Chip Design and Graph Representation

Neil Damle
ndamle@ucsd.edu

Wai Siu Lai
wslai@ucsd.edu

Pranav Rebala
prebala@ucsd.edu

Sahith Cherumandanda
scherumandanda@ucsd.edu

Lindsey Kostas
lkostas@qti.qualcomm.com

Abstract

Chip design optimizations have become increasingly challenging due to rising complexity, resulting in recent research towards machine learning methods and other techniques to improve the design process. One such example of a machine learning technique is the DE-HNN model, which uses a directed hyper-graph to improve the outcome of place-and-route tools, such as by modeling the congestion in a given netlist. We build upon the results and architecture of this model to identify specific features in a netlist that are highly related to congestion through exploratory analyses and the implementation of explainable AI tools, such as SHAP (SHapley Additive exPlanations) values. In addition, we propose new approaches to the partitioning process using causal analysis techniques. We also explore alternatives to the data processing step, such as undersampling and oversampling, to improve the model's performance on outliers. Finally, we consider reframing the model architecture as a classification problem to improve the identification of congested locations.

Code: `https://github.com/prebala123/chip-design`

1 Introduction

Chips are key components of many applications and tools, from mobile phones to self-driving cars. Chip design involves defining the product requirements for a chip's architecture and system, as well as the physical layout of the chip's individual circuits, a task which is becoming increasingly challenging as these technologies continue to develop. Due to Moore's Law, rising complexity is pushing the limits of existing chip design techniques, and machine learning offers a possible avenue to new progress. One specific area where chip designers face problems is congestion; often there are certain areas in a chip through which large amounts of information must pass, creating bottlenecks and reducing efficiency. Although electronic design automation tools have been useful to ensure scalability, reliability and time to market, our group aims to improve the process by exploring self-supervised learning techniques that will create useful features to learn effective graph representations to improve performance in predicting congestion and demand.

Given the relatively unexplored domain of machine learning for chip design, many recent attempts have tried to apply tools from different fields, which may not perfectly fit this specific use-case. For example, chip circuits are often represented as graphs in machine learning, and researchers must choose a specific kind of graph and the features within the graph. Our project aims to identify possible shortcomings in current circuit representations and suggest possible improvements, which will allow for deeper and more accurate research in the future.

2 Methods

2.1 Generating Baselines

Our main method of determining the performance of the model is using the F score, which combines precision and recall. Since our dataset has a large class imbalance, we decided that this was better than using accuracy as it tests whether the model trains to predict the less seen class. We used ten superblue chips as the training data, and kept one for validation and one for testing. First, we ran the base DE-HNN model for 200 epochs to get baseline results for how well the model performs with the original features from the original paper's work. Then we did an ablation study where we removed groups of correlated features like the persistence diagram or eigenvectors one at a time, to see how they would impact the model's performance. Our hypothesis was that if removing a feature caused the F score of the model to decrease by a larger amount than another feature, then it is more important to the congestion prediction.

2.2 Variational Auto-Encoding

Variational Graph Auto-Encoders (VGAEs) is an unsupervised technique for learning latent representations of graphs. We want to use it to incorporate node features into the embedding. Their probabilistic nature allows them to capture uncertainty and find any extra information such as "curvature" of the latent space. One of the goals of our project is to highlight areas where the current representation is lacking in order to design novel feature engineering using domain knowledge. We investigated how VGAEs can be effective at reconstructing graph connectivity (for instance we suspect parts of the netlist where the additional directional or physical features can play an important role) and also highlight areas where the current representation is lacking. Along the long quarter, our group discussed some hypotheses to understand what feature representations are lacking to improve the congestion classification. We suspect the lack of directional (driver to sink) information leads to latent embeddings that fail to capture timing critical paths, causing lower classification accuracy. We suspect the lack of directional (driver to sink) information leads to latent embeddings that fail to capture timing critical paths, causing lower classification accuracy. Also, missing detailed physical placement and timing information (something clocks can provide) causes sample classes with high congested regions to be misrepresented in the latent space. Our project aims to remedy this by comparing the performance of models between the baseline results from the DEHNN paper and results from including the augmented feature set. If adding directional features improves reconstruction quality or downstream task performance (e.g. reduced error in congestion prediction), this supports the first hypothesis. And if models that include physical layout and timing attributes give embeddings that cluster more clearly into high- and low-performance groups (e.g. achieve better regression/classification metrics), that supports the second hypothesis. Using the VGAE framework is a step towards understanding how to use an unsupervised learner for our graph structured data. The aim is to "denoise" and interpret the latent structure and by evaluating the latent clusters and comparing them to the DEHNN baseline performance outcomes, we hope to gain insights into which features are most indicative of performance — and therefore which aspects of the netlist representation need to be enhanced.

Our group aims to explore the method of a Variational Graph Auto-Encoder to learn a richer latent space in hope to improve downstream predictive because we hypothesize the current undirected hypergraph representation of netlists may omit key aspects such as directionality, physical layout, and timing constraints, as suggested by mentors. Augmenting the representation with directional and physical/timing information—and using a Variational Graph Auto-Encoder to learn a richer latent space, we find this approach helps in building more accurate models and also offer explainability into what design aspects most influence performance.

