Enhancing ML model accuracy for Digital VLSI circuits using diffusion models

Prasha Srivastava
IIIT, Hyderabad, India

Pawan Kumar
IIIT, Hyderabad, India

Zia Abbas
IIIT, Hyderabad, India

Abstract

Generative AI has seen remarkable growth over the past few years, with diffusion models being state-of-the-art for image generation. This study investigates the use of diffusion models in generating artificial data generation for electronic circuits for enhancing the accuracy of subsequent machine learning models in tasks such as performance assessment, design, and testing when training data is usually known to be very limited. We utilize simulations in the HSPICE design environment with 22nm CMOS technology nodes to obtain representative real training data for our proposed diffusion model. Our results demonstrate the close resemblance of synthetic data using diffusion model to real data. We validate the quality of generated data, and demonstrate that data augmentation certainly effective in predictive analysis of VLSI design for digital circuits.

1 Introduction

In recent times, generative AI has experienced remarkable growth, propelled by advances in machine learning (ML) models like Generative Adversarial Networks (GANs) [1, 2], Variational Autoencoders (VAEs) [3], and the most recent addition, Denoising Diffusion Probabilistic Models (DDPM) [4]. These models excel at crafting realistic, high-quality data, gaining traction across domains like image synthesis, text generation, and music composition. This surge in generative AI holds immense potential to reshape ML in diverse fields, including VLSI design, testing, and optimization, particularly in data-sparse scenarios. Diffusion models have demonstrated excellence in computer vision, natural language processing, and interdisciplinary realms such as medical image reconstruction. Precise diffusion models generating high-quality artificial circuit data can revolutionize ML-driven VLSI tasks—performance assessment, design, and testing [5, 6, 7, 8, 9, 10, 11, 12]. An effective data augmentation approach will help addresses concerns such as privacy, proprietary data access, computation costs, and data acquisition constraints. Our study focuses on practical application of VLSI circuit design. We study the effectiveness of synthetic data generation method for VLSI circuit data using Denoising Diffusion Probabilistic Model (DDPM). We show that diffusion models can generate high-quality samples even for circuit data. We present a detailed analysis of their performance over delay estimation of twelve fundamental 22nm CMOS technology-based digital cells, see Table 5.

2 Related Works

Data scarcity is a universal issue which affects the deployment of AI/ML in multiple domains as discussed in [13]. Many of the proposed AI/ML applications in VLSI design domain [8, 10, 12] rely on a large amount of training data. In [12] and [14], authors have achieved precise training with $15K$ and $50K$ samples respectively. However, obtaining such a substantial volume of training data may not always be feasible for various applications. Hence in VLSI domain, scarcity of training data due to cost, time, and quality constraints poses a challenge which needs to be addressed. Many studies have proposed synthetic data generation [15, 16, 17, 18, 19, 20, 21, 22, 23]. Generative models such

Machine Learning for Systems Workshops at 37th NeurIPS 2023.

as VAE [3], GAN [1], and Diffusion Probabilistic models [4] are known to produce large and diverse synthetic data for image datasets. Diffusion models have been observed to have an edge over other generative models in terms of quality for image data generation [24].

3 Synthetic circuit-data generation using Diffusion Models

Parametric data from VLSI designs is continuous data that holds valuable insights for ML-driven automation of VLSI performance assessment, design, and testing. As mentioned before, this paper aims to apply denoising diffusion probabilistic models for generating synthetic data for VLSI designs, enhancing the training of ML models in data-scarce scenarios. This indirectly advances automation in VLSI design tasks. Our work illustrates the development of an accurate diffusion model-based synthetic data generation method for delay estimations in various 22nm CMOS technology-based digital VLSI circuits. The details on the dataset can be found in appendix (Section 7.1). The methodology can be divided broadly into the following steps:

3.1 Formulation of a Denoising Diffusion Probabilistic Model tailored for VLSI circuit-data

Diffusion models define a Markov chain of diffusion steps to slowly add random noise to data and then learn to reverse the diffusion process to construct desired data samples from the noise. Diffusion models have two processes to follow:

Forward Process - Here, random noise is incrementally added to data over multiple time steps. This sequential noise introduction is mathematically characterized as follows: $x_t = \sqrt{1-\beta_t} \cdot x_{t-1} + \sqrt{\beta_t} \cdot \epsilon$. Where x_t represents the data at time step t, where $0 < t < T$ and T are the total number of steps. This process starts with the original data x_0 and iteratively adds noise over T steps, with β_t controlling the amount of noise added at each step. The variance schedule β_t determines the trade-off between introducing noise and maintaining data fidelity. As t increases, the noise contribution becomes more significant due to the increasing value of β_t.

