text{MIN}\{D(t-i),D(t-i+1),\ldots,D(t)\} \tag{3}$$

$$\text{amplified}(t)=\frac{D(t)-\text{average}(t)}{\max(t)-\min(t)}\times 100. \tag{4}$$

Fig.2showstheamplifieddataalongwiththeoriginal.The orange curve is the original data and the blue curve is the amplified data.

*3) Improved Loss Function:*Our initial binary model still has a significant issue: it is trained with imbalanced data. Normal samples (i.e., noncritical temperatures) account for nearly 99.5 % of the training data. Due to the low ratio of failuresamples(i.e.,criticaltemperatures),themodelishighly confident in identifying critical samples, which is misleading. Weaugmenttheclassicbinarycross-entropylossfunctionwith weights in order to increase model sensitivity to normal sam- ples.$y$isthepredictedvalueand$\hat{y}$ istheactualvalue.The

[1]Ouruse-casesystem,containingthedescribedSoC.

weight factor $\alpha$is determined empirically based on the rate of failure samples in the training data

$$\text{Loss}=-(\alpha y\log\hat{y}+(1-\alpha)(1-y)\log(1-\hat{y})) \tag{5}$$

$$\alpha=0.992. \tag{6}$$

*4) Model Structure:*We propose the simplest structure of an RNN prediction model that provides the required accu-racy in order to minimize the hardware overhead. The LSTM internalstructureisdefinedinthefollowingequations.$x$refers totheinputfeatures,$h$istheoutputresult,W,$b$aretheweights and bias, and c are the intermediate variables

$$i_t=\sigma(W_{xi}x_t+W_{hi}h_{t-1}+b_i) \tag{7}$$

$$f_t=\sigma(W_{xf}x_t+W_{hf}h_{t-1}+b_f) \tag{8}$$

$$o_t=\sigma(W_{x0}x_t+W_{h0}h_{t-1}+b_0) \tag{9}$$

$$\tilde{c}_t=\tanh(W_{xc}x_t+W_{hc}h_{t-1}+b_c) \tag{10}$$

$$c_t=f_t\odot c_{t-1}+i_t\odot\tilde{c}_t \tag{11}$$

$$h_t=o\odot\tanh(c_t). \tag{12}$$

Fig. 3 (black and blue) illustrates the architecture of the proposedRNN/LSTMmodelwhichcontainstwoRNN/LSTM layers (the RNN and LSTM structure provide comparable accuracy,showninSectionIV):onefullyconnectedlayerand one binary classification layer based on sigmoid activation. The input features are time sequences of temperature, per-core utilization, and power. After calculation of time step t in thefirst layer, the result is conveyed to step $t+1$ in the same layerandthestep$t$inthesecondlayer.Atthesametime, step$t+1$ data is added into the next step calculation. In each RNN/LSTMlayer,thereare8timestepsand64hiddenlayers. In the last time step, the result is passed to a fully connected layer and a sigmoid layer for classification. The output resultis the failure probability. When the value is greater than 0.9, we define it as failure and output 1.

B. Long-TermRegressionModel

The long-term regression model is used to predict behav-iorinthenormalstate.Inthisstate,temperaturevariesin a large range depending on how the system is being exer- cised. Our goal is to predict the temperature sufficiently in advance to make runtime decisions in order to avoid critical states completely while also optimizing performance. In order toproactivelyavoidcriticalstateswithoutunnecessarilysacri-ficingperformance,itisnecessarytoensurethattheprediction engine can be applied during normal execution. As the system state is noncritical, precision can be sacrificed for universal-ity. To this end, we build a regression model for long-term prediction.

1) ModelDefinition:

1) *Inputs:*Temperature, power,per-coreutilization.
2) *Outputs:*Temperature.

