Turkish Online Journal of Qualitative Inquiry (TOJQI)
Volume 12, Issue 6, July, 2021: 5297 – 5313

Research Article

DICHA - DNN based Intelligent classification for Workload Characterization on Heterogeneous Architecture

R.Sivaramakrishnan[1], Dr.G.Senthilkumar[2]

Abstract

Nowadays Heterogeneous system on-chip (HSoC) become highly essential. IoT, Industry 4.0, intelligent vehicles, embedded devices, and cyber-physical framework applications are broadly utilizing such equipment models for workload processing. These continuous applications include a miscellaneous set of workloads with various attributes which highly influences the computational cycles. Moreover, asset organization become a basic issue in HSoC. In this paper DNN based classifier is proposed for HSoC stages to predict ideal computational asset for every responsibility at runtime. Deep Neural Networks (DNN), with deep layers and extremely high element of boundaries, have exhibited get through learning capacity in Machine learning region. Nowadays DNN with Big Data input are driving another heading in enormous scope object acknowledgment. The proposed classifier analysed the execution of a few HSoC stages to comprehend the functioning guideline of ongoing responsibilities at runtime. The noticed attributes are outlined as continuous data set and the equivalent is used to train and test the DNN classifier. The proposed classifier is assessed on raspberry-pi HSoC and re-enacted on the python with ML library. Precision, throughput, affectability, selectivity measurements are distinguished to break down the exhibition of the proposed calculations. The proposed DICHA framework accomplished the precision up to 96% contrasted and outrageous ML predictor for and furthermore saved the execution energy up to 30% for real-time embedded benchmark workloads like MiBench, IoMT.

keywords*: dnn, accuracy, heterogeneous architecture, machine learning, workload characterization, throughput*

[1]Research Scholar, [2]Associate professor in ECE SCSVMV University

[1]sivaram6685@gmail.com, [2]Gskkanchi@gmail.com

Introduction

In the modern era, the smart mobile phones has become an essential component that integrates multiple processors and chips for operating, managing and controlling the application [2]. The communication, lower power operation and performance are affected by the design and deployment of older mobile processors. The processors becomes more complex for supporting the transition to digital for delivering the data services. Every mobile

phones consists of application processors, Random Access Memory (RAM), Digital Signal Processor (DSP) and Control Processing Unit (CPU) which is also known as ARM processor

[3]. The performance of a device is based on the processor as it is in need to experience any application on the device like smart phone [4]. Execution of any tasks in the smart devices are done through processor that is made up of multiple cores. Single-core, dual-core, quad core, hexa-core, and octa-core are the distinct cores that could differentiate itself by its speed. Each processor is independent in the architecture of the smart phones [4]. Workloads in smart phones is shown in Figure 1.

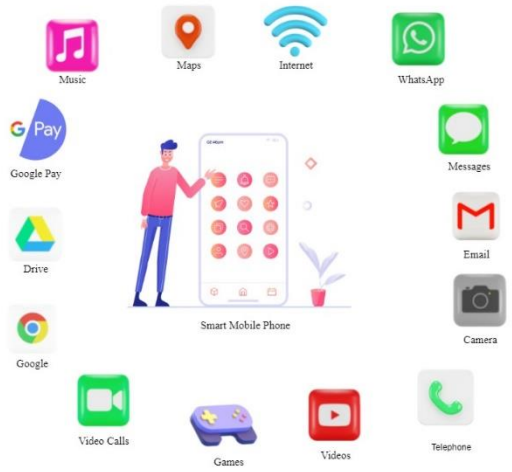

Figure.1 Workload in smart mobile phone

Execution of memory, computation and graphical threats in a single program is not easy. Hence characterization of workload at runtime, recognizing hardware configurations for resources optimally, load balancing and maximizing the performance becomes challenging task for the computer architects [1]. The amount of task given to the processor, time sharing, batch, number of transaction processing, higher consumption of energy and applications that are multi-threaded is termed as workload characterization that can affect the performance of the system [5].

Genetic algorithm, ant colony optimization and swarm optimization are the optimization techniques for enhancing the performance. Integer Linear Programing (ILP) helps in scheduling and task scheduling problems for optimizing the energy consumption and performance constraints. Linear regression, Artificial Neural Network (ANN), decision tree algorithms, reinforcement learning, Support Vector Regressions (SVR) are the deep learning techniques that helps in optimizing the consumption of energy, Voltage Frequency (VF) pairs, hardware configurations for scheduling the workload at run time [1].

Contribution of the proposed work

In this paper, we use DNN classifier to improve a scheduler's capacity to distinguish the mapping methodology which brings about the most noteworthy framework throughput than other regular schedulers. The primer outcomes show huge execution benefits between 25%-30% contrasted with a proficient state-of-the art heterogeneous scheduler. We additionally show how expanding the intricacy of the DNNs classifier may bring about

consistent losses contrasted and the more lightweight ANNs which are simpler to execute and have less overheads. These underlying outcomes help to approve the evidence of idea that as

far as anyone is concerned is quick to use ML to anticipate thread execution at the scheduling quantum granularity as well as to improve scheduling in heterogeneous framework.

- A heterogeneous scheduling model utilizing a next quantum thread behavior predictor, ML based execution predictors, and a framework throughput augmentation scheduling strategy.
- The plan, executions, and assessment of two lightweight and DNN based execution classifier for two distinctive core sorts which produce an estimation IPC (Instruction per cycle) esteem per scheduling quantum for various threads.

The paper arrangement is as follows. Detailed discussion of related works for the idea of technical background is presented in Section II. Section III presents our proposed DNN scheduling model. The experimental setup and results and discussion is discussed in Section IV, V respectively. Conclusion and future direction of this work is presented in Section VI.

Related Works

Gomatheeshwari B et.al [1] develops a lightweight-deep neural network (LW-DNN) that includes phases such as selection of feature, optimizing the feature and predicting the features. The first two phases extracts and optimizes the workload by pre-processing algorithm and the last phase will predict the cores at runtime for enhancing the efficiency in energy and performance. The core prediction phase comprises linear regression, SVM, a light weight DNN and testing the predictive models. The framework are evaluated in distinct asymmetric multicore platforms like ARM. This proposed prediction model achieves the higher accuracy in predicting the core, improvement in average energy consumption and the execution time also gets improved when compared to the existing prediction models.