2.3 Partitioning

Partitioning this graph into substructures is critical for identifying groups of nodes that may represent functionally cohesive blocks within the chip. Such compartmentalization greatly aids in design analysis. Mathematically, the goal of partitioning the graph is to isolate regions of high internal connectivity. The quality of a given partitioning is evaluated through the conductance metric. For a partition S, conductance is defined as:

$$\phi(S) = \frac{\mathrm{cut}(S, \bar{S})}{\min\{\mathrm{vol}(S),\, \mathrm{vol}(\bar{S})\}},$$

where:

- $\mathrm{cut}(S, \bar{S})$ denotes the number of edges connecting nodes in S to nodes in its complement, $\bar{S}$,
- $\mathrm{vol}(S)$ is the total number of incident edges of the nodes in S.

A lower conductance value indicates that a partition is well-separated from the rest of the graph, suggesting a strong internal cohesion. In our approach, both the maximum and average conductance values across partitions are computed. The maximum conductance reflects the worst-case quality among all partitions, whereas the average provides an overall measure of separability.

We utilize a METIS-based partitioning approach, noted for its scalability and efficiency. Taking advantage of this, our initial partitioning experiments involved a grid search over two key parameters: ufactor (controls the allowable imbalance among partitions) and the number of partitions. For each combination of these parameters, we computed the corresponding maximum and average conductance values to assess partition quality. Based on our evaluation, we identified that using an ufactor of 600 with 10 partitions yielded the optimal balance for the global graph.

Observing that the global partitioning will hide finer substructures, we applied a hierarchical refinement strategy. Specifically, after partitioning the full graph using the optimal parameters, each of the 10 resulting subgraphs were independently subjected to the same parameter grid search. For each subgraph, conductance metrics were generated over the same range of parameter settings. This multi-level partitioning identified nested structures and provided a clear evaluation of the partition quality within each segment.

2.4 Persistence Diagrams

In the graph representation of the chip design, nodes correspond to components, while edges indicate connectivity. To capture the local structure around each component, we define a neighborhood based on physically nearby nodes and then summarize its overall shape using a topological summary called the persistence diagram. This summary, which records when structural features appear and disappear as the neighborhood expands, is transformed into a fixed-size image using a PersistenceImager. This uniform image serves as the feature vector for subsequent analysis.

Initial causal discovery analyses - utilizing the PC algorithm - revealed that one particular feature within these persistence images is strongly associated with chip demand. This feature appears repeatedly with high intensity across samples. To understand its origin, we backtracked through the persistence pipeline by mapping this pixel back to the original persistence diagram. This reverse mapping suggests that the critical feature aggregates topological elements that form very early and vanish quickly in the encoding process. We interpret these characteristics to represent some common local connectivity structure that may cause operational demand in the chip design.

2.5 Undersampling

During an exploratory analysis of the dataset, we found that the spread of the demand variable, which is what our model tries to predict for regression, was heavily skewed, with a very small proportion of nodes and nets having a high demand. We also conducted an error analysis on the baseline model, where we found that a significant amount of the total error stemmed from underperformance on the high-demand samples. More specifically, we realized that the model nearly always predicted a value between -1 and 1, despite the fact that there were samples which had demand greater than 10.

The approach we took to solving this problem was undersampling the dataset, which is a technique to balance the dataset and help the model generalize better across the range of possible values. Although there are a variety of undersampling algorithms, we chose to begin with a simpler approach to serve as an initial attempt to improve the performance on high-demand samples. We began by defining a set of bins (using arbitrary bounds which will require tuning in the coming weeks), and binning the dataset based on the demand values. From there, we randomly sampled each bin to ensure that they all had the same size, creating a much more balanced dataset that was approximately 20% the size of the original dataset. Given the graph structure of the dataset, we also had to filter the edges of the dataset to ensure they only contained nodes/nets that were included in the new dataset.

3 Results

Table 1 presents some summaries of the performance of our model.

Table 1: Model Results

Feature	Precision	Recall	F score
Baseline DE-HNN	0.7975	0.9968	0.8860
No Persistence Diagram	0.7976	0.9967	0.8861
No Eigenvectors	0.8003	0.9678	0.8760
No cell based features	0.8013	0.9863	0.8842

4 Contributions

- Neil: This quarter, my main task was developing the undersampling data pipeline in addition to generating benchmarks and baselines. This involved working with the data pre-processing steps to ensure that we were able to modify the dataset and make it balanced. I also performed EDA on the dataset, and modified the original code to identify high-error samples.
- Pranav: I mainly worked on modifying the code to make congestion classification predictions as well as demand regression, and I trained the model multiple times to get baseline results. I also trained the model while removing some groups of features to determine their importances. I also kept the repository up to date and documented so others can easily run the code.
- Sahith: I focused on systemically developing a better understanding of the chip design, and trying to extract features algorithmically. Much of my work dealt with different partitioning algorithms and tweaking the hyperparameters for optimal partitions. A more explorartory part of my work involved analyzing the persistence diagrams of the graph and applying causal discovery algorithms to them to try to extract the effects of certain substructures.
- Wai Siu Lai: Experiment with different feature representations, investigated using VGAEs and SHAP. Compare the baseline performance metrics with the augmented feature representation. Embed additional specific domain knowledge from mentors into the nodes and edges of dataset.

5 Appendix

- Previous Project Proposal
- ”DE-HNN: An effective neural model for Circuit Netlist representation. Z. Luo, T. Hy, P. Tabaghi, D. Koh, M. Defferrard, E. Rezaei, R. Carey, R. Davis, R. Jain and Y. Wang. 27th Intl. Conf. Artificial Intelligence and Statistics (AISTATS), 2024.” arxiv