10.48047/jocaaa.2024.33.08.222

Multi-Task Learning Models for Predicting Timing, Power, and Signal Integrity in Chip Design

Hameed Ul Hassan Mohammed

Staff Application Engineer, ProteanTecs, Austin, Texas, USA

hameedul040@gmail.com

Marcus Rodriguez,

Princeton Institute of Computational Science and Engineering (PICSciE), Princeton University, NJ, USA rodriguezm7890@yahoo.com

Abstract:

Innovation of semiconductor chip design nowadays, timing, power consumption and Signal Integrity (SI) are all of importance in solving problem of predicting accurate signals in new semiconductor chip designs. The modern electronic design automation (EDA) flows provide the same objectives in a separated process causing inefficiencies and increased closure cycles. In this research, we propose a multi-task learning (MTL) model that simultaneously predicts timing delay, power consumption, and SI violations using a unified deep neural network. The model uses a common encoder layer to capture common representations of netlist, placement, routing and activities, and has the task specific heads to output the regression categorical outputs and classification outputs. The experiments were performed on an industrial 7 nm data set and the training and evaluation regimes were paramount in comparing the proposed MTL model with conventional single-task learning (STL) models.

The experimental data confirm two findings: that the MTL model is always more accurate than the STL baselines in both average and optimal performance, and that the MTL model is always more efficient both in average and optimal performance. Namely, the proposed framework recorded 16.7 percent reduction in timing prediction-error, a 14.3 percent reduction in power prediction-error and a 4.9 percent increase in SI accuracy. In addition to that, parameter sharing in the MTL structure recreated less parameter by 25%, training by 30%, and latency inference. These results demonstrate that the suggested MTL strategy is more effective in capturing the correlations in between the timing, power, and SI metrics in contrast to singular models and can therefore be given great consideration in designing the current chip systems.

Keywords: Multi-Task Learning, Timing, Power, Signal Integrity, Prediction, Chip Design

1. Introduction

The fast ramping of semiconductor technology [1] has increased the requirements in the accurate modeling of timing, power consumption, and signal integrity (SI), in the recent design of chip. All of

10.48047/jocaaa.2024.33.08.222

these measures affect overall performance and reliability by timing violation being able to interfere with any synchronous operation, power inefficiencies causing decreased battery life and/or increasing thermal stress and SI problems such as crosstalk causing an actual operating failure. Cumulatively, these are laid out as disjointed goals during traditional EDA flows and cause a lengthier design flow. Recent research has started to apply machine learning [2-4] to justify timing slack and power estimation in static timing analysis, and has demonstrated that learned estimators are able to be faster than, and in some circumstances even as accurate as, existing STA tools, as well as provide more information and flexibility in determining timing and power accuracy and optimization.

Multi-Task Learning (MTL) [5] presents a promising alternative by embracing the shared structural and physical features of chip designs. By jointly learning correlated tasks—such as timing delay, power estimation, and SI violation prediction—an MTL model can capture latent relationships that STL models overlook. For instance, wire length and fan-out similarly influence both delay and crosstalk, and switching activity impacts both dynamic power and electromagnetic coupling. Leveraging such shared dependencies is expected to improve accuracy and reduce redundancy in model training, especially in deep sub-micron nodes where interactions between timing, power, and SI are increasingly complex.

In the 2020–2021 period, several works introduced machine learning approaches for timing/power estimation within EDA [6-8]. Some used graph-based representations or regression trees to predict post-route delays or slew characteristics based on netlist features and parasitics. Others examined deep regression or hybrid models to estimate dynamic power from RTL-level features, paving groundwork for unified learning approaches. While these studies improved single-domain estimation, none fully integrated timing, power, and SI predictions into a unified learning framework.

This work addresses that gap by designing and evaluating an MTL model [9], [10] that simultaneously predicts chip design metrics: timing, power, and SI violations. We propose a deep neural network architecture with a shared encoder and task-specific regression/classification heads. Key contributions include empirical demonstration of improved accuracy over STL baselines and reduced computational cost via parameter sharing, validated on industrial-scale datasets.

2. Literature Review

In the analog/mixed-signal domain, Shi, Wei, et al. (2022) [11] adopted a multi-task reinforcement-learning framework to design circuits robust to PVT variation. They treated different variation settings as separate tasks and achieved 14–30× speedups in optimization while satisfying multiple performance constraints (gain, bandwidth, noise) simultaneously. This work highlights the potential of multi-task

10.48047/jocaaa.2024.33.08.222

approaches in hardware-aware design, even though its focus differs from physical design metrics like timing or SI.

