Synthesis QoR Prediction

Ankur Sharma, Praharsha Mahurkar, Chinmay Samudra

Abstract

In the process of semiconductor chip development Logical Synthesis plays a very important role. It translates the behavioral description of the circuit (which is specified using HDLs) to boolean gate level netlists. In the prevalent industry flows, every critical **design IP is taken through multiple rounds of synthesis to determine the optimal gate level representation**. Each of these rounds make use of a different synthesis "recipe". Furthermore, these rounds are time consuming due to design complexities and large tool run times. This adds to the total design cycle time and affects the chip's time to market. We **explore** ML driven approaches to predict the Quality of Result (QoR) after using a new synthesis recipe on a design IP. This approach aims to reduce the run time of the multiple design iterations.

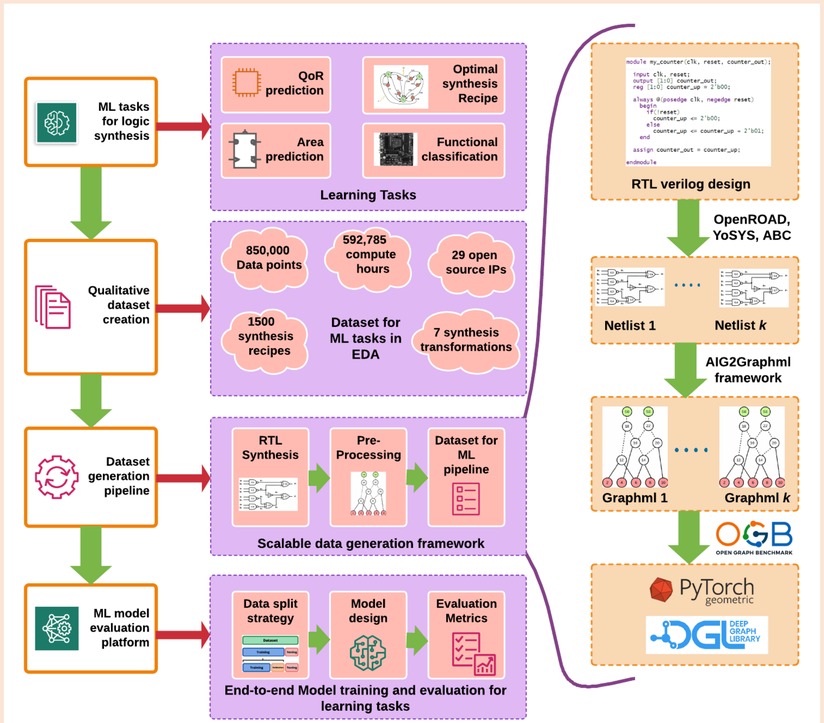

Figure 1. Pipeline : ML for EDA QoR Prediction

Methods

We explore multiple methods for prediction:

i. **Simple Linear Regression**: This regression model uses 4 numerical input features given in the first dataset (# of AND gates, # of NOT gates, IP length and Area). The design recipe and design name are encoded using one-hot-encoding as they are categorical features. As a result, our total input consists of 1544 features.

ii. **Neural Network based regression**: we tried multiple architectures with multiple hidden layers and neurons for this task, however all suffer from vanishing gradient problem. We used the first dataset for these trials. We then tried batch normalization, ReLU and Leaky ReLU, and various mini batch size iterations but to no avail.

iii. **Graph Convolutional Network:** we use a GNN that is comprised of 2 graph convolution layers that expect adjacency matrices (derived from a node encoder model). This then goes through batch normalization layers and ReLU to give us an encoding of the graph. Post this step, the model flows as depicted in the image below. We're using Adam Optimizer here.