Kyle M. Tarplee et.al [6] designs a novel linear programming for a heterogeneous system to compute efficiently and to reduce energy and makespan. The divisible load theory (DLT) approach is applied on a single task for calculating the lower-bound solution. The computation of lower-bound solution, the feasible solutions that is tight upper bound are recovered from the infeasible lower-bound by using the two-phase algorithms. The linear vector optimization comprises of multiple Non-Dominated solutions and inner and outer approximation. The pareto front generation comprises weighted sum, genetic algorithm for non-dominated sorting, convex fill algorithm and pareto front solution quality. This approach suits perfectly for large scale scheduling problems and helps the decision makers to reduce the operational cost and increases the efficiency by trade offing the energy and makespan.

Sarad venugopalan et.al [7] proposes a novel mixed integer linear programming (MILP) for scheduling problems and also eliminates the requirement of linearization and bilinear equations from communication delays. This paper comprises of MILP, model for task scheduling, bi-linear reductions and the proposed formulation. This formulation includes SHD basic formulation, SHD relaxed formulation, and SHD reduced formulation. The MILP is compared based on the 1 minute and 12 hour timeouts. The size of MILP formulation is

independent to the number of processors. This formulation reduces the constraint complexity and improves in graph structures when compared to the existing formulation.

Zhuo tang et.al [8] reduces the energy and the quality of service are maintained by DVFS-enabled energy efficient workflow task scheduling algorithm (DEWTS). Initially this algorithm calculates the order of scheduling of tasks and the heterogeneous earliest finish time (HEFT) algorithm acquires the deadline and makespan. The processors that are underutilized are combined with unique task number and utilization of energy in resorting processors by closing the last node. The slacking phase of this technique distributes the tasks to the idle slot without rebelling any dependency constraints and increases the makespan. This technique reduces the total consumption of power and also balances the performance in scheduling.

Cha V.Li et.al [9] introduces a method for thread characterization for exploring the performance of counters in hardware and techniques that automates the evaluation of the workload on processors that are heterogeneous. The thread characterization comprises phases for pre-processing, filtering and resampling, counter selection, extracting and reducing the feature. The linear regression model and structural similarity metric are the machine learning techniques used for the evaluation. This method furnishes higher accuracy when compared to the existing methods and also have low overhead.

Daniel nemirovsky et.al [10] presents a cpu scheduling model for predicting the performance of multiple threads by using machine learning at the quantum. The authors demonstrates the light weight artificial neural network (ANNs) to improve the efficiency in scheduling. Scheduling based on machine learning comprises of parameter engineering, next quantum thread behaviour predictor, and predictor model of ANN. This model maximizes the improvement around 21% of throughput and also predicts the performance more accurately than the existing methods.

E.jayanthi et.al [11] adopts the distinct deep learning and machine learning mechanisms for mapping the workload individually on quad-core platforms. Naïve baize, support vector machine and random forest algorithm are the machine learning algorithms whereas Recurrent Neural Network and long short term memory are the deep learning algorithms. These techniques helps in predicting the workload of the core before execution and also to optimize the consumption of energy. The accuracy and consumption of energy are the prediction metrics that are compared with the traditional mechanisms. Among all the deep and machine learning model the LSTM deep learning model predicts workload accurately and outperforms the other models.

A.Gopinath et.al [12] calculates the workload of the processor by using micro architecture independent workload classifier (MIWC). The MIWC comprises phases such as setup, mechanism for cognition classifier and rules for the allocation of intelligent core. The allocation of intelligent core includes the enhanced support vector machine (ESVM) with fuzzy rule sets (FRS). Estimating the energy for various workloads, calculating the branch

fitness ratio and evaluating the clock frequency by IPC calculator. The evaluation of speed and the analysis for distinct workloads algorithms by computati on time and in testing phase. This EVM method reduces the energy around 25% of original consumption.

Monir zaman et.al [13] discusses the state of the art for predicting the workloads and introduces the support vector regression (SVR) for predicting the behaviour of the workload. Predicting the last value, predicting the history table, predicting the average moving for autoregressive are the techniques for predicting the characteristics of workload. The SVM in recursive and direct models are the prediction approaches for forecasting the workloads. Predicting the performance in counter and predicting the temperature and power accurately by using this method. This model reduces the absolute error and outperforms the other traditional methods in prediction error.

Vivienne sze et.al [14] reviews the recent advances of deep neural network (DNN) along with architectures and hardware that support it with lower computational costs. The artificial intelligence, neural networks, training versus inference, cloud versus embedded are the background of DNNs. Accelerating the computation of kernel on GPU and CPU platforms, dataflow of energy for accelerators are DNN processing in hardware. DRAM, SRAM, memories of non-volatile resistive and sensors are the processing of near-data. Reducing the precision, reducing the size of model and number of operations are the codesign and hardware of DNN models. This article surveys the avenues for optimizing the DNN processing. It also focuses on maintaining the throughput, accuracy and cost.

PROPOSED SYSTEM

DICHA - Working Mechanism

For the effective scheduling in multicore heterogeneous architecture, the proposed framework incorporates DNN with ELM classifier for prediction of energy and other features .The detailed theory behind the DNN and ELM classifier is explained as follows.

Deep Neural Networks

Deep learning technique was initially applied for classification of image [24], which was used to limit the dimensionality of information portrayal and perceive targets adequately. For extraordinary learning capacity, the profound learning strategy has likewise been utilized effectively for Workload characterization on heterogeneous stage. The DNN architecture is represented in Figure.2. Thusly, a DNN can be prepared to gain valuable component data straightforwardly from a dataset without requiring master information in the predetermined fields. The DNN working mechanism and assessment of anticipated outcomes are shown in Figure 3.

