Two-stage framework for corner case stimuli generation Using Transformer and Reinforcement Learning

Chung-An Wang, Chiao-Hua Tseng, Chia-Cheng Tsai, Tung-Yu Lee, Yen-Her Chen,
Chien-Hsin Yeh, Chia-Shun Yeh, Chin-Tang Lai
MediaTek Inc., Hsinchu, Taiwan
{CA.Wang, Imogen.Tseng, Joseph.Tsai, Edison.Lee, Dale.Chen, Chien.Yeh, Jason.Yeh,
citi.lai}@mediatek.com

***Abstract*- Constrained-Random Verification (CRV) is a common method to achieve full function coverage by generating test cases randomly with some meaningful constrained setting. Some corner cases verification relies on a deep understanding of the DUT and component design from experts. The constrained creation and tuning are usually the time-consuming tasks. Furthermore, some of the corner cases are the "difficult" corner cases, which means the targets are hard to be hit even if experts tune the constraints many times. We deal with the difficult corner case verification of FIFO full condition in this paper. The two-stage process was introduced. The framework combines the concept of CRV and machine learning, including attention mechanism and Reinforcement Learning (RL). Our experiment results demonstrate that we improve the hit rate up to 380x of corner case for FIFO verification.**

I. INTRODUCTION

Constrained-Random Verification (CRV) generates numerous stimulus based on the constrained setting by expert. The generated stimulus are used to verify performance of Design Under Test (DUT) in the testbench [1]. As the chip design becomes more and more complicated in recent years, the functional verification tasks also become more challenging. Some corner conditions are very difficult to be verified by CRV within the complex design. Hence, the coverage closure of functional verification require more computing resources and domain knowledge to complete the verification task.

According to the analysis from the simulation results produced by numerous stimulus, it might be more efficiency on the coverage closure if we could utilize these stimulus and corresponding outputs to control the constrained settings. Thus, with the development of machine learning and deep learning, there are more and more researches about utilizing learning-based technique to learn from simulated data to accelerate the process of verification. For instance, [2] [3] train a supervised model by machine learning to select the effective stimulus.

In the past, the constrained range setting of different coverpoints would be given by experts based on their experience. Within these constrained range, [4] optimizes the constrained setting to improve the quality of stimulus. Furthermore, because some stimulus generation are involved in time series problem, the successively constrained settings also need to be considered. Such as [5], [6] use RL to generate a sequence of input settings based on the interactions between stimulus and DUT.

The advantages of the above methods are combined in this paper. We consider how to select an appropriate constrained sub-range then generate the successive constraints to produce the effective sequence of stimulus. Therefore, we design a two-stage framework to deal with these two problems, respectively. We leverage Transformer model to select constrained sub-range, then apply RL to generate the sequence of constraints based on the selected constraints. We take some FIFO full conditions in MMU (memory management unit) as the corner case examples. The background of our problem and machine learning are introduced at next section.

II. BACKGROUND

Problem background. The Memory Management Unit (MMU) is a common design in System on Chip (SoC) and it usually contains many FIFOs in the design. A well-known design verification corner case nowadays is to verify functional operations is working or not when the stack is full of specific FIFO. Since the behavior of FIFO include PUSH and POP, and the influence to these two mechanisms depends on many factors of other components responses. It's hard to control the FIFO directly to fill up the stack of FIFO. The input address is one of the dominating factor to effect FIFO PUSH and POP. So we design a set of constraints to control the alternation of input address. The problem is how to determine a sequence of constraints to trigger FIFO-PUSH as possible as it could to

fill up the stack. We use multiple deep learning techniques to automatically generate the stimulus for increasing the probability of difficult corner case hit.

Machine learning (ML). Machine learning is an algorithm based on the experience as sample data to build the model in order to make the prediction of unseen data. It's also a kind of artificial intelligence. Supervised learning is a type of machine learning. It must contain the input X and the corresponding label Y as training data. The model learns the pattern between training pairs (X, Y) according to different data. And the model is optimized by the objective function.

Transformer. Transformer is one of the deep learning model especially effective for many natural language processing tasks. This model is also a sequence to sequence model (Seq2Seq) which means the input and output of model are both with sequence format, it usually consists of an encoder and a decoder. Hence, it's also a recurrent network, the model predicts an element in each step until the output becomes complete. As the paper title of model Transformer is "Attention Is All You Need" [7], it's obvious that the attention mechanism is the most important part of this model. Indeed, the encoder and decoder of this model are both build by Self-Attention blocks. The use of attention mechanism is to decide in each step which other parts of sequence is important or not.

