Modeling of Adaptive Receiver Performance Using Generative Adversarial Networks

Priyank Kashyap*, Yongjin Choi†,Sumon Dey†, Dror Baron*, Chau-Wai Wong*,
Tianfu Wu*, Chris Cheng†, Paul D. Franzon*
* Electrical and Computer Engineering, North Carolina State University
†Hewlett Packard Enterprise

2022 IEEE 72nd Electronic Components and Technology Conference (ECTC) | 978-1-6654-7943-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/ECTC51906.2022.00307

Abstract—**As the development of IBIS Algorithmic Modeling Interface (IBIS-AMI) models gets complex and requires time-consuming simulations, a data-driven and domain-independent approach can have tremendous value. This paper presents a data-driven approach to modeling a high-speed serializer/deserializer (SerDes) receiver through generative adversarial networks (GANs). In this work, the modeling considers multiple channels, random bitstreams, and varying decision feedback equalizer (DFE) tap values to predict an accurate bit error rate (BER) contour plot. We employ a discriminator structure that improves the training to generate a contour plot that makes it difficult to distinguish the ground truth. The generated plots' bathtub curves strongly correlate to the ground truth bathtub curves and have a root-mean-squared error (RMSE) of 0.014, indicating a good fit.**

Index Terms—**SerDes, receiver, behavior modeling, adaptive, generative, GAN, DFE, IBIS-AMI**

I. Introduction

With increasing clock frequencies, signal integrity and power integrity play an essential role in deciding whether a device will function as desired. Engineers must consider problems that may arise due to signal integrity and power integrity from the device's conception to production. To do so, engineers utilize computationally expensive and time-consuming simulations at each step of the design phase. Often, designs consist of different intellectual property (IP) blocks from different vendors. The vendors need to obfuscate and conceal these design blocks to protect their IP; thus, the industry utilizes IBIS-AMI models to share relevant transmitter and receiver design properties and compatibility with different electronic design automation (EDA) tools.

As the frequency of operation goes up, the design complexity of the transmitter and receiver circuitry increases. Transmitters consist of finite impulse response (FIR) filters with multiple taps, and receivers consist of continuous-time linear equalizer (CTLE) and DFE filters to alleviate the signal integrity issues arising from intersymbol interference (ISI). However, the development of IBIS-AMI models requires multiple time-consuming design iterations to develop an accurate model of their IP block as it requires detailed circuit-level simulations. Hence, it is crucial to find alternative strategies to improve simulation times and effectively assist engineers.

A rich swath of prior work attempts to solve the earlier issues of iterative long simulation times through data-driven modeling. All of the preceding work belongs to one of two major categories: ones where different methods predict characteristics such as eye height (EH) and eye width (EW), and others where the proposed method predicts the underlying time series or eye diagram. Machine learning (ML) algorithms such as support vector machines can predict the system performance metrics characteristics [1]. Artificial neural networks (ANNs) show how system parameters can predict specific eye characteristics [2]. However, in works that predict eye characteristics, specifically EH and EW, there is a limitation in the information an engineer obtains as the model only conveys information about 4 different eye-opening points. Additionally, these approaches use domain knowledge to choose system characteristics, thus reducing the problem's scope.

Predicting the underlying time series is the alternative to predicting EH/EW from the proposed approaches. Non-linear system identification (SID) models can effectively model a receiver's behavior and predict the output time series [3], [4]. Li et al. show how a non-linear SID model can predict the receiver behavior by recovering the underlying time series [3]. They expand on this by utilizing SID models for different channel conditions but a fixed CTLE configuration and go a step further by predicting the eye diagram of the system [4]. Each receiver configuration requires a unique model in both cases, making the development process cumbersome.

Further, [5] and [6] model the impact of variable tap settings to an effect. Li et al. use a deep neural network (DNN) to learn a receiver's CTLE adaptation process and evaluate the results using eye characteristics [5]. Whereas Nguyen and Schutt-Aine recover a pulse waveform using a recurrent neural network (RNN) that demonstrates the impact of the DFE tap setting. The RNN learns the initial state through a DNN that takes the tap settings and learns a latent space [6].

