Hewlett Packard Enterprise

REVOLUTIONIZE EDA WITH AI: UNLOCKING THE POWER OF GENERATIVE ADVERSARIAL NETWORKS IN HIGH-SPEED RECEIVER MODELING

Priyank Kashyap
Y. Choi, C. Cheng – Hewlett Packard Enterprises
C.W. Wong, D. Baron, T. Wu, P. Franzon – ECE Dept., North Carolina State University

18th April 2024

NC STATE UNIVERSITY

Outline

Generative AI (GenAI) has proliferated in different domains. Electronic design automation (EDA) is no exception! In this talk, we'll examine how two different models are finding use in different areas of EDA.

Motivation

- Current methods to simulate high-speed receivers are time-consuming
  - Multiple iterations that human designers to develop models that correlate to actual hardware
  - Can take months of a human designer's time
- Simulations are computationally expensive
  - Running bit-by-bit simulations take time
  - Macro-models can take upwards of 4-5 minutes as they perform transient and statistical analysis
- EDA tools might not support specific macro-model versions
  - Certain tools are not backward compatible and won't work with older macro-models
  - Need expensive licenses for EDA tools

High-Speed Link Signaling

- As channel speed goes higher, the channel's loss increases
- Simulation becomes more challenging due to the complexity of the circuit and reduced noise margin
- Requires accurate modeling of multiple parameters (eye heights and widths)

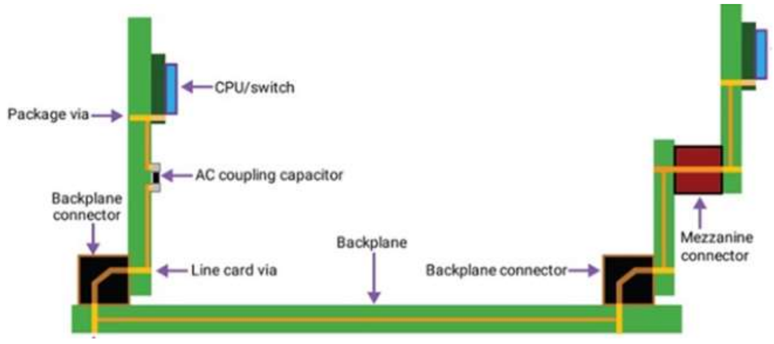


Fig. A complex backplane channel**

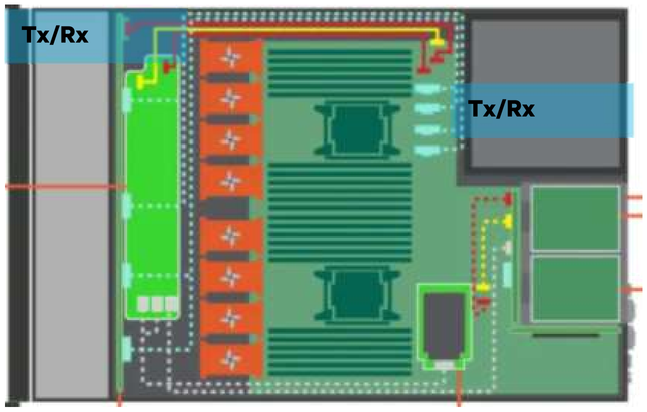


Fig. Inside the server

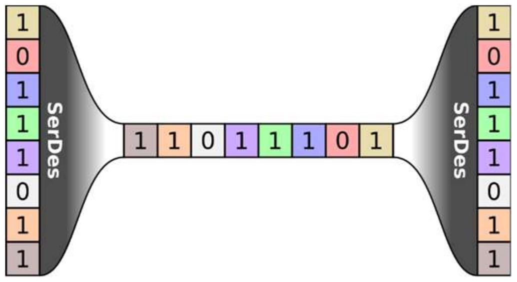


Fig. Parallel to Serial data conversion for transmission over a channel*

* https://en.wikipedia.org/wiki/SerDes
** https://www.amphenol-icc.com/connect/backplane-connectors-for-pcie-applications.html

Signal Integrity: NRZ

*https://www.synopsys.com/designware-ip/technical-bulletin/pam4-400g-ethernet-2019q3.html

- For NRZ one-bit symbol with 2 distinct amplitude levels
- Different ways to evaluate the performance of a link

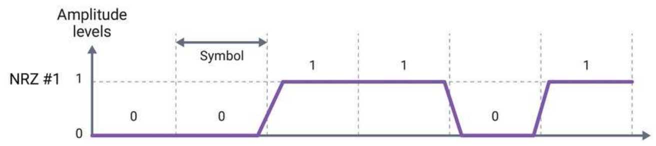
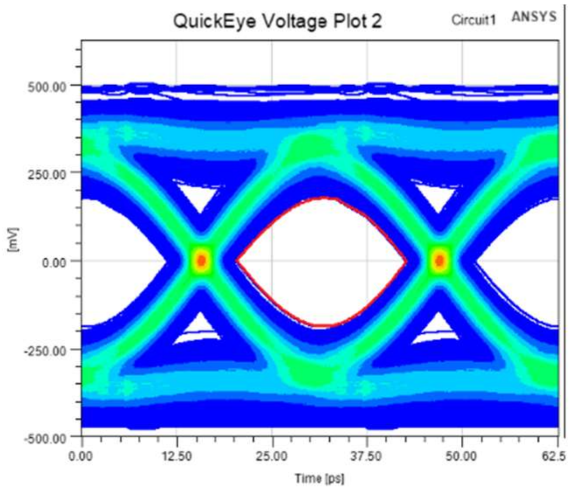
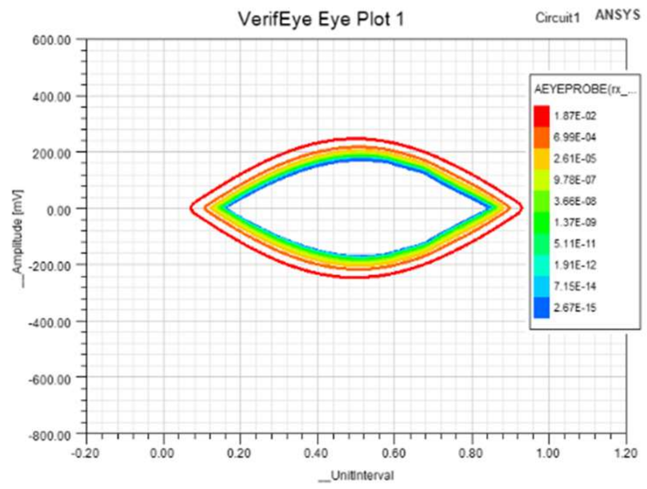


Fig. PAM-2 data transmission*





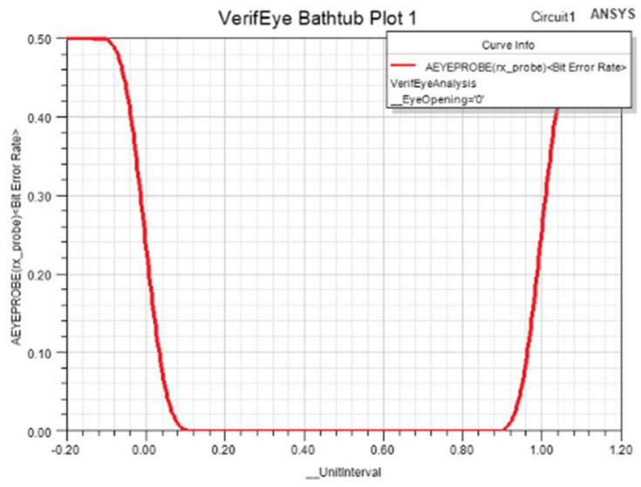


Fig. Eye diagram (left) with persistence, BER contour plot (center), and horizontal bathtub curve (right) for one set of measurements

Hewlett Packard Enterprise | NC STATE UNIVERSITY

Signal Integrity: Pulse Amplitude Modulation Level 4 (PAM-4)

- PAM4 signaling is an effective way to double the bandwidth while keeping the same Nyquist frequency at the expense of decreasing signal to noise ratio
- A two-bit symbol with 4 distinct amplitude levels
- Enables the use of existing channels at high bit rates without doubling the baud rate and increasing channel loss
- 12 different transitions possible (6 rise/fall)
  - Reduces the SNR by 9.5db
  - Susceptible to crosstalk and reflections

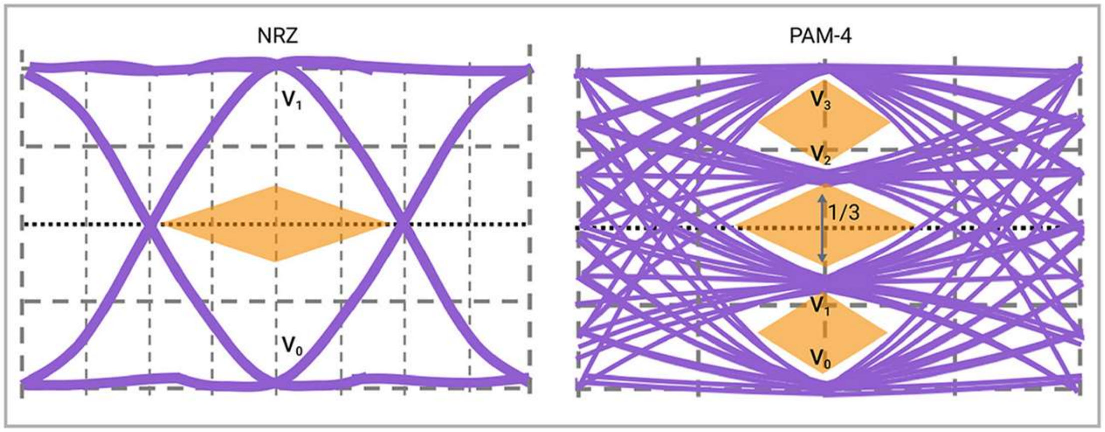


Fig. PAM-4 vs NRZ eye openings compared*

*https://www.synopsys.com/designware-ip/technical-bulletin/pam4-400g-ethernet-2019q3.html

Signal Integrity: Crosstalk

- Energy induced on a *victim* line by a switching signal on a neighboring *aggressor* line
- It is edge rate dependent and occurs due to mutual coupling between the lines
- Two kinds:
  - NEXT/ Backward: Amplitude depends on T_r of the aggressor and saturates; additive couple
  - FEXT/ Forward: Pulse with a duration of the T_r and amplitude dependent on length of trace with coupling; subtractive coupling

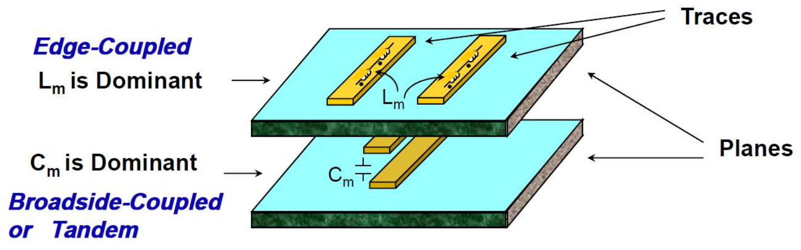


Fig. Crosstalk due to coupling edge or tandem coupling

High-Speed Receiver Equalization

Decision Feedback Equalizer (DFE)

- Slicer quantizes input as 0 or 1
- Feedback FIR filter directly subtracts the Intersymbol interference (ISI) from the incoming signal
- Can only remove post-cursor ISI
- Tap values are dependent on how much equalization is needed

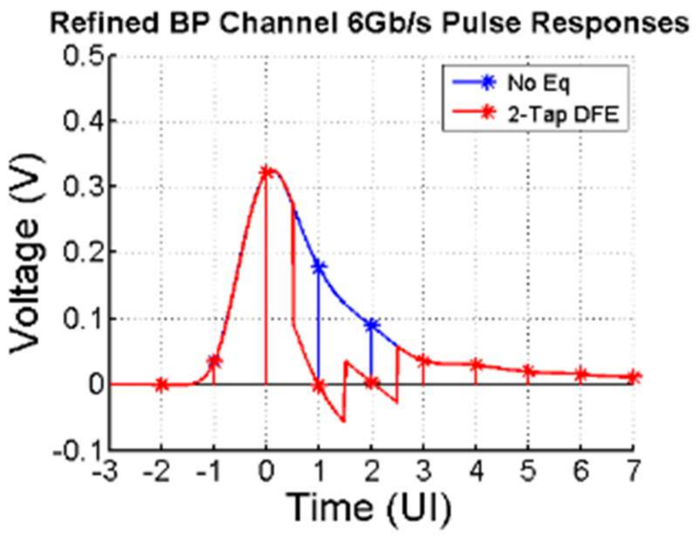


Fig. DFE removing post-cursor ISI

Hewlett Packard Enterprise | NC STATE UNIVERSITY

High-Speed Receiver Equalization

Continuous Time Linear Equalizer (CTLE)

- Can be implemented as passive or active
- Implement a high-pass filter to compensate for channel loss*
- Amplifies noise and crosstalk

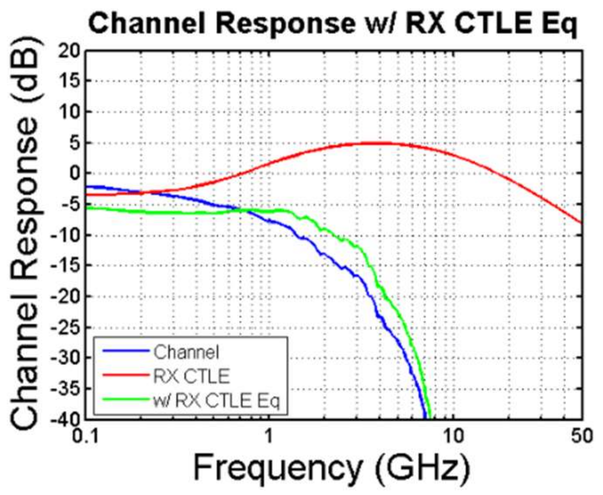


Fig. Frequency response curves of channel, CTLE, and Channel + CTLE

*W. T. Beyene, "The design of continuous-time linear equalizers using model order reduction techniques," *2008 IEEE-EPEP Electrical Performance of Electronic Packaging*, San Jose, CA, USA, 2008, pp. 187-190, doi: 10.1109/EPEP.2008.4675910.