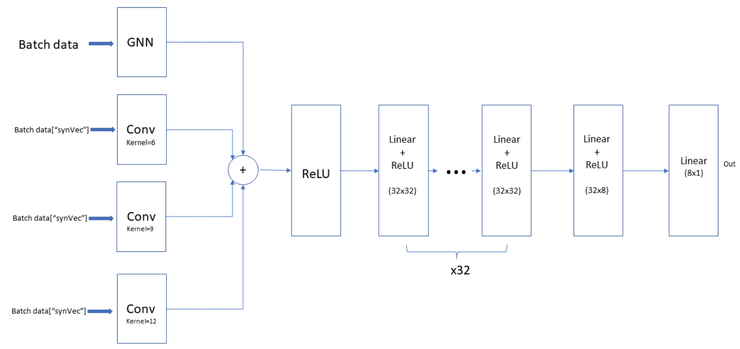

Figure 4. GCN Network

Results Discussion

The results for our methods:

i. **Simple Linear Regression**: This method gives us an accuracy of 88.59%. The threshold considered for this accuracy is 10ps. The plot given below depicts the difference of the predicted and expected outputs for the test datapoints. It is seen that majority of the outputs lie within 25ps of the expected ones.

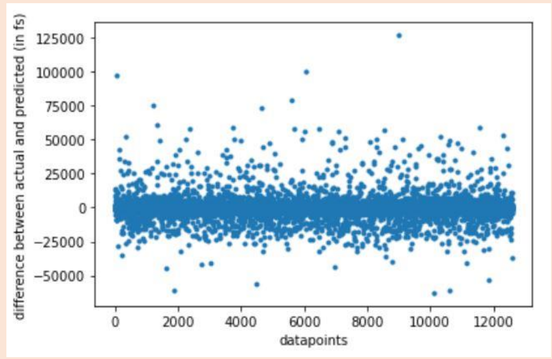

Figure 5. Linear Regression Predicted – Expected output

ii. **Neural Network based regression**: Vanishing gradient causes our network output accuracy to be very low. We see around 30% accuracy.

iii. **Graph Convolutional Network:** We see an MSE of 1.32 after a couple of epochs. Complete training still remains.

In all of the approaches, we have used the MSE loss, which is depicted in the snippet below.

.

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Figure 6. MSE Loss

Dataset explanation

We used the dataset from NYU-MLDA's OpenABC-D. Two types of dataset representation:

1. Total AND gate count, total not gate count, IP max length, Total Area, Delay
2. And-Invert-Graph Representation using Nodes and Edges.

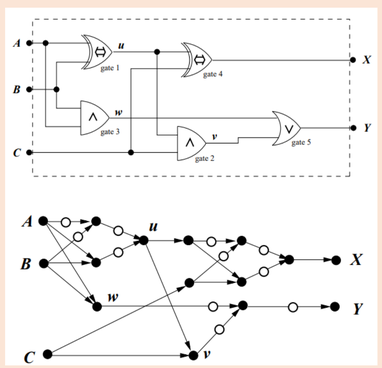

Figure 2. AND-INVERT Graphs

	design	recipe	ANDgates	NOTgates	ipLen	area	delay
0	Rocket_csr	0	6671	7234	22	4758.21	3369.65
1	Rocket_csr	1	6689	7159	19	4923.93	6041.40
2	Rocket_csr	2	6632	6791	21	5051.87	2811.36
3	Rocket_csr	3	6636	6813	21	5073.42	2920.17
4	Rocket_csr	4	6641	7132	20	4921.80	2879.31

Figure 3. Dataset

Future Work

- Immediate future work which is in the pipeline for us is the training of GCN. GCN data is 19GB and hence takes time to predict.
- Current NN model doesn't perform well due to vanishing gradient. Although multiple approaches have been tried to resolve it, many more remain.
- Current GCN as well as Linear Regression model only predict Delay. An extended model for the prediction of Power and Area could be conceptualized.
- The Open ABCD presents a more comprehensive dataset totaling 1.4TB. This can be trained for more accurate results as well more varied target variables.

References

[1] Animesh Basak Chowdhury, Benjamin Tan, Ramesh Karri, and Siddharth Garg. Openabc-d: A large-scale dataset for machine learning guided integrated circuit synthesis. arXiv preprint arXiv:2110.11292, 2021a.

[2] Animesh Basak Chowdhury, Benjamin Tan, Ramesh Karri, and Siddharth Garg. Openabc-d: A large-scale dataset for machine learning guided integrated circuit synthesis, 2021b.

[3] Animesh Basak Chowdhury, Benjamin Tan, Ryan Carey, Tushit Jain, Ramesh Karri, and Siddharth Garg. Too big to fail? active few-shot learning guided logic synthesis. arXiv preprint arXiv:2204.02368, 2022. 27–29.