Prediction of Protein-Protein Interactions using Graph Convolutional Networks

Advisor: Ms.Fatemeh Salehi Rizi

Supervisor: Prof. Dr. Michael Granitzer

Akshat Sharma (87620),

Amit Manbansh (87622),

Lovesh Bishnoi (87738),

Mihir Shah (87568)

Agenda

1. Introduction

- Protein-Protein Interactions (PPIs): When two proteins interact with each other then that interaction between them is termed as a PPI.
- A set interacting protein pairs can be graphically represented.
- Machine Learning techniques can be used to solve the problems related to the graphical data.
- For example the classification problem in Zachary's karate club network [1].

Introduction (Zachary's karate club network [1])

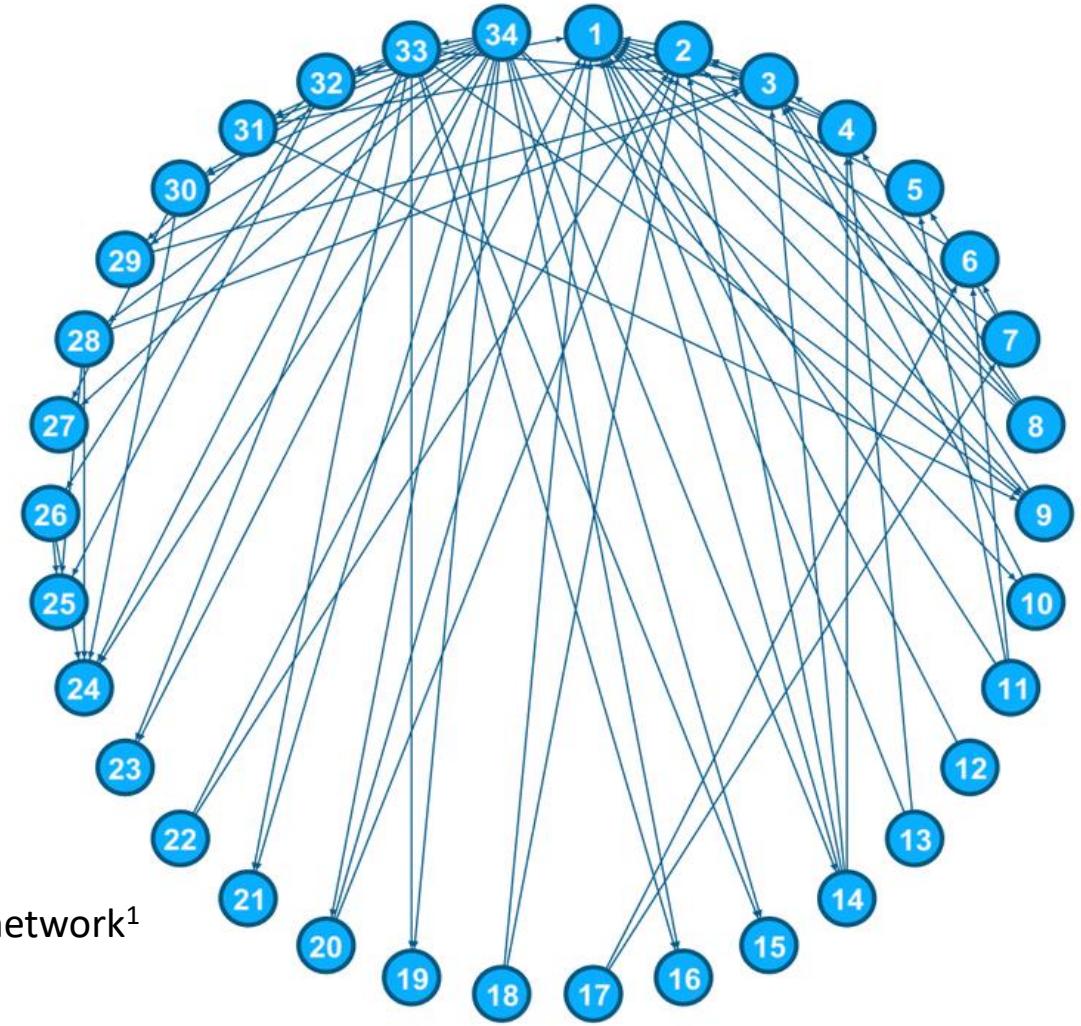


Figure 1: Zachary's karate club network[1]