What is a Neural Network?

Fully Connected Network

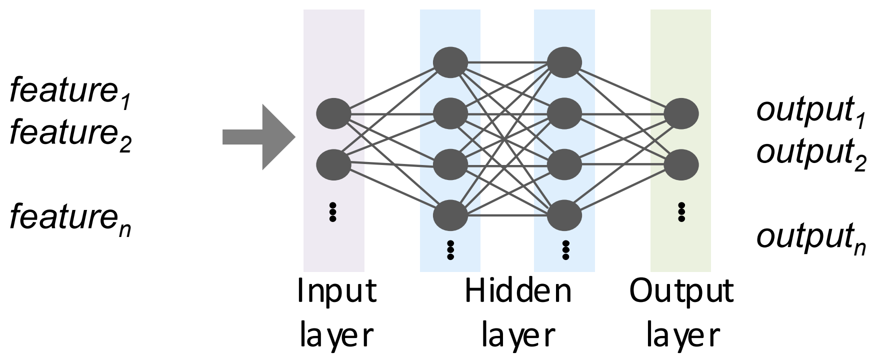


Fig. A fully connected network with hidden layers and multiple inputs

Convolutional Neural Network

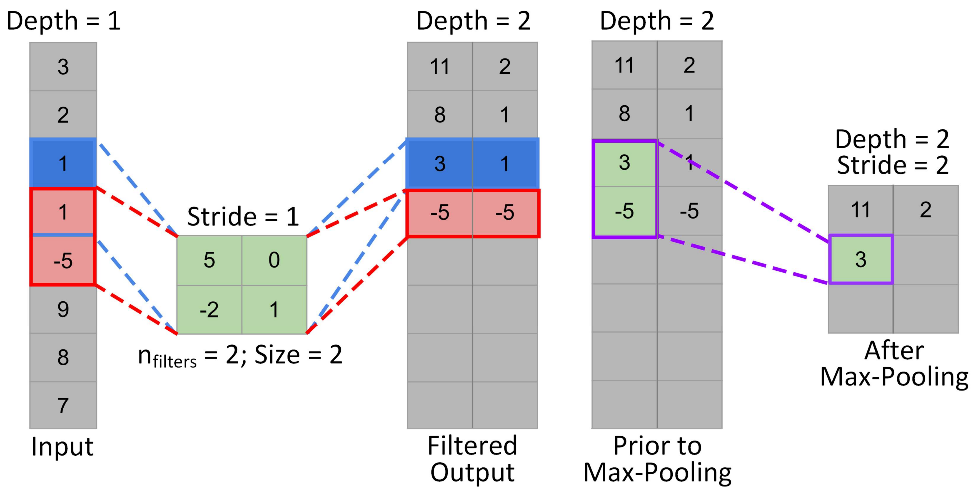


Fig. Convolutional and pooling layers of a CNN. The input passes 2 filters of size 2, after which the output is sub-sampled using maximum pooling with a stride of 2.

Hewlett Packard Enterprise | NC STATE UNIVERSITY

What are Generative Adversarial Networks?

- GANs consist of two modules that play a zero-sum game and try to outperform each other until they reach a state of equilibrium
  - Generator (G): Creates new examples from a learned latent distribution
  - Discriminator (D): Discerns whether an example provided to it comes from the generator or dataset
- Generates data with the same statistical behavior as the training data

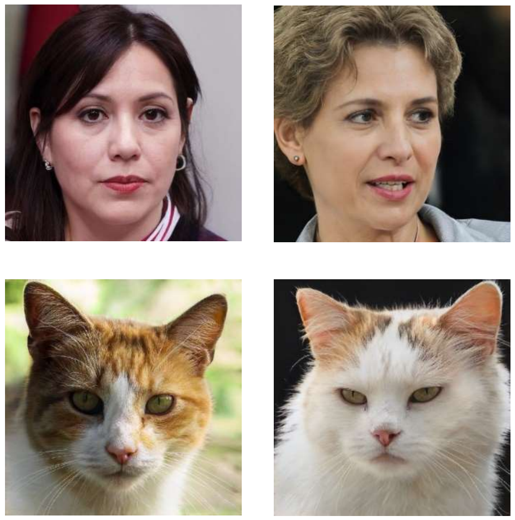


Fig. GAN generated outputs*

*https://thispersondoesnotexist.com/ & https://thiscatdoesnotexist.com/

What are Conditional Generative Adversarial Networks?

- cGANs consist of the same two modules, however,
  - The generator learns to output, y, from a given input, x, in addition to a random vector from a latent space
  - The discriminator discerns whether a sample provided to it comes from the generator or the dataset, given the same additional input
- The objective function for the cGAN is as follows:

$$L_{cGAN}(G,D) = \mathbf{E}_{\mathrm{x\,y}}[\log(\mathrm{D}(y|x))] + \mathbf{E}_{\mathrm{x}\,z}[\log(1 - D(G(z|x)|x)]$$

  - where, z is randomness introduced by dropout in our implementation

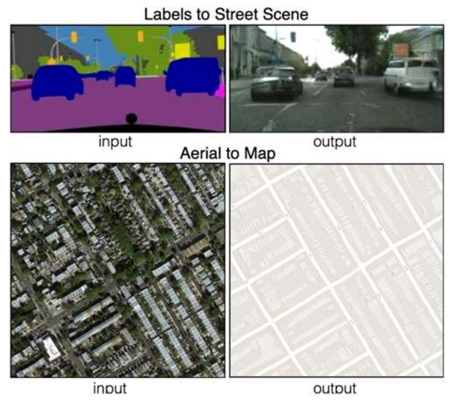


Fig. cGAN generated with pix2pix performing image to image translation**

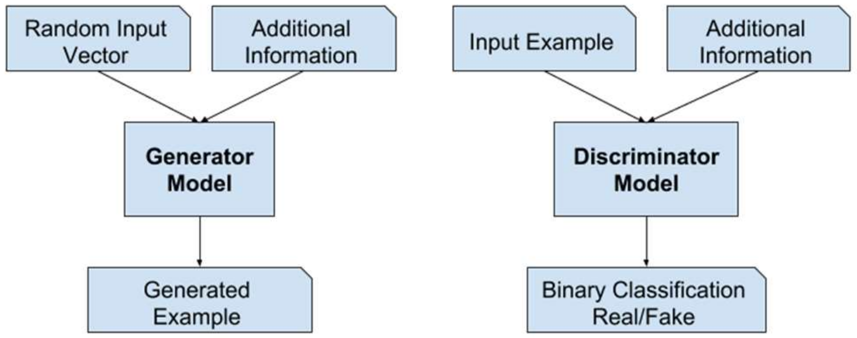


Fig. Overview of the cGAN training process*

* https://machinelearningmastery.com

** P. Isola, J. Zhu, T. Zhou, and A. A. Efros. 2017. Image-to-image translation with conditional adversarial networks. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 5967–5976. https://doi.org/10.1109/CVPR.2017.632

How Does Time-Series Work in all of this?

- Encode time series as an image by transforming to a Gramian Angular Sum Fields (GASF)*
  - Rescale the measurements between the interval [-1,1]
  - Convert to the polar coordinate system by taking the arccosine of each time step
- Trigonometric sum between cosine of the sum of the ith and the jth angular points to form a GASF
  - Takes < 10ms to finish the GASF generation over the entire dataset
  - Captures the temporal dependency between different time steps
- Captures the ISI effectively

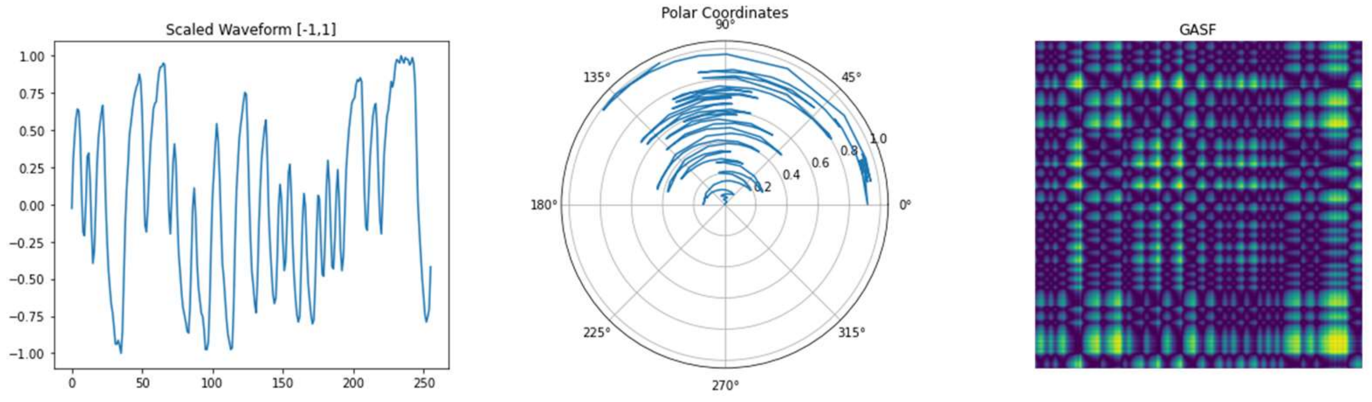


Fig. Conversion of time series to a polar encoding and a GASF

* Z. Wang and T. Oates. 2015. Imaging time-series to improve classification and imputation. In International Joint Conference on Artificial Intelligence (IJCAI). 3939–3945.

U-Net Generator (2-Encoder)

- The generator is given the input waveform encoded as a GASF and taps weights to predict the bit error rate (BER) contour plot
  - U-Net-based generator as suggested by Pix2Pix *
  - Additional encoder network to learn the tap settings
- The tap network is a 4-layer fully connected network where the outputs are latent variables

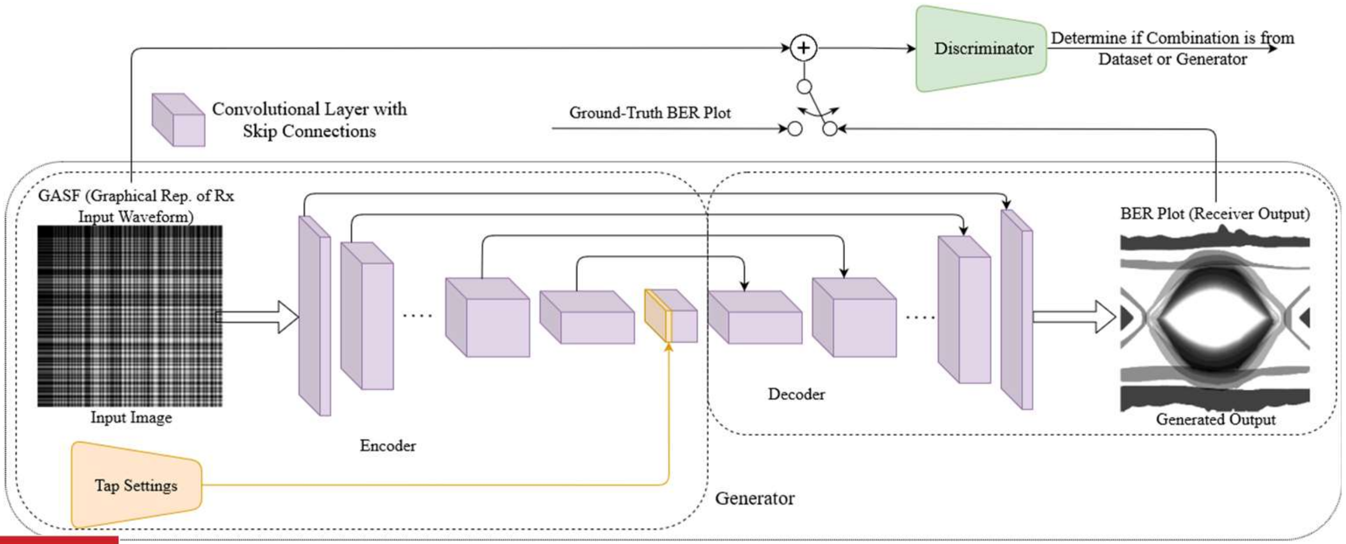

Discriminator
Determine if Combination is from Dataset or Generator
Convolutional Layer with Skip Connections
Ground-Truth BER Plot
GASF (Graphical Rep. of Rx Input Waveform)
BER Plot (Receiver Output)
Input Image
Encoder
Decoder
Generated Output
Tap Settings
Generator

Fig. Generator of the cGAN conditioned on the GASF and DFE tap configurations to predict an eye diagram

*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.

