DE-HNN: An effective neural model for Circuit Netlist representation

Zhishang Luo[1], Truong Son Hy[2], Puoya Tabaghi[1], Donghyeon Koh[4], Michael Defferrard[4], Elahe Rezaei[3], Ryan Carey[3], Rhett Davis[5], Rajeev Jain[3], Yusu Wang[1]

University of California San Diego [1] Indiana State University [2]
Qualcomm Technologies, Inc. [3] Qualcomm Wireless GmbH [4] North Carolina State University [5]

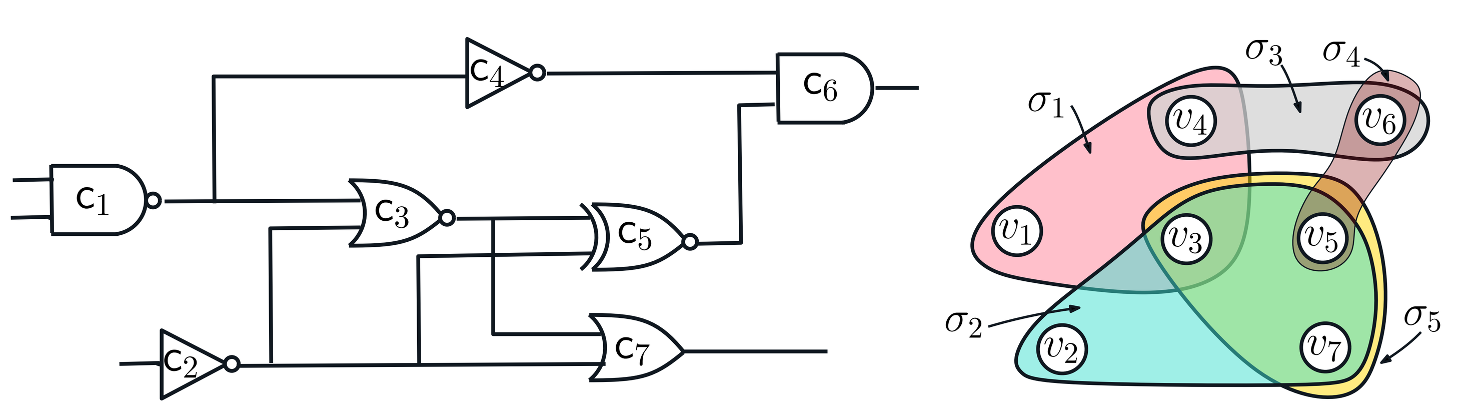
Motivation & Main Results

Chip design's growing complexity leads to the need of a machine learning model that can provide fast feedback. Such model's accuracy is affected by the representation of the design data, which is usually a "netlist" that describes the cells and nets and how they are connected in a design. Our works:

- We represent a netlist as a **directed hypergraph** to seperate the roles of driver and sinks cells.
- We propose a learning model **DE-HNN** for directed hypergraphs which can universally approximate any node or hyperedge based function that satisfy *equivariant* and *invariant* properties.
- We use a **hierarchy of virtual nodes (VNs)** to aid the learning of large-scale long-range interactions and a topological summary called **persistence diagram (PD)** to encode the "shape" of graph motif around each node.
- We compare our **DE-HNN** with several SOTA machine learning models for (hyper)graphs and netlists, and our model outperforms them in predicting properties of netlists.

DE-HNN: A Neural Network for Directed Hypergraphs

- A *netlist* $\mathcal{H}$ consists of a collection of **cells** (logic gates) $\mathcal{C} = \{c_1, \ldots, c_n\}$, and a set of **nets** $\mathcal{N} = \{\sigma_1, \ldots, \sigma_m\}$, see Figure (a) below.
- A *directed hypergraph* $\vec{H} = (V, \vec{\Sigma})$ has *directed hyperedge* $\sigma \in \vec{\Sigma}$ consists of an ordered pair $\sigma = (v_\sigma, S_\sigma)$ with $v_\sigma \in V$ and $S_\sigma \subseteq V$, see Figure (b) below.
- A *netlist* $\mathcal{H}$ thus can be represented as a *directed hypergraph* where we have: **cell** $\Leftrightarrow$ **node**, and **net** $\Leftrightarrow$ **directed hyperedge**.