*2) ModelTraining:*Inthiscase,weutilizealargerrangeof training data (60 °C—85 °C). We observe temperature varia-tion generally due to change in core utilization and operating frequency. We categorize training workloads as follows: uni-core,multicore,andshifting.Weexecutecombinationsof

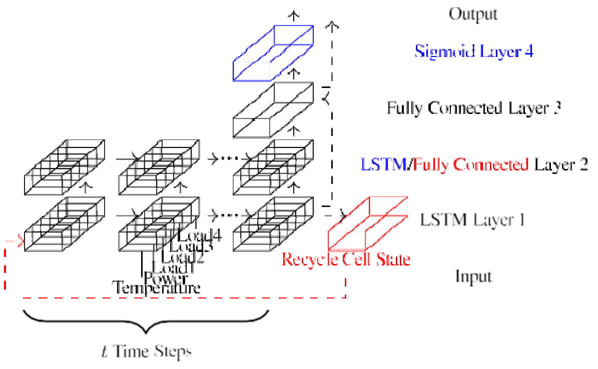


Fig. 3.Integrated model structure. The structures are shared between theshort-term binary model and the long-term regression structure, depending onwhich is active. Functionality and structure specific to the short-term binarymodel is in blue, and specific to long-term regression model is in red.

synthetic benchmarks to compose our workloads. The bench-marks vary in instructions-per-cycle (IPC), utilization, and cache miss rate, exercising the processor in a wide range.

Forunicore workloads,we first run eachbenchmarkon one core to emulate stable workload state. Then, we combine multiple benchmarks and start them one by one to emulate changingworkloadstateononecore.Formulticoreworkloads, we assign different benchmarks on different cores and start them simultaneously. For shifting workloads, we assign the samebenchmarksondifferentcoresandstartthematdifferent times.

Raw data collected from the ODROID-XU3 does not ini-tially appear stable, making filtering essential.[2] After trying several filters to smooth the raw data and considering the hardware feasibility, we conclude that the data preprocessed by recursion average filter produces the most accurate model. Filter sizes of each input are determined empirically.

*3) Model Structure:*LSTM has the nature of storing long-termmemory,therefore,todealwiththelong-termcases, we choose LSTM structure for our model. Compared to a short-termmodel,increasedhistoricaldataisneededtoensure precision when predicting a large temperature range far in advance. This leads to increased model time step and exe- cution time. Therefore, we apply a stateful LSTM theory inthe cell structure, fitting output cell state as the initial state. In this way, the structure can remember long-term memory and better adapt.

Fig. 3 (black and red) illustrates the architecture of the proposed LSTM model. The input features aretimesequences of temperature, per-core utilization, and power. After calcula-tionofstep$t$,thecellstateisrecycledtonexttermcalculation. There are 8 time steps in the LSTM layer and 64 hidden lay- ers in each cell. We need 16 previous steps for prediction, therefore, the cell state will be passed for initialization every second iteration.

C. HardwareImplementationFramework

Tointegratetheshort-andlong-termmodels,wespecify a single shared-hardware implementation that supports all of Fig. 3. A judgement module receives temperature values from

[2]Data is stored in a userspace buffer, sampled from sensors via kerneldrivers every 5 ms.

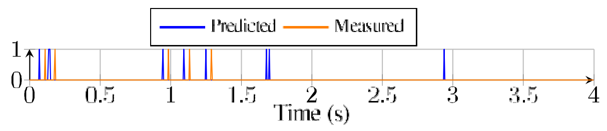


Fig. 4.Sample prediction of one workload. Binary events (i.e., experiencingcritical temperature) are predicted and observed.

thesensoranddecideswhichmodeltoactivate.Iftemperature is ≥85 °C, the short-term prediction model is activated and its weights are loaded into the model structure. If it is <85 °C,the long-term prediction model is activated.

To reduce the structural overhead, the core LSTM and fully connected layers are partially shared, composed with the least common parameters (LSTM: 8 time steps, 64 hidden layers; fully connected: 4 hidden layers). The excess time steps canbe stored in a state buffer and fed back (Fig. 3).

Using the LSTM implementation of Chang *et al.* [1], we calculate 12960 FFs, 7201 LUTs, and 16 BRAM overhead. The LSTM hardware is 20 times faster than the Zync ZC7020 ARM-based hard-core processor (4.4 μs per inference), 44 times more power efficient than a software implementation with the Zync ZC7020 (performance-per-watt).

IV. Evaluation

We evaluate the effectiveness of both our short-term binary model and long-term regression model separately, using additional measured data from the ODROID-XU3. The measured data consists of the model input data measured at 5 ms intervals. We perform sensitivity analyses of LSTM/RNN models for different parameters and structures.