Reinforcement Learning. Reinforcement Learning (RL) is another type of machine learning algorithm. The goal of RL is to train an agent which learns good behavior to earn highest rewards for specific environment. The agent always generates an action to interact with environment. And the environment returns a state and reward back to agent. The agent is according to the state and reward to determine the next action. Through continuous iterations to explore the environment, the agent can learn from different actions and the corresponding rewards to become a smarter agent ultimately.

A novel framework that integrates the above methodologies is proposed in our work. And our contributions are as follows:

1. We refactor the constrained range set from CRV approach to several discrete sub-range sets, and utilize the Self-Attention mechanism from Transformer to select the outstanding sub-range set.
2. We redesign the MMU verification problem to be similar to the RL setting in game-playing. After RL agent learns good patterns through trial and error, it is suitable for generating effective stimulus with the selected sub-range sets.
3. We combine the above two methodologies as two-stage verification framework. The reason is that determine a successive stimulus from large space is very difficult. But after the constrained setting space was reduced at the first stage, it becomes more possible to generate effective stimulus to speed up coverage closure.

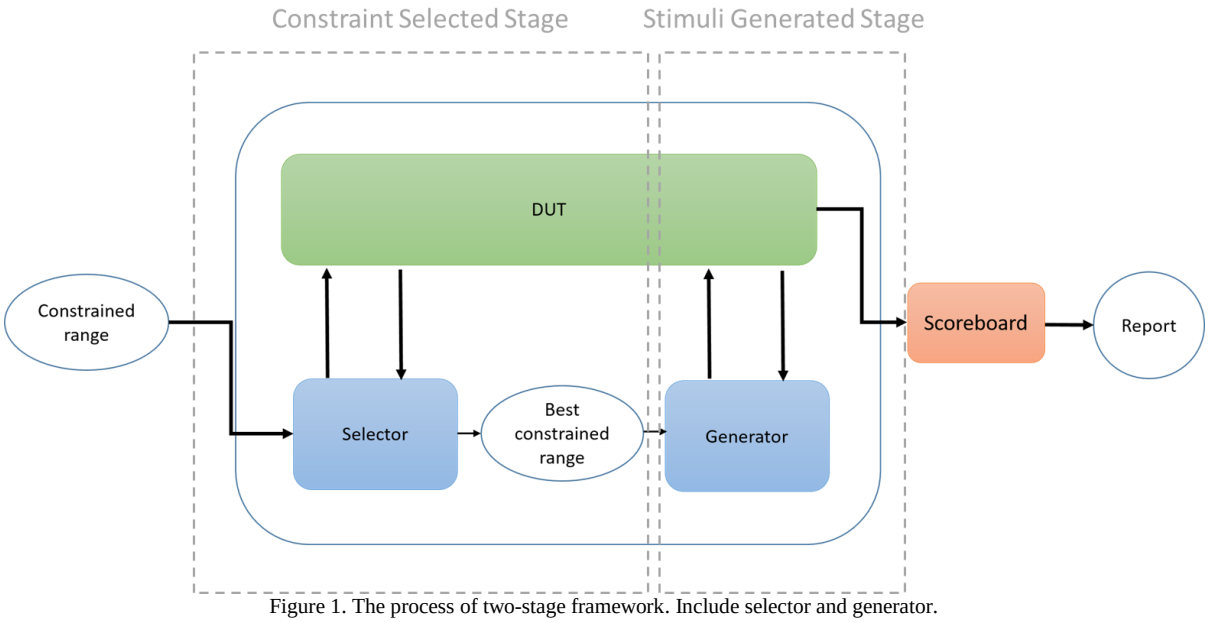

Figure 1. The process of two-stage framework. Include selector and generator.

III. APPROACH

This section introduces the two-stage framework which includes constraint selected stage and stimuli generated stage. At constraint selected stage, we present how we inspired by the module Transformer to build the selector model. And next, we describe the details of how the generator model is implemented with Reinforcement Learning.