Reverse process – This phase involves predicting the noise added to each data point during the forward process. A neural network, denoted as f_θ, predicts noise ϵ_t from noisy data x_t as $\epsilon_t = f_\theta(x_t)$

Generating new data - New data samples are generated by performing the reverse process on random noise samples ϵ_t drawn from $N(0, I)$. The reverse process reconstructs data by iteratively removing predicted noise contributions: $x_{t-1} = \frac{x_t - \sqrt{\beta_t} \cdot \epsilon_t}{\sqrt{1-\beta_t}}$. The number of steps T controls the balance between data fidelity and noise injection, influencing the quality of generated samples. Existing models deal with very complex data modalities such as images. For image generation a UNET architecture with residual connections was proposed. Since our target dataset is relatively less complex than images, we propose a simple encoder-decoder architecture [25] instead of a UNET for reverse denoising process.

3.2 Qualitative evaluation of generated artificial data

The quality assessment of synthetic data involves evaluating measures like inception score, Frechet inception distance, average log-likelihood, Parzen window estimates, and visual fidelity. However, these metrics primarily cater to image data, making it unclear which measure is optimal for other data modalities. Theis et al. [26] suggested that evaluation for generative models should align with the intended application. Thus, we evaluate diffusion models for circuit data by directly comparing them to the source of our data, which is the simulator. We extract specific features from the synthetic dataset, designating them as input features(for example supply voltage, temperature, channel length, transistor width, load capacitance etc.), while the rest are deemed output features(propagation delays). Subsequently, we feed the input features into the simulator and compare the resulting output feature values with the generated output feature values. Our evaluation metric in this study is the mean absolute percentage error. Refer to Figure 2 for a visual representation of the comprehensive evaluation procedure for the generated data. The diffusion model is trained using just 500 real data samples. Subsequently, artificial samples are generated and used for performance evaluation. Training continues through epochs until desired performance is reached, with hyperparameter adjustment for subpar results.

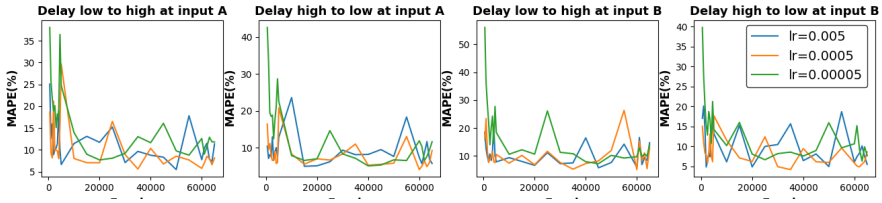

Figure 1: Performance of model with different learning rates w.r.t. HSPICE for delay in AND gate dataset. It can be seen that learning rate of 0.0005 ensures low percentage errors across all features.

4 Experimental setup and model architecture

We use `Python-3.8.16` and Google Colab for the training of Diffusion Denoising Probabilistic models. Moreover, our implementation uses `Keras-2.9.0` and `Tensorflow-2.9.2`. As discussed in 3.1, a diffusion model is devised for each dataset, encompassing forward and reverse processes for circuit data. Complete architectural details can be found in appendix (Section 7.2).

5 Results

As discussed in 3.2, the output feature values from the model and the simulator are compared to get the right idea of performance i.e MAPE is calculated to evaluate how closely the model follows the simulator. We start the hyper-parameter search by finding the optimal number of layers. Table 1 depicts the model's performance across varying hidden layer counts. A five-layer architecture emerges as the best choice for the NOT gate dataset featuring 17 attributes. This architecture also holds well for datasets up to 19 attributes. It was observed that for datasets featuring 21 attributes, a six-layered architecture boosts the learning capacity effectively. Our subsequent exploration involves different learning rates. Figure 1 indicates that a learning rate near 0.0005 ensures consistently low percentage errors across all features. The Table 3 provides the mean absolute percentage errors (MAPE) attained for all datasets, it can be seen that low MAPE is obtained for the various datasets. Additionally, Figure 2 showcases that density distribution of generated and original data for the NOT dataset exhibit a high degree of proximity. Furthermore Table 4 shows a significant improvement in a gradient-boosting regression (GBR) model using artificial data, to predict CMOS NOT gate delays. Thus, validating the proposed approach's efficacy in predicting parameters that concern digital circuit design. For brevity, the data generation with other generative models such as GANs or VAEs were not as effective, and hence not shown in results.