A. Short-Term BinaryModelEvaluation

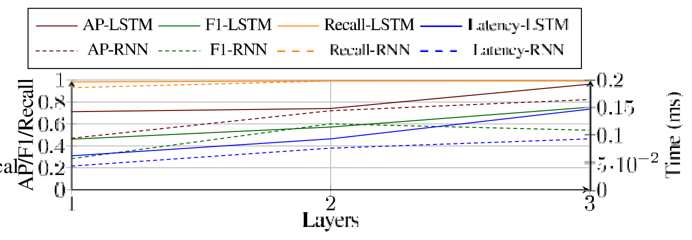
*1) Evaluation Metrics:*The output of the short-term binary model is a binary classification. We evaluate the model by average precision score (AP) and F1-score. The average precision score summarizes a precision–recall curve as the weighted mean of precision achieved at each recall thresh-old, with the increase in recall from the previous threshold used as the weight

$$\text{AP} = \sum_n (R_n - R_{n-1}) P_n \tag{13}$$

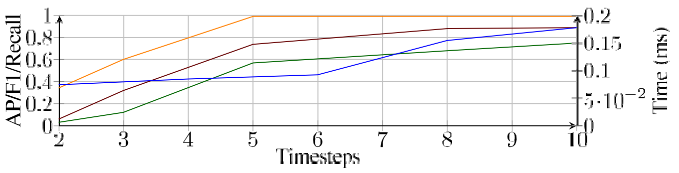
where $P_n$and $R_n$are the precision and recall at the nth thresh-old. F1-score is a measure of a test's accuracy and is defined as the weighted harmonic mean of the precision and recall of thetest.F1-scoreconveysabalancebetweenprecision(P)and recall (R)

$$F1 = \frac{2\times P\times R}{P+R}. \tag{14}$$

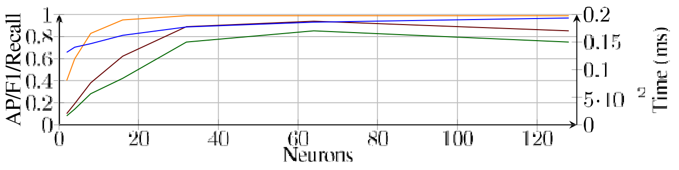
*2) Evaluation Results:*The model can predict up to 8 steps (40ms)ahead.TheF1-scoreis0.43andtheAPscoreis 0.78.Thelatencyofshort-termbinarymodelis0.088ms (based on execution in Python, no hardware acceleration).Fig. 4 shows the prediction result of one dataset. The orange showsmeasuredfailuresandtheblueshowspredictedfailures. Observethattherearanumberofmispredictedfailures(false positives). This is preferable to false negatives (nonpredicted



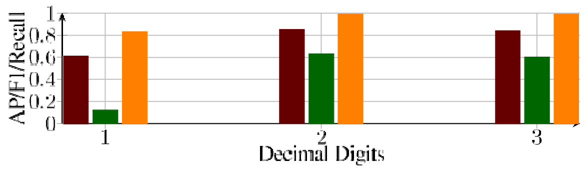
(a)



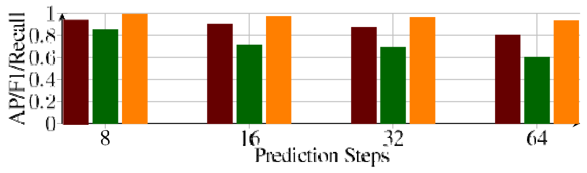
(b)



(c)



(d)



(e)

Fig. 5.Sensitivity analysis of model structure. (a) Comparison for numberof network layers. (b) Comparison for number of time steps considered in thenetwork. (c) Comparison for number of neurons. (d) Comparison for numberof decimal places (precision) used in the model. (e) Comparison for variousdegrees of prediction (i.e., how many steps in advance). One time step in ourcase is 5 ms.

failures), as we are trying to anticipate and potentially avoid undesirable system state. In fact, in the experiment shown in Fig. 4, the recall value is 1, which means that all measured failures are predicted—i.e., we have no false negatives.