U-Net Discriminator (2-Encoder)

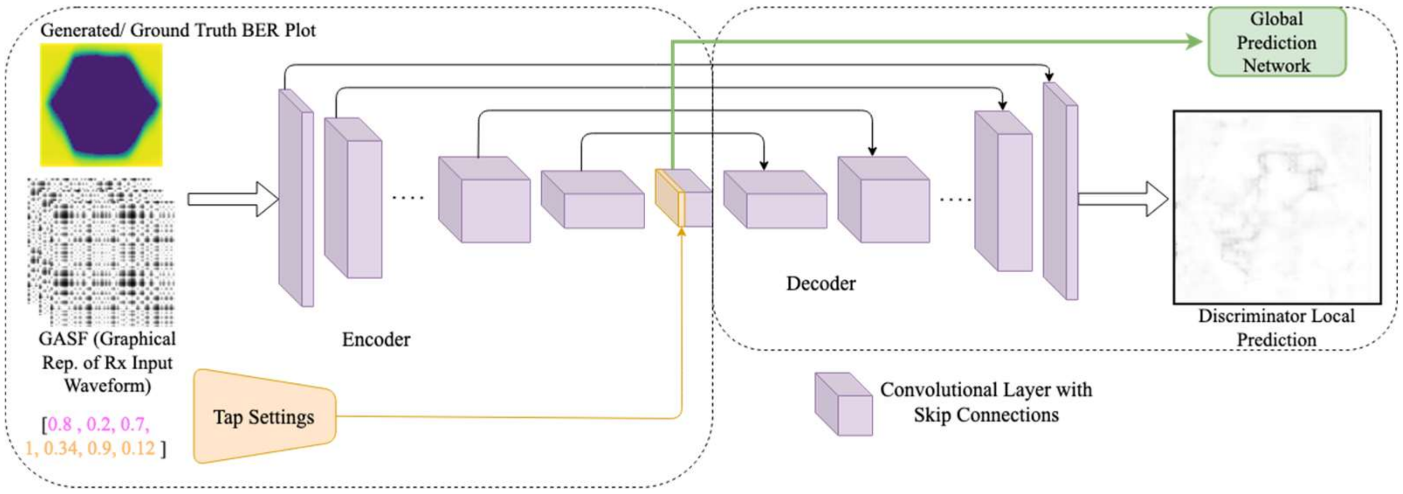

Generated/ Ground Truth BER Plot
GASF (Graphical Rep. of Rx Input Waveform)
[0.8 , 0.2, 0.7, 1, 0.34, 0.9, 0.12]
Tap Settings
Encoder
Decoder
Global Prediction Network
Discriminator Local Prediction
Convolutional Layer with Skip Connections

Fig. cGAN discriminator where the GAN is conditioned on a GASF and tap weigths and predicts a local and global value

- The discriminator is a U-Net architecture that predicts both a full pixel map (at decoder output) and a single true/false prediction (at the bottleneck) for a given input*
  - Takes the input GASF, tap values, and either the ground truth BER plot or generated BER plot
  - Predicts whether the concatenated image is from the dataset or generator using two levels of prediction

* E. Schonfeld, B. Schiele, and A. Khoreva. 2020. A U-Net based discriminator for generative adversarial networks. In IEEE/CVF Conference on Computer Vision and Pattern Recognition. 8207–8216.

Metric Network

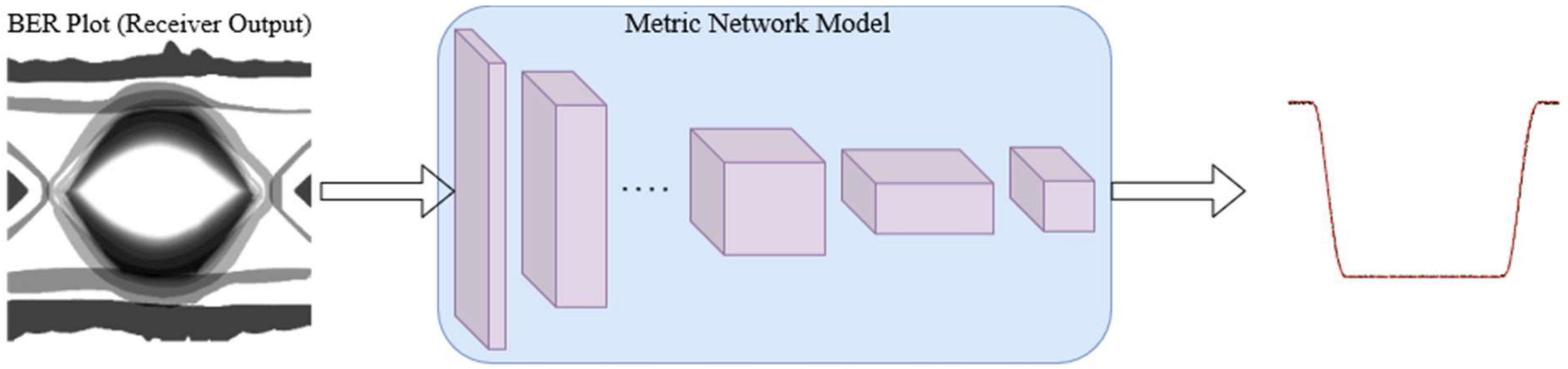


Fig. Neural network predicts the bathtub curves from the ground truth BER/eye-diagram

- To evaluate the quality of generated images, we use a deep neural network (DNN) trained on the ground truths eye diagram and their corresponding characteristics
  - Uses encoder model architecture from the GAN generator/discriminator
  - Consists of successive Convolution and Batch Normalization layers to reduce the dimension of the input image
- Output neurons correspond to the number of characteristics to evaluate the generated plots
  - 700 points for the bathtub curve
    - We calculate the Pearson Correlation Coefficient (PCC) as well as the root mean squared error (RMSE)of the generated bathtub curve to the ground-truth bathtub curve to evaluate the generated BER plot
  - 2 for the eye characteristics: eye height and width

Datasets

Name	Sim/ Meas.	DFE	CTLE	Bitstreams	Channel Conditions	Crosstalk	Characteristics
NRZ (no Xtalk)	Meas.	Yes	Yes	Yes	Yes	No	BTC/Eye
NRZ (Xtalk)	Meas.	Yes	Yes	Yes	Yes	Yes	BTC/Eye
PAM4 (Xtalk)	Meas.	Yes	Yes	Yes	Yes	Yes	BTC

Table. Description of the different datasets

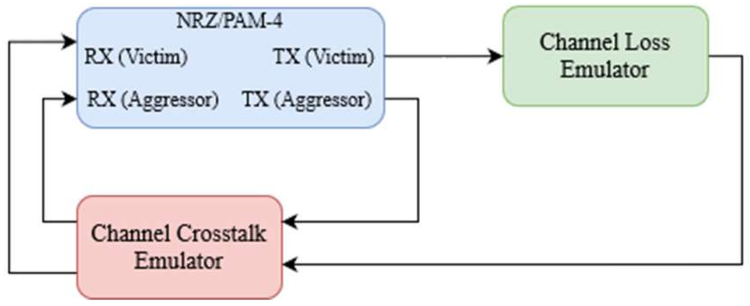


Fig. Device setup for measurement based

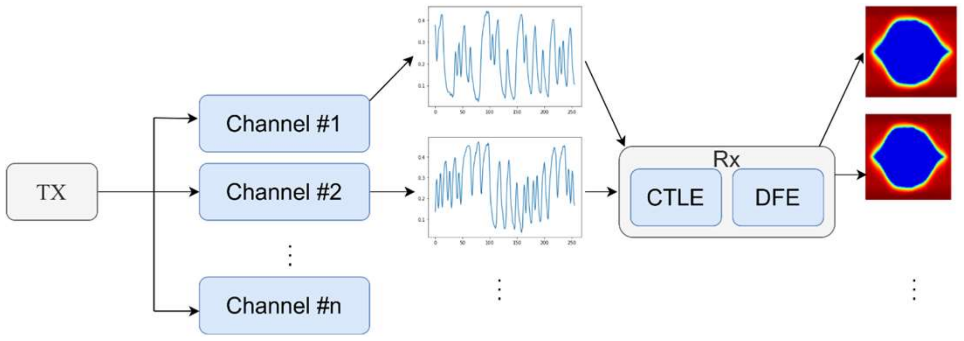


Fig. Alternate view of the data collection setup

BER Pre-processing

- For NRZ
  - 5 channels with varying loss conditions
- For NRZ with crosstalk
  - 4 channels with varying loss conditions
  - 3 crosstalk configuration per loss
- For PAM-4
  - 4 channels with varying loss conditions
  - 7 crosstalk configuration per loss
- 7 receiver tap configurations
  - Includes CTLE and DFE tap values
  - Uniform sampling across all taps
- Captured BER contour plot
  - Original dimension is 330x330 to 256x256
  - Log_{10} of the original value and scaled [0,1]

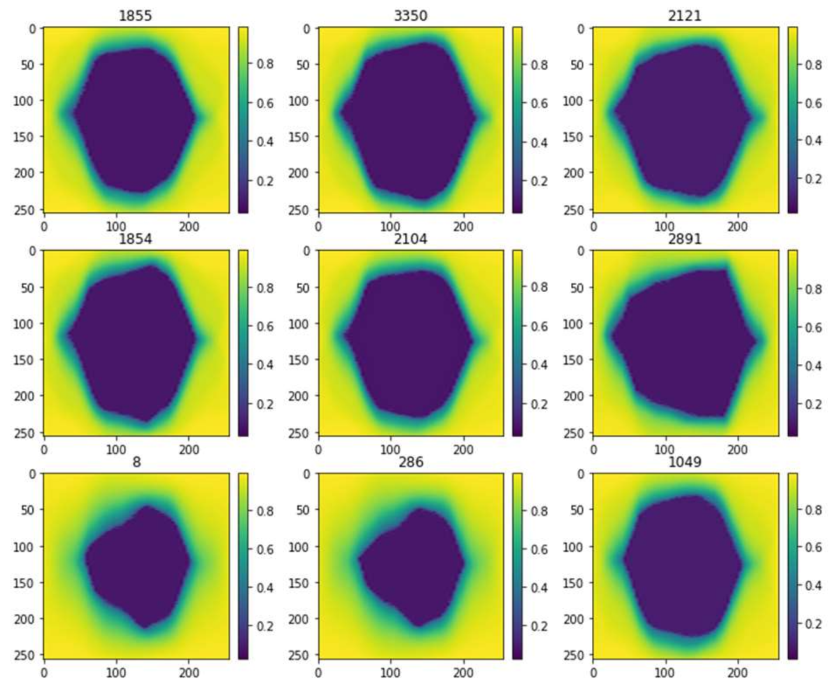


Fig. Random BER contour plots from dataset; range of channel conditions shown

Hewlett Packard Enterprise | NC STATE UNIVERSITY

Time Series Pre-processing

- Capture the receiver waveform for a SerDes running at 15 Gb/s at 20 ps intervals with a PRBS-15 bitstream
- Select the number of bits from PRBS and create a window using 256 time-steps
  - Include overlaps with previous windows to form a GASF input with multiple channels that maintains dependencies across multiple channels
- Raw time series used to generate the GASF
  - 25% overlap between windows
  - Can extend to a more extensive bit sequence without performance degradation
- Each window ends up being treated as a channel (like RGB) and together forms the input for the GAN

