LICHA – WORKLOAD CHARACTERIZATION USING LSTM BASED INTELLIGENT CLASSIFICATION FOR HETEROGENEOUS ARCHITECTURES

R.Sivaramakrishnan\textsuperscript{1} Dr.G.Senthilkumar\textsuperscript{2} \\
\textsuperscript{1}Research Scholar, \\
\textsuperscript{2}Associate professor/ECE SCSVMV University

ABSTRACT
Heterogeneous system-on-chip (HSoC) become ubiquitous in recent days. Industry 4.0, IoT, embedded devices, intelligent vehicles, cyber-physical system applications are extensively using such hardware architectures for processing its workloads. These real-time applications comprise a miscellaneous set of workloads with different characteristics which affects the computational processes extremely. Additionally, resource administration become a critical issue in HSoC. This paper presents a novel technique for workload characterization that enhances application performance in terms of both hardware-software resource mapping. A long short term memory (LSTM) based RNN classifier is proposed for HSoC platforms in order to predict ideal computational resource for each workload at runtime. The proposed classifier examined the implementation of several HSoC platforms in order to understand the working principle of real-time workloads at runtime. The observed characteristics are framed as real-time database and the same is utilized to train and test the LSTM classifier. The proposed classifier is evaluated on raspberry-pi HSoC and simulated on the python with ML library. Accuracy, throughput, sensitivity, selectivity metrics are detected to analyse the performance of the proposed algorithms. LICHA achieved the accuracy up to 96% compared with extreme machine learning predictor and also saved the execution energy up to 30% for real-time embedded benchmark workloads such as MiBench, IoMT.

Keywords: Accuracy, Heterogeneous architecture, Machine learning, Workload Characterization, Throughput

I. INTRODUCTION
Heterogeneous models become universal in current embedded frameworks. The construction of heterogeneous stage remember different centers for terms of CPUs, GPUs and Accelerators are incorporated on a similar chip. ARM cortex-53 just as incorporated GPUs and different gas pedals. Despite the fact that these models performed well, proficient string distribution on the separate centers become a basic issue because of the heterogeneity qualities of strings which burns-through additional time, energy and by implication influences the whole presentation. Viable schedulers ought to know about a framework's assorted computational assets, the changes in behaviours of thread, and have the option to distinguish designs identified with a threads presentation on various centres. Moreover, since applications may perform distinctively on particular centres sorts; a proficient scheduler ought to have the option to assess exhibitions to recognize an ideal planning plan. Planning figures out which thread to ship off which centre and is a difficult that imparts likenesses to proposal frameworks and route frameworks the two of which have profited utilizing ML. To conquer these issues at runtime, barely any articles zeroed in on responsibility portrayal for symmetric processors [1]. ML based supervised classifiers are created to foresee the string qualities of every responsibility in earlier. ML and DL methods have been applied with extensive achievement in the fields of PC vision, normal language handling for recommender frameworks. LSTMs specifically, are starting to be used in a wide assortment of fields because of their proficient learning in the middle of info information and downright yields.

The connections learned by the LSTMs are regularly difficult to recognize and program for physically yet can give phenomenal forecast correctness’s. Despite the fact that ML strategies have been acquiring footing in the course of the most recent couple of years, its application toward improving equipment execution stays in its
soonest organizes. According to our best of knowledge, applying ML for foreseeing string execution on heterogeneous cores and augmenting framework throughput is a novel strategy. The past examination offers excellent degree for energy saving applications in these multicore processor structures. "Miniature design Independent Workload Classifier" (MIWC) is utilized for the computation of processor jobs. New test seat has been proposed with the guideline of "Enhanced Vector Machines" (EVM) system and the energy utilization is quite diminished by carrying out these sorts of estimations. MIWC has been approved with the accompanying various benchmarks like Internet of Medical Things (IoMT), BEEBS, Mi Bench (Media Bench). EVM system demonstrates energy decrease in these kinds of gadgets by 25-30% of its unique utilization according to the above seat marks standard. The proposed work is the evidence of idea of the chances that emerge by applying ML to PC engineering plans. To accomplish this, the proposed framework selects the ML methodologies for scheduling in for heterogeneous frameworks since it is a territory which is shares likenesses with other people who have profited utilizing ML.

**Contribution of the proposed work**

In this paper, we use LICHA 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 LSTMs 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 one deep LSTM 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 LICHA 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.