a) Model structure tradeoffs: To ensure the practical utility of our hardware predictor in low-power embedded systems, it is important to balance precision and complexity. Considering the feasibility constraints, we explore the impact of several hyper-parameters and layer structures on the model performance. Parameters include RNN type, model structure, number of hidden neurons, decimal digits, and number oftime steps. We evaluate the RNNs and LSTMs based on AP, F1, recall (performance), runtime, and degree of prediction. Fig. 5 shows how different hyperparameters affect the model performance. The left y-axes measure AP score, F1-score, and recall score. The right y-axes measure the time it takes to gen-erateoneprediction.Thesolidlinesrefertothemodelwith

Fig.6.LSTMpredictionaccuracyfor64-step(320 ms)prediction,comparedto measured behavior.

LSTM layers and the dotted lines refer to the model withRNN layers. Fig. 5(a) shows how the number and type of lay- ers affect the performance. It indicates that LSTM has better accuracy. Prediction time increases with the number of layers. Therefore, it is best to apply 2-layer LSTM. Fig. 5(b) shows how the number of previous timesteps affects the performance. After five timesteps, the accuracy plateaus and predictiontimeincreases,therefore,usingfivetimestepsisthe bestchoice.Fig.5(c)showshowthenumberofneuronsaffects the performance. Accuracy pleateus beyond 32 neurons, thus we choose 32 neurons in the network. Fig. 5(d) shows howthe decimal digit influences performance. Two digits is the minimum number to maintain accuracy. Fig. 5(e) shows how accuracy degrades as the prediction moves further in advance.

B. Long-TermRegressionModel

Fortheregressionmodel,weusemeanabsoluteerror (MAE)toevaluatetheaccuracy,where$y_i$isthepredicted temperature$k$stepsinadvance(P_i),and$\hat{y}_i$isthemeasured temperature at step $i + k(M_{i+k})$

$$\mathrm{MAE}=\frac{1}{n}\sum_{i=1}^{n}|P_i-M_{ik}|_+ \quad (15)$$

Fig. 6 shows a sample time plot of one experiment. The orange dashed line shows the measured temperature 64 steps (320ms)inadvance.Thelatencyofthelong-termregres- sion model is 0.108 ms (no hardware acceleration). The blueis the predicted temperature in realtime. The MAE achievedbythepredictorfor320 msinadvanceis0.018.Thehigh- est accuracy achieved by existing prediction methods is 0.024 MAE[17],andthelongestpredictionstepis500ms[4],which we improve by 25% and 36%, respectively.

V. Conclusion

We presented a novel long short-term prediction engine (LSTM) based approach for hardware hazard prediction. Two models, each with distinct prediction criteria, are used by the prediction engine to produce predictions for both normal and urgent conditions. The ODROID-XU3 platform's data is used to train and evaluate the integrated model. The short-term model achieves an average precision score of 0.78 and generates accurate binary predictions 40 ms ahead of critical conditions.With an MAE of 0.018, the long-term model generates temperature values up to 320 ms ahead of time.

References

[1] A. X. M. Chang, B. Martini, and E. Culurciello. (2015). *RecurrentNeuralNetworksHardwareImplementationonFPGA*.[Online]. Available:https://arxiv.org/abs/1511.05552

[2] Z.Chen,Y.Liu,andS.Liu,"MechanicalstatepredictionbasedonLSTMneura lnetwok,"in*Proc.Chin.ControlConf.*,2017, pp.3876–3881.

[3] A. Chigurupati, R. Thibaux, and N. Lassar, "Predicting hardware failure using machine learning," in *Proc. Rel. Maintainability Symp.*, 2016, pp.1–6.

[4] R. Cochran and S. Reda, "Consistent runtime thermal prediction andcontrol through workload phase detection," in *Proc. ACM/IEEE DesignAutom. Conf.*, 2010, pp. 62–67.

[5] A.K.Coskun,T.S.Rosing,andK.C.Gross,"Utilizingpredic-tors for efficient thermal management in multiprocessor SoCs," *IEEETrans.Comput.-AidedDesignIntegr.CircuitsSyst.*,vol.28,no.10, pp.1503–1516,Oct.2009.