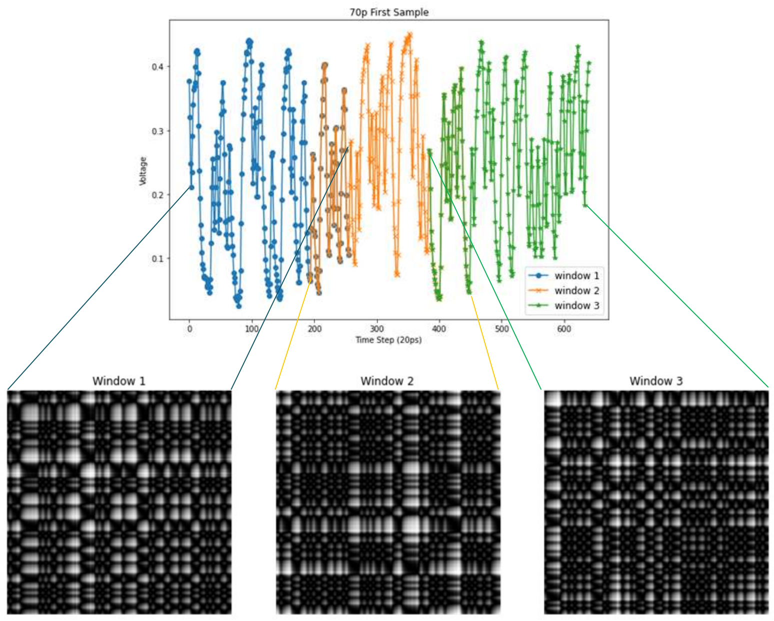


Fig. Windowing the time series and its corresponding GASF for 150 bits

Training Progression with the U-Net

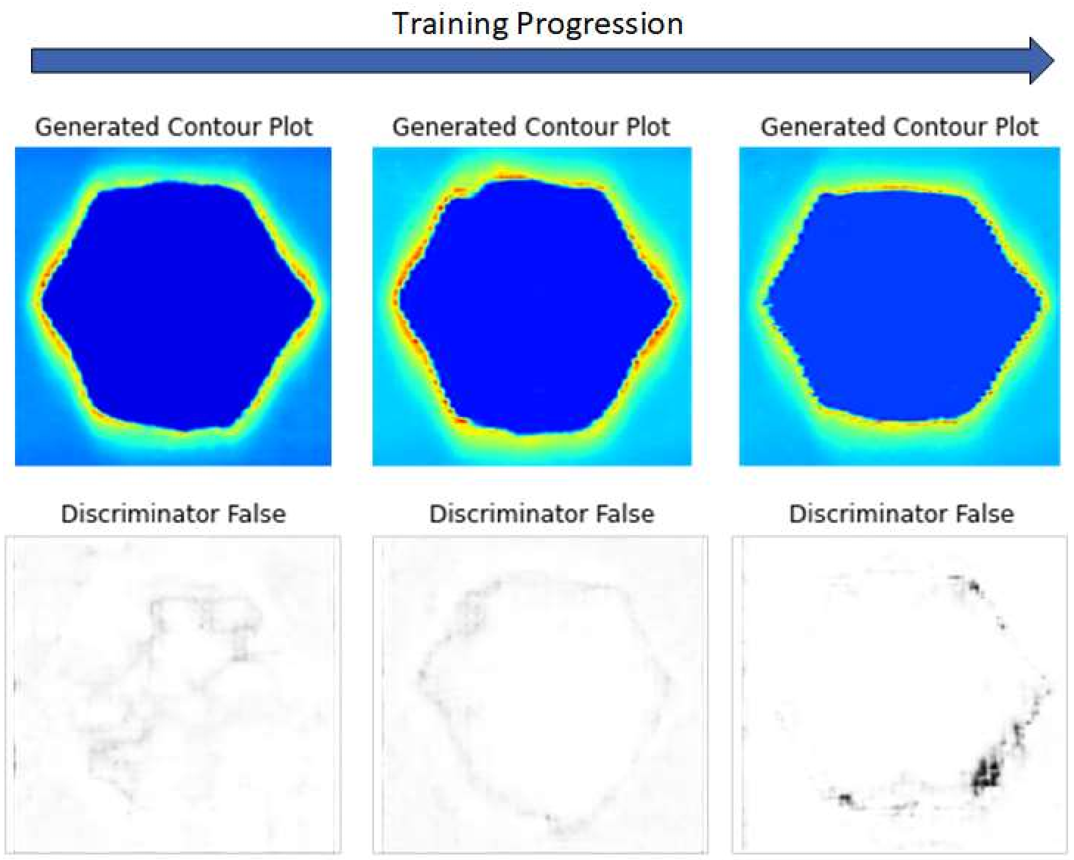
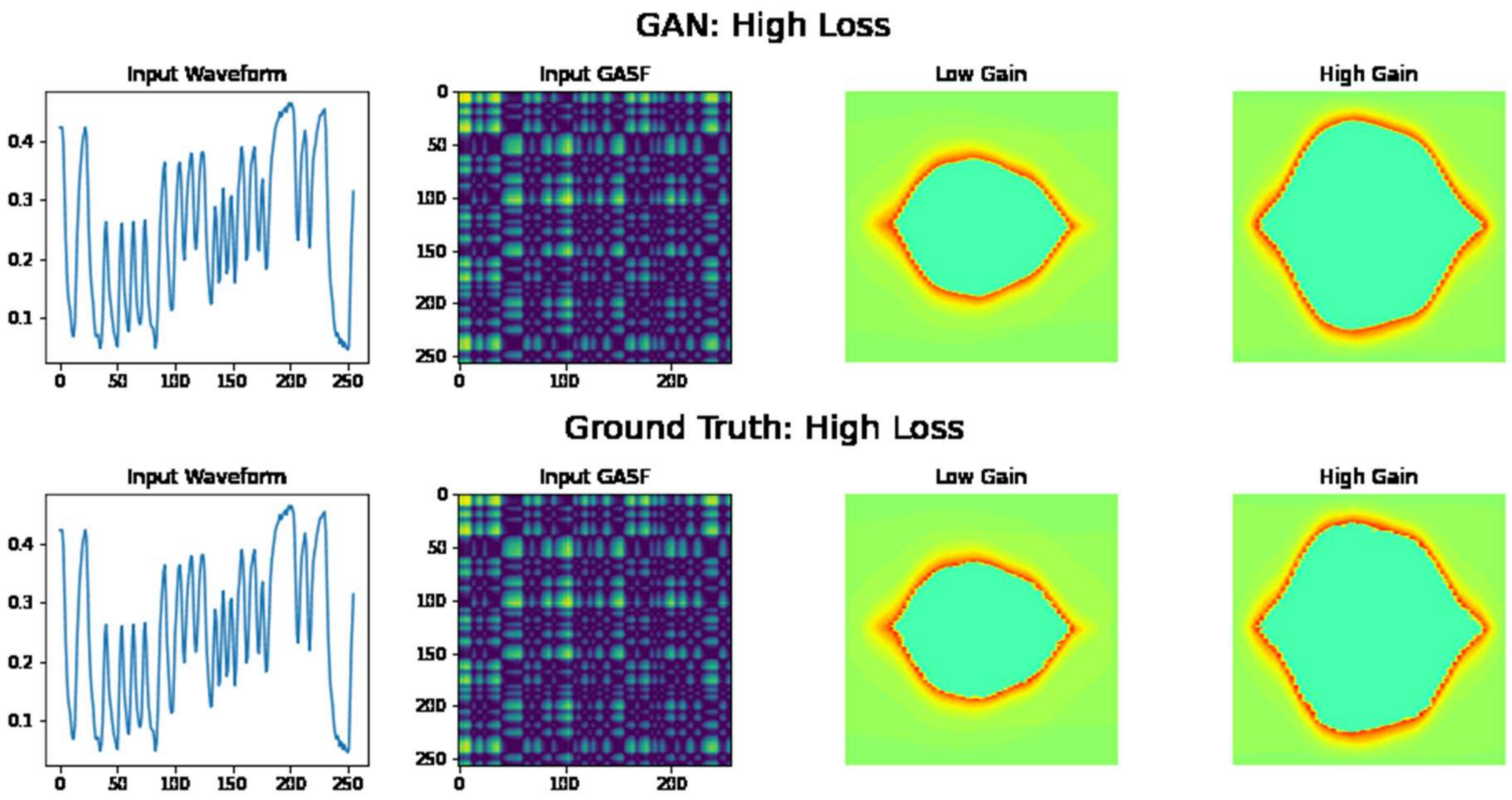
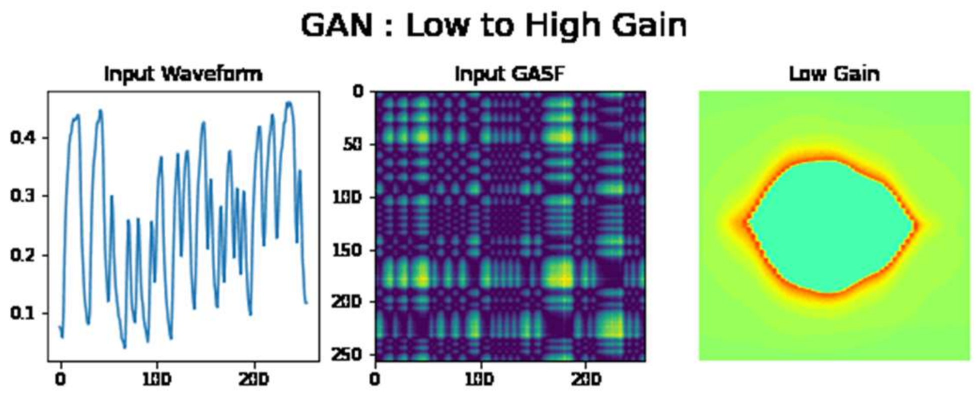


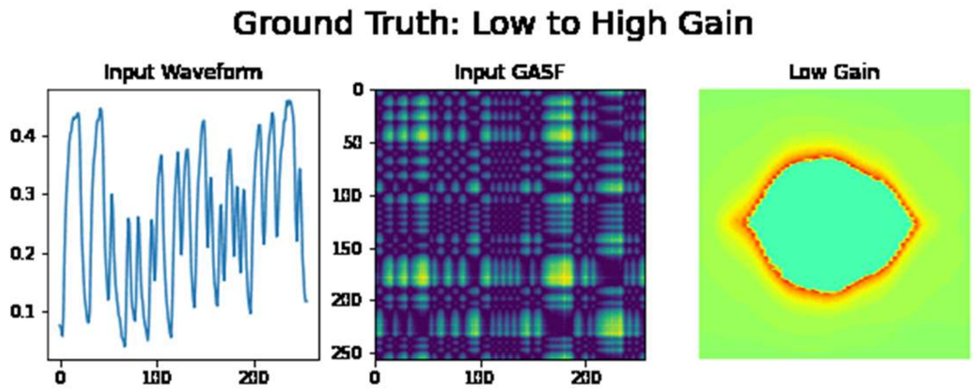
Fig. Evolution of the discriminator pixel prediction and the corresponding generated contour plot for a known GASF

Interpolation Between Channels



Interpolation Gain Conditions





NRZ with Crosstalk

- Additional tap to indicate whether there is crosstalk
  - 300 bits with a 25% overlap
- Metric network error (EH/EW)
  - MAPE: 4.47% and 3.68%
- Time series overlapping
  - Without overlapping EH/EW error increases to 18.14% and 9.96%

# of Bits	MAPE (EH/ EW)	PCC
300	**<u>6.83</u>** **<u>5.16</u>**	0.99
400	18.38 7.67	0.98
500	18.95 7.89	0.98
600	9.57 6.05	0.99