[1] http://www.learndatasci.com/k-means-clustering-algorithms-python-intro/

2. Motivation

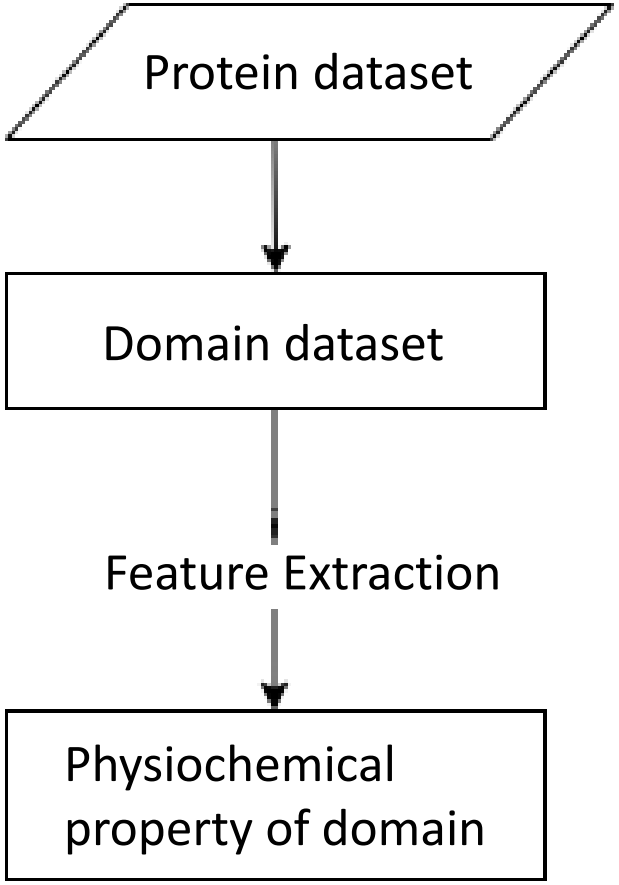
- Many undiscovered Protein interactions can be identified, computationally.
- These Protein interactions can be later verified experimentally thus requiring less human resource.
- Knowledge of protein interactions can help pharmacists and microbiologists understand the function and behaviour of protein.
- These new interactions can help the chemists and the biologists in the area of medicine.

3. Related Works

- Some of the works done in the Prediction of the Protein-Protein Interactions (PPIs) are Prediction of Protein-Protein Interactions Based on Domain by Xue Li et. al [2].
- Domain is the structural component of the protein, it could be an amino acid or pairs of amino acids.
- They used The Adhesome[1] as the Protein dataset and the Domain dataset was extracted from the protein dataset from Pfam database.[2]
- They used Support Vector Machine classifier to predict the PPIs based on the physiochemical properties of the domains.

[1] The Adhesome: A Focal Adhesion Network (http://www.adhesome.org/)
[2] Pfam database (http://pfam.xfam.org/)

Related Works (cont.)