Generative methods are gaining traction in different EDA flows for image-based tasks through GANs. LithoGAN is an example where the authors use GANs to go from an input mask to an output resist pattern [7]. WellGAN is another example of the design flow where a GAN converts the input layout pattern to one that has the well regions defined [8].

This paper proposes a generative approach to model a SerDes receiver accurately. We utilize a data-driven approach to train a GAN for a receiver with various bitstreams, channel conditions, and different DFE configurations to generate a BER contour plot. The bathtub curves of the resultant plots are highly correlated to the actual bathtub curves of the system and

2377-5726/22/$31.00 ©2022 IEEE
DOI 10.1109/ECTC51906.2022.00307

show an RMSE of 0.014. Unlike prior work that uses GANs to recover an eye diagram, this work can handle different DFE configurations through an embedding network [9].

The rest of the paper is organized as follows. Section II consists the necessary background and prior work. Section III describes the proposed method and the metrics utilized for evaluation. Section IV discusses the problem that was analyzed, and how the dataset was generated. Section V shows the experimental results. Section VI concludes the article and presents a brief overview of future work.

II. Background

A. Bit-Error Rate (BER) Contour Plots

In today's high-speed links, the device's success depends on its BER, the number of bits that arrive at the receiver in error per unit time. Eye diagrams tell the engineer about the performance of a communication link. An eye diagram contains all possible transitions overlaid on top of each other; thus, it provides insight into which transitions may present an issue or how jitter in the link plays a role in determining the eye-opening. Though eye diagrams are sufficient, it has become increasingly challenging to measure BER of 1×10^{-15} using a sampling oscilloscope. So many have started to use bathtub curves and their 2D representations, BER contour plots, as a measure of performance [10].

Unlike the case of the eye diagram, a bit error rate tester (BERT) creates the bathtub curve by generating and passing data through a device under test [10]. The BERT compares the transmitted and received data for errors across the unit interval. BER contour plots are a 2D extension of the bathtub curves and provide us a view of the eye closure as the BER increases and reveal cases that an eye diagram might not recover as a sampling scope relies on a sparse sampling strategy, making it difficult to catch a bit error. Fig. 1 shows the BER contour plot and its corresponding bathtub curve.

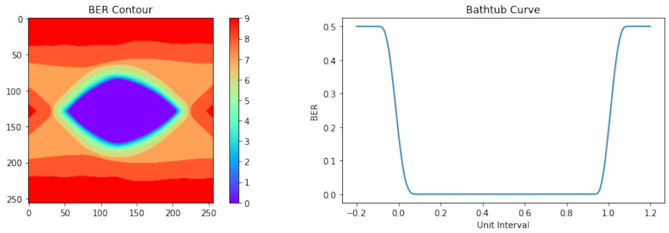


Fig. 1: BER contour plot and corresponding bathtub curve.

B. Generative Adversarial Network (GAN)

Generative networks have found tremendous use for tasks such as synthetic dataset generation, super-resolution, and image translation, to name a few. The GAN itself consists of two modules: the generator, G, which generates samples similar to the ground truth, and a discriminator, D, whose job is to determine if a sample presented to it is the ground truth or not. Mathematically, the generator attempts to learn a mapping from some noise vector, z, to the desired image, y, given parameters θ_g of the neural network, i.e., $G : z \rightarrow y$. The discriminator's function $D(y|\theta_d)$ returns the probability that the sample y presented to it is from the dataset or the generator. Together the generator and discriminator play a min-max game where they try to outperform each other until they reach a state of equilibrium [11]. To that extent, the loss function is

$$\begin{aligned} L_{\text{GAN}}(G, D) = &\mathbb{E}_y \left[\log D(y)\right] \\ &+ \mathbb{E}_{x,z} \left[\log(1 - D(G(z|x)))\right]. \end{aligned} \tag{1}$$