Table. Error with respect to increasing bit sequence

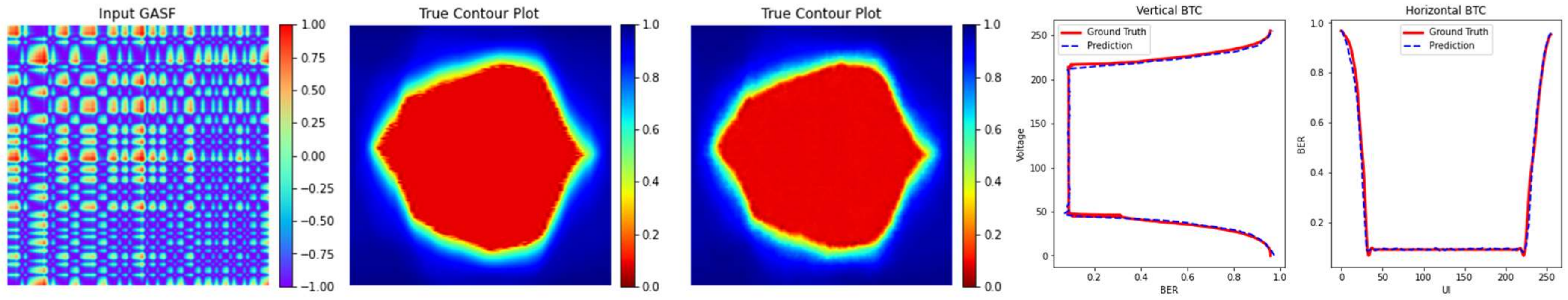
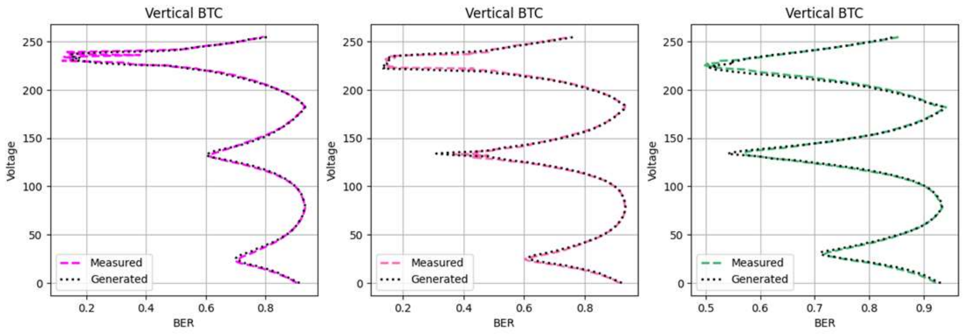
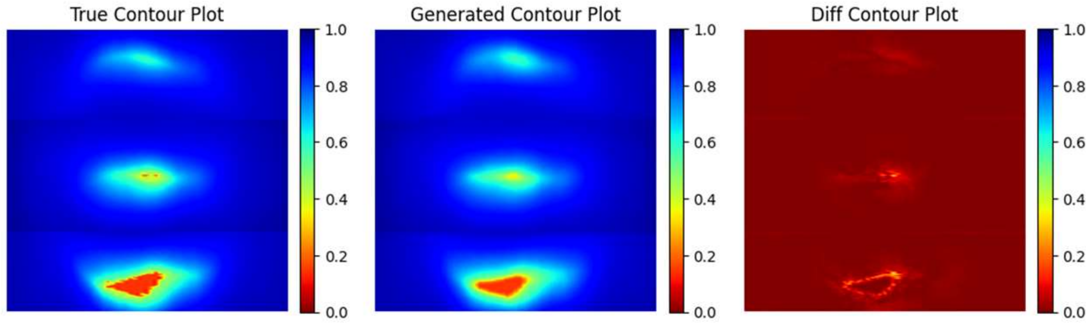


Fig. Results (left to right) the GASF input, ground-truth, cGAN-generated BER contour plot, vertical and horizontal BTC

Hewlett Packard Enterprise | NC STATE UNIVERSITY

PAM-4





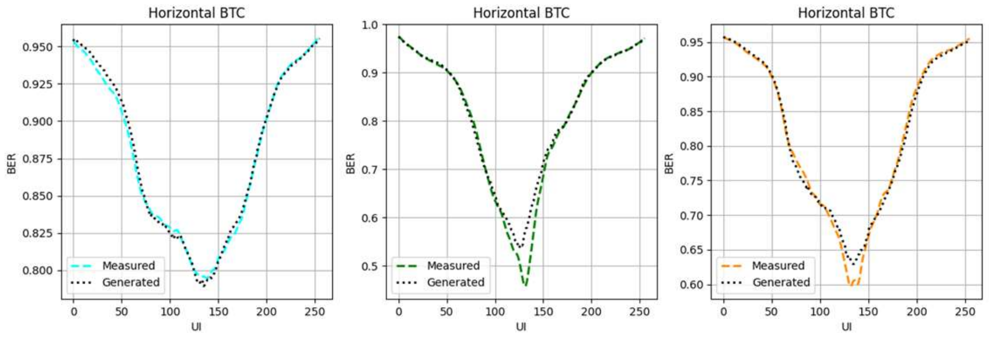


Fig. Evaluation of the horizontal and vertical bathtub curves for the ground truth and generated BER contour plots

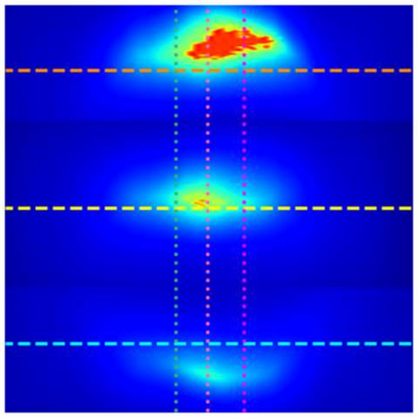


Fig. Reference for where each bathtub curve slice is taken

- U-Net Generator (2-Encoder) & U-Net Discriminator (2-Encoder)
  – Generator loss function looks at random selection of BTCs

Hewlett Packard Enterprise | NC STATE UNIVERSITY

PAM-4 Continued

- Calculate the image difference of the generated vs. the ground truth BER contours
- MAPE (Mean Absolute Percentage Error) as the image difference

$$MAPE(l,c) = \frac{1}{M_{lc}} \sum_{m=1}^{M_{lc}} \left(100 * \frac{1}{N} \sum_{n=1}^{N} \left| \frac{A_{mn} - g_{mn}}{A_{mn}} \right| \right)$$

- The worst MAPE is 2.5%
  - When loss = 40% & Xtlk = 60%

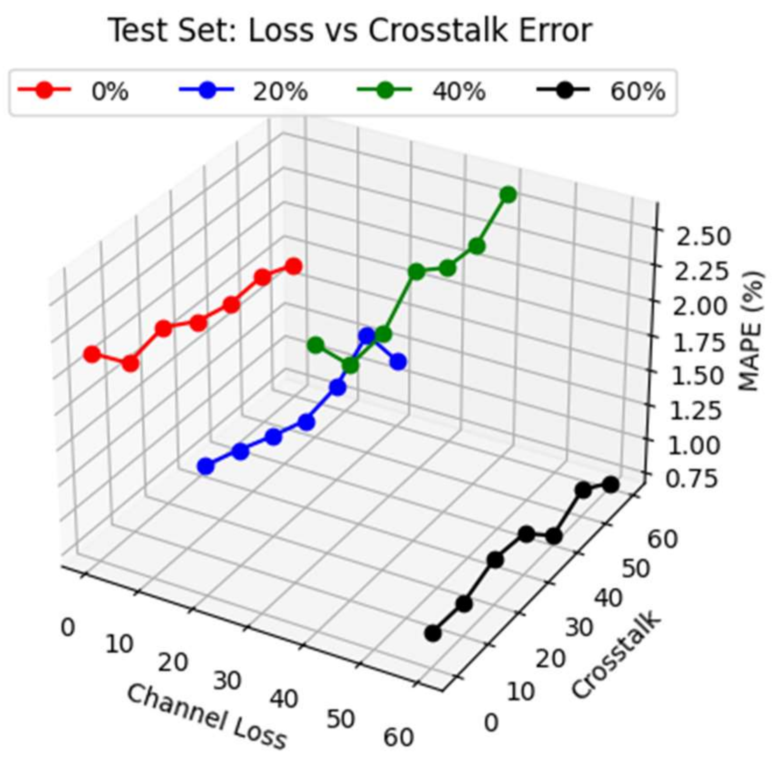


Fig. Channel loss vs Crosstalk MAPE Error on test set

Training and Inference Times

Name	Training Size	Test Size	Training Time (per iteration) (sec)	Testing Time (per batch of 4) (ms)
NRZ (no Xtalk)	14,400	3,600	134.51	135
NRZ (Xtalk)	6,917	1,730	107.35	127
PAM4	11,468	2,868	178	129

Table. Comparing dataset sizes with training and inference times

Thermal Problems

- Like high-speed receiver modeling, thermal simulations can take up significant resources
  - –The compute depending on the size of the board
  - –Requires iterative simulations during development to ensure thermal performance is met
- The tools themselves are iterative
  - –They perform power and electrical simulations and
  - –Feed the temperature solution back to recalculate the electrical parameters
  - – The electrical stimulation restarts with new parameters until the power and thermal solution reach equilibrium.

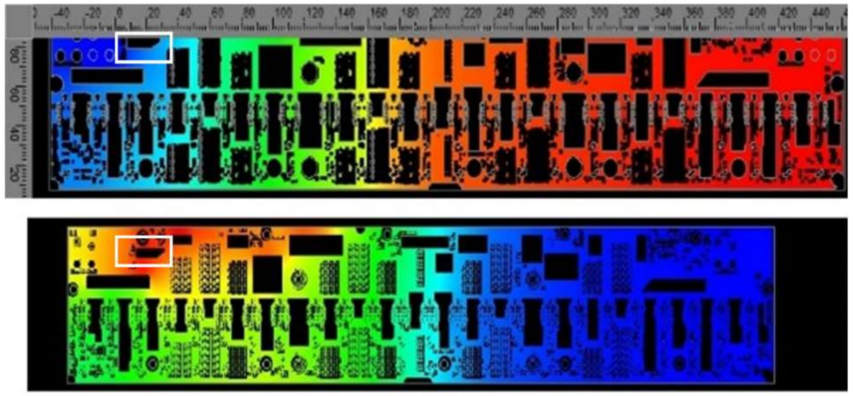


Fig. The backplane thermal hotspot (bottom) due to excessive power drop (top).

Hewlett Packard Enterprise | NC STATE UNIVERSITY

System Level Data

- Vary the sink current between [0-40] A at uniform intervals for one chip with the other off
  - Then switch which chip is *on*
- Run Cadence PowerDC©
  - First collect single IR current density maps – no heating effects
    - Is the input for the cGAN
  - Run E/T Co-Simulation flow
    - Captures the heating effects
    - Is the output for the cGAN
- Preprocessing
  - Scale the power and heat maps to [-1,1]
  - Split into 80% for training-validation and the remaining 20% for testing

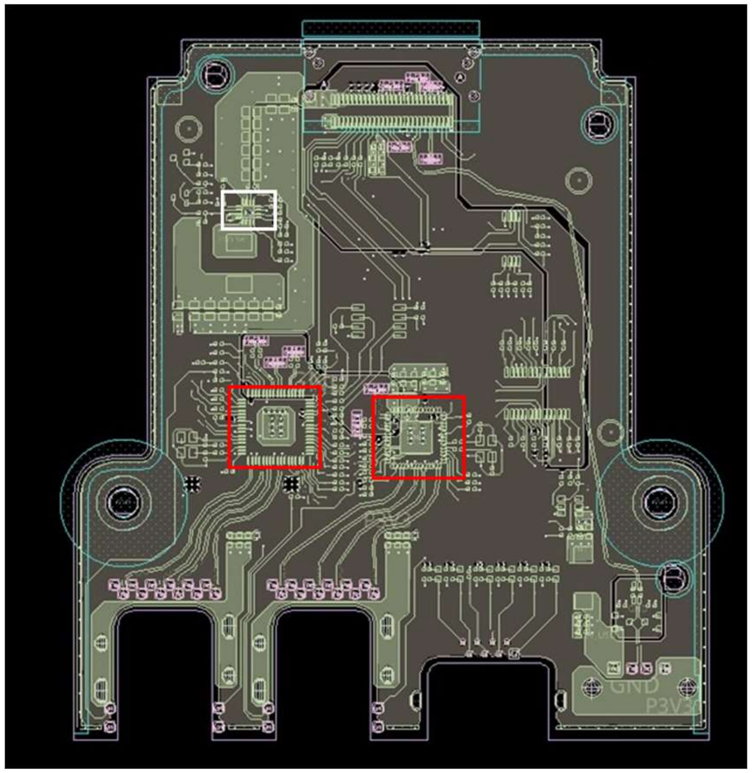
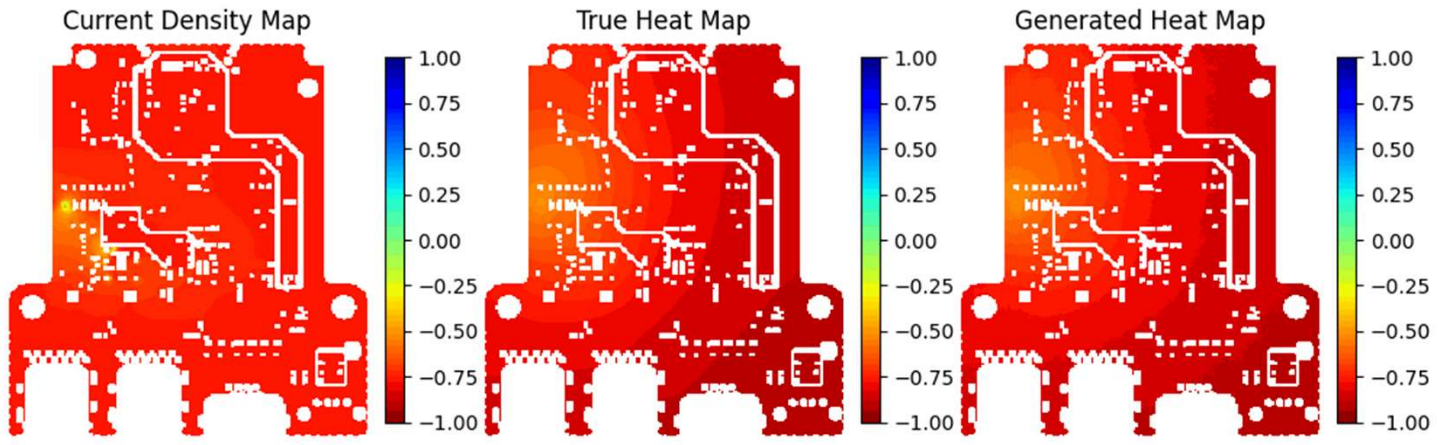