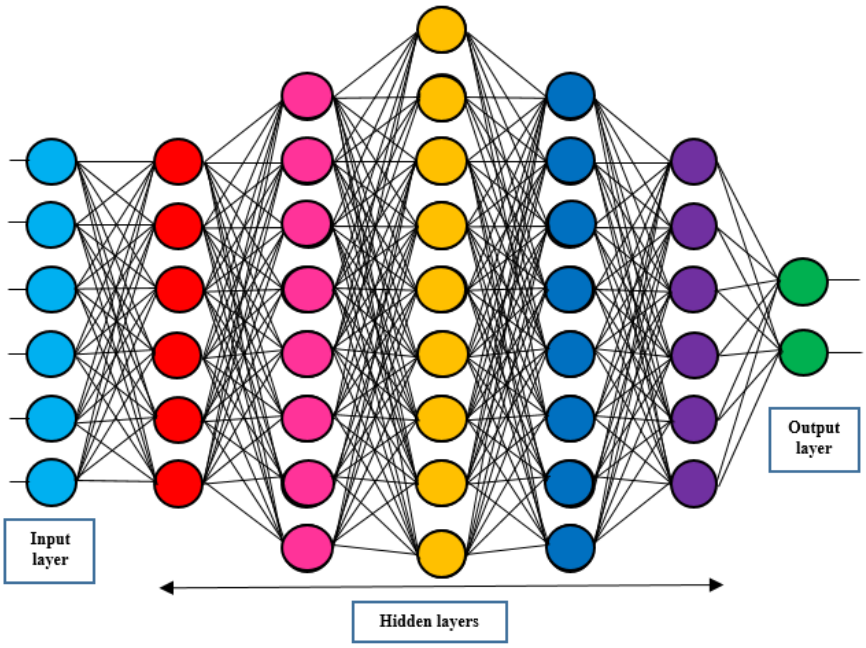

Figure.2 DNN architecture

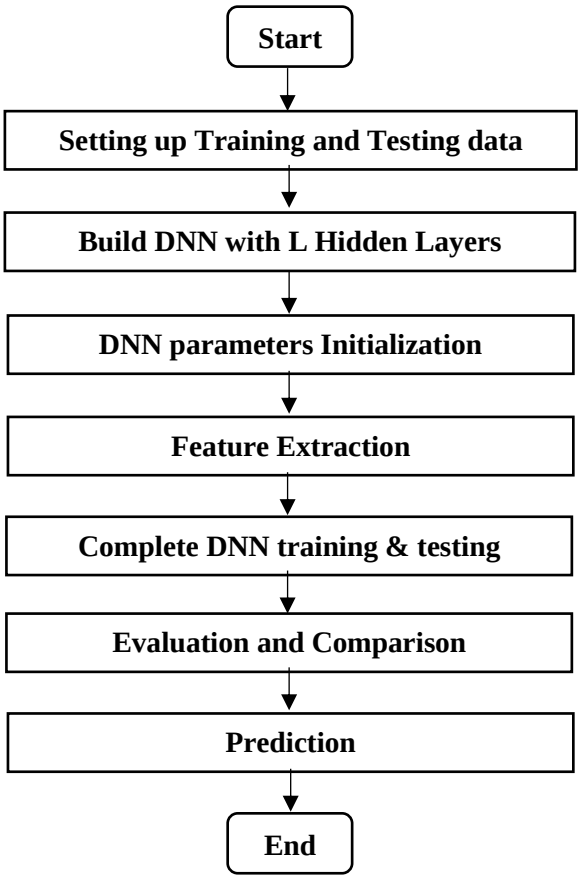

Figure.3 The flowchart of the proposed deep neural network (DNN) approach for prediction

DNN is a stacked layer model in which the layers are associated together and there are no associations of hubs inside a similar layer. A DNN incorporates an information layer, a yield layer, and a couple of covered up layers put between them in the model. The quantity of hubs of an info layer is set relating to the dimensionality of the information. Essentially, the quantity of hubs of a yield layer is characterized comparing to the dimensionality of the objective information. The quantity of hubs of each secret neural layer is set as per the organization work, for which there are no necessary exacting guidelines. Every hub in the following layer is straightforwardly connected to all hubs in the past layer. Hubs of the primary layer get the info information and communicate them to different layers, while hubs of the last layer yield the objectives. The nonlinear connection between the DNN layers is demonstrated by the accompanying conditions:

$$z_j^l = \sum_i w_{ij}^l x_i^{l-1} + b_j^l \quad (1)$$

$$h_{W,b}(x) = x_j^l = f\left(z_j^l\right) = f(\sum_i w_{ij}^l x_i^{l-1} + b_j^l) \quad (2)$$

Where

x_j^l → activation value of neuron j in layer l;

z_j^l → Linear activation combination of neurons in the previous layer;

b_j^l → Bias value of neuron j in layer l;

w_{ij}^l → Weight parameter between nodes i in layer l-1 and j in layer l;

f(.) → activation function, which is usually chosen to be sigmoidal of the form $\mathrm{f}fz) = \frac{1}{1+e^{-z}}$ and mostly used in DNN.

Given the fixed setting of the model parameters (W, b) to our data, a nonlinear form of hypotheses $h_{W,b}(x)$ denoted the output of the neural network, which gave us a real number. The network outputs were serially calculated layer by layer on the model's parameters, which were initialized appropriately.

Dataset Feature Extraction of Workloads

In this framework, for the analysis of threat execution without any intelligent allocation, the raspberry pi which is quad core heterogeneous architectures to test workloads like MiBench, IoMT, Beeps. LinuX tools is utilized to identify characteristics of workloads regards of cache miss rate, branch predictor, instruction and, memory by measuring 60+ performance counter values [16]. The measure values are utilized as a dataset to train the prediction network. This framework incorporates the optimization algorithm same like [15] because the extracted features are large in size. Table 1 presents the thread analysis features. The training model selects different benchmarks which are mentioned above.

Table.1. *Important Features of Thread Analysis*

Instructions	Category
L1, L2 Cache miss rates	Memory
Branch misprediction	Branch
ALU/floating point instructions	Operational
Start time, finish time	Timestamp
Power consumption of each function	In watts
Pipelined stages	As per core type
Micro-architecture instructions	ILP
TLBs	Transition blocks

DICHA MECHANISM

DICHA is a hybrid model which incorporates DNN intelligent learning mechanisms from the testbeds features are extracted for the inputs of different learning algorithms. For the Energy prediction (Ep), at different iterations, the inputs AME, RMSE, ACCURACY, MRE and THROUGHPUT given as Inputs to DNN. To prove better classification and to detect various attacks, these outputs along with the other features are given as the inputs to the ELM. The overall DICHA mechanism is illustrated in Figure 4