(a) (b)

(a) A netlist $\mathcal{H}$ with 7 cells $\mathcal{C} = \{c_1, \ldots, c_7\}$ and 5 nets. For example, the output of gate c_2 flows into cells c_3, c_5, and c_7, giving rise to the net $\sigma = (c_2, \{c_3, c_5, c_7\})$. That is, the driver cell of σ is $\mathsf{v}_\sigma = c_2$, while its sink-set being $\mathsf{S}_\sigma = \{c_3, c_5, c_7\}$. (b) The corresponding directed hypergraph $\vec{H} = (V, \vec{\Sigma})$ with 7 nodes and 5 hyperedges $\vec{\Sigma} = \{\sigma_1, \ldots, \sigma_5\}$. Each node v_i corresponds to cell c_i, and each hyperedge is marked as a shaded region.

The input to our base-DE-HNN is the directed hypergraph $\vec{H} = (V, \vec{\Sigma})$ that represents the netlist $\mathcal{H}$. For the ℓ-th layer, base-DE-HNN will compute cell/node feature $m^\ell(v)$ and net/directed hyperedge feature $M^\ell(\sigma)$ as:

- Node Update:

$$m^\ell(v) = \mathrm{Agg}^\ell_{\sigma \to v}(\{\{M^{\ell-1}(\sigma')\}\}_{\sigma' \in \mathcal{I}(v)}) \tag{1}$$

Implementation:

$$m^\ell(v) = \sum_{\sigma' \in \mathcal{I}(v)} \mathrm{MLP}^\ell_1\big(M^{\ell-1}(\sigma')\big), \tag{2}$$

- Net Update:

$$M^\ell(\sigma) = \mathrm{Agg}^\ell_{v \to \sigma}(m^\ell(\mathsf{v}_\sigma), \{\{m^\ell(v')\}\}_{v' \in \mathsf{S}_\sigma}) \tag{3}$$

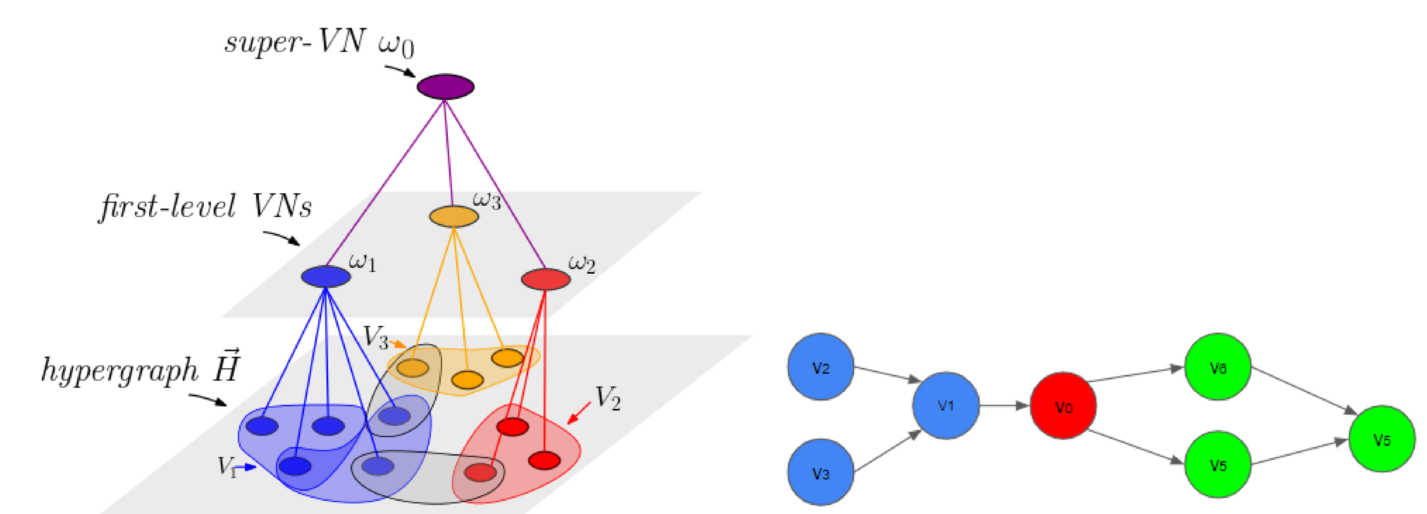
Implementation:

$$M^\ell(\sigma) = \mathrm{MLP}^\ell_3\Big[m^\ell(\mathsf{v}_\sigma) \oplus \Big(\sum_{v' \in \mathsf{S}_\sigma} \mathrm{MLP}^\ell_2(m^\ell(v'))\Big)\Big] \tag{4}$$

Augmenting Base DE-HNN to Full DE-HNN

We further augment our base-DE-HNN with following strategies to full-DE-HNN to capture long-range interactions and the multi-scale graph topology.