A. Two-stage Framework

The two-stage framework is designed for the FIFO related verification to replace manual tuning of constraints and random generation of stimulus. As Fig.1 shows, after providing the constrained range settings for the specific FIFO to the selector, the selector is used to find the best sub-range set within the original constrained setting at constraint selected stage. Afterwards the generator starts to learn to generate the successive stimulus based on the best sub-range set at stimuli generated stage. The pseudo code of our two-stage framework is provided in Fig. 2 to show the implemented details.

B. Implementation of Constraint Selected Stage

In this stage, our purpose is to reduce huge searching space by selecting an appropriate range of constraints. Due to the characteristic of FIFO, it's hard to get enough records of FIFO full condition. To address this problem, we transform the target to the number of PUSH counts instead of FIFO is full or not. Through the observation of the relationship between FIFO PUSH and different combination of constraints, we refactor the initial constrained range setting to several discrete sub-range sets. Because more PUSH counts can make FIFO full easily, the selector chooses the one with highest PUSH counts as the best sub-range set. Hence, we have a better constrained setting space to learn how to generate stimuli in the second stage. Next, we describe the design of our selector and the detail of our model.

1. Selector: We process several times of following steps to select the best sub-range set. First, we perform the simulations for one of sub-range sets and get the number of PUSHs as the target variable for ML model. Next, we train the model with this sub-range set and calculate target value. After the model fits previous data, we can predict the number of PUSHs of residual sub-range sets and select the one with highest prediction value to perform next round of simulation. ML model can learn patterns between sub-range sets and know what kind of sub-range could trigger higher PUSH counts. Hence, we can get the best constrained range set in a few iterations instead of performing simulations for all the sub-range sets.
2. Transformer: Inspired by the module Transformer, especially the ability of encoder part to learn the semantic relation between the words in sentence. We utilize the main idea Self-Attention of encoder part to build our selector in order to learn the influence between constraints to select the outstanding sub-range sets effectively. Because all the sub-range set is ascending order, the positional index becomes very important. We also encode the position by the index from each sub-range set at the beginning to utilize the position information. As the Fig. 3(a) shows, there is a positional encoding before the neural network. And the main block of our selector is a Self-Attention block. At the end, we pass the output from self-attention block to a linear layer to predict the PUSH counts of simulation.

C. Implementation of Stimuli Generated Stage

After we receive the best sub-range set by first stage, we aim to generate the successive stimulus to fill up the stack of FIFO. As the successive stimulus are composed of a sequence of constraints. The problem is redesigned to be similar to the Reinforcement Learning setting in game-playing scenario. The generator learns to generate effective stimulus based on the selected sub-range sets and the DUT provides feedback to the generator. Next, we will define the State, Action and Reward in our RL model.

1. Action: There are total N cycles for each stimuli to be executed, where N is the length of stimuli. For each cycle, the generator selects a constraint based on the selected sub-range set from the first stage. Given T as the length of the one sub-range set, the action space will be T^N. For example, if the total constraints in the initial range set is the set {0, 1, 2, …, M-1}, the original action space will be M^N without selector. However, once we reduce the range and get smaller sub-range set $\{m_1, m_1+1, \ldots, m_2-1\}$, where $m_1 \geq 0$ and $m_2 \leq M$, the action space can be reduced to $(m_2-m_1)^N$.
2. State: The initial state is encoded as a list of 0 with the same length as the length of stimuli. Through each cycle, the state will be updated to the collection of past actions. For example, $state_n$ will be the collection of $action_0$, $action_1$, …, and $action_{(n-1)}$, where $n \in N$.

Algorithm 1: Two-stage Framework

Data: ConstraintRange//[0, T]
Result: StimuliHit

```
Env = Init(DUT);
Selector = Init(Transformer);
TrainingData = Init();
ResidualRanges = CreateAllRangeSet(ConstraintRange)//[0, 1], [0, 2], ..., [0, T], [1, 2], [1, 3],
  ... ;
for iter ← 0 to N do
    RangeSelected = Selector.SelectRange(ResidualRanges);
    ResidualRanges.Pop(RangeSelected);
    TrainingData.Add([RangeSelected, Env(RangeSelected.RandomGenerateStimuli())]);
    Selector.Fit(TrainingData);
BetterRange = TrainingData.X[argmax(TrainingData.Y)];
Generator = Init(RL, BetterRange);
TrainingData = Init();
StimuliHit = [ ];
while True do
    States = Init();
    Reward = Init();
    Stimuli = Generator.GenerateStimuli(States, Reward);
    Reward, Log = Env(Stimuli);
    TrainingData.Add([Stimuli, Reward]);
    Generator.Fit(TrainingData);
    if Log.HitTarget() then
        StimuliHit.Append(Stimuli);
```

Figure 2. The algorithm of two-stage framework.