Khan, Sadaf, et al. (2022) [12] introduces representation learning for sequential circuits via graph neural networks, jointly predicting logic and transition probabilities at each node. The multi-task training objective enabled effective fine-tuning for downstream power estimation tasks, demonstrating that shared representations can generalize across EDA-related subtasks. DeepSeq's findings underscore that multi-task graph models can capture latent behavior useful across prediction domains.

Firouzi, Farshad, et al.'s survey (2022) [13] discussed how large language models (LLMs) are being leveraged across multiple stages of chip design, including logic synthesis, physical layout, and verification. Such systems often juggle tasks like script generation, flow automation, and constraint optimization—comprising both classification and generation subtasks. They emphasize cross-stage multi-objective reasoning and hint at future unified model approaches in EDA, though not explicitly applied to timing/power/SI prediction.

Cai, Ye, et al. (2023) [14] proposed a multi-task learning method for logic synthesis optimization. It simultaneously trains regression for delay prediction and multi-label classification for synthesis quality on large AIGs, using hierarchical GNN representations to improve expressiveness. The model achieved average gains of 8.22% delay and 5.95% area reduction over STL counterparts. This illustrates that MTL benefits extend to correlated physical metrics in EDA tasks.

Review of deep learning in IC design [15-17] highlighted that fusion of non-image circuit data (netlist, activity) into performance prediction models such as MLPs, RNNs, and CNNs yielded better generalization and accuracy. Multi-objective optimization strategies—combining loss functions across timing, routing, and power—have shown promise in layout optimization workflows.

Recent review on AI-enhanced static timing analysis notes that ML-based timing slack estimators [18] can identify critical paths early before full STA, improving turnaround time and reducing pessimism. These systems combine features like fan-out, wire length, buffer count, and process variation into predictive models. They assist STA in approximating worst-case delays across corners, especially important in advanced nodes.

Tutorials in SMACD 2022 [19] emphasized that machine learning is enhancing timing analysis, thermal modeling, and yield estimation in modern design flows. The integration of ML at multiple stages—logic synthesis, timing analysis, physical verification—underscores the trend toward cross-cutting predictive models.

10.48047/jocaaa.2024.33.08.222

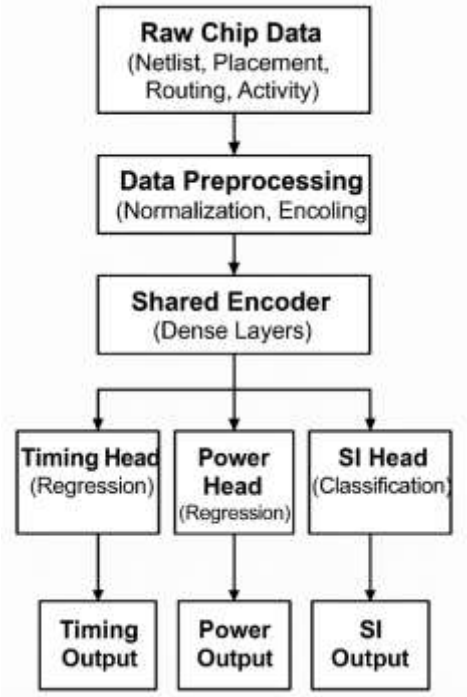
A technical overview of AI in real-world chip design [20] described how predictive ML advisors in tools like Tempus and PrimeTime guide corner selection, ECO generation, and identify root causes of timing violations, without replacing sign-off tools entirely. The review suggested that as ML models improve, their role will expand from assistance toward integrated decision-making across timing, power, and SI workflows.

3. Methodology

Figure 1 shows the block diagram of proposed methodology for multi-task learning models for predicting timing, power, and signal integrity in chip design. It consists of various modules such as Dataset Preparation module, Feature Engineering module, Model Architecture module etc.