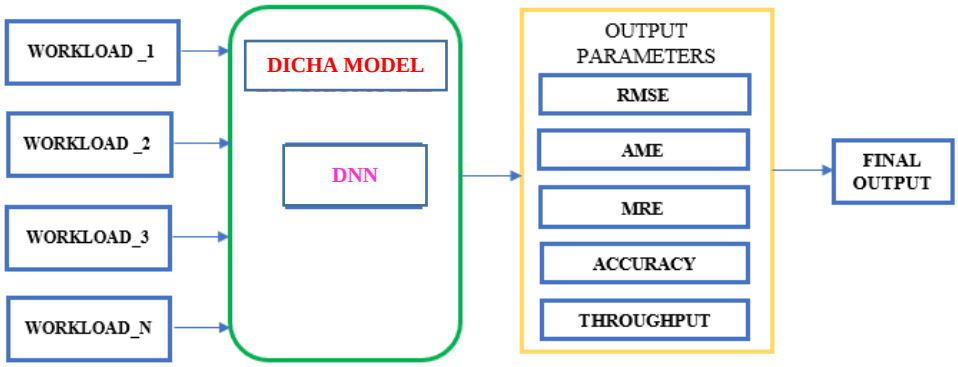

Figure.4 working mechanism of DICHA framework

Consider that, for the i^{th} input P^l in the training dataset is

$$\{(P^l; y^l) | P^l \in R^l, y^l \in R\}_{l=1}^{L} \quad (3)$$

Y_s (P^l) is the LSTM practical output

Linear classification of training dataset is presented as

$$\{(y_s(P^l), y^l)\}_{l=1}^{L}, \quad (4)$$

The hybrid model (classification model) is represented as

$$\hat{y}(p) = C_0 + C_1 Y_s(P) \quad (5)$$

Newly generated training dataset is

$$\{(y_s(P^l); y^l)\}_{l=1}^{L}, \quad (6)$$

We assume that

$$C_0 + C_1 Y_s(P^l) = y^l \quad l = 1,2,3 \ldots \ldots L \quad (11)$$

Then above mentioned formula is modified as

AC = Y

$$A = \begin{bmatrix} 1 & Y_s(P^1) \\ 1 & Y_s(P^2) \\ \vdots & \vdots \\ 1 & Y_s(P^L) \end{bmatrix} \quad (7)$$

$C = [C_0 \quad C_1 \quad C_2]^T$ (8)

$Y = [y^1 \quad y^2 \quad y^L]^T$ (9)

Hybrid model parameters is represented as follows

$$\hat{C} = A^+ \quad (10)$$

Where A^+ → Moore-Penrose Pseudo inverse Matrix of A

EXPERIMENTAL SET-UP

This framework adopts different HSoCs such as Raspberry pi -3 which includes ARM cortex-7 & ARM cortex- 53. Another HSoC comprises the ARM -v9 processors. Different HSoC platforms are mentioned in Table 2. FPGAs are used to extract the characteristics of workloads.

Table.2. *DIFFERENT TYPE OF ARM ARCHITECTURES*

Sl. No	Core's type tested			
	Test_Bed_1	Test_Bed_2	Test_Bed_3	Test_Bed_4
01	Cortex-A5 Series	ARM-9 Series	Cortex-A5 Series	Cortex-A8 Series
02	ARM-9 TDMI Series	ARM-9 TDMI Series	Cortex-R Series	Cortex A-5 Series
03	Cortex-A5 Series	CortexA-5 Series	Programmable FPGA	
04	Cortex A Series		Programmable FPGA	

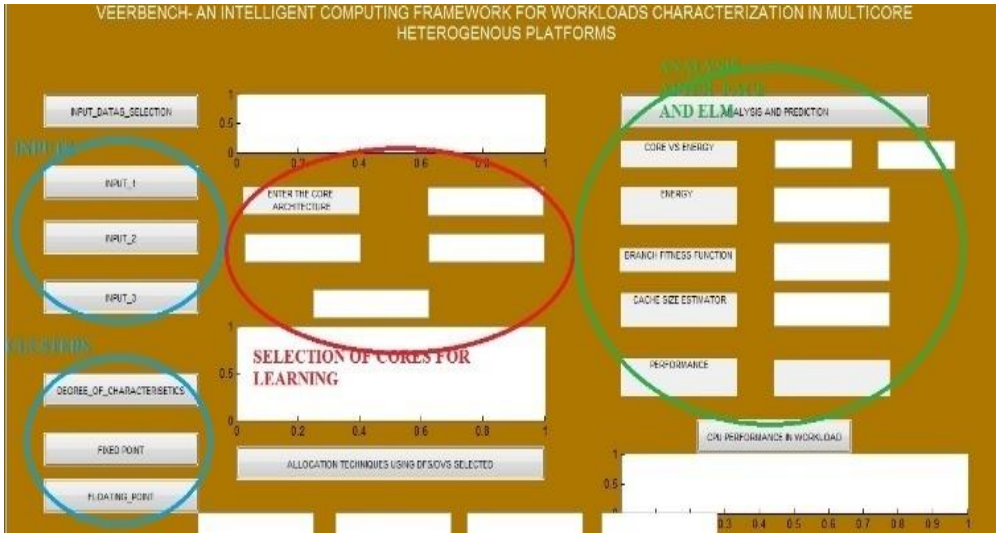

Figure 5. DICHA Framework designed for the Multi core Heterogeneous Architectures

MATLAB toolboxes version > R2012 is utilized for the DICHA framework and this framework view is presented as follows

Real-time Benchmarks

For the evaluation purpose, benchmarks such as SPLASH-2, PARSEC, and SPEC 2010 are utilized. The application workloads like as “vpr, perlbench, gcc, mgrid, equake, art, applu, gap, soplex, twolf, mesa, wupwise, crafty, povray, milc, swim, vortex, mcf, leslied, bwaves, libquantum, lbmastar, blackholes, body track, sphinx, facesim, ferret, canneal, swaptions, fluidanimate, fft, fmm, radix,H.264, ocean”, for the testing the HSoC working

principle. These (programs) workloads are run on each processor for the different iterations and also witnessed various metrics for database creation.

Results and Discussion

This section gives the performance analysis of DICHA framework with the comparison of other exiting algorithms

Performance Metrics

To evaluate the DICHA model performance, "Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Square Error (MSE), Throughput and Accuracy" are estimated.

The below figure illustrate that proposed DNN improved 4% of performance of workload execution by compared with state-of-art algorithms. This is achieved by appropriate utilization of resources and workloads mapping. The below figures 6 to 12 shows the other performance metric enhancements achieved. X-axis the performance metrics such as "MAE, RSME, MRE, Throughput and Accuracy" and Y-axis denotes the application workloads.

