Graph Neural Network *for predicting drug-drug interactions*

Stephen Horn[1] (Stephen.Horn@unh.edu) (instructor, Laura Dietz[1])
[1]Department of Computer Science, University of New Hampshire, Durham, NH

CS752.01 / 852
Foundations of Neural Networks
Fall 2022

Abstract

We use a Graph Neural Network to predict drug-drug interactions.

The GNN model was trained from a large database of known interactions between FDA-approved and experimental drugs, encoded as an undirected Graph.

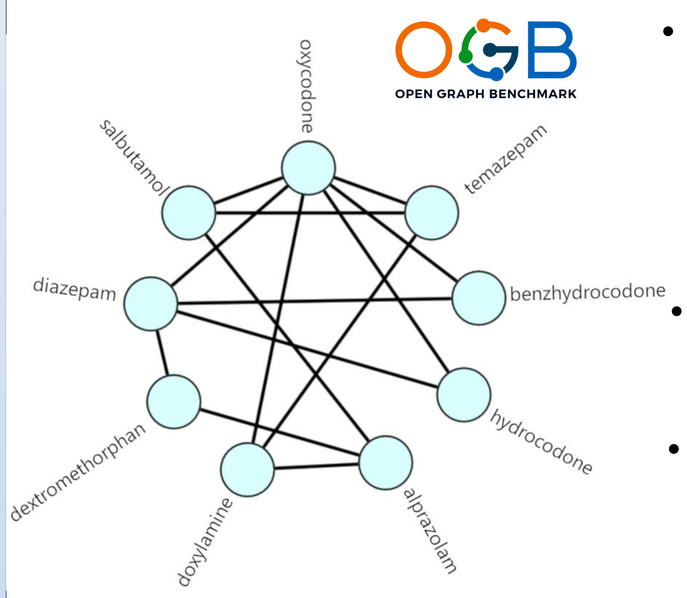
A Graph convolutional approach was used with GraphSAGE as the aggregator. We find that shallow aggregators with few layers outperform deep aggregators with many layers, whenever the total number of model parameters is held constant between them.

The Shallow Aggregator yields an accuracy of 0.586 (hits @K=60) on un-partitioned test split. Yields accuracy 0.786 +/- 0.01 on partitioned test splits (hits@K=770). The model size was 817 thousand parameters.

Background

GNN models are valued for graph induction. A new node is introduced to the Graph, and with it, several known edges. The GNN then predicts possibly new edges between any pre-existing node and the new node. For example, if a node for benzodiazepine is introduced into the Graph with edges indicating known drug-drug interactions, the model will predict perhaps other edges with a strong weight. Any non-trivial predictions guide decision-making in clinical trials, or flag contraindications for prescriptions. [3]

Prediction Task



- The `ogbl-ddi` dataset is an unweighted, undirected Graph G ={V,E}. Each node in V represents an FDA-approved or experimental drug. Each edge in E represents interactions between drugs : joint effect of taking the two drugs together is considerably different from the expected effect of the drugs taken independently.
- Task is to predict new drug-drug interactions given information on already known drug-drug interactions.
- 4267 drugs. 2.1 million interactions. From Open Graph Benchmark archive, hosted by Stanford. [1][2]

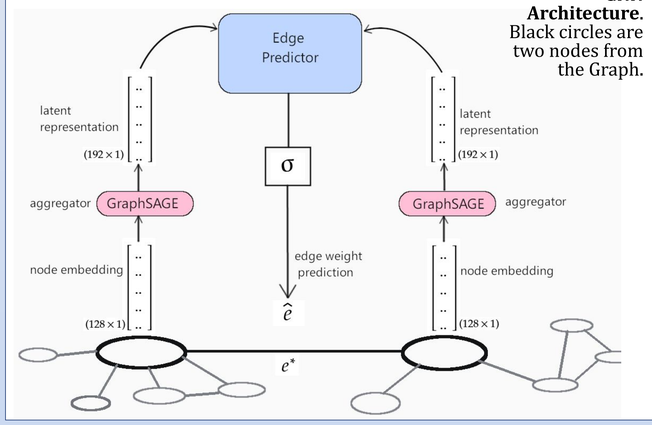
GNN layers

$$\mathbf{x}'_i = W_1\mathbf{x}_i + W_2 \cdot \mathbf{mean}_{n \in N(i)}\mathbf{x}_n$$