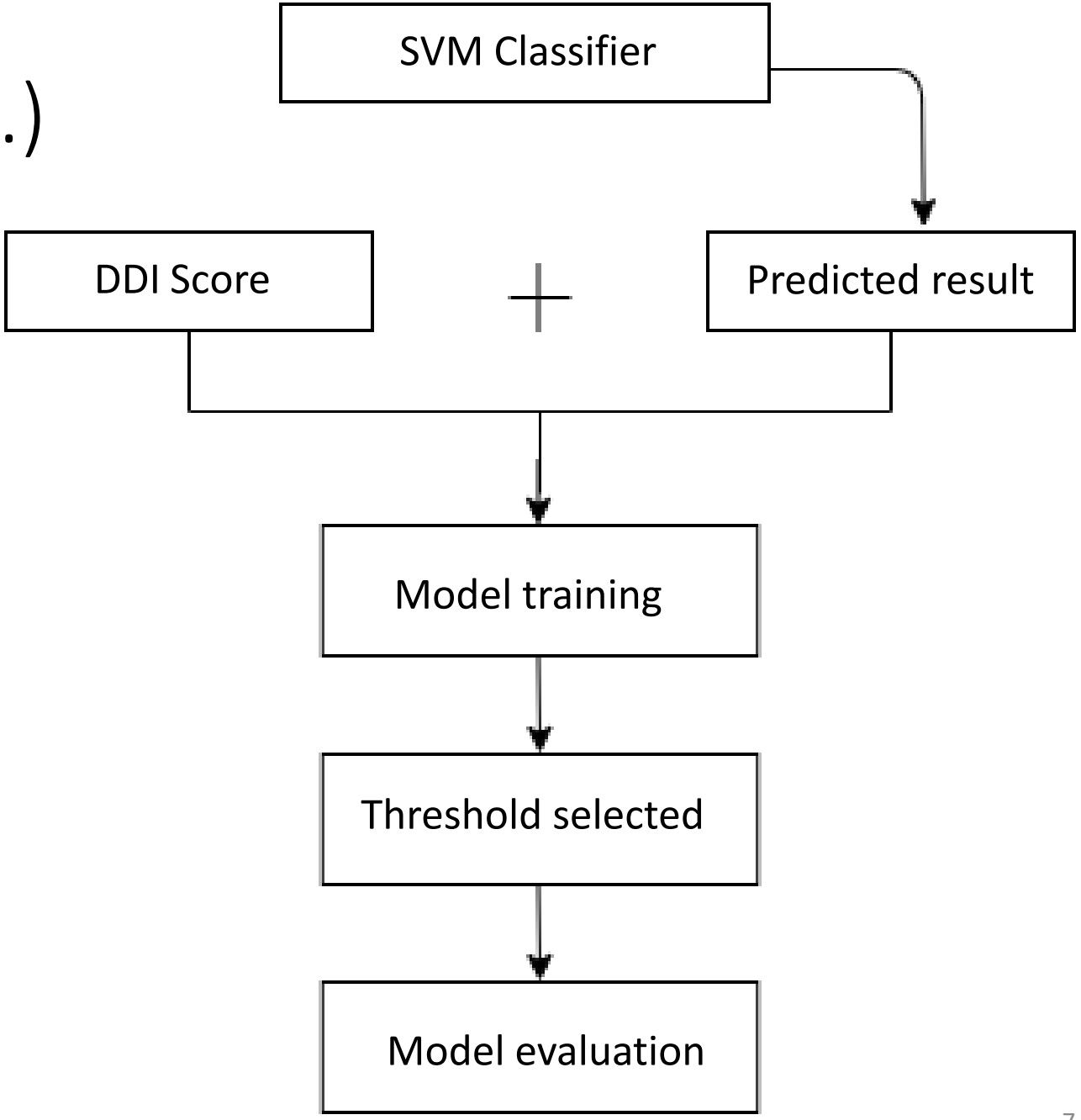


Figure 2: Prediction of Protein-Protein Interactions Based on Domain Flowchart [2]

4. Problem Statement

- What is the impact of hyperparameters on link prediction of PPIs in yeast dataset using Graph Convolutional Networks (GCN)?
- Mathematically, for a graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$, where V are nodes and E are edges, two nodes $\mathrm{x}, \mathrm{y} \subseteq \mathrm{V}$, we want to predict if $\mathrm{Edge}\{\mathrm{x}, \mathrm{y}\} \subseteq \mathrm{E} \mid \mathrm{Edge}\{\mathrm{x}, \mathrm{y}\} \not\subset \mathrm{E}$.
  1. How does increase in hidden-layers in GCN impact the accuracy in a link prediction task?
  2. What is the impact of various hyperparameters on the performance of GCN for a link prediction task?

5. Methodology

Graph Convolutional Network (GCN)

Let $\mathbf{G}$ be a graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ which is summarised in an Input Feature Matrix $\mathbf{X}^{(\mathbf{N}\times\mathbf{C})}$ to the GCN which produces an Output node level feature Matrix $\mathbf{Z}^{(\mathbf{N}\times\mathbf{C})}$, where

- **V** – Nodes
- **E** – Edges
- **N** – Number of nodes in **G**
- **C** – High dimensional node features
- **F** – Reduced number of node features
- $\mathbf{C} > \mathbf{F}$

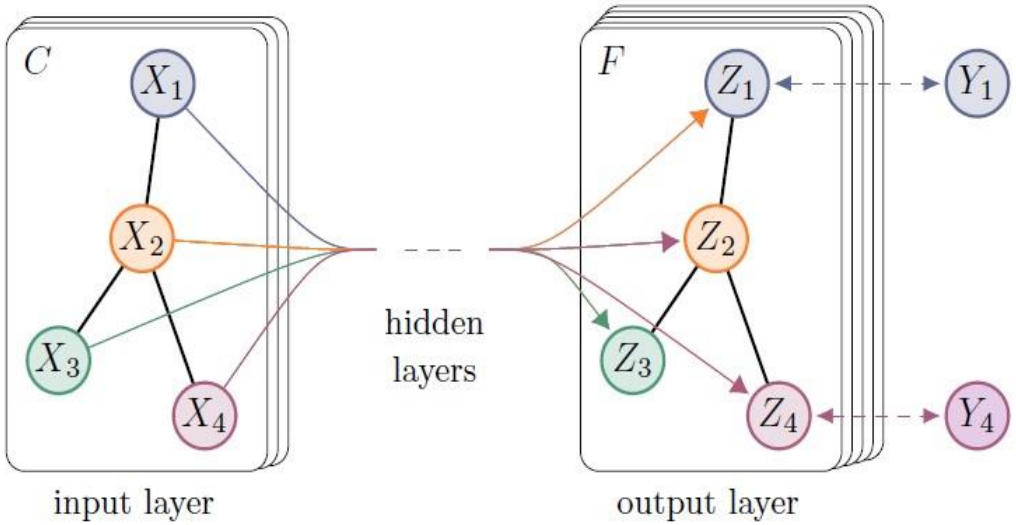


Figure 3: Graph Convolutional Network [3]

Mathematically each GCN layer can be represented as:

$$f(\mathbf{H}^{(l)}, \mathbf{A}) = \sigma(\mathbf{A}\mathbf{H}^{(l)}\mathbf{W}^{(l)})$$

Where,

- $\mathbf{H}^{(0)} = \mathbf{X}$
- $\mathbf{H}^{(L)} = \mathbf{Z}$
- $\mathbf{L}$ – Number of layers.
- $\mathbf{A}$ – Adjacency Matrix of the graph structure.
- σ – Activation Function