[6] F.D.D.S.Lima,G.M.R.Amaral,L.G.D.M.Leite,J.P.P.Gomes, and J. D. C. Machado, "Predicting failures in hard drives with LSTMnetworks," in *Proc. Brazil. Conf. Intell. Syst.*, 2017, pp. 222–227.

[7] Y.Ge,Q.Qiu,andQ.Wu,"Amulti-agentframeworkforthermalawaretask migration in many-core systems," *IEEE Trans. Very Large ScaleIntegr. (VLSI) Syst.*, vol. 20, no. 10, pp. 1758–1771, Oct. 2012.

[8] I. Giurgiu, J. Szabo, D. Wiesmann, and J. Bird, "Predicting dram reliability in the field with machine learning," in *Proc. ACM/IFIP/USENIXMiddleware Conf. Ind. Track*, 2017, pp. 15–21.

[9] "ODROID-XU,"Hardkernel,Seoul,SouthKorea,Rep.[Online]. Available:https://www.hardkernel.com/shop/odroid-xu3/

[10] R. Kumar, S. Vijayakumar, and S. A. Ahamed, "A pragmatic approachto predict hardware failures in storage systems using mpp database andbigdatatechnologies,"in*Proc.IEEEInt.Adv.Comput.Conf.*,2014, pp.779–788.

[11] S. Liu, G. Liao, and Y. Ding, "Stock transaction prediction modelingandanalysisbasedonLSTM,"in*Proc.IEEEConf.Ind.Electron.Ap pl.*,2018, pp. 2787–2790.

[12] T. MÃijck, S. Sarma, and N. Dutt, "Run-DMC: Runtime dynamic heterogeneous multicore performance and power estimation for energyefficiency,"in*Proc.Int.Conf.Hardw.Softw.CodesignSyst.Synth.*,2015 , pp.173–182.

[13] S. Huang, C. Fung, K. Wang, P. Pei, Z. Luan, and D. Qian, "Usingrecurrentneuralnetworkstowardblack-boxsystemanomalyprediction,"in *Proc. IEEE/ACM Int. Symp. Qual. Service*, 2016, pp. 1–10.

[14] S. Sharifi, D. Krishnaswamy, and T. S. Rosing, "PROMETHEUS: Aproactive method for thermal management of heterogeneous MPSoCs,"*IEEETrans.Comput.-AidedDesignIntegr.CircuitsSyst.*,vol.32,no.7, pp.1110–1123,Jul.2013.

[15] G.Singla,G.Kaur,A.K.Unver,andU.Y.Ogras,"Predictivedynamic thermal and power management for heterogeneous mobileplatforms,"in*Proc.DesignAutom.TestEuropeConf.Exhibit.*,2015, pp.960–965.

[16] X. Sun *et al.*, "System-level hardware failure prediction using deeplearning," in *Proc. ACM/IEEE Design Autom. Conf.*, 2019, pp. 1–6.

[17] E.W.Wächter,C.deBellefroid,K.R.Basireddy,A.K.Singh, B. M. Al-Hashimi, and G. Merrett, "Predictive thermal management forenergy-efficient execution of concurrent applications on heterogeneousmulticores," *IEEETrans. Very Large ScaleIntegr.(VLSI)Syst.*, vol.27,no. 6, pp. 1404–1415, Jun. 2019.

[18] Y. Wang, J. Zhou, K. Chen, Y. Wang, and L. Liu, "Water qualityprediction method based on LSTM neural network," in *Proc. Int. Conf.Intell. Syst. Knowl. Eng.*, 2017, pp. 1–5.

[19] B.WojciechowskiandJ.Biernat,"Temperaturepredictionformulti-coremicroprocessors with application to dynamic thermal management," in*Proc. Int. Workshop Thermal Investigat. ICs Syst.*, 2012, pp. 1–6.

[20] C. Xu, G. Wang, X. Liu, D. Guo, and T. Liu, "Health status assess-ment and failure prediction for hard drives with recurrent neuralnetworks,"*IEEETrans.Comput.*,vol.65,no.11,pp. 3502–3508,Nov. 2016.

[21] I. Yeo, C. C. Liu, and E. J. Kim, "Predictive dynamic thermal management for multicore systems," in *Proc. ACM/IEEE Design Autom. Conf.*,2008, pp. 734–739.