MAE – Mean Absolute Error

Average of the absolute errors between the predicted values and actual observed values and it is given as follows

$$Mean\ Absolute\ Error\ (MAE) = \frac{1}{L}\sum_{y=1}^{L}|\hat{y}^{l} - y^{l}| \quad (11)$$

Where y^{l} → predicted value of the workload , $\hat{y}^{l}$ → real values of the workload, L → number of test or training samples.

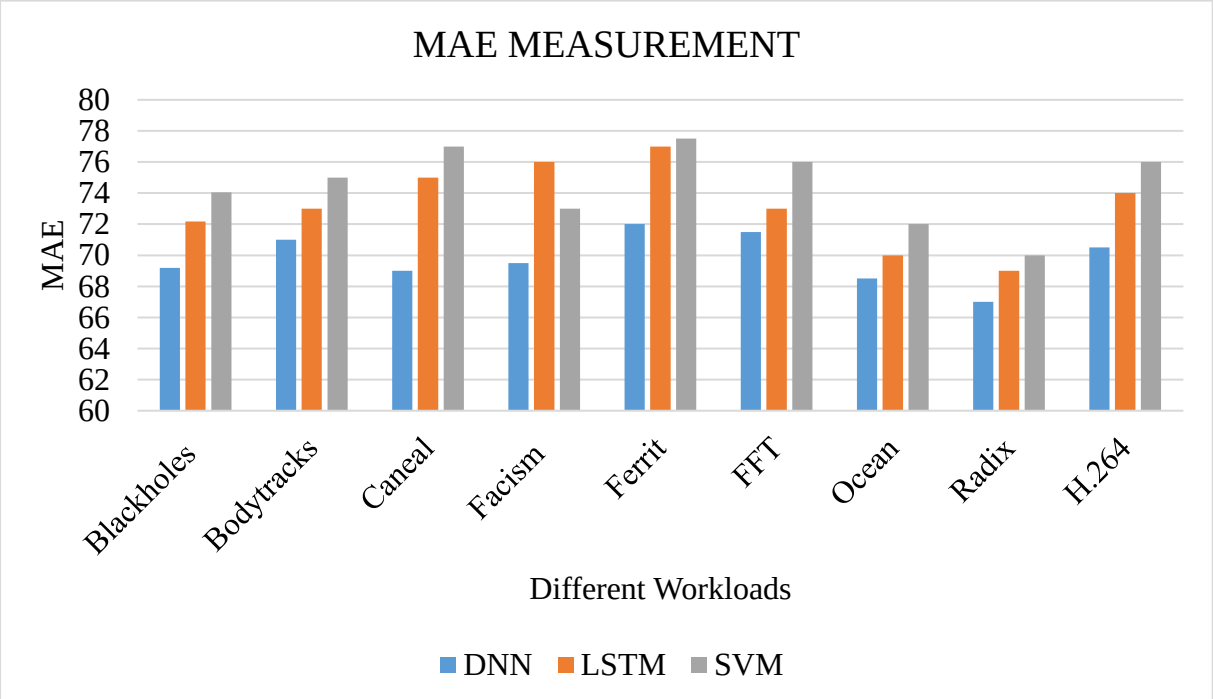

Figure 6. MAE measurement for different workloads

Root Mean Square Error (RMSE)

It is the predicted and actual values difference as mentioned as follows,

$$Root\ Mean\ Square\ Error\ (RMSE) = \sqrt{\frac{1}{L}\sum_{y=1}^{L}|\hat{y}^l - y^l|}$$

(12)

Where y^l → predicted value of the workload , $y\hat{}^l$ → real values of the workload, L → number of test or training samples.

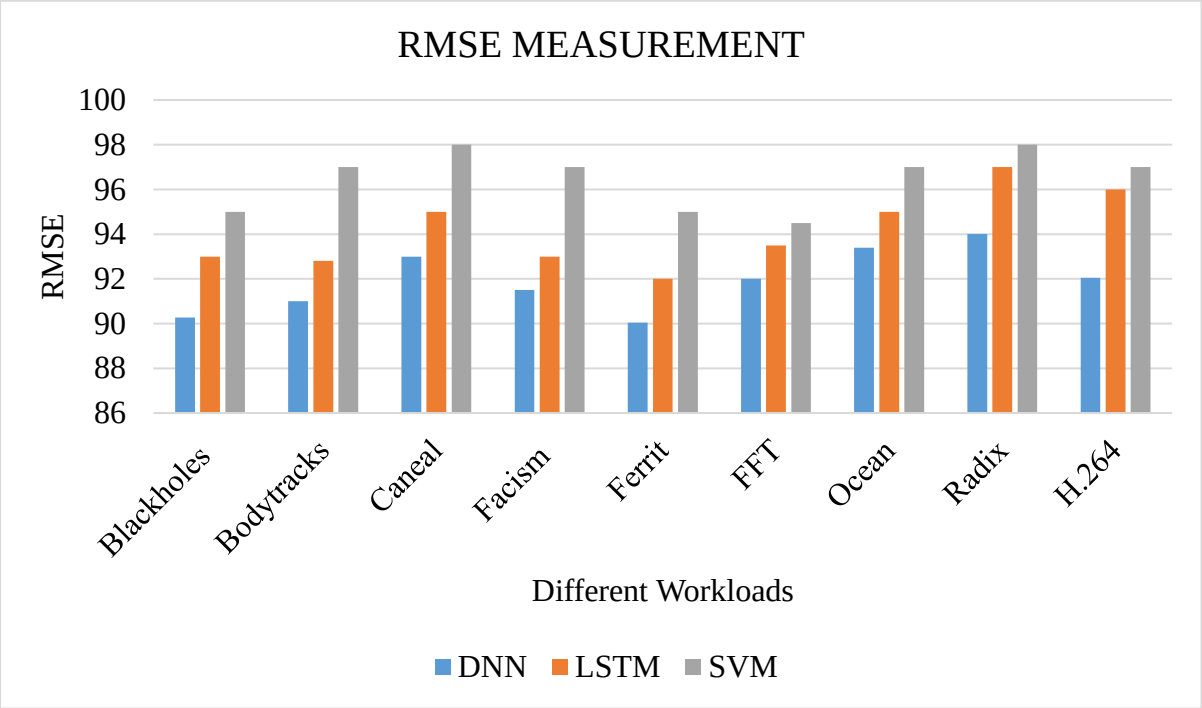

Figure.7. RMSE Measurement for different Workloads

MRE – Mean Relative Error

This represents the accuracy in % by dividing absolute errors by corresponding actual values.