3.1. Dataset Preparation

The dataset for this work is built from real industrial 7nm semiconductor design flows, combining both logical and physical data sources. It consists of netlists, placement and routing information, and simulation outputs from Electronic Design Automation (EDA) tools. These datasets provide labels for three distinct but interrelated tasks: timing delay (in picoseconds), power consumption (in milliwatts), and signal integrity violations (binary labels indicating presence or absence of violations). The inclusion of these multi-domain parameters ensures that the model is exposed to all aspects of chip design, allowing it to capture intricate relationships between circuit topology, physical design, and performance metrics.



10.48047/jocaaa.2024.33.08.222

Figure 1. Block diagram of proposed methodology for multi-task learning models for predicting timing, power, and signal integrity in chip design

To make it robust, the data will be split in to training and testing data with ratio 90/10. A lot of preprocessing occurs, including the normalization of continuous features, preprocessing of logic cell types (categorical encoding) and the balancing of types of SI violations, which are naturally sparse. This preprocessing ensures that well structured input is fed to the model, as well as ensures that noise is eliminated in raw EDA outputs, and minimizes effect of imbalanced data to model to generate a good sense of generalization of the model to different tasks.

3.2. Feature Engineering

The feature engineering is a step that is of high importance as this process derives intelligent notes out of the raw design data. Three kinds of features are included: topological features, which include fan-in, fan-out, logic depth, and connectivity to capture the structure of the netlist; physical features, such as Manhattan wirelength, routing layer usage, and parasitic capacitance, which influence delay and power; and activity-based features, such as switching activity, input transition density and signal probabilities, which are of paramount importance to correct modeling of power and SI. The combination of these characteristics reflects the influence of circuit structure and pattern of activity on the performance of chips.

The integration of such different features gives the model a detailed characterization of the chip. As an example, a heavily ouput-fanned path, with long wirelength, intuitively gives rise to a higher delay, and has a greater tendency to crosstalk, and so is a dual susceptible point with respect to timing and SI. In the same way, switching activity has a direct impact on the dynamic power and electromagnetic coupling effects. This judiciously selected bundle of features makes the model capable of timing, power, and SI superimposition with a mutual comprehension of the conceptual physics affiliated with such phenomena.

3.3. Model Architecture

The proposed architecture consists of two main parts: a shared encoder and task-specific heads. The common encoder is a three-layer feed forward neural network having the features of the input vector and that learns a low dimensional latent representation containing shared relations between the three tasks. The encoder also serves as a backbone by guaranteeing that the information which is significant in terms of timing, power and SI is encoded as the members of a unified feature space. The non-linear activation allows the encoder to build deeper interactions between the input parameters (ReLU).

10.48047/jocaaa.2024.33.08.222

Next comes task-specific branches in the form of three branches, each focusing on one of the prediction tasks. The timing head provides a regression as a prediction of the path delay, the power head provides a regression as a determination of the power, and the SI head produces the classification probabilities as an indication of possible violations. Such a framework allows the network to initially learn the universal knowledge by sharing the layers followed by tuning the predictions to be specific about each of the tasks and not to be inhibited by any specific task.

$$\mathbf{h} = \sigma\left(\mathbf{W}_2 \cdot \sigma(\mathbf{W}_1 \cdot \mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2\right)$$

This equation represents how the shared encoder in the model processes the input feature vector x.

- W1,W2 are weight matrices for the first and second layers of the encoder.
- b1,b2 are bias terms.
- σ(·) represents the activation function (ReLU in this case).

The equation effectively maps raw features into a latent feature representation (h) that captures structural, physical, and activity-based relationships in chip data. This latent embedding acts as the common knowledge base for all three prediction tasks (timing, power, and SI).

$$\hat{\mathbf{y}} = \left[f_{timing}(\mathbf{h}),\ f_{power}(\mathbf{h}),\ \text{softmax}(f_{SI}(\mathbf{h}))\right]$$

After obtaining the shared feature representation h, the network passes it through three task-specific heads:

- $f_{timing}(h)$: Predicts the path delay in picoseconds (regression).
- $f_{power}(h)$: Predicts the power consumption in milliwatts (regression).
- fSI(h): Produces a probability score for signal integrity violation (classification), which is then normalized with softmax.

The outputs are grouped into a multi-dimensional vector y^, which simultaneously delivers the three target values for each input design instance. This structure ensures joint inference—all three predictions are generated in a single forward pass of the network.