Unlike regular GANs, which are a form of unsupervised learning, the conditional GAN (cGAN) uses labeled data to enable the model to extend the latent space z and thus generate and discriminate images better. The generator learns a mapping given some input x, and noise vector z, to generate an output y. The discriminator trains to discern whether y is from the dataset or the generator for a given input x [12]. To that extent, the loss function in Eq. 1 changes to provide the loss function of the cGAN

$$\begin{aligned} L_{\text{cGAN}}(G, D) = &\mathbb{E}_{x,y} \left[\log D(y|x)\right] \\ &+ \mathbb{E}_{x,z} \left[\log(1 - D(G(x, z)|x))\right]. \end{aligned} \tag{2}$$

Additionally, prior work has shown that adding an ℓ_1 loss term to the generator training improves the quality of the images generated by the model [12]. To control the impact of the ℓ_1 loss, we multiply it by a factor λ. Thus the GAN's final objective is as follows:

$$L = L_{\text{cGAN}}(G, D) + \lambda \ell_1. \tag{3}$$

C. Gramian Angular Field (GAF)

SerDes modeling is inherently a time series problem; thus, to utilize GANs, which work well with images, we convert the time-series waveforms to a Gramian angular field (GAF). The GAF consists of the angular sums across different time steps and exploits the temporal relationships enabling it to capture channel effects such as ISI. Unlike other conversions, such as a Markov transition field (MTF) and recurrence plots (RPs), which keep the transition dynamics and signal trajectories over an area, a GAF does not require additional tuning. By converting the time series to images, a DNN can utilize previously unavailable features by converting the time series to an image.

To convert a waveform to a GAF, we scale the original data between $[-1, 1]$. Then the scaled time series is expressed in the polar coordinate system by taking the arccosine of the value of the scaled time series at each time step. After converting to the polar coordinate system, we make an $m \times m$ Gramian matrix, where m is the length of the time sequence. The (i, j)th entry of the Gramian matrix is the trigonometric sum or difference between the ith and jth step to form the Gramian angular sum field (GASF) or Gramian angular difference field (GADF), respectively. The GASF is defined as follows:

$$G = \begin{bmatrix} \cos(\phi_1 + \phi_1) & \dots & \cos(\phi_1 + \phi_n) \\ \cos(\phi_2 + \phi_1) & \dots & \cos(\phi_2 + \phi_n) \\ \vdots & \ddots & \vdots \\ \cos(\phi_n + \phi_1) & \dots & \cos(\phi_n + \phi_n) \end{bmatrix} \tag{4}$$