Fig. Thermal problem for simulation

Hewlett Packard Enterprise | NC STATE UNIVERSITY

Results on System Level



Animation: (Left to right) Ground truth current density map, simulated heat map and the generated heat map

Thermal System Level – Over a Dataset

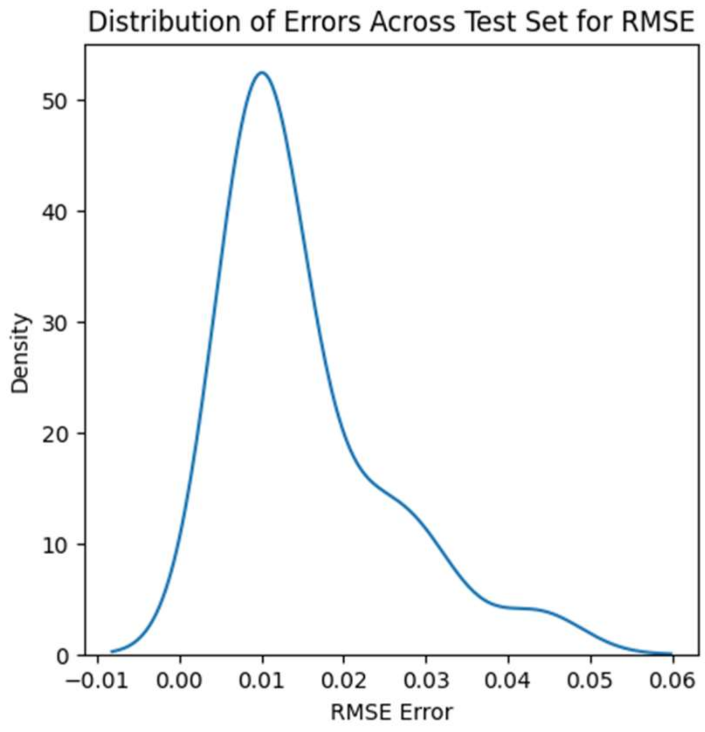


Fig. Distribution of test set errors

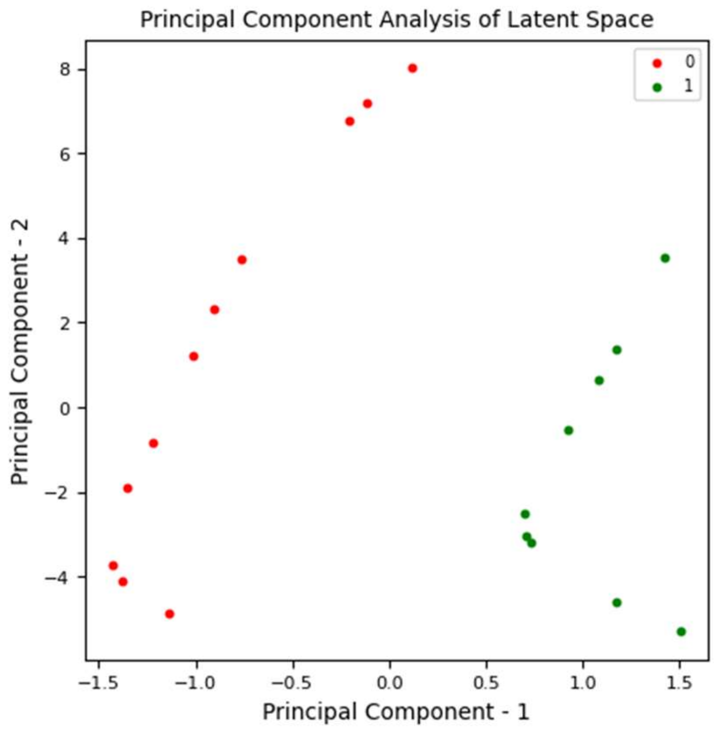


Fig. Exploring the latent space to understand the underlying physics

Generative Digital Twins

- Digital twins are computational models that cover the solution space within a targeted design limit
- Asymmetrical training vs. inference speed
  - Large dataset and high computational demand to train a generative model
  - Lightweight and fast computation for inference
- Realtime prediction of SI or multi-physics (power/thermal) systems
- Allow dynamic performance tuning based on changing input condition
- Generative models may have invertible solutions, i.e., given a desirable output, what is the most likelihood input condition
  - It can be constrained by power, space etc.

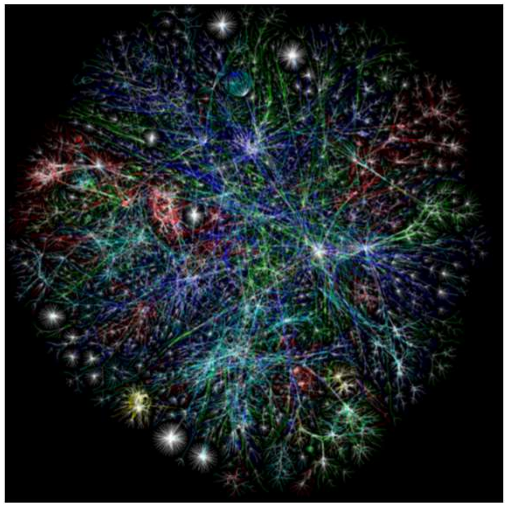
Demo!

Outline

- Generative Adversarial Networks for Modeling
- LLMs for EDA
  - What are Large Language Models?
  - How do they Work?
  - LLM Applications for EDA
  - Challenges on Using LLMs in EDA

Hewlett Packard Enterprise | NC STATE UNIVERSITY

What are Large Language Models?



Chunk of the internet, ~10TB of text

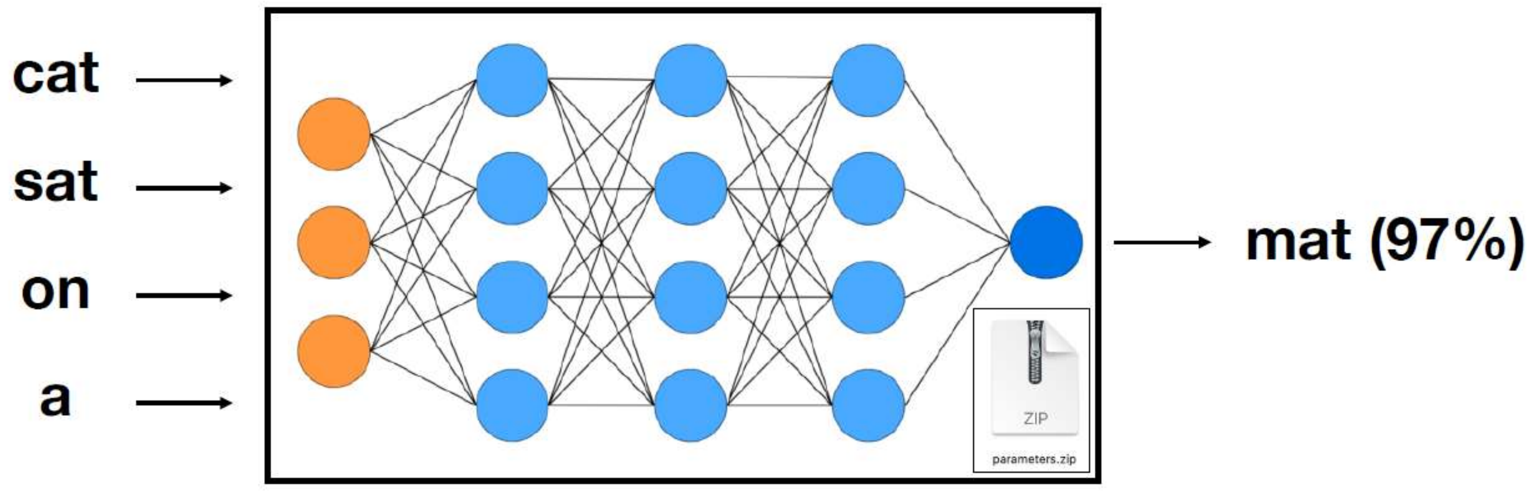
6,000 GPUs for 12 days, ~$2M ~1e^{24} FLOPS

parameters.zip
~140 GB file

Andrej Karpathy – Introduction to Large Language Models; https://www.youtube.com/watch?v=zjkBMFhNj_g

*numbers for Llama 2 70B

How do they Work?



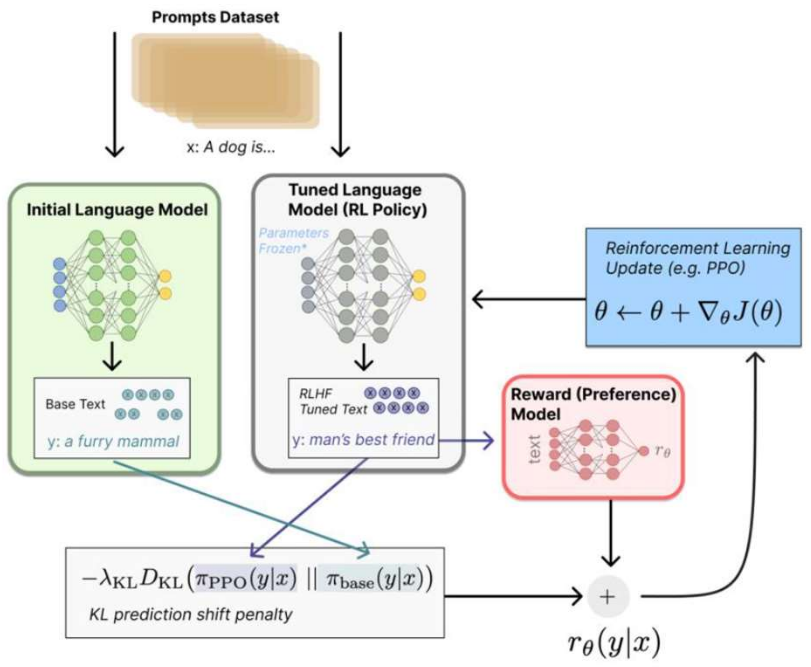
e.g. context of 4 words

predict next word

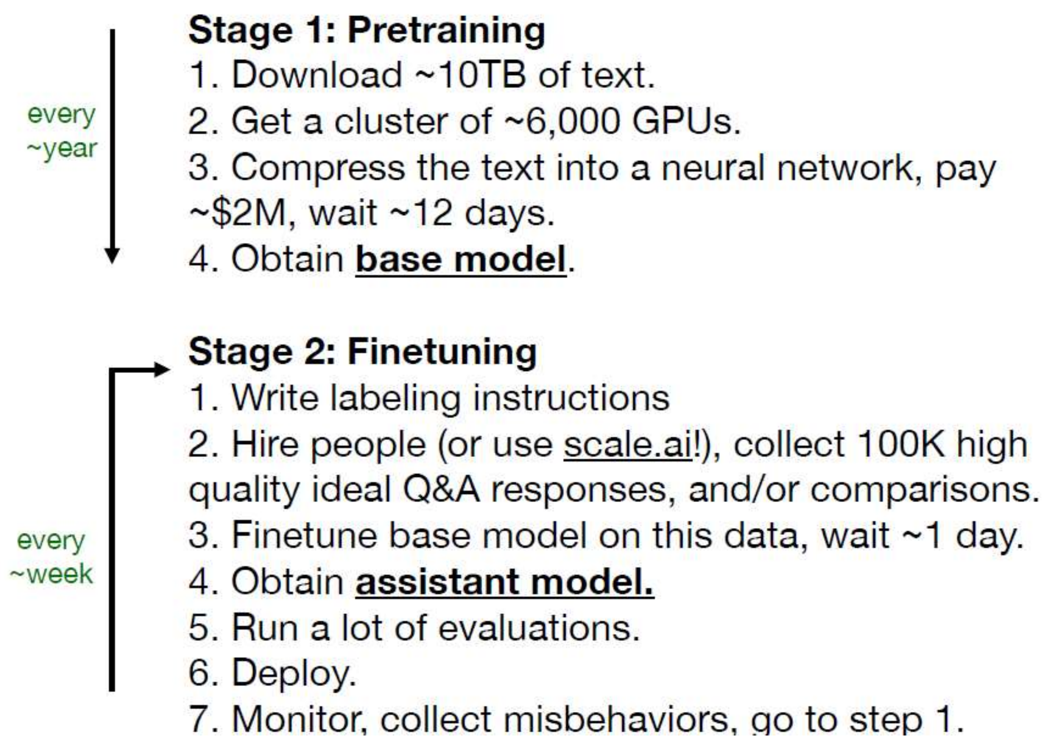
How do they Work?