**II. RELATED WORKS**

Hua Zheng proposed an “Optimizing Task Heterogeneous Assignment calculation with Probability Algorithm” for energy saving by using a probabilistic methodology on multi core architecture. Creators planned a model which utilizes least energy to accomplish ideal errand tasks for the time requirement and energy imperative. The outcomes infers that it is profoundly powerful in parallelism approach and giving unprecedented yield quality on multicore heterogenous recreation climate when contrasted and Integer Linear Programming (ILP) calculation [2]. Lei Yang presented another methodology known as various leveled programming way to deal with settle the issues on temperature and execution struggle on multicore framework. Scarcely any issues have been tended to like execution on application, correspondence on entomb processor, processor energy utilization. This methodology improves the idleness decrease on correspondence (.57%), energy productivity (23.04%), and decrease temperature (6.11° C) when contrasted and best in class strategies [3]. Dongjoo Shin proposed a reconfigurable profound learning energy-proficient processor in versatile stages particularly for intermittent neural organizations and convolution neural organizations. the proposed strategy utilizes 65nm innovation to manufacture the processor by using 3 key highlights. The main component is this design needs to help both RNNs as well as CNNs. Then reconfigurable multiplier is utilized unique fixed point for online variation. Finally, framework augmentation which utilizes quantization table to diminish of chip memory access and copied duplication. At long last, the outcomes are thought about different approaches and the work shows higher energy productivity in multicore stages [4]. Hergys Rexha examined energy acquire by using the recurrence and voltage scaling methods. The proposed philosophy fabricates a model and table for proficiency it totally relies upon stage designs. Assessment done on the varieties of energy productivity under different stage setups and giving comparable degree of execution. This framework results exhibit that trading the sort of center and number as per
the recurrence can prompt impressive increase in energy, the solitary restriction of the framework is it utilizes the group stage level so the effectiveness of the energy must be expanded for run time applications [5]. Antonio Pullini presented SoC known as MIA Wallace which is comprised of utilizing 65nm innovation and it is essentially utilized for IoT applications. This is utilized to run different ML calculations like neural organizations, since it has the ability to give signal, sound, video arrangement which is profoundly helpful in IoT applications [6].

Michel A. Kinsy acquainted Hermes multicore design with coordinate both secure and non-secure centre handling components in to a similar chip. This chip configuration is significant while keeping up the security of handling components, avoidance of debasement, information spillage and preparing components cooperation. The proposed engineering relies upon trust careful controlling estimation and secure interface of switch. This framework was empowered utilizing dispersed key administration framework for the execution of multicore framework with the handling components [7]. Juan Fang presented the calculation known as store substitution calculation for the streamlining and this calculation utilizes powerfully changing approach of CPU-GPU architecture. The calculation has been tried under different conditions to gauge the presentation. The proposed framework results show that this calculation makes huge effect on heterogeneous multicores for the presentation advancement. Reserve substitution calculation has a high effectiveness when contrasted with different calculations [8]. Loris introduced a novel registering engineering for the investigation of super low force bio signals. Its heterogeneous design executed on CGRA (coarse-grained reconfigurable exhibit) and contained with various processors which client shared speed increase asset. The CGRA work effectively upholds bio signals for the serious applications while requiring low design overhead. Furthermore, both the processors and the reconfigurable SIMD (Single guidance/various information) execution models to grow adequacy when various data streams are at the same time handled. on account of lightweight equipment, the run time conduct on the structure is arranged which all the while synchronizes the processors for execution of SIMD and control admittance to the reconfigurable gas pedal. by using the SIMD execution and run-time reconfiguration the proposed framework accomplishes when executing complex bio signal assessment applications, generally energy reserve funds up to 37.2 % sped up to 11.3x on the bits for the super low force multicore stage, which doesn't feature CGRA speeding up [9].

Yi-Jung Chen, presented new processor portion strategy for MPSoCs with single ISA heterogeneous multicore design. This framework consequently incorporates center portion for some random responsibility so it accomplishes the exhibition advancement when the asset compel is met. The processor dissemination issue for MPSoCs handled by this framework with design target. To get a best presentation of an equipment arrangement the proposed procedure integrated the plan of programming for the best errand planning. Result accomplishes 8.25% improvement in execution when it is contrasted and homogeneous engineering and when the territory obliges is met. Differentiated withal superior exhibitions, however the centers are excessively enormous, the quantity of cases is likewise set to comparable to the greatest parallelism level of the responsibility, the proposed method gas all things considered 11.5 % of corruption in execution while cost diminished by 60.7% [10]. Rakesh Kumar showed that HMCA (“Heterogeneous Multicore Architecture”) can give best throughput over homogeneous multicore framework. Evaluation is performed for not many string booking components to know the possible increase for execution open structure heterogeneity. Over a wide extent of stringing parallelism, the heterogeneous engineering performs 18% when contrasted and homogeneous design. The results infers that, this framework has less reaction timing than homogenous design. In like manner, the heterogeneous structure was consistent at passage rates that were up to 43% higher [11].