$$Mean\ Relative\ Error\ (MRE) = \frac{1}{L}\sum_{y=1}^{L}\frac{|\hat{y}^l - y^l|}{|y^l|}$$

(13) Where y^l → predicted value of the workload, $y\hat{}^l$ → real values of the workload, L → number of test or training samples

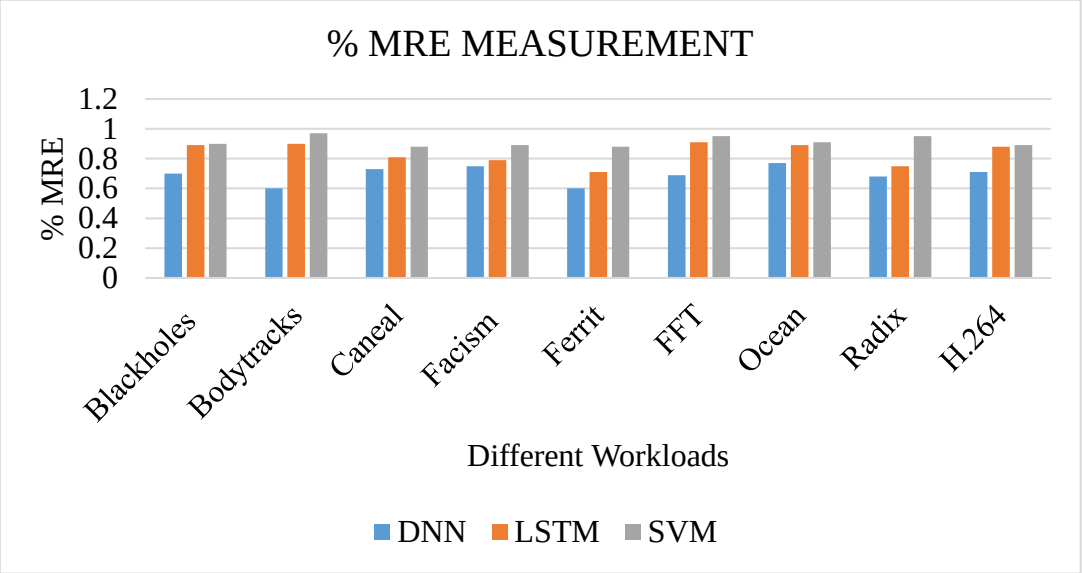

*Figure.8.*MRE Measurement with different Workload

Throughput Measurement

Throughput of the proposed framework has been measured with the different parameters taken from the workloads.

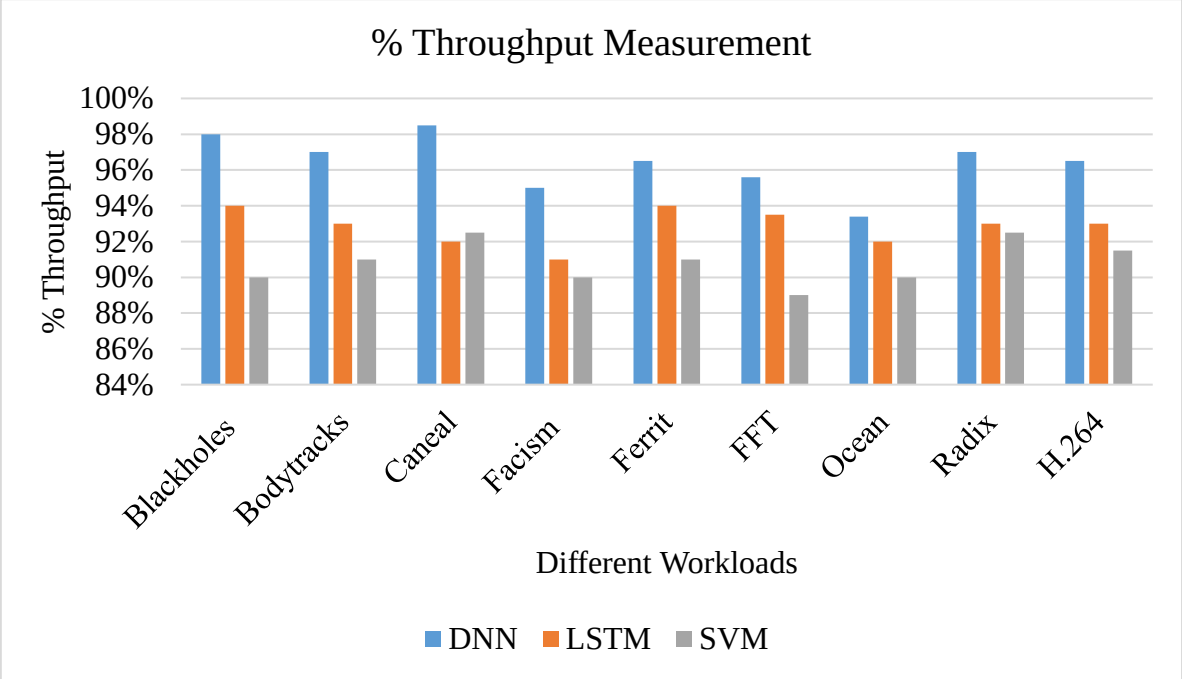

Figure 9. Throughput Measurement with different workloads

Accuracy

It is the ratio of number of correct predictions to the total number of input samples which is measured for different workloads by using formula given below

$$Accuracy = \frac{Number\ of\ Correct\ Predictions}{Total\ NUmber\ of\ Predictions\ made}$$

(14)