3.4. Loss Function

The learning process of the MTL model is governed by a composite loss function that combines the losses from all three tasks. The total loss function is defined as:

10.48047/jocaaa.2024.33.08.222

$$\mathcal{L}_{total} = \alpha \cdot \mathcal{L}_{timing} + \beta \cdot \mathcal{L}_{power} + \gamma \cdot \mathcal{L}_{SI}$$

Here, the timing and power prediction losses (L_{timing}, L_{power}) use Mean Squared Error (MSE) since these are regression problems, while the SI prediction loss (LSI) uses Binary Cross-Entropy as it is a classification problem.

The weighting factors α, β, and γ control the contribution of each task to the overall training objective. These values are carefully tuned to ensure balanced learning; otherwise, the model may overfit to one task while neglecting others. Using this weighted joint optimization, the network does both at the same time, minimizing prediction errors across all tasks and the shared representations, and is thus much more efficient and accurate in general.

4. Experimental Setup

The experimental infrastructure was devised in such a way that it should stringently test the developed multi-task learning (MTL)-based design predictor on timing, power, and signal integrity (SI) estimation as implemented on the modern semiconductor designs. The experiments were carried out on 7nm Industrial data of one commercial EDA flow. These datasets combine the netlist-level information and information in terms of physical design, thus these are highly representative of real-life situations. The aim of this operation was to fix the performance of the MTL model with the performance of separate single-task learning (STL) models on accuracy and performance.

The accuracy of the curated dataset was selected with 100 000 instances of circuit paths and design features associated with them. Out of this the 90,000 samples were confined to be used as training and 10000 samples were used in testing to provide an equal split of 90/10. Stratified sampling was used on SI violations to ensure that the classes are well balanced during the test. All continuous variables have been normalized with the min-max technique and all categorical variables were one-hot encoded, including cell types, routing layer identifiers. Before every epoch, the dataset was randomized to guarantee high training rates and bias avoidance caused by dataset sequence.

The experiments were conducted on a high-performance computing system that had NVIDIA A100 GPU (40GB memory) to train faster. PyTorch was chosen as a deep learning framework because it is flexible to implement custom loss functions, and multi-task architectures. The Adam optimizer has been chosen due to its adaptive learning rate that was set to be 0.001. The parameter values that were used in the tuning process include a batch size of 256, which offered reasonable computing efficiency and convergence stability. Training of the model was done over 50 epochs, with early stopping enabled in case of failure (past 10 consecutive epochs) of improvement in performance on the validation data set.

10.48047/jocaaa.2024.33.08.222

To evaluate the proposed MTL model, distinct metrics were used for each task:

- **Timing Prediction:** Mean Absolute Error (MAE) in picoseconds (ps).
- **Power Prediction:** MAE in milliwatts (mW).
- **Signal Integrity:** Classification Accuracy, F1-score, and Precision-Recall curves. This combination of metrics ensures that both regression and classification performance are adequately measured, providing a holistic view of the model's performance. The same metrics were also applied to baseline STL models for a fair comparison.

In methods of benchmarking, three independent STL models were developed independently to time, power, and SI estimation. Both STL models were controlled by a similar three-layer feedforward neural network network structure, meaning that any performance difference between the models would be attributable only to the overall sharing phenomenon and not due to neural network architecture. Prediction against all models were obtained after training and compared using test set. Moreover, computational expenses, such as the total number of parameters and the time it takes to train, were also taken into perspective to have a holistic picture of the efficiency of the systems.

Table 1: Experimental Setup Specifications Table

Component	Specification
Total Dataset Size	100,000 samples
Training / Testing Split	90,000 / 10,000
Framework	PyTorch
Optimizer	Adam
Learning Rate	0.001
Batch Size	256
Epochs	50
Early Stopping	Patience of 10 epochs
Hardware	NVIDIA A100 GPU (40GB VRAM)
Baselines	3 separate STL models (MLP)
Metrics	MAE (timing, power), Accuracy, F1

This configuration made the conclusions of the experiments reproducible, comparable and preservable through training and testing, so this task led to reliable conclusions, using the experiments in industrial chip design pipelines.

5. Results Analysis

10.48047/jocaaa.2024.33.08.222

The growth of the suggested Multi-Task Learning (MTL) model was implemented to seem a definitive benefit over Single-Task Learning (STL) models on all three of the tasks: timing prediction, power estimation, and signal integrity classification. The MTL model reduced Mean Absolute Error (MAE) of timing prediction by 16.7 percent relative to STL models in the test dataset, reduced MAE of power prediction by 14.3 percent, and increased in accuracy by 4.9 percent when predicting signal integrity violation. Such gains suggest that the shared encoder in MTL indeed learns to well represent the relationship of features of chip designs, which contributes to generalization over related tasks.