- Hierarchy of virtual nodes. A *virtual node (VN)* is an additional node we add that is connected to all input nodes. Adding a single VN effectively reduces the graph diameter to 2. In the case of large (hyper)graphs, we partition the node set and assign one local VN to each subset, and we add a global VN that connects all local VNs, see Figure (c).
- Positional and structural encodings. Besides Laplacian positional encoding, to capture the "shape" of the neighborhood, similar to to (Yan et al., 2021; Zhao et al., 2020), we use extended persistence diagram (PD) induced by shortest path distance function within the 6-hop directed neighborhood of each node. In Figure (d) it's a smaller example of 2-hop directed neighborhood.



(c) (d)

DE-HNN for Netlist Properties Predictions

We apply our base-DE-HNN and full-DE-HNN to tasks including Net-based wirelength regression, Net-based demand regression, and Cell-based congestion classification, similar to [1] and [2]. Tables below shows part of the single-design and cross-design empirical results, last row "Improvement" refers to the improvement of our full DE-HNN model over the best baseline for each metric.

Single Design

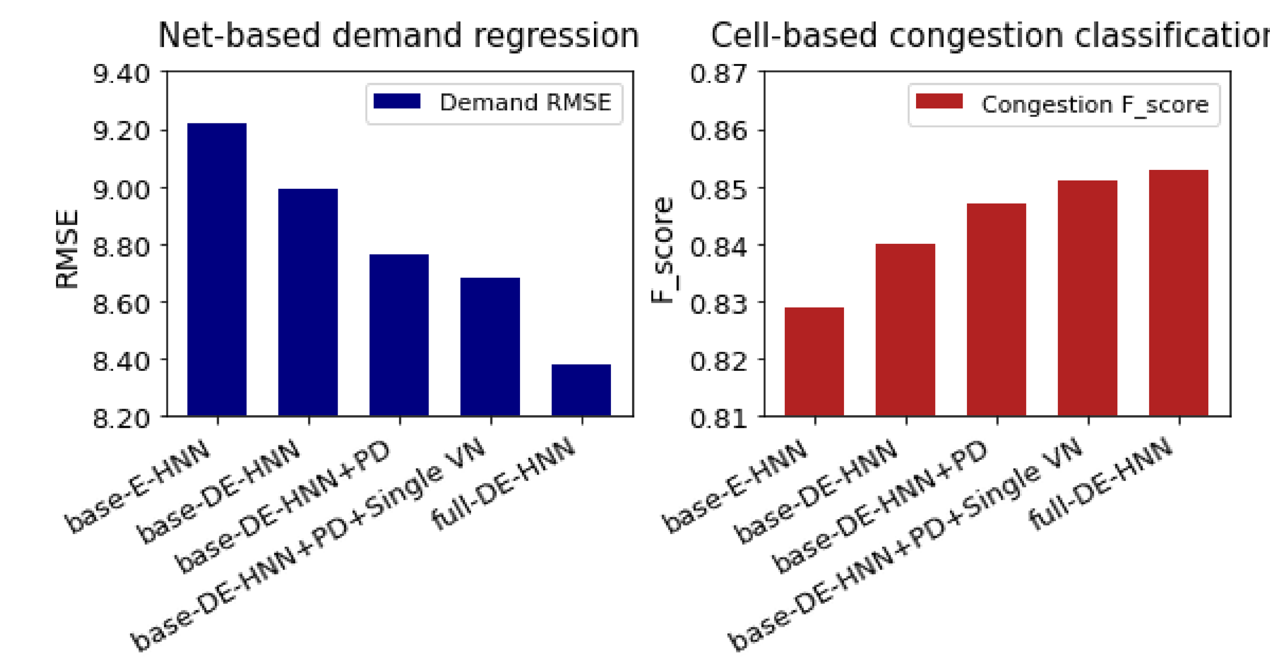
	net-based wirelength regression			net-based demand regression			cell-based congestion classification		
Model	**RMSE ↓**	**MAE ↓**	**Pearson ↑**	**RMSE ↓**	**MAE ↓**	**Pearson ↑**	**Precision ↑**	**Recall ↑**	**F$_{score}$↑**
GCN	1.762	1.276	0.750	9.321	6.163	0.570	0.761	0.857	0.802
GATv2	1.812	1.330	0.687	9.342	6.118	0.561	0.810	0.864	0.835
AllSet	1.718	1.264	0.760	9.072	5.745	0.632	0.782	0.837	0.804
HMPNN	1.841	1.368	0.710	9.342	6.118	0.561	0.774	0.826	0.792
HNHN	1.852	1.368	0.717	9.119	5.885	0.594	0.792	0.869	0.826
NetlistGNN	1.773	1.320	0.740	9.063	5.839	0.623	0.812	0.860	0.831
base DE-HNN	1.751	1.269	0.748	8.997	5.764	0.630	0.824	0.860	0.840
full DE-HNN	**1.689**	**1.245**	**0.770**	**8.381**	**5.334**	**0.683**	**0.833**	**0.876**	**0.853**
Improvement	**1.7%**	**1.6%**	**1.3%**	**7.5%**	**7.2%**	**8.1%**	**2.6%**	**0.8%**	**2.2%**