III. PROPOSED SYSTEM

3.1 LICH A - Working Mechanism

For the effective scheduling in multicore heterogeneous architecture, LICHA framework incorporates RNN with LSTM for prediction of energy and other features .The detailed theory behind the LSTM and ELM are explained as follows.

RNN - Recurrent Neural Network

A RNN is known to be outstanding kind of ANN and comprising input, covered up, yield layers. As indicated by FFN (Feed Forward Network), layer by layer availability of hubs shows up in the secret layer there are no association between the hubs. Be that as it may, agreeing RNN, similar secret layers hubs are associated. One of the significant attributes of RNN is that, it can effectively get familiar with the time arrangement information
since it has the capacity of encoding the earlier information into the way toward learning in the current secret layer. Design of RNN model is given in figure [11]

Node mapping is referred as $Z_m$ and which is represented as follows

$$Z_t = f(Ap_t + KZ_{t-1})$$  \hfill (1)

Where $Z_t$→ hidden state time [memory unit for the network], $F(.)$→ nonlinear function, $P_t$→ input time, 

A → shared parameters of every layers,

In RNN system, the immediate type of charts can be framed by hubs alongside its groupings. Thus unique conduct can be displayed for season of arrangements.

This uses inner memory for the cycle of groupings of info. So RNN network uses the previous information for the expectation of future qualities. In the event that the Interval time between the past information and current information for expectation is enormous in down to earth applications, at that point this procedure not ready to remember the previous information fundamentally, so still there is an evaporating slope issue, subsequently the anticipated results are not good in some constant situation. To defeat this issue, RNN execution has been improved, LSTM network has presented.

**Long-Short-Term Memory Network**

A LSTM is known to be an integration of RNN and LSTM mechanism which is illustrated in Figure 2. This framework is built by an input, output, and forget gates (IG, OG, FG) and cells. Cells involves in remembering few value in certain time intervals hence it is known as memories. Gates are considered as special structure with inputs 0’s and 1’s. The information gets passes through gate if the output gate value is equal to 1. The value does not gets passes through gate if the output value is equal to 0. The current input $P$, output vectors $O$ and gate calculation $Z(p)$ are represented as follows.

$$P= (P_1, P_2, \cdots, P_{t-1}, P_t)$$

$$O = (O_1, O_2, \cdots, O_{t-1}, O_t)$$
Fig. 2. Structure – LSTM unit

\[ Z(p) = \sigma(Kp + b) \]  

(2)

Where \( K \rightarrow \) weight matrix (WM), \( \sigma(P) = \frac{1}{1 + e^{-P}} \), \( b \rightarrow \) Bias Vector (BV).

Every time the current states are documented by the Cell states. This referred as calculation node and represented as follows

\[ C_t = f_t \cdot C_{t-1} + i_t \cdot \text{tanh} \left( Zc \cdot [P_{t-1}, P_t] + b_c \right) \]  

(3)

Where, \( bc \rightarrow \) BC of cell state, \( I \rightarrow \) IG, \( f \rightarrow \) FG, \( Zc \rightarrow \) WM of cell state

The Input and forget calculation can be respectively expressed as:

\[ i_t = \sigma(Zi \cdot [P_{t-1}, P_t] + b_i) \]  

(4)

\[ f_t = \sigma(Zf \cdot [P_{t-1}, P_t] + b_f) \]  

(5)

Where \( Z_i \rightarrow \) IG WM, \( Z_f \rightarrow \) FG WM, \( b_i \rightarrow \) IG BC, \( b_f \rightarrow \) FG BC. LSTM OG controls information about current time of cells state to current output.