No. of layers	E(%)
6 hidden layers	12.5
5 hidden layers	**3.51**
4 hidden layers	10.5

Table 1: Comparison of performance of models with different layers across all features.

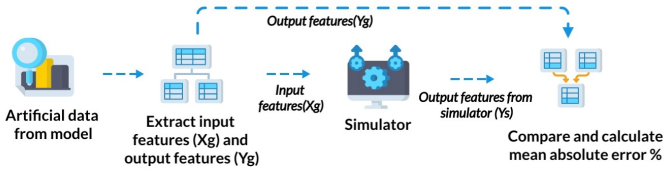

Table 2: Evaluation process for artificially generated data.

6 Conclusion

Achieving model accuracy hinges on high-quality training data, yet obtaining ample data for electronic circuits can be expensive or practically difficult to obtain. Our study proposes a variant of diffusion model for generating a) synthetic data, b) validation method, and c) subsequently predicting delay estimations in 22nm CMOS technology-based digital VLSI circuits. More specifically, the training data is obtained by various simulations in the HSPICE with 22nm CMOS technology nodes. The technique's efficacy is proven through extensive experiments on twelve essential digital circuit designs, showcasing notably low mean absolute percentage errors with respect to HSPICE circuit simulator. It is also observed that artificial data distribution closely resembles the original data distribution. Improvement in a GBR's performance using the proposed augmented data is also demonstrated.

Table 3: Percentage error obtained for different digital circuit datasets used in this work. Here, the error is calculated with respect to the HSPICE. Here A=delay lh node a, B=delay hl node a, C=delay lh node b, D=delay hl node b, E=delay lh node c, F=delay hl node c.

Delay dataset	Mean Absolute Percentage Error					
	A	B	C	D	E	F
NOT gate	3.9	3.12	-	-	-	-
Two input NAND gate	4.41	6.3	4.54	7.52	-	-
Two input AND gate	5.76	4.12	5.13	5.84	-	-
Two input NOR gate	5.14	2.52	3.92	6.05	-	-
Two input OR gate	3.77	3.5	5.57	4.42	-	-
Two input XOR gate	0.34	3.44	4.04	2.94	-	-
Three input AND-OR circuit	7.96	4.83	4.74	8.33	4.3	8.55
FULL ADDER	2.85	2.85	3.62	5.93	2.15	4.42
2:1 MULTIPLEXER	3.81	3.27	2.91	5.53	3.68	4.03
Three input NAND gate	7.73	4.66	6.15	5.404	7.37	3.26
Three input AND gate	4.04	4.64	5.33	2.92	3.91	3.02
Three input NOR gate	4.57	5.007	5.11	6.46	3.74	5.13

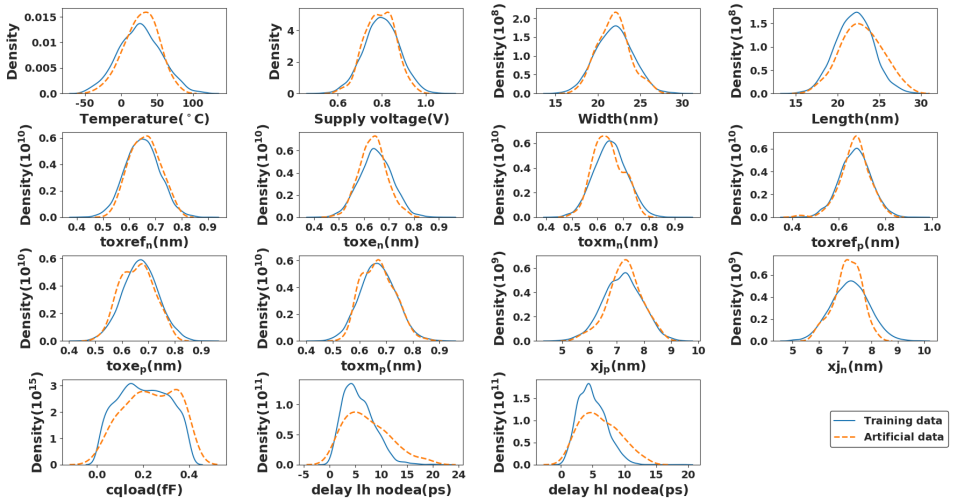

Figure 2: Distribution plots of artificially generated data and original data for delay dataset from NOT gate showing close resemblance between two.

Table 4: Performance of a predictive gradient boosting regression model with and without artificial data. (A higher R2 score denoted by ↑ and a lower MSE, RMSE, MAE and MAPE denoted by ↓ are preferred.)