Cross Design

	net-based wirelength regression			net-based demand regression			cell-based congestion classification		
Model	**RMSE ↓**	**MAE ↓**	**Pearson ↑**	**RMSE ↓**	**MAE ↓**	**Pearson ↑**	**Precision ↑**	**Recall ↑**	**F$_{score}$↑**
GCN	1.691	1.276	0.746	6.571	5.024	0.365	0.633	0.997	0.773
GATv2	1.717	1.281	0.737	6.623	5.137	0.363	0.630	0.999	0.765
NetlistGNN	1.762	1.324	0.718	8.328	6.839	0.367	0.647	0.953	0.771
Allset	1.837	1.348	0.695	6.120	4.820	0.345	0.645	0.964	0.773
HMPNN	1.785	1.335	0.710	6.979	5.356	0.306	0.633	0.999	0.773
HNHN	1.754	1.333	0.701	6.390	4.870	0.358	0.648	0.939	0.767
base **DE-HNN**	1.731	1.291	0.730	6.778	5.085	0.337	0.653	0.990	0.774
full DE-HNN	1.677	1.242	0.754	6.037	4.670	0.372	0.660	0.986	0.780
Improvement	1.9%	2.6%	1.8%	1.4%	4.1%	1.4%	0.7%	-	0.3%

Ablation Study

We carried out an ablation study and compare the performance of the following versions: (a) base-E-HNN is similar to base-DE-HNNbut without direction. (b) base-DE-HNN is the base model for directed hypergraph with **neither** PDs **nor** VNs. (c) base-DE-HNN+PD is the base model with only PDs. (d) base-DE-HNN+PD+single VN is the base model with PD and a single global VN. (e) full-DE-HNN is our full model with PDs and a two-level hierarchy of VNs. The results for net-based demand regression and cell-based congestion classification are shown in Figure below.



Ablation study for net-based demand regression (left, RMSE) and cell-based congestion classification (right, F-score).

	net-based wirelength regression			net-based demand regression			cell-based congestion classification		
Model	**RMSE ↓**	**MAE ↓**	**Pearson ↑**	**RMSE ↓**	**MAE ↓**	**Pearson ↑**	**Precision ↑**	**Recall ↑**	**F_score ↑**
GCN with no PD	1.809	1.326	0.735	9.698	6.453	0.547	0.746	0.837	0.784
GCN+PD	1.762	1.276	0.750	9.321	6.163	0.570	0.761	0.857	0.802
Improvement	1.9%	3.6%	5.2%	3.9%	4.5%	4.2%	2.0%	2.4%	2.3%
GATv2 with no PD	1.920	1.401	0.659	9.710	6.392	0.539	0.802	0.856	0.811
GATv2+PD	1.812	1.330	0.687	9.342	6.118	0.561	0.810	0.864	0.835
Improvement	0.7%	0.6%	1.6%	3.8%	4.3%	4.1%	1.0%	1.0%	3.0%

Ablation Study: the effect of using persistence diagrams (PDs) to two baselines. For each method, the 3rd row shows the percentage of improvement after using PD as part of the input features. The results we reported are those baselines+PD.

Software

Our source code and netlists data used are publicly available. Scan barcode below or https://github.com/tilos-ai-institute/dehnn.

SCAN ME

Reference

[1] Yang et al., *Versatile multi-stage graph neural network for circuit representation.*, NeurIPS 2022.
[2] Wang et al., *Lhnn: Lattice hypergraph neural network for vlsi congestion prediction.*, IEEE DAC 2022.
[3] Chien et al., *You are allset: A multiset function framework for hypergraph neural networks.*, ICLR 2022.
[4] Dong et al., *Hnhn: Hypergraph networks with hyperedge neurons.*, Arxiv 2020.
[5] Heydari et al., *Message Passing Neural Networks for Hypergraphs.*, Springer Nature Switzerland 2022.