The output \( (D_t) \) can be expressed as:

\[ D_t = \sigma(Zo \cdot [O_{t-1}, P_t] + b_o) \]  

(6)

where \( Zo \rightarrow \) OG WM \( b_o \rightarrow \) OG BC

LSTM training phase incorporates the back propagation algorithm and the final output is represented as follows,

\[ O_t = D_t \cdot \text{tanh} \left( c_t \right) \]  

(7)

3.2 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 [16] because the extracted features are large in size. Table 2 presents the thread analysis features. The training model selects different benchmarks which are mentioned above.
Table 2. Important Features of Thread Analysis

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

IV. LICHA MECHANISM

LICHA is a hybrid model incorporates different 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 LSTM. 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 LSTM mechanism is illustrated as follows:

The hybrid model (classification model) is represented as

\[ \hat{y}(p) = C_0 + C_1 Y_s(P) \]  

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

\[ \{(P^i, y^i) | P^i \in R^l, y^i \in R \}_{i=1}^L \]  

\(Y_s(P^i)\) is the LSTM practical output

Linear classification of training dataset is presented as

\[ \{(y_s(P^i), y^i)\}_{i=1}^L \]  

Newly generated training dataset is

www.turkiphysiotherrehab.org
We assume that

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

Then above mentioned formula is modified as

\[ AC = Y \]

\[ A = \begin{bmatrix} 1 & Y_s (P^1) \\ 1 & Y_s (P^2) \\ \vdots & \vdots \\ 1 & Y_s (P^L) \end{bmatrix} \]  

\[ C = [C_0 \ C_1 \ C_2]^T \]  

\[ Y = [y^1 \ y^2 \ y^L]^T \]

Hybrid model parameters is represented as follows

\[ \hat{C} = A^+ \]

Where \( A^+ \rightarrow \) Moore-Penrose Pseudo inverse Matrix of A

4.1 Data Pre-Processing

For the LSTM network, the output value ranges from -1 to 1, if the activation function is selected as tanh. In out experimentation, Normalization is required to ensure the correctness of the results for the electrical load.

V. 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>
<thead>
<tr>
<th>Sl. No</th>
<th>Core’s type tested</th>
<th>Test_Bed_1</th>
<th>Test_Bed_2</th>
<th>Test_Bed_3</th>
<th>Test_Bed_4</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Cortex-A5 Series</td>
<td>ARM-9 Series</td>
<td>Cortex-A5 Series</td>
<td>Cortex-A8 Series</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>ARM-9 TDMI Series</td>
<td>ARM-9 TDMI Series</td>
<td>Cortex-R Series</td>
<td>Cortex A-5 Series</td>
<td></td>
</tr>
<tr>
<td>03</td>
<td>Cortex-A5 Series</td>
<td>CortexA-5 Series</td>
<td>Programmable FPGA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>04</td>
<td>Cortex A Series</td>
<td>Programmable FPGA</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
MATLAB toolboxes version > R2012 is utilized for the LICHA framework and this framework view is presented as follows.

### 5.1 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, lbmaster, 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.

## VI. RESULTS AND DISCUSSION

This section gives the performance analysis of LICHA framework with the comparison of other exiting algorithms **Performance Metrics**

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

The below figures illustrate that proposed LSTM improved 3% 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 5 to 11 shows the other performance metric enhancements achieved. X-axis the performance metrics such as “MAE, RMSE, MRE, Throughput and Accuracy” and Y-axis denotes the application workloads.

(i) **MAE – Mean Absolute Error**

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

\[
\text{Mean Absolute Error (MAE)} = \frac{1}{L} \sum_{y=1}^{L} |\hat{y}^i - y^i| \tag{17}
\]

Where \(\hat{y}^i\) \rightarrow predicted value of the workload, \(y^i\) \rightarrow real values of the workload, \(L\) \rightarrow number of test or training samples.
(ii) **Root Mean Square Error (RMSE)**

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

\[
\text{Root Mean Square Error (RMSE)} = \sqrt{\frac{1}{L} \sum_{l=1}^{L} (\hat{y}^l - y^l)^2}
\]  

(19)

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

(iii) **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|}

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

(iv) Throughput Measurement
Throughput of the proposed framework has been measured with the different parameters taken from the workloads.

(v) 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.
\[ \text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions made}} \]  \hspace{1cm} (17)

Fig. 9. Accuracy Measurement for various workloads

From the above figures infers that, LICHA 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.

\[ \text{SENSITIVITY} = \frac{\text{TP}}{\text{TP} + \text{TN}} \]

**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.

\[ \text{Selectivity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \]
VII. CONCLUSION

In this paper we have spearheaded applying ML to a throughput amplifying heterogeneous by LICHA model fit for anticipating the presentation of different threads on assorted framework assets at the scheduling quantum. We have shown how LSTM 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% to 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 LSTM 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