Metric	Real data		Real + Artificial data		Improvement (%)	
	delay lh	delay hl	delay lh	delay hl	delay lh	delay hl
R2 score	0.93	0.95	0.976	0.973	4.95 ↑	2.42 ↑
MSE	4.58×10^{-25}	2.22×10^{-25}	2.25×10^{-25}	1.38×10^{-25}	50.87↓	37.84↓
RMSE	6.77×10^{-13}	4.71×10^{-13}	4.75×10^{-13}	3.72×10^{-13}	29.84↓	21.02 ↓
MAE	5.16×10^{-13}	3.32×10^{-13}	3.44×10^{-13}	2.52×10^{-13}	33.33↓	24.10↓
MAPE	0.106	0.103	0.099	0.083	6.60↓	19.42↓

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems* (Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, eds.), vol. 27, Curran Associates, Inc., 2014.

[2] S. K. Danisetty, S. R. Mylaram, and P. Kumar, "Adaptive consensus optimization method for gans," in *2023 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2023.

[3] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, pp. 307–392, 2019.

[4] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," 2020.

[5] D. Amuru, A. Zahra, H. V. Vudumula, P. K. Cherupally, S. R. Gurram, A. Ahmad, and Z. Abbas, "Ai/ml algorithms and applications in vlsi design and technology," *Integration*, vol. 93, p. 102048, 2023.

[6] A. Jafari, S. Sadri, and M. Zekri, "Design optimization of analog integrated circuits by using artificial neural networks," in *2010 International Conference of Soft Computing and Pattern Recognition*, pp. 385–388, 2010.

[7] Y. Li, Y. Wang, Y. Li, R. Zhou, and Z. Lin, "An artificial neural network assisted optimization system for analog design space exploration," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 10, pp. 2640–2653, 2020.

[8] S. Devi, G. Tilwankar, and R. Zele, "Automated design of analog circuits using machine learning techniques," in *2021 25th International Symposium on VLSI Design and Test (VDAT)*, pp. 1–6, 2021.

[9] E. Afacan, N. Lourenço, R. Martins, and G. Dündar, "Review: Machine learning techniques in analog/rf integrated circuit design, synthesis, layout, and test," *Integration*, vol. 77, pp. 113–130, 2021.

[10] N. Kahraman and T. Yildirim, "Technology independent circuit sizing for fundamental analog circuits using artificial neural networks," in *2008 Ph.D. Research in Microelectronics and Electronics*, pp. 1–4, 2008.

[11] K. Agarwal, A. Jain, D. Amuru, and Z. Abbas, "Fast and efficient resnn and genetic optimization for pvt aware performance enhancement in digital circuits," in *2022 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, pp. 1–4, 2022.

[12] D. Amuru, M. S. Ahmed, and Z. Abbas, "An efficient gradient boosting approach for pvt aware estimation of leakage power and propagation delay in cmos/finfet digital cells," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2020.

[13] A. Munappy, J. Bosch, H. H. Olsson, A. Arpteg, and B. Brinne, "Data management challenges for deep learning," in *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 140–147, IEEE, 2019.

[14] D. Amuru, A. Zahra, and Z. Abbas, "Statistical variation aware leakage and total power estimation of 16 nm vlsi digital circuits based on regression models," in *VLSI Design and Test* (A. Sengupta, S. Dasgupta, V. Singh, R. Sharma, and S. Kumar Vishvakarma, eds.), (Singapore), pp. 565–578, Springer Singapore, 2019.

[15] S. C. Wong, A. Gatt, V. Stamatescu, and M. D. McDonnell, "Understanding data augmentation for classification: when to warp?," in *2016 international conference on digital image computing: techniques and applications (DICTA)*, pp. 1–6, IEEE, 2016.

[16] A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," in *2018 international interdisciplinary PhD workshop (IIPhDW)*, pp. 117–122, IEEE, 2018.

[17] M. Asif, O. Nazeer, N. Javaid, E. H. Alkhammash, and M. Hadjouni, "Data augmentation using biwgan, feature extraction and classification by hybrid 2dcnn and bilstm to detect non-technical losses in smart grids," *IEEE Access*, vol. 10, pp. 27467–27483, 2022.

[18] W. Tan and H. Guo, "Data augmentation and cnn classification for automatic covid-19 diagnosis from ct-scan images on small dataset," *arXiv*, 2021.

[19] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, pp. 1–48, 2019.

[20] A. Kortylewski, A. Schneider, T. Gerig, B. Egger, A. Morel-Forster, and T. Vetter, "Training deep face recognition systems with synthetic data," *arXiv preprint arXiv:1802.05891*, 2018.