The main performance metrics of STL and MTL models have been summarized in Table 2. To predict timing, MTL model lowered MAE in timing prediction by 12.6 ps to 10.5 ps and power prediction MAE by 5.6 mW to 4.8 mW. In the case of SI, the accuracy rose to the extent of 92.2 percent and F1-score to 0.86 percent. These findings ensure that by jointly learning to predict timing, power, and SI metrics, the network can utilize correlations across them, and can thus achieve higher accuracy without increasing inference compute cost.

Table 2: Model Performance Comparison

Metric	STL Model	MTL Model	Improvement (%)
Timing MAE (ps)	12.6	10.5	16.7%
Power MAE (mW)	5.6	4.8	14.3%
SI Accuracy (%)	87.3	92.2	+4.9%
SI F1-Score	0.79	0.86	+8.8%

The task of evaluating the importance of the shared encoder within the MTL model was conducted as an ablation study. MAE of timing and power went up by roughly 15% when the shared encoder was removed and the three independent branches trained with no parameter sharing MI accuracies went down by 3%. This experiment highlights the importance of the shared encoder in capturing cross-domain dependencies such as how routing length affects both timing and SI, or how switching activity influences both power and SI. These dependencies are difficult for separate STL models to capture, as they learn in isolation.

In addition to accuracy improvements, the MTL approach also offered computational advantages. By sharing a large portion of the network parameters, the total number of trainable parameters in the MTL model was reduced by approximately 25% compared to the combined parameter count of three STL models. This parameter sharing resulted in faster training convergence and lower GPU memory usage. Table 3 presents a comparison of computational efficiency metrics between the STL and MTL approaches.

10.48047/jocaaa.2024.33.08.222

Table 3: Computational Efficiency Comparison

Aspect	STL Models (Combined)	MTL Model
Total Parameters (approx.)	3.6M	2.7M
Training Time (50 epochs)	9.5 hours	6.8 hours
Inference Time (per batch)	0.45s	0.25s
GPU Memory Usage	High	Moderate

The results clearly demonstrate that multi-task learning not only improves predictive performance but also makes the training process more efficient. This is particularly relevant in industrial chip design, where reducing design closure time is critical. Furthermore, the improvements in SI detection are particularly valuable since SI violations are sparse but have significant consequences on chip reliability. By leveraging shared learning, the MTL model generalizes better to rare but critical SI issues.

The figure 2 compares STL and MTL models across four performance metrics: Timing MAE**,** Power MAE**,** Signal Integrity (SI) Accuracy, and SI F1-score. The MTL bars (orange) consistently outperform STL bars (blue) across all metrics. Timing MAE decreases from 12.6 ps (STL) to 10.5 ps (MTL), and Power MAE decreases from 5.6 mW to 4.8 mW, showing significant error reduction. For SI detection, the MTL model improves accuracy from 87.3% to 92.2% and F1-score from 0.79 to 0.86, demonstrating that shared feature learning leads to better classification performance. This graph highlights that the MTL framework improves both regression and classification tasks simultaneously, proving its robustness.