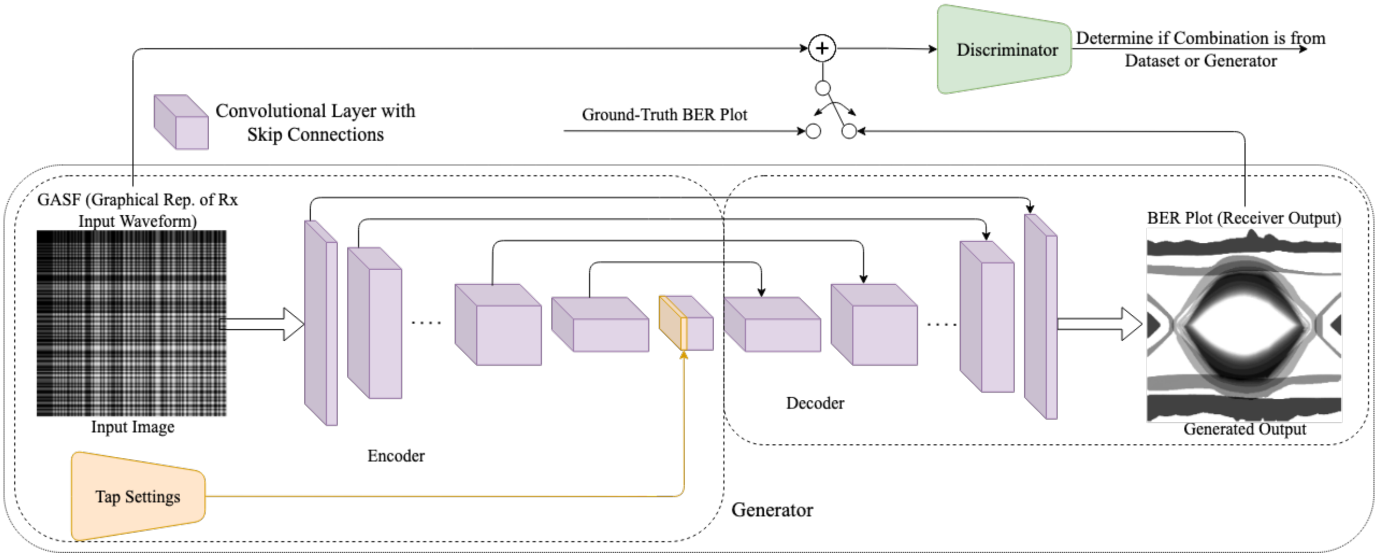

Discriminator
Determine if Combination is from Dataset or Generator
Convolutional Layer with Skip Connections
Ground-Truth BER Plot
GASF (Graphical Rep. of Rx Input Waveform)
Input Image
Encoder
Tap Settings
Decoder
BER Plot (Receiver Output)
Generated Output
Generator

Fig. 2: GAN architecture used for training all implementations. The generator encoder and decoder with skip connections to forward information for the reconstruction of the eye diagram. The discriminator takes in the concatenation of the GASF and either the synthetic or ground-truth eye diagram.

where ϕ_n is the nth time step encoded in the polar coordinate system.

III. Methodology

Like the Pix2Pix network, we use a U-Net-based network as a starting point for the generator [12]. The generator uses the waveform at the receiver's input as a GASF and the DFE tap settings to predict the appropriate BER contour plot. The network itself consists of three sub-networks, a time encoder network, a tap encoder network, and a decoder network. The time encoder network learns a latent representation from the GASF input and consists of convolution layers that downsample the input. The tap encoder is a fully connected network that learns a latent space from the tap configurations. The decoder network takes in the combined latent representation from the two encoder models and attempts to recreate the correct BER contour plot. The decoder network has ConvTranspose layers to reconstruct the higher resolution BER plots and has skip connections from the time encoder network such that there is no information loss in the reconstruction process. The number of filters used in the time encoder and decoder network aligns such that the dimensions of the output image at each layer match appropriately.

Unlike prior work, where a discriminator predicts patches to determine whether the sample presented is from the dataset or the generator, we employ a U-Net-based discriminator with two levels of prediction. By having two levels of prediction, the discriminator can focus on both global features and local details [13]. The global prediction from the network comes from the bottleneck in the U-Net, and it predicts whether the combination of the GASF, tap settings and BER contour plot are from the dataset or the generator. The local prediction determines if each pixel value at each location is from the dataset or the generator and, as such, has a size equal to the output image size. However, unlike Schonfeld et al., we use regular GAN loss functions instead of hinge loss as the generated images are similar to the ground truth [13].

Fig. 2 shows the GAN model with the complete architecture of the generator shown. As shown from the figure, the tap encoder network's learned representation adds to the time encoder's learned latent space, and together, the decoder learns to transform the inputs to a BER contour plot. The generator uses this local prediction and the global prediction from the discriminator as feedback to modify its predictions for the next iteration.

IV. Dataset Creation

We created a parametric differential stripline model where the length of line and spacing between signal traces are tuneable parameters to create a practical problem. We uniformly sample the design space, obtain the s-parameters of 10 different transmission lines, and then use them as a communication link to simulate a SerDes running at $32\,\mathrm{Gb/s}$ with a 10% rise and fall time. Before the data collection, we determine the optimal tap settings for the DFE that yields the maximum eye-opening for each transmission line through Ansys Electronic Desktop. We use Gaussian sampling around the optimal taps with a standard deviation equivalent to half of the optimal tap weight to determine which DFE configurations to include in the dataset. We then capture the waveforms going into the receiver and the BER contour plots generated from the simulation.

The original waveform consists of 32000-time steps, sampled at every $0.5\,\mathrm{ps}$. We downsample the waveform by determining the appropriate Nyquist frequency, which was

10 ps. Thus, the preprocessing flow downsamples the original waveform to 3200-time steps before converting it to a GAF. The downsampling step is crucial as the GAF is an $m \times m$ matrix, making the GAF generation a significant bottleneck because its size increases exponentially with an increase in the time series length.