3. Reward: The reward function is defined as the number of PUSHs because the more PUSH counts could have higher probability to fill up FIFO.
4. Environment: The environment could be any kind of DUT. In our study, we take a multi-media MMU as our environment.
5. Generator: The generator plays the role of "Agent" in RL. Generator learns how to generate the best action to maximize the reward earned from DUT. The interaction between generator and DUT as Fig. 3(b) shows is as follows:
    a. Generator generates the stimuli by RL Agent.
    b. Perform the simulation and acquire FIFO PUSH times as reward.
    c. Optimize Generator based on the action, state and the reward.

We apply Actor-Critic method [8] as our generator. The actor model will predict action of N cycles every iteration. The critic model is used to judge every action predicted by actor. We use PPO2 principle [9] as objective function to optimize these two models.

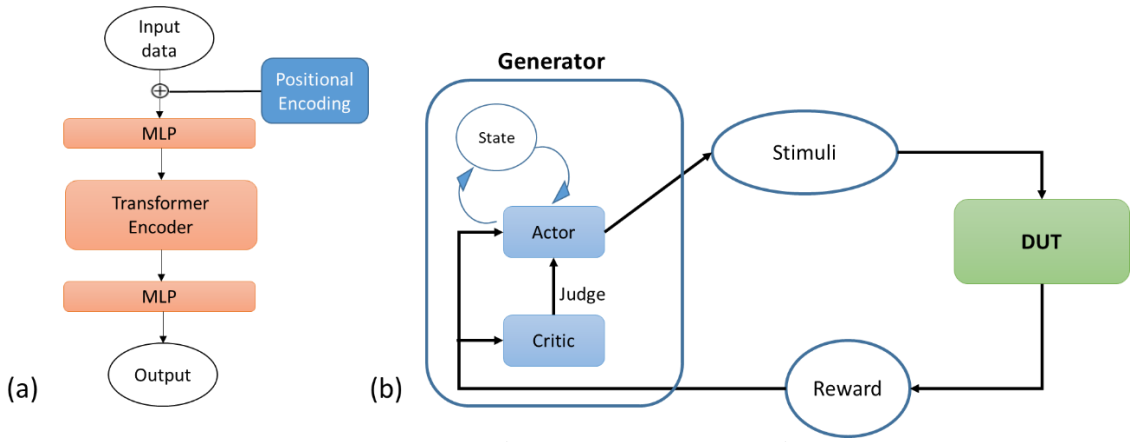

Figure 3. (a) The architecture of Selector. (b) The architecture of Generator.

IV. EXPERIMENTAL RESULTS

We use the multi-media MMU which includes 6 FIFOs {A, B, C, D, E, and F} in the design. The depth of full conditions for each FIFO is {A: 8, B: 16, C: 16, D: 16, E: 4, and F: 16}. Because the variety of FIFO condition is always dominated by input addresses, a set of constraints is designed to control the change of input address. We explain our results into two parts. First, we show the constrained Selector is high efficiency to find the best constrained range. Second, we compare the overall performance of two-stage framework with CRV approach.

D. Find the best constrained range by Selector

We separate a set of constrained setting into 120 sub-range sets. If we use the different sub-range to generate successive stimulus, the corresponding occupancy of FIFO is changeable. In order to choose the best occupancy of FIFO from all sub-range sets, the successive stimulus are generated within each sub-range set. We generate 25 stimulus randomly, because a few stimulus are not enough to represent the occupancy of FIFO. Then we sum up the total number of PUSHs as constrained Selector labels. After perform training and prediction alternately many times during constraint selected stage, the best sub-range set with the highest PUSH counts of FIFO is selected with high efficiency. To validate the ability and stability of Transformer model, we conduct 100 experiments with different initial weights of neural nets. The result is shown as Fig. 4. More than 70% of experiments show that the selector could find the best sub-range set within 10 iterations. Although the worst cases spent 15 iterations, it is still far less than 120 iterations if we want to find the best range without selector. This indicates that the selector only needs 375 simulation times rather than 3000 simulation times, which reduces approximately 10 times simulation resources.