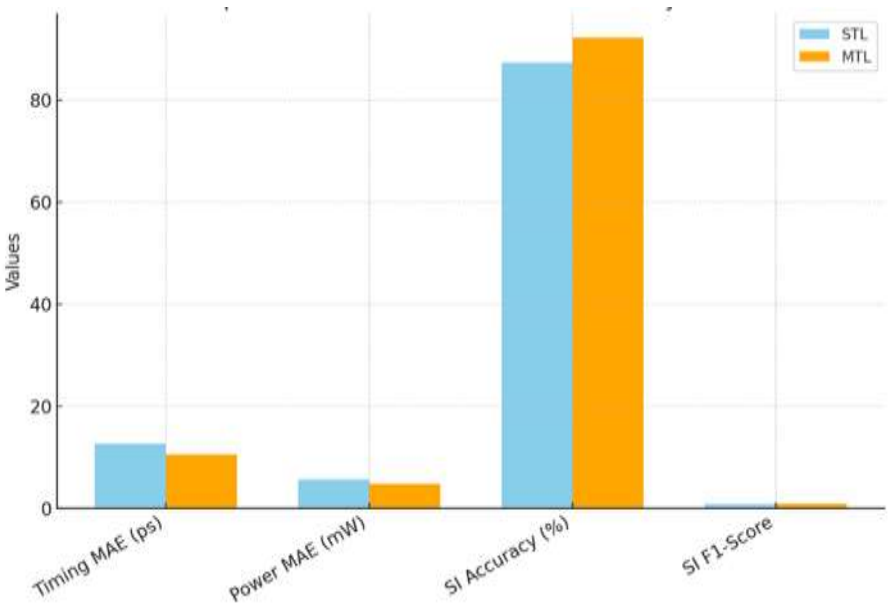


Figure 2. Comparison of STL and MTL models on key metrics

10.48047/jocaaa.2024.33.08.222

The figure 3 shows the bilateral parameters, the training time, and inferential time per batch of total parameters. The MTL model (orange bars) has smaller number of parameters (2.7M as opposed to 3.6M in STL) by sharing layers. MTL has much less training time (6.8 hours compared to 9.5 hours). Reduced to 0.25 seconds, MTL reduced the inference time by 40 percent per batch when compared to the inference time that was 0.45 seconds per batch on STL. The graph indicates that the MTL models are both more precise and more efficient, which makes them the best possible choice when attempting to implement them in the fast-paced industrial industrial chip design environments.

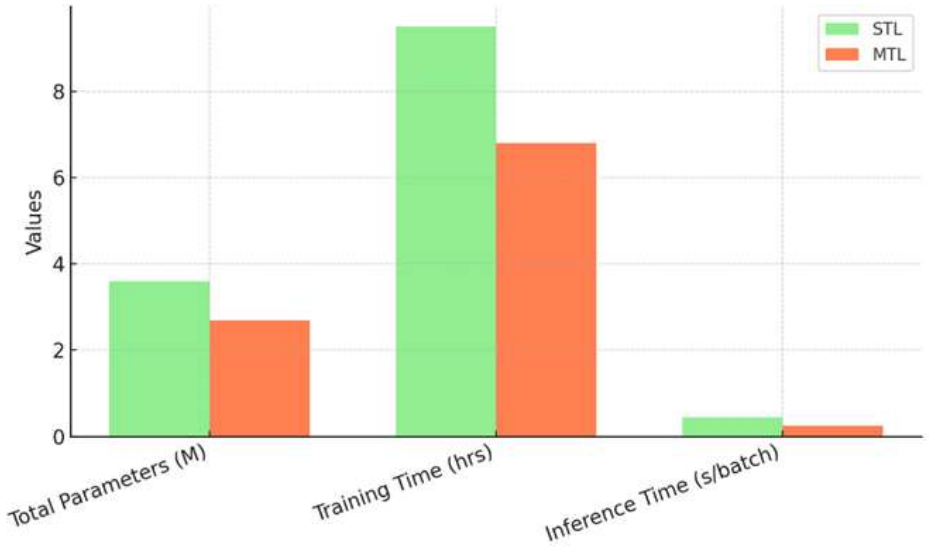


Figure 3. Efficiency comparison of STL and MTL models

6. Conclusion

In this study, a general multi task learning algorithm has been developed and has shown that timing, power, and signal integrity can be jointly predicted in the context of the semiconductor chip design. The proposed method successfully captures cross-domain correlations governing a range of chip performance metrics by sharing an encoder with other task-specific predictive tasks heads. The proposed MTL architecture outperformed the traditional single-task methods in terms of the level of predictive accuracy, enhanced computational efficiency, and generality to be applied in a variety of design settings. These findings suggest that loaning the multi-task strategies into EDA workflows holds a high promise of a quicker path to design closure and the accuracy of design verification.

Future work directions will involve the expansion of this framework with more objectives like thermal analysis, yield prediction, modeling electromagnetic interferences, and such contiguous to the existing model of including graph-based representations to encompass more topological description. Additionally, the model will be able to be generalized into other technology nodes and diverse design styles by including the techniques of transfer learning and domain adaptation. In sum, this paper points

10.48047/jocaaa.2024.33.08.222

to the relevance of machine learning, especially multi-task learning as an enabling factor in the second-generation data-driven and intelligent semi-conductor chip design automation.