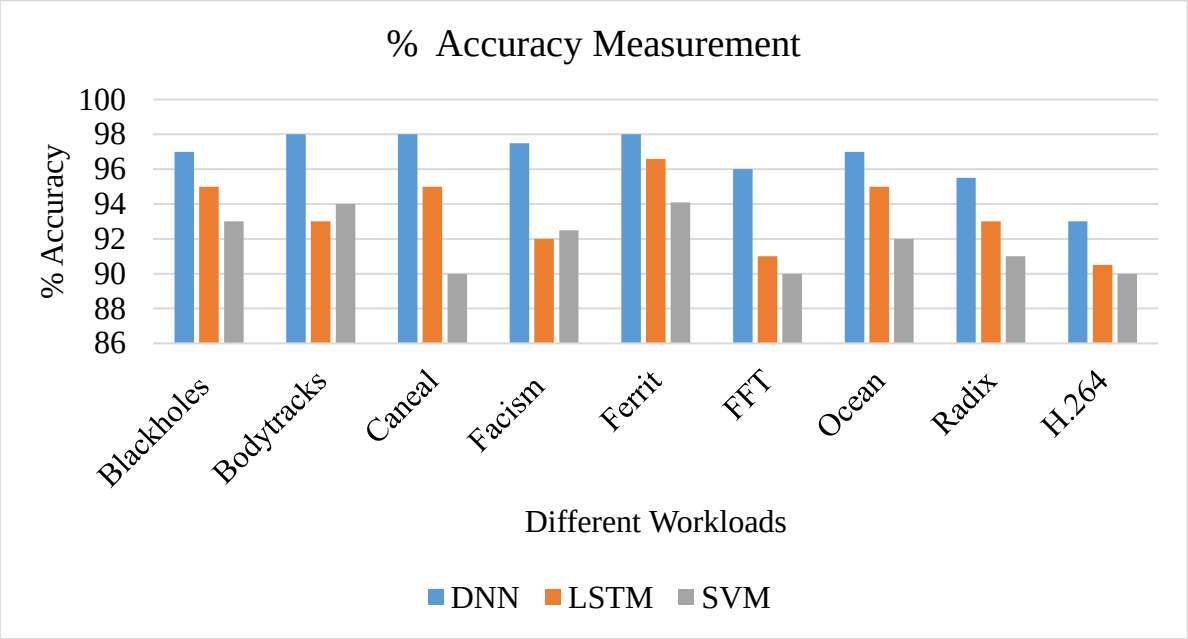

Figure 10. Accuracy Measurement for various workloads

From the above figures infers that, DICHA framework proves better regards of MAE, RMSE, MRE, Accuracy, Throughput.

Sensitivity

It is the ratio of number of correct predictions to the total number of input samples which is measured for different workloads by using formula given below

.Sensitivity = TP/TP+TN

(15)

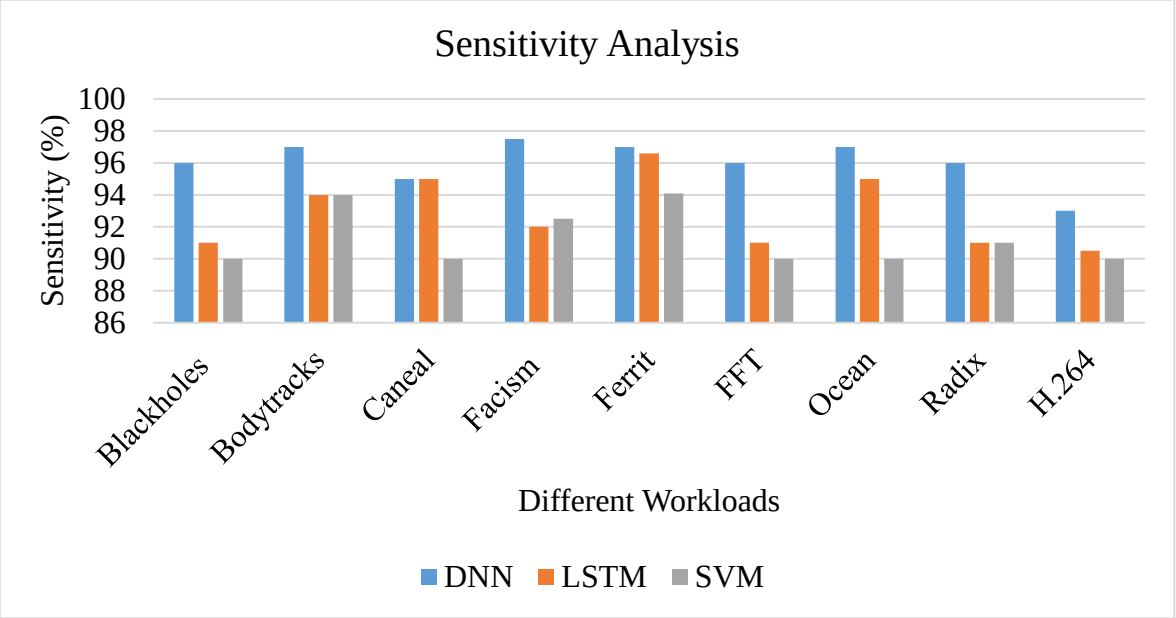

Figure. 11 Sensitivity Analysis of Proposed Model

(vii) Selectivity

The selectivity of the framework has been measured with the different parameters taken from the workloads. It is the ratio of number of correct predictions to the total number of input samples.

Selectivity = TN/TN+FP (16)