Step 1: Pretrain on next word prediction for large volumes of text (trillions of examples of the form below):
A dog is man's best _____ => friend

Step 2: Perform **Reinforcement Learning from Human Feedback (RLHF)** to train the model to do what humans want, rather than just predict next word. - *finetuning*


Prompts Dataset
x: A dog is...
Initial Language Model
Base Text
y: a furry mammal
Tuned Language Model (RL Policy)
Parameters Frozen*
RLHF Tuned Text
y: man's best friend
Reinforcement Learning Update (e.g. PPO)
$\theta \leftarrow \theta + \nabla_\theta J(\theta)$
Reward (Preference) Model
text
r_θ
$-\lambda_{KL} D_{KL}(\pi_{PPO}(y|x) \,||\, \pi_{base}(y|x))$
KL prediction shift penalty
+
$r_\theta(y|x)$

Summary of LLMs



every ~year

Stage 1: Pretraining

1. Download ~10TB of text.
2. Get a cluster of ~6,000 GPUs.
3. Compress the text into a neural network, pay ~$2M, wait ~12 days.
4. Obtain **base model**.

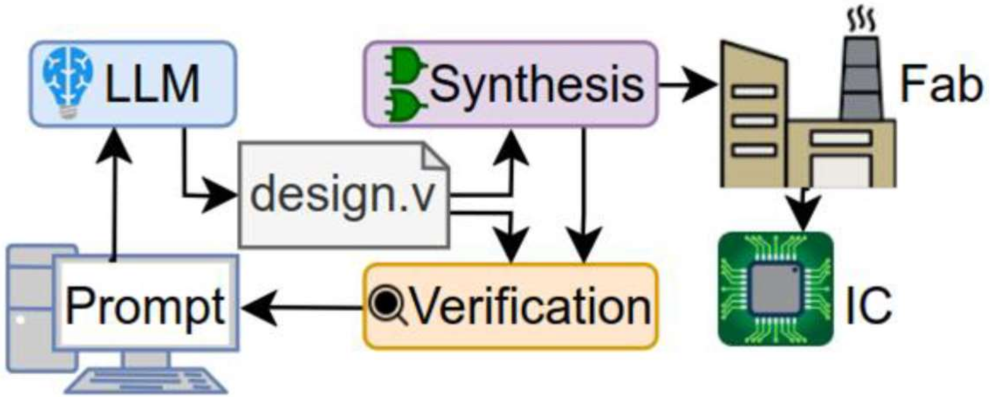
every ~week

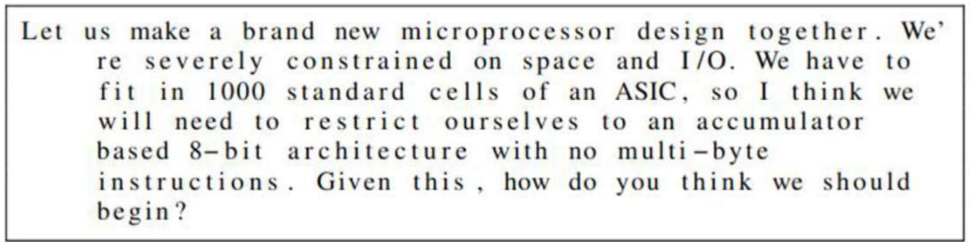
Stage 2: Finetuning

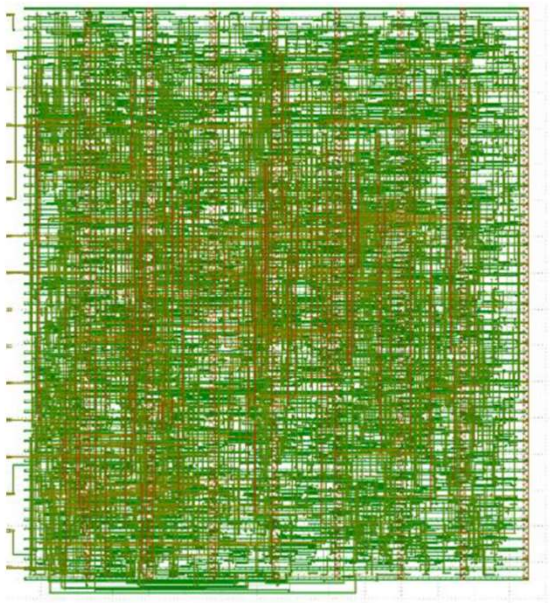
1. Write labeling instructions
2. Hire people (or use scale.ai!), collect 100K high quality ideal Q&A responses, and/or comparisons.
3. Finetune base model on this data, wait ~1 day.
4. Obtain **assistant model.**
5. Run a lot of evaluations.
6. Deploy.
7. Monitor, collect misbehaviors, go to step 1.

Andrej Karpathy – Introduction to Large Language Models; https://www.youtube.com/watch?v=zjkBMFhNj_g

LLM Designed Chip to Tape out




Let us make a brand new microprocessor design together. We're severely constrained on space and I/O. We have to fit in 1000 standard cells of an ASIC, so I think we will need to restrict ourselves to an accumulator based 8-bit architecture with no multi-byte instructions. Given this, how do you think we should begin?



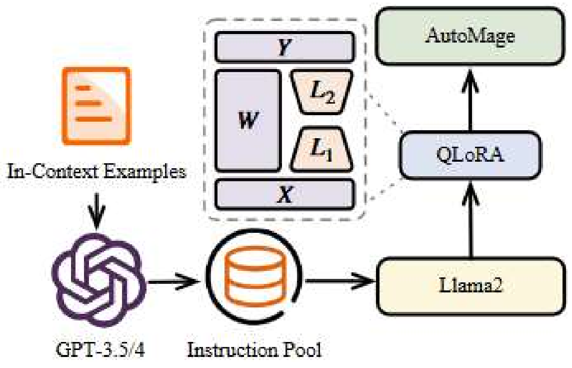
Component	Count
Comb. Logic	999
Diode	4
Flip Flops	168
Buffer	126
Tap	300

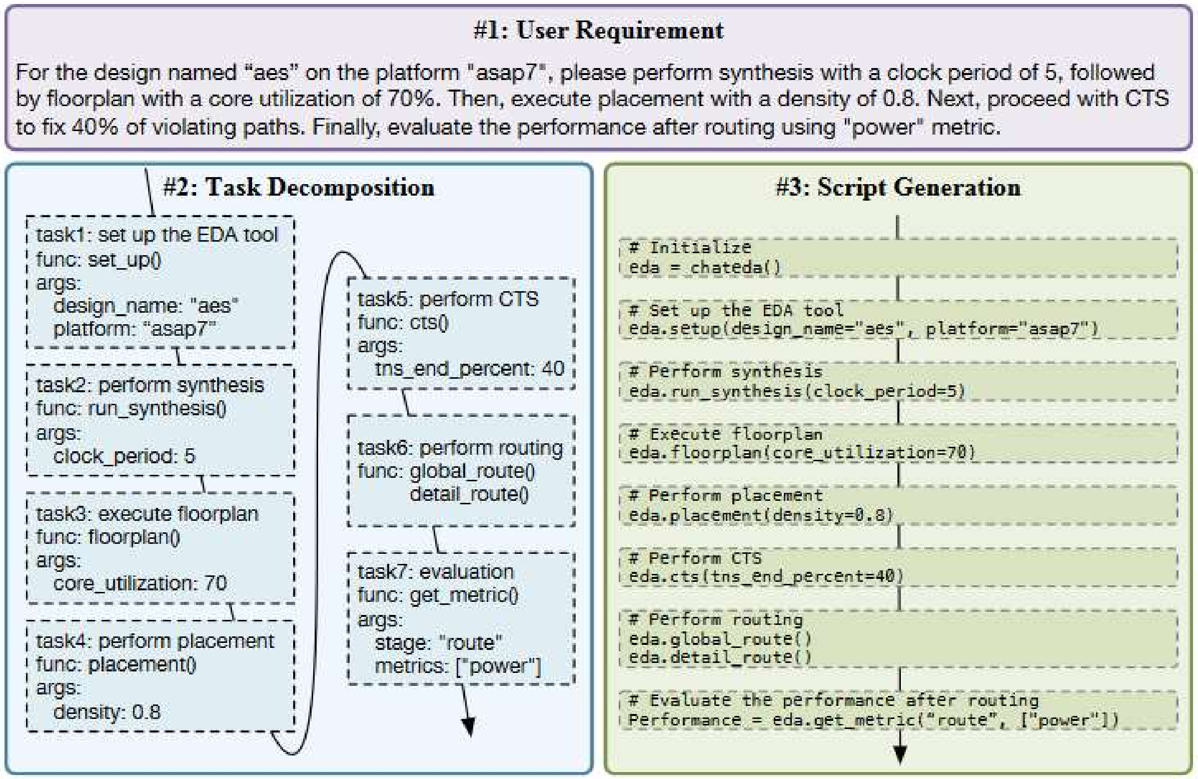
Above: (a) Components.

Left: (b) Final processor GDS render by 'klayout', I/O ports on left side, grid lines = 0.001 um.

Hewlett Packard Enterprise | NC STATE UNIVERSITY

Script Generation and Automation





#1: User Requirement

For the design named "aes" on the platform "asap7", please perform synthesis with a clock period of 5, followed by floorplan with a core utilization of 70%. Then, execute placement with a density of 0.8. Next, proceed with CTS to fix 40% of violating paths. Finally, evaluate the performance after routing using "power" metric.

#2: Task Decomposition

```
task1: set up the EDA tool
func: set_up()
args:
  design_name: "aes"
  platform: "asap7"

task2: perform synthesis
func: run_synthesis()
args:
  clock_period: 5

task3: execute floorplan
func: floorplan()
args:
  core_utilization: 70

task4: perform placement
func: placement()
args:
  density: 0.8

task5: perform CTS
func: cts()
args:
  tns_end_percent: 40

task6: perform routing
func: global_route()
      detail_route()

task7: evaluation
func: get_metric()
args:
  stage: "route"
  metrics: ["power"]
```

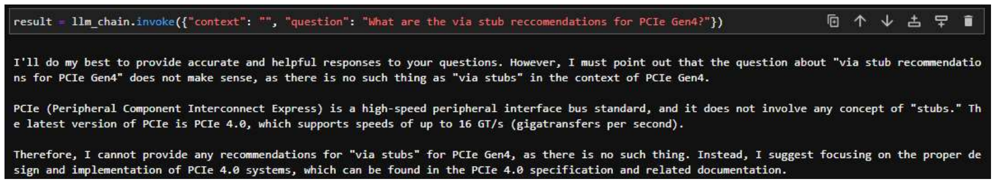
#3: Script Generation

```
# Initialize
eda = chateda()

# Set up the EDA tool
eda.setup(design_name="aes", platform="asap7")

# Perform synthesis
eda.run_synthesis(clock_period=5)

# Execute floorplan
eda.floorplan(core_utilization=70)

# Perform placement
eda.placement(density=0.8)

# Perform CTS
eda.cts(tns_end_percent=40)

# Perform routing
eda.global_route()
eda.detail_route()

# Evaluate the performance after routing
Performance = eda.get_metric("route", ["power"])
```

Challenges with LLMs in EDA

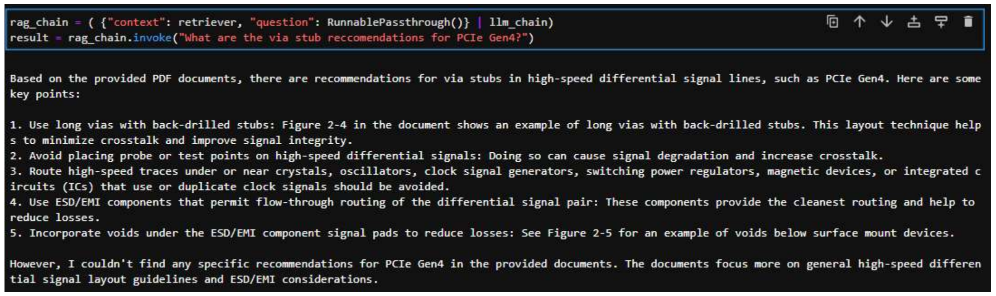
- Verification
  - Code generation is good with the LLMs
  - However, we need to ensure that the generated code is accurate
    - Easy for high-resource languages like Python harder for HDL
- Data-problem
  - Data needs to be on-premise due to its sensitive nature
  - Code for designs are stored in-house
- Optimization
  - Numerous problems on the backend require further optimization
  - Where do LLMs come in?