GraphSAGE layer
- with mean over neighbors

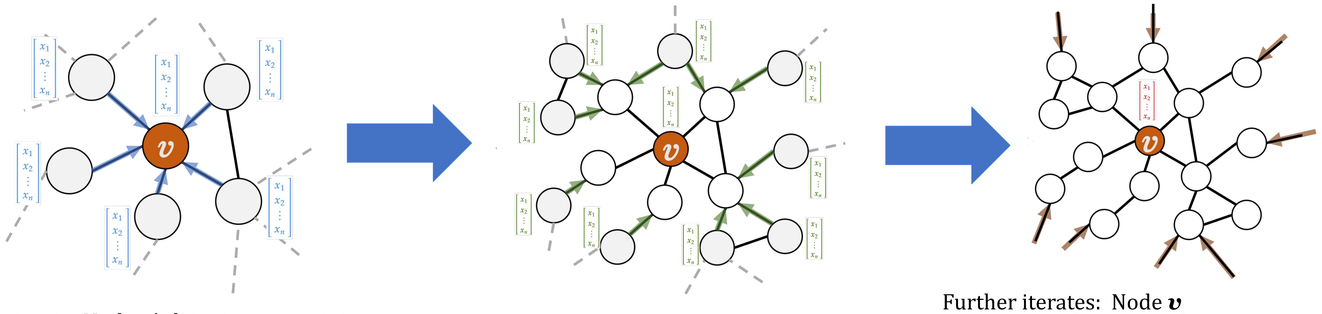
$$L_1(i,j) = \varphi\left(\left(t'_i \odot t'_j\right) W_1^{\mathsf{T}} + b_1 \right)$$
$$L_2 = \varphi\left(L_1 W_2^{\mathsf{T}} + b_2 \right)$$
$$\hat{e}_{i,j} = \sigma\left(L_2 W_o^{\mathsf{T}} + b_o \right)$$

Edge predictor
- ReLU activation
- sigmoid probability



GNN Architecture. Black circles are two nodes from the Graph.

Graph Convolution



First iterate: Node $\boldsymbol{v}$'s latent representation is itself and immediate neighbors.

2nd iterate: Node $\boldsymbol{v}$'s representation gains neighbors of neighbors.

Further iterates: Node $\boldsymbol{v}$ represents information from an ever larger section of the Graph.

Shallow and Deep Aggregators

Models	AGG *layers*	embedding params	AGG params	Edge predictor params	Total params
Shallow Aggregator	2	546,176	197,376	74305	817,857
Deep Aggregator	6	546,176	197,732	74305	818,213

Custom Deep , narrow aggregator,
- Increase aggregation layers to 6.
- Decrease hidden dimensions between them.
- same total model parameters.

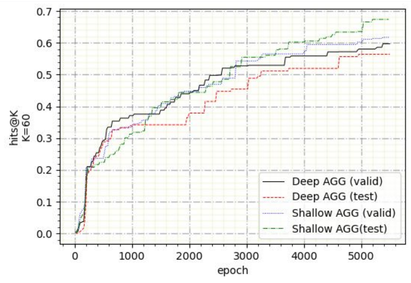
Algorithm 1: GraphSAGE algorithm

Input : Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$; input features $\{\mathbf{x}_v, \forall v \in \mathcal{V}\}$; depth K; weight matrices $\mathbf{W}^k, \forall k \in \{1, ..., K\}$; non-linearity σ; differentiable aggregator functions $\text{AGGREGATE}_k, \forall k \in \{1, ..., K\}$; neighborhood function $\mathcal{N} : v \to 2^{\mathcal{V}}$
Output : Vector representations $\mathbf{z}_v$ for all $v \in \mathcal{V}$