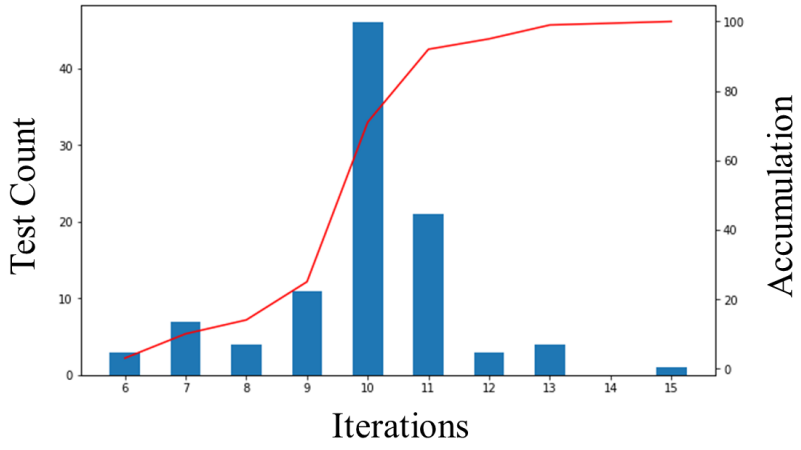

Figure 4. The distribution of the iteration count to find the best sub-range.

E. Comparison of Traditional CRV and Two-stage Framework

We compare the performance of our two-stage framework with traditional CRV, i.e. the stimulus are randomly generated based on the constrained settings we mentioned above. We performed 30,000 simulations on traditional CRV and 1,000 simulations on our approach. As Fig. 5 shows, the blue bars are the probability of FIFO depth reached by CRV, and the orange bars indicate the probability by our two-stage framework. Our approach clearly shifts the distribution to the right-side, which indicates we can increase the probability of occurrence of FIFO full.

As Table I shows, the hit rate of original CRV approach is extremely low, only one FIFO's hit rate exceeds 6%, the others are all less than 0.3%. Take FIFO A as an example. Through our two-stage framework, we increase hit rate from 6.2% to 87%, which is 14 times improvement. For the FIFO E and F which are very difficult to reach full condition (less than 0.02%), the improvements are even better. The proposed framework can increase hit rate about 380 and 202 times than CRV respectively.

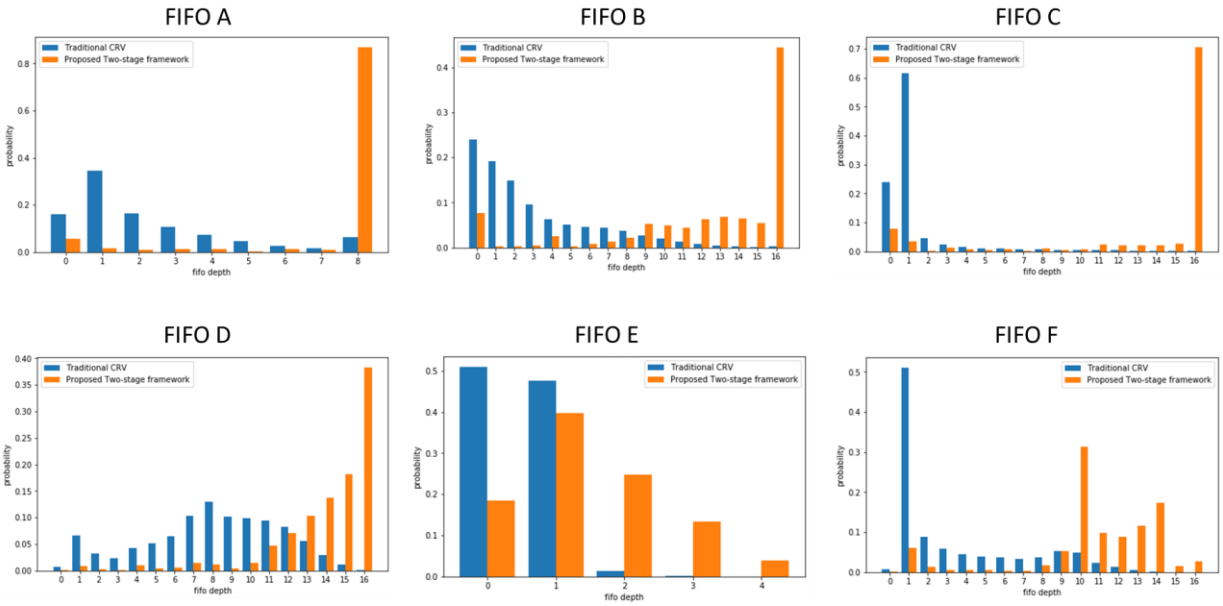

Figure 5. The probability distribution of FIFO depth for each FIFO.