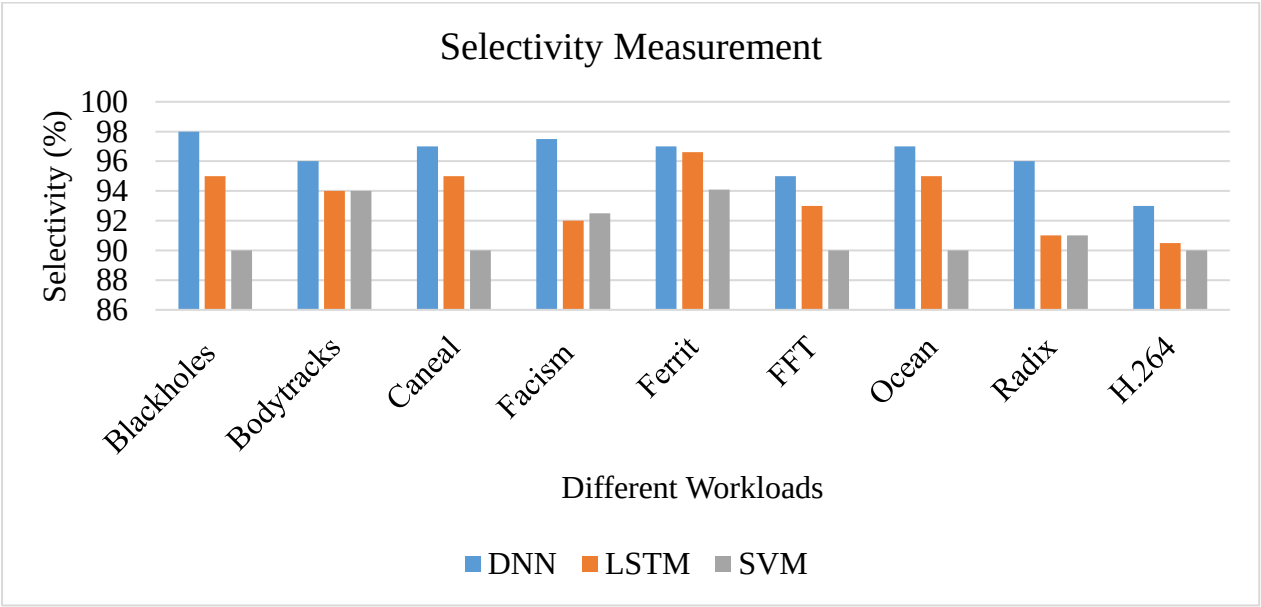

Figure.11.Selectivity Measurement

From the above figures infers that, DICHA framework proves better regards of sensitivity and selectivity

Conclusion

In this paper we have spearheaded applying ML to a throughput amplifying heterogeneous by DICHA model fit for anticipating the presentation of different threads on assorted framework assets at the scheduling quantum. We have shown how DNN can give profoundly exact execution forecasts to an assorted arrangement of uses even while just being prepared on a little subset. Likewise, we portrayed how the indicator exactness can be significantly developed using internet learning and profound learning. Our methodology yields critical outcomes with normal throughput upgrades between 25% - 31% over ordinary heterogeneous schedulers for CPU and memory serious applications. We try to grow the extent of our work later on by a further examination of DNN hyper boundaries and elective ML models just as testing our methodology on a physical heterogeneous CMP. We trust that the novelty of this work has uncovered a portion of the energizing chances accessible by applying ML procedures in the field of PC engineering.

References

1. Amarnath.K.P,Dijo Joseph, Shiv Prasad T.M, “A Comparative Study on Recent Mobile Phone Processors”, International Conference on Emerging Trends in Engineering & Management (ICETEM-2016), pp:42-45, 2020.
2. Calzarossa. M and G. Serazzi, "Workload characterization: a survey," in Proceedings of the IEEE, vol. 81, no. 8, pp. 1136-1150, Aug. 1993, doi: 10.1109/5.236191.
3. Gomatheeshwari B, J. Selvakumar, “Appropriate allocation of workloads on performance asymmetric multicore architectures via deep learning algorithms”, Microprocessors and Microsystems, vol:73, 2020, doi:10.1016/j.micpro.2020.102996.
4. Jayanthi. E. and A. L. Vallikannu, "A Review on Workload Characteristics for Multi Core Embedded Architectures using Machine Learning and Deep Learning Techniques," 2021 Sixth International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), 2021, pp. 456-461, doi: 10.1109/WiSPNET51692.2021.9419405.
5. Li CV., Vinicius, D. Mossae,”Exploring machine learning for thread characterization on heterogeneous multiprocessors”, ACM SIGOPS Operating System Review, September 2017, doi:10.1145/3139645.3139664.
6. Nemirovsky.D, T. Arkose, N. Markovic, M. Nemirovsky, O. Unsal and A. Cristal, "A Machine Learning Approach for Performance Prediction and Scheduling on Heterogeneous CPUs," 2017 29th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD), 2017, pp. 121-128, doi: 10.1109/SBAC-PAD.2017.23.
7. Nguyen, H., Cheol-Hong Kim and J. Kim. “Effective Prediction of Bearing Fault Degradation under Different Crack Sizes Using a Deep Neural Network.” *Applied Sciences, Vol.* 8, No.2332, pp.1-13, 2018.
8. PratapSingh, Mahendra & Kumar, Manoj,”Evolution of Processor Architecture in Mobile Phones”,International Journal of Computer Applications,Vol:90, doi: 10.5120/15564-4339.

9. Premchander, Perumal,"Study of Independent Workload Calculation for Various Multicore Embedded Processors using Machine Learning Algorithm", international journal of scientific research and review, 2019.
10. Surana.P, N. Madhani and T. Gopalakrishnan, "A Comparative Study on the Recent Smart Mobile Phone Processors," 2020 7th International Conference on Smart Structures and Systems (ICSSS), 2020, pp. 1-3, doi: 10.1109/ICSSS49621.2020.9202174.
11. Sze. V, Y. Chen, T. Yang and J. S. Emer, "Efficient Processing of Deep Neural Networks: A Tutorial and Survey," in Proceedings of the IEEE, vol. 105, no. 12, pp. 2295-2329, Dec. 2017, doi: 10.1109/JPROC.2017.2761740.
12. Tang Z., L. Qi, Z. Cheng, K. Li, S.U. Khan, K. Li, "An energy efficient task scheduling algorithm in DVFS-enabled cloud environment", J Grid Computing, vol:14 (1),55–74, 2016,doi: 10.1007/s10723-015-9334-y.
13. Tarplee. K. M., R. Friese, A. A. Maciejewski, H. J. Siegel and E. K. P. Chong, "Energy and Makespan Tradeoffs in Heterogeneous Computing Systems using Efficient Linear Programming Techniques," in IEEE Transactions on Parallel and Distributed Systems, vol. 27, no. 6, pp. 1633-1646, 1 June 2016, doi: 10.1109/TPDS.2015.2456020.
14. Venugopalan. S. and O. Sinnen, "ILP Formulations for Optimal Task Scheduling with Communication Delays on Parallel Systems," in IEEE Transactions on Parallel and Distributed Systems, vol. 26, no. 1, pp. 142-151, Jan. 2015, doi: 10.1109/TPDS.2014.2308175.\
15. Zaman. M, A. Ahmadi and Y. Makris, "Workload characterization and prediction: A pathway to reliable multi-core systems," 2015 IEEE 21st International On-Line Testing Symposium (IOLTS), 2015, pp. 116-121, doi: 10.1109/IOLTS.2015.7229843.