[21] T. Chalongvorachai and K. Woraratpanya, "3dvae-ersg: 3d variational autoencoder for extremely rare signal generation," in *2021 13th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pp. 177–182, 2021.

[22] X. Zhang and J. Zhou, "A heavy-tailed distribution data generation method based on generative adversarial network," in *2021 IEEE 10th Data Driven Control and Learning Systems Conference (DDCLS)*, pp. 535–540, 2021.

[23] I. Abbasnejad, S. Sridharan, D. Nguyen, S. Denman, C. Fookes, and S. Lucey, "Using synthetic data to improve facial expression analysis with 3d convolutional networks," in *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 1609–1618, 2017.

[24] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," 2021.

[25] D. Bank, N. Koenigstein, and R. Giryes, "Autoencoders," 2021.

[26] L. Theis, A. v. d. Oord, and M. Bethge, "A note on the evaluation of generative models," 2015.

[27] "Hspice circuit simulator." `https://cseweb.ucsd.edu/classes/wi10/cse241a/assign/hspice_sa.pdf`.

7 Appendix

7.1 The dataset

We gathered extensive datasets covering design, process, and performance parameters for twelve core digital cells, see Table 5. These parameters are chosen to enable subsequent performance assessment using a predictive ML model, validating the utility of dataset. For training, we employ simulated data from Electronic Design Automation (EDA) tool HSPICE [27]. Training data for Digital cells (Table 5) comprises vectors of random values, drawn from Gaussian distributions of process parameters. We account for ±10% variations at 3σ in CMOS standard cells at 22nm High-K MGK via Predictive Technology Models (PTM). Twelve process parameters (PMOS and NMOS) are included. In addition to statistical distributions, temperature samples spanning $-55°C$ to $125°C$ and supply voltage deviations of ±10% from the nominal (0.8V) are integrated. Load capacitance varies similarly to process parameters. We perform propagation delay estimations via HSPICE Monte-Carlo simulations to obtain training data, encompassing PVT (Process, Voltage, Temperature) variations.

Table 5: List of digital circuit datasets used in this work.(**Input parameters for evaluation:** Supply voltage, Temperature, Channel length, Transistor width, Physical and electrical equivalent of oxide thickness, Nominal gate oxide thickness, Source/Drain junction depth, and Channel doping concentration, Load capacitance; **Output parameters for evaluation:** Propagation Delays)

Dataset	Parameters	Dataset	Parameters
NOT gate delay	17	Three input AND-OR circuit delay	21
Two input NAND gate delay	19	Full adder delay	21
Two input AND gate delay	19	2:1 Multiplexer delay	21
Two input NOR gate delay	19	Three input NAND gate delay	21
Two input OR gate delay	19	Three input AND gate delay	21
Two input XOR gate delay	19	Three input NOR gate delay	21

7.2 Architecture Details

A diffusion model is devised for each dataset, encompassing forward and reverse processes for circuit data.

Forward process: We set T = 100 for all experiments. We adopt a variance of β_t, transitioning linearly from 0.001 to 0.02, following the approach by Ho et al. [4]. Thus the noised inputs are created to feed the network representing reverse process.

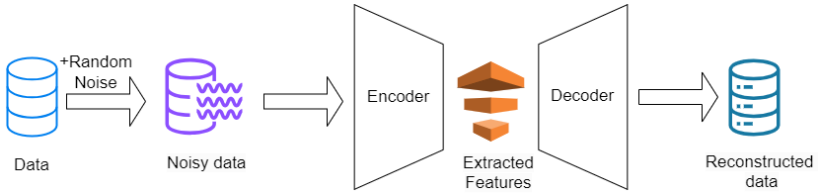

Figure 3: Encoder-Decoder architecture of the network performing reverse process

Reverse process: To realize the reverse process a neural network is used. The network follows an encoder-decoder structure [25] with several layers that progressively reduce and then increase the number of units, culminating in the output layer that generates the denoised data. Since our target dataset is relatively less complex than images, we propose a simple encoder-decoder architecture instead of a UNET for reverse denoising process. The use of batch normalization and leaky ReLU activation functions in the hidden layers aims to improve the network's stability and facilitate better learning and convergence during training. Figure 3 shows the reverse process architecture.

Training data undergoes noise addition through the forward process, gradually transforming to pure noise, and the reverse process learns to de-noise and predict original distribution. After training the diffusion model, the synthetic data is then generated by first sampling a pure random noise then using trained reverse diffusion model to generate the desired sample.