Chat with Your Documents



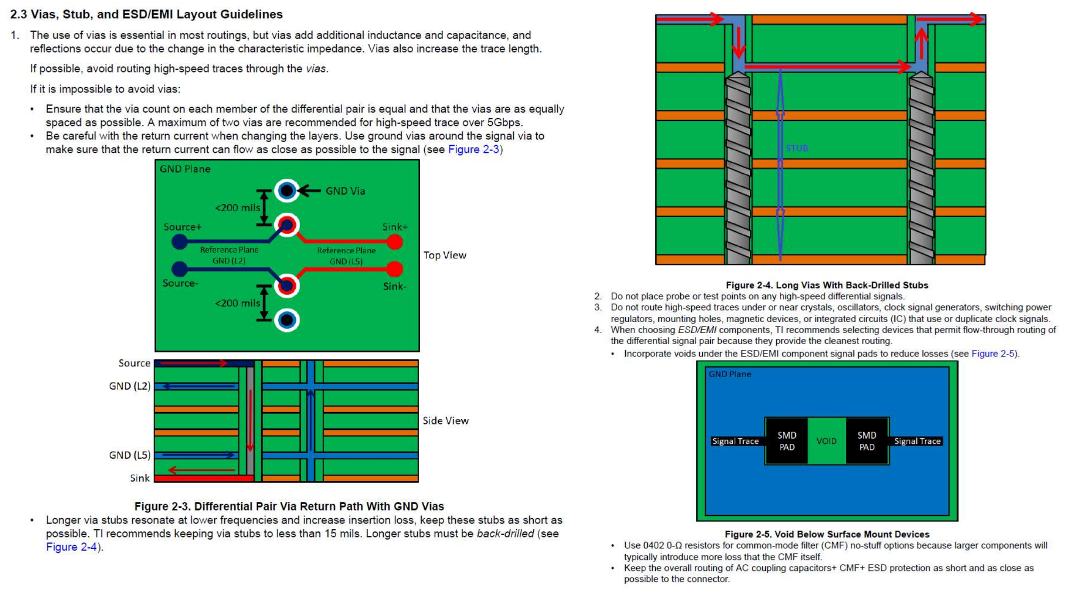
```
result = llm_chain.invoke({"context": "", "question": "What are the via stub reccomendations for PCIe Gen4?"})

I'll do my best to provide accurate and helpful responses to your questions. However, I must point out that the question about "via stub recommendatio
ns for PCIe Gen4" does not make sense, as there is no such thing as "via stubs" in the context of PCIe Gen4.

PCIe (Peripheral Component Interconnect Express) is a high-speed peripheral interface bus standard, and it does not involve any concept of "stubs." Th
e latest version of PCIe is PCIe 4.0, which supports speeds of up to 16 GT/s (gigatransfers per second).

Therefore, I cannot provide any recommendations for "via stubs" for PCIe Gen4, as there is no such thing. Instead, I suggest focusing on the proper de
sign and implementation of PCIe 4.0 systems, which can be found in the PCIe 4.0 specification and related documentation.
```

Fig: No RAG model response about via stub recommendations



```
rag_chain = ( {"context": retriever, "question": RunnablePassthrough()} | llm_chain)
result = rag_chain.invoke("What are the via stub reccomendations for PCIe Gen4?")

Based on the provided PDF documents, there are recommendations for via stubs in high-speed differential signal lines, such as PCIe Gen4. Here are some
key points:

1. Use long vias with back-drilled stubs: Figure 2-4 in the document shows an example of long vias with back-drilled stubs. This layout technique help
s to minimize crosstalk and improve signal integrity.
2. Avoid placing probe or test points on high-speed differential signals: Doing so can cause signal degradation and increase crosstalk.
3. Route high-speed traces under or near crystals, oscillators, clock signal generators, switching power regulators, magnetic devices, or integrated c
ircuits (ICs) that use or duplicate clock signals should be avoided.
4. Use ESD/EMI components that permit flow-through routing of the differential signal pair: These components provide the cleanest routing and help to
reduce losses.
5. Incorporate voids under the ESD/EMI component signal pads to reduce losses: See Figure 2-5 for an example of voids below surface mount devices.

However, I couldn't find any specific recommendations for PCIe Gen4 in the provided documents. The documents focus more on general high-speed differen
tial signal layout guidelines and ESD/EMI considerations.
```

Fig: Model response with RAG on a PCIe Gen4 document



2.3 Vias, Stub, and ESD/EMI Layout Guidelines

1. The use of vias is essential in most routings, but vias add additional inductance and capacitance, and reflections occur due to the change in the characteristic impedance. Vias also increase the trace length.

   If possible, avoid routing high-speed traces through the *vias*.

   If it is impossible to avoid vias:

   - Ensure that the via count on each member of the differential pair is equal and that the vias are as equally spaced as possible. A maximum of two vias are recommended for high-speed trace over 5Gbps.
   - Be careful with the return current when changing the layers. Use ground vias around the signal via to make sure that the return current can flow as close as possible to the signal (see Figure 2-3)

Figure 2-3. Differential Pair Via Return Path With GND Vias

   - Longer via stubs resonate at lower frequencies and increase insertion loss, keep these stubs as short as possible. TI recommends keeping via stubs to less than 15 mils. Longer stubs must be *back-drilled* (see Figure 2-4).

Figure 2-4. Long Vias With Back-Drilled Stubs

2. Do not place probe or test points on any high-speed differential signals.
3. Do not route high-speed traces under or near crystals, oscillators, clock signal generators, switching power regulators, mounting holes, magnetic devices, or integrated circuits (IC) that use or duplicate clock signals.
4. When choosing *ESD/EMI* components, TI recommends selecting devices that permit flow-through routing of the differential signal pair because they provide the cleanest routing.
   - Incorporate voids under the ESD/EMI component signal pads to reduce losses (see Figure 2-5).

Figure 2-5. Void Below Surface Mount Devices

   - Use 0402 0-Ω resistors for common-mode filter (CMF) no-stuff options because larger components will typically introduce more loss that the CMF itself.
   - Keep the overall routing of AC coupling capacitors+ CMF+ ESD protection as short and as close as possible to the connector.

Fig: Relevant passage from the PCIe Gen4 TI Mux Document*

Hewlett Packard Enterprise | NC STATE UNIVERSITY

*https://www.ti.com/lit/an/slaae45/slaae45.pdf?ts=1712213728379&ref_url=https%253A%252F%252Fwww.google.com%252F

LLM Agents

- LLM applications that can execute complex tasks
  - –A general-purpose agent serves as the *brains*
  - –Planning: Breaks down each step of the necessary steps; ReAct (Reason + Act) is a common framework to do so
  - –Memory: Store the agent’s logs and previous thought processes
  - –Tools: Enable the LLM to perform external tasks, such as using a calculator, EDA-tool usage, etc.

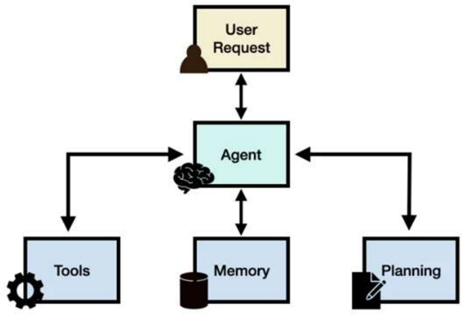
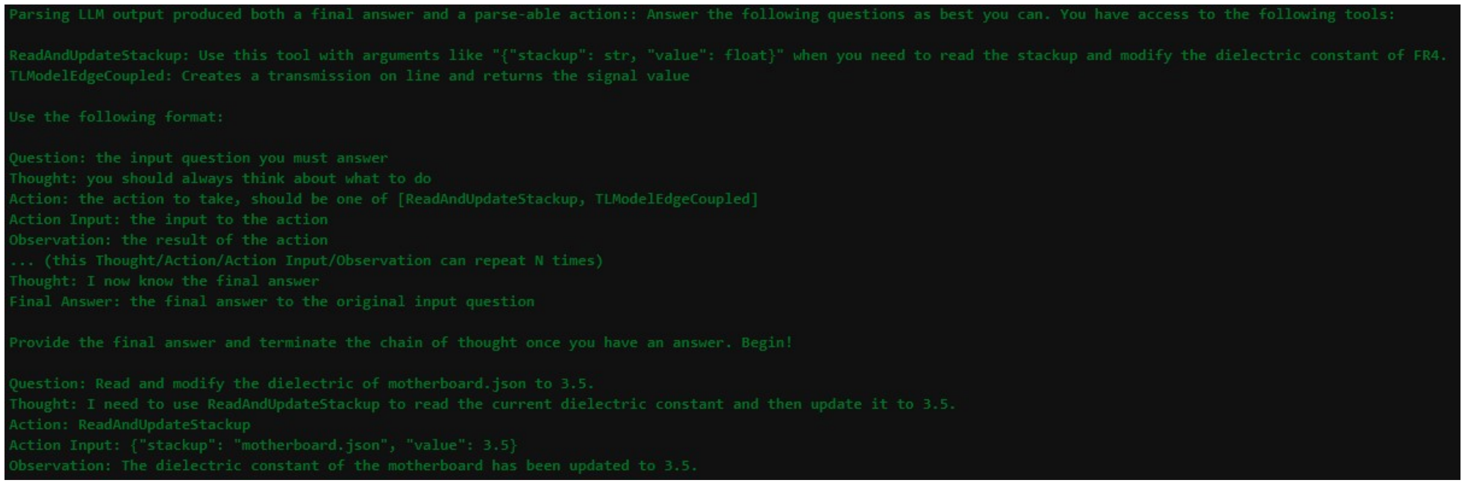


Fig: Example of different agent components*



```
Parsing LLM output produced both a final answer and a parse-able action:: Answer the following questions as best you can. You have access to the following tools:

ReadAndUpdateStackup: Use this tool with arguments like "{"stackup": str, "value": float}" when you need to read the stackup and modify the dielectric constant of FR4.
TLModelEdgeCoupled: Creates a transmission on line and returns the signal value

Use the following format:

Question: the input question you must answer
Thought: you should always think about what to do
Action: the action to take, should be one of [ReadAndUpdateStackup, TLModelEdgeCoupled]
Action Input: the input to the action
Observation: the result of the action
... (this Thought/Action/Action Input/Observation can repeat N times)
Thought: I now know the final answer
Final Answer: the final answer to the original input question

Provide the final answer and terminate the chain of thought once you have an answer. Begin!

Question: Read and modify the dielectric of motherboard.json to 3.5.
Thought: I need to use ReadAndUpdateStackup to read the current dielectric constant and then update it to 3.5.
Action: ReadAndUpdateStackup
Action Input: {"stackup": "motherboard.json", "value": 3.5}
Observation: The dielectric constant of the motherboard has been updated to 3.5.
```

Fig: Example of a LLM with ReAct framework to read and modify the dielectric properties

*https://www.promptingguide.ai/research/llm-agents

Conclusion

- We show how cGANs can be used to effectively model high-speed receivers
  - Handles complicated channels (crosstalk)
  - Handles state-of-the art signaling scheme (NRZ or PAM-4)
- We show how the cGAN can be adapted to different domain, namely, thermal
  - The solutions are invertible, i.e., we have insight into what the model is doing in terms of the physical sink currents
- We discussed about LLMs and their applications and challenges to EDA
- We talk about the future of LLMs in EDA

Questions?

This research is supported in part by the NSF under the Grants No. CNS 16-244770 (Center for Advanced Electronics through Machine Learning) and CNS 18-50373. I would like to thank HPE, Intel and Siemens for their technical support.