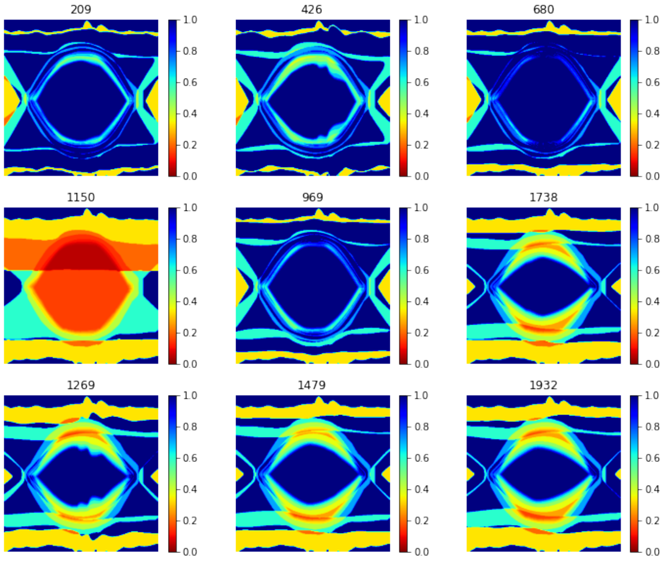
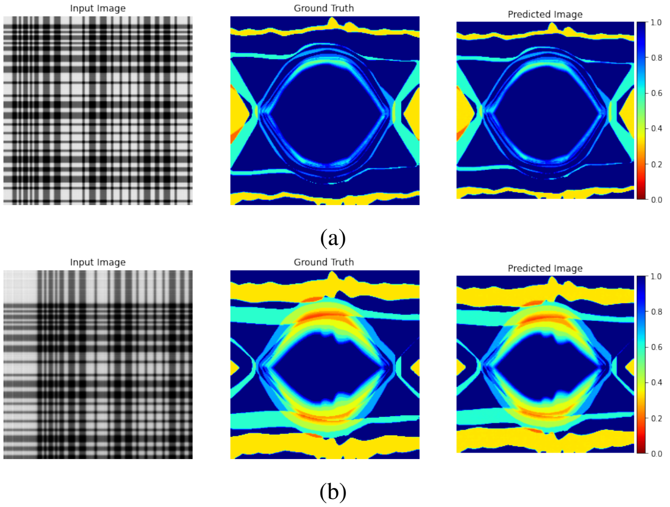


Fig. 3: Randomly selected BER plots from the dataset which showcases the wide range of solutions possible. The color bar next to each of the BER plots gives the pixel intensities for the 256×256 image.

After we downsample the waveforms, we transform them to a GASF, one of the GAN inputs. The BER contour plots captured by Ansys default to a resolution of 700×500, and the preprocessing flow digitizes and resizes them to a 256×256 image used by the GAN as a target. Fig. 3 shows randomly selected BER contour plots targeted at BER-6 with a 256×256 resolution from the dataset and illustrates the wide range of possible outputs the GAN must learn to generate. Lastly, the preprocessing flow normalizes the DFE tap settings $[0, 1]$ to help the model's convergence.

V. Experimental Results

To validate the generator's results, we use a separate neural network, the metric network, which produces a bathtub curve for a given BER contour plot. We employ this approach as it is not feasible to put the generated results back into a simulator to determine if they are significant. Additionally, metrics such as Frechet inception distance score, which compares image quality between a dataset and GAN generated images, do not reveal much about the system's performance. We train the metric network on the ground truth BER contour plots and bathtub curves and then use it to obtain the bathtub curves for the generated BER contour plots. The metric network constitutes multiple convolutional layers and then has a fully connected layer with the number of output neurons corresponding to the number of steps in the bathtub curve, in our case 700.



(a)

(b)

Fig. 4: Results using the cGAN with the left indicating the GASF input, middle representing the ground-truth and right representing the generated BER contour plot for two cases.