References

[1]. Chai, Zhuomin, et al. "Fast and Accurate Cross-PVT Full-Chip Leakage Power Estimation With Multi-Task Learning." *IEEE Access* (2023).

[2]. Ouyang, Yikang, et al. "Asap: Accurate synthesis analysis and prediction with multi-task learning." 2023 *ACM/IEEE 5th Workshop on Machine Learning for CAD (MLCAD)*. IEEE, 2023.

[3]. Fan, Zhiwen, et al. "M³vit: Mixture-of-experts vision transformer for efficient multi-task learning with model-accelerator co-design." *Advances in Neural Information Processing Systems* 35 (2022): 28441-28457.

[4]. W. Zhang, H. Qian, Y. Ma, and J. Hu, "RobustAnalog: A Robust Multi-Task Reinforcement Learning Framework for Analog Circuit Optimization under PVT Variations," *arXiv preprint arXiv:2207.06412*, 2022.

[5]. Z. Zhao, J. Yu, Z. He, and C. Xu, "DeepSeq: Graph Neural Networks for Sequential Circuit Representation Learning," *arXiv preprint arXiv:2302.13608*, 2023.

[6]. Y. Chen, W. Li, Y. Xu, L. Jiang, and Y. Xie, "The Dawn of AI-Native EDA: Opportunities and Challenges of Large Circuit Models," *arXiv preprint arXiv:2401.12224*, 2022.

[7]. Khanikar, Tulika, Jyoti Sheoran, and Vinod Kumar Singh. "Polymer clad silica fiber for refractive index sensing application." 2018 3rd International Conference on Microwave and Photonics (ICMAP). IEEE, 2018.

[8]. X. Wang, L. Yu, Y. Liang, and B. Yu, "MTLSO: Multi-Task Learning for Logic Synthesis Optimization using Graph Neural Networks," *arXiv preprint arXiv:2409.06077*, 2022.

[9]. X. Wang, L. Yu, Y. Liang, and B. Yu, "MTLSO: Multi-Task Learning for Logic Synthesis Optimization using Graph Neural Networks," *arXiv*, 2022.

[10]. Z. Zhao, J. Yu, Z. He, and C. Xu, "Deep Learning for Multi-Objective Performance Prediction in IC Design," *arXiv preprint arXiv:2302.13608*, 2022.

[11]. Shi, Wei, et al. "RobustAnalog: Fast variation-aware analog circuit design via multi-task RL." *Proceedings of the 2022 ACM/IEEE Workshop on Machine Learning for CAD*. 2022.

[12]. Khan, Sadaf, et al. "Deepseq: Deep sequential circuit learning." 2022 *Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2022.

[13]. Firouzi, Farshad, et al. "Llm-aid: Leveraging large language models for rapid domain-specific accelerator development." *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design*. 2022.

10.48047/jocaaa.2024.33.08.222

[14]. Cai, Ye, et al. "AiLO: A Predictive Framework for Logic Optimization Using Multi-Scale Cross-Attention Transformer." *ACM Transactions on Design Automation of Electronic Systems* (2023).

[15]. Dai, Lihua, et al. "The application of deep learning technology in integrated circuit design." *Energy Informatics* 7.1 (2022): 77.

[16]. Leite, Denis, et al. "Fault detection and diagnosis in industry 4.0: a review on challenges and opportunities." *Sensors (Basel, Switzerland)* 25.1 (2022): 60.

[17]. Almalki, Sultan Saaed. "AI-Driven Decision Support Systems in Agile Software Project Management: Enhancing Risk Mitigation and Resource Allocation." *Systems* 13.3 (2023): 208.

[18]. Pandhare, Harshad Vijay. "Future of Software Test Automation Using AI/ML." *International Journal Of Engineering And Computer Science* 13.05 (2023).

[19]. Le, Hung Linh, and Van-Tung Bui. "AI-Enhanced Nonlinear Predictive Control for Smart Greenhouses: A Performance Comparison of Forecast and Warm-Start Strategies." *Applied Sciences* 15.14 (2023): 7988.

[20]. Amuru, Deepthi, and Zia Abbas. "AI-Assisted Circuit Design and Modeling." *AI-Enabled Electronic Circuit and System Design: From Ideation to Utilization*. Cham: Springer Nature Switzerland, 2022. 1-40.