TABLE I
EFFICIENCY OF TWO-STAGE FRAMEWORK

FIFO Name	A	B	C	D	E	F
Traditional CRV (hit times/simulation times)	1849/30000	83/30000	84/30000	57/30000	3/30000	4/30000
Proposed Two-stage framework	869/1000	443/1000	706/1000	383/1000	38/1000	27/1000
Hit rate (compare to traditional CRV) increase about x (times)	14	160	252	201	380	202

Finally, we calculate the total runtime of our approach. The average runtime per simulation is about 3 minutes. To perform 1,000 simulations, it costs about 3,000 minutes. As Fig.6 shows, to train selector and generator only costs about 7 minutes, it is about 0.2% of overall runtime. According to the above experiment results, our two-stage framework not only reduces a huge amount of simulations but also significantly increases hit rate for different kinds of FIFOs.

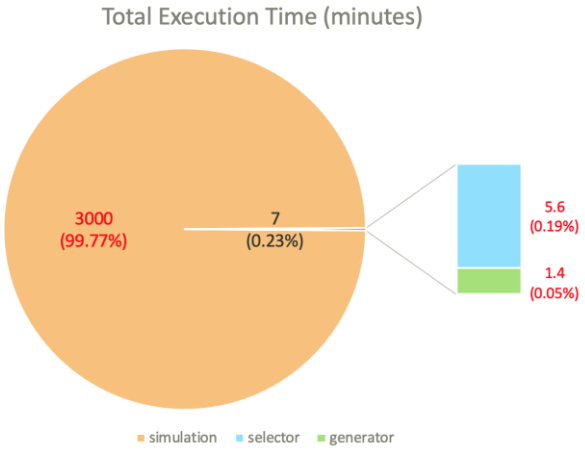

Figure 6. The distribution of system runtime.

V. CONCLUSION

In this paper, we propose a two-stage framework which applies supervised learning model Transformer and Reinforcement Learning methodology. As the results we demonstrated, we can significantly increase the hit rate of the corner case in design verification, the most improvement even achieves 380 times better than the traditional CRV. This impressive hit rate brings the benefits to optimize the verification quality and also shortens the time-to-market. The approach has already applied to the MMU product verification, and we believe this two-stage framework can also be adapted to the other designs.

REFERENCES

[1] J. Yuan, C. Pixley, A. Aziz, and K. Albin, "A framework for constrained functional verification," in ICCAD-2003. International Conference on Computer Aided Design (IEEE Cat. No. 03CH37486). IEEE, 2003, pp. 142-145.

[2] F. Wang, H. Zhu, P. Popli, Y. Xiao, P. Bodgan, and S. Nazarian, "Accelerating coverage directed test generation for functional verification: A neural network-based framework," in Proceedings of the 2018 on Great Lakes Symposium on VLSI, 2018, pp. 207-212.

[3] S. Gogri, J. Hu, A. Tyagi, M. Quinn, S. Ramachandran, F. Batool, and A. Jagadeesh. Machine Learning-Guided Stimulus Generation for Functional Verification, DVCON 2020.

[4] R. Roy, M.S. Benipal, S. Godil. Dynamically Optimized Test Generation Using Machine Learning, DVCON 2021.

[5] N. Pfeifer, B. V. Zimpel, G. A. G. Andrade and L. C. V. dos Santos, "A Reinforcement Learning Approach to Directed Test Generation for Shared Memory Verification," 2020 Design, Automation & Test in Europe Conference & Exhibition (DATE), 2020, pp. 538-543, doi: 10.23919/DATE48585.2020.9116198.

[6] William Hughes, Sandeep Srinivasan, Rohit Suvarna, and Maithilee Kulkarni. Optimizing design verification using machine learning: Doing better than random, 2019.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention Is All You Need, 2017.

[8] V. R. Konda and J. N. Tsitsiklis, "On Actor-Critic Algorithms," SIAM Journal on Control and Optimization, vol. 42, no. 4, pp. 1143–1166, 2003.

[9] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017.