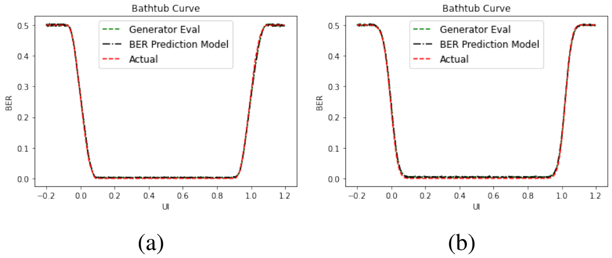


Fig. 5: Bathtub curve prediction on BER plot from ground truth, prediction and cGAN.

In Fig. 4, the first column contains the input waveform to the receiver, which we express as a GASF, the second column shows the ground-truth BER contour plot that we obtain from Ansys Electronic Desktop, and the third column is the generated BER contour plot. Both Fig. 4a and Fig. 4b are results based on the test set from our dataset. It is hard to determine whether any difference exists between the BER contour plots based on simple visual inspection. All the features are present in the correct locations, to the degree that we observe that transition between different BERs is correct.

We further the visual inspection by quantifying their underlying statistics rather than image similarity. As discussed before, we evaluate the generated results by using our trained metric network to predict the horizontal bathtub curves for the generated BER contour plot. Fig. 5 shows the ground truth bathtub curve, the metric model's prediction on the actual BER contour plots, and the metric model's prediction on the generated BER contour plots for the cases discussed previously. The generated bathtub curve and the ground truth curve show a high correlation. Additionally, the RMSE between the two

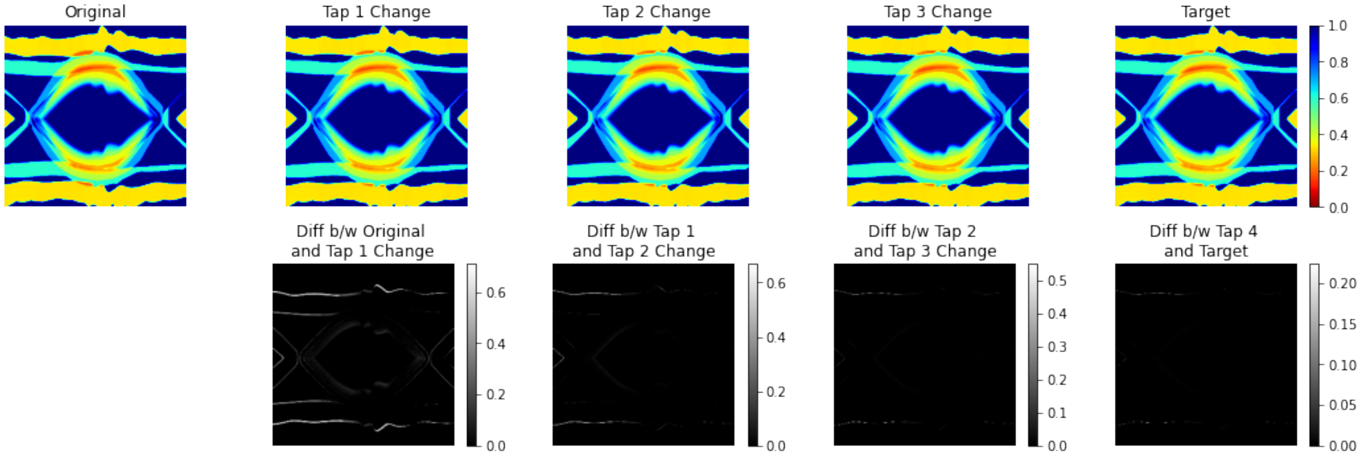


Fig. 6: Varying the tap configuration from left to right with one tap adjusted. The difference between the tuned tap configuration is shown on the second row.

curves over the entire testing dataset is 0.014, indicating a good fit for the generated plots.

An impressive ability of GAN is their ability to interpolate the conditioning values presented during training. We fix the channel and input bitstream while changing the tap configuration values for two cases in the test set to verify this. In Fig. 6, the original and the final tap configurations are seen on the left and right most of the top row. In between, we adjust one tap value as we move from left to right and determine its impact by comparing the difference of the two consecutive BER plots and highlighting regions of interest. The bottom row shows that the first tap produces the most significant change and impacts the opening region, whereas subsequent tap changes have minor changes across the region. Moreover, when comparing the difference between the initial and final BER plots for the ground truth and generated case, the difference occurs at identical locations.