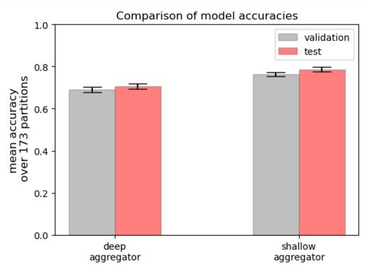
$\mathbf{h}_v^0 \leftarrow \mathbf{x}_v, \forall v \in \mathcal{V}$;
for $k = 1...K$ **do**
  for $v \in \mathcal{V}$ **do**
    $\mathbf{h}_{\mathcal{N}(v)}^k \leftarrow \text{AGGREGATE}_k(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\})$;
    $\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k)\right)$
  end
  $\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \|\mathbf{h}_v^k\|_2, \forall v \in \mathcal{V}$
end
$\mathbf{z}_v \leftarrow \mathbf{h}_v^K, \forall v \in \mathcal{V}$

depth K in the algorithm are recursive iterations which correspond to the layers of the aggregator. [4]

Experiments



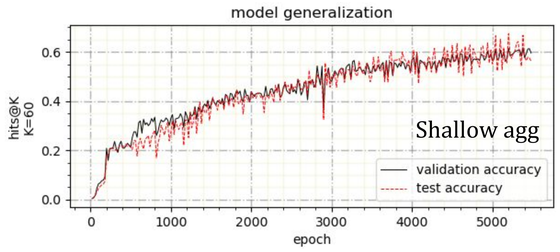
ACCURACY Best accuracy yet seen over timed course of training.

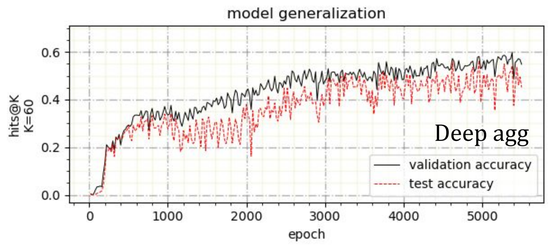


Shallow aggregator yields better accuracy over training and test sets than the Deep counterpart.

RELIABILITY Degree to which the validation accuracy is a good indicator of test accuracy.

- Shallow aggregator exhibited greater reliability over training.





Models	Validation set accu. (un-partitioned)	Test set accu. (un-partitioned)
Shallow Aggregator	0.603	0.586
Deep Aggregator	0.548	0.451

Accuracies over validation and test sets without partitioning.

	Validation set (partitioned)		Test set (partitioned)	
	Shallow Aggregator (base)	Deep Aggregator	Shallow Aggregator (base)	Deep Aggregator
arith mean	0.764 +/– 0.01	0.690 +/–0.01▼	0.786 +/–0.01	0.707 +/– 0.01▼
paired T-Test		7.8e–157		3.2e–156
significant?		TRUE		TRUE

The difference in accuracy over test and validation sets is significant with respect to a paired T-test with a confidence level of alpha=1% .
173 partitions hits@K=770.

Conclusion and Future Work

- State-of-the-art solutions on `obgl-ddi` use Graph Diffusion Networks with model sizes of 3.5 million parameters.[5]
- GNN here is 818 thousand parameters. Future work would increase model parameters and continue training beyond 5500 epochs.
- Discover whether these results hold.

References

➢ [1] https://ogb.stanford.edu/docs/linkprop/#ogbl-ddi
➢ [2] Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., ... & Leskovec, J. (2020). Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, *33*, 22118-22133.
➢ [3] Guney, E. (2017). Reproducible drug repurposing: When similarity does not suffice. In *Pacific Symposium on Biocomputing 2017* (pp. 132-143).
➢ [4] Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. *Advances in neural information processing systems*, *30*.
➢ [5] Sun, C., & Wu, G. (2020). Adaptive graph diffusion networks with hop-wise attention. *arXiv preprint arXiv:2012.15024*.