We train all the models discussed here on an Nvidia 2080-Ti GPU. As with most ML algorithms, the training time varies based on the hyper-parameters. For the hyper-parameters, we found that training takes $18\,\mathrm{s}$ per epoch and, on the whole, takes $30\,\mathrm{min}$ to train. During inference, the model takes $192\,\mathrm{ms}$ ms to obtain a prediction from the GAN and the metric network, showing its performance advantage over other methods.

VI. Conclusion

This paper presents a data-driven approach to modeling a high-speed SerDes receiver through GANs. The approach converts the time series to an intermediate representation from which the GAN performs a domain translation task based on the different DFE tap settings. We demonstrate that the bathtub curves for the generated images and ground truths are correlated and have an RMSE of 0.014. Additionally, we show that the model can interpolate between unseen tap values while generating images similar to the ground truth. This modeling requires a one-time effort in terms of the data collection required but amortizes over the model's lifetime as such an effort takes $192\,\mathrm{ms}$ during inference.

The impact of nonlinearities introduced by the CTLE in the receiver can be explored in future work. Additionally, the modeling aspect can include the impact of crosstalk and different signaling modes. Another active area of research revolves around using learned metrics to quantify the results of GANs to make them suitable for a wide variety of tasks.

Acknowledgment

This research is supported in part by the NSF under Grant No. CNS 16-244770 (Center for Advanced Electronics through Machine Learning) and the industry members of the CAEML IUCRC.

References

[1] R. Trinchero and F. G. Canavero, "Modeling of eye diagram height in high-dpeed links via support vector machine," in *IEEE 22nd Workshop on Signal and Power Integrity (SPI)*, 2018, pp. 1–4.

[2] T. Lu, J. Sun, K. Wu, and Z. Yang, "High-speed channel modeling with machine learning methods for signal integrity analysis," *IEEE Transactions on Electromagnetic Compatibility*, vol. 60, no. 6, pp. 1957–1964, 2018.

[3] B. Li, P. Franzen, Y. Choi, and C. Cheng, "Receiver behavior modeling based on system identification," in *2018 IEEE 27th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS)*, 2018, pp. 299–301.

[4] B. Li, B. Jiao, M. Huang, R. Mayder, and P. Franzon, "Improved system identification modeling for high-speed receiver," in *IEEE 28th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS)*, 2019, pp. 1–3.

[5] B. Li, B. Jiao, C. .-H. Chou, R. Mayder, and P. Franzon, "CTLE adaptation using deep learning in high-speed serdes link," in *IEEE 70th Electronic Components and Technology Conference (ECTC)*, 2020, pp. 952–955.

[6] T. Nguyen and J. Schutt-Aine, "A tunable neural network based decision feed-back equalizer model for high-speed link simulation," in *IEEE 29th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS)*, 2020, pp. 1–3.

[7] W. Ye, M. B. Alawieh, Y. Lin, and D. Z. Pan, "LithoGAN: End-to-end lithography modeling with generative adversarial networks," in *56th ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 1–6.

[8] B. Xu, Y. Lin, X. Tang, S. Li, L. Shen, N. Sun, and D. Z. Pan, "WellGAN: Generative-adversarial-network-guided well generation for analog/mixed-signal circuit layout," in *56th ACM/IEEE Design Automation Conference (DAC)*, 2019, pp. 1–6.

[9] P. Kashyap, W. S. Pitts, D. Baron, C.-W. Wong, and T. W. P. D. Franzon, "High speed receiver modeling using generative adversarial networks," in *2021 IEEE 30th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS)*, 2021, pp. 1–3.

[10] *Bridging the gap between ber and eye diagrams*, May 2019. [Online]. Available: https://download.tek.com/document/65W_26019_0_Letter.pdf.

[11] 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. Q. Weinberger, Eds., vol. 27, Curran Associates, Inc., 2014. [Online]. Available: https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

[12] P. Isola, J. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5967–5976.

[13] E. Schonfeld, B. Schiele, and A. Khoreva, "A U-Net based discriminator for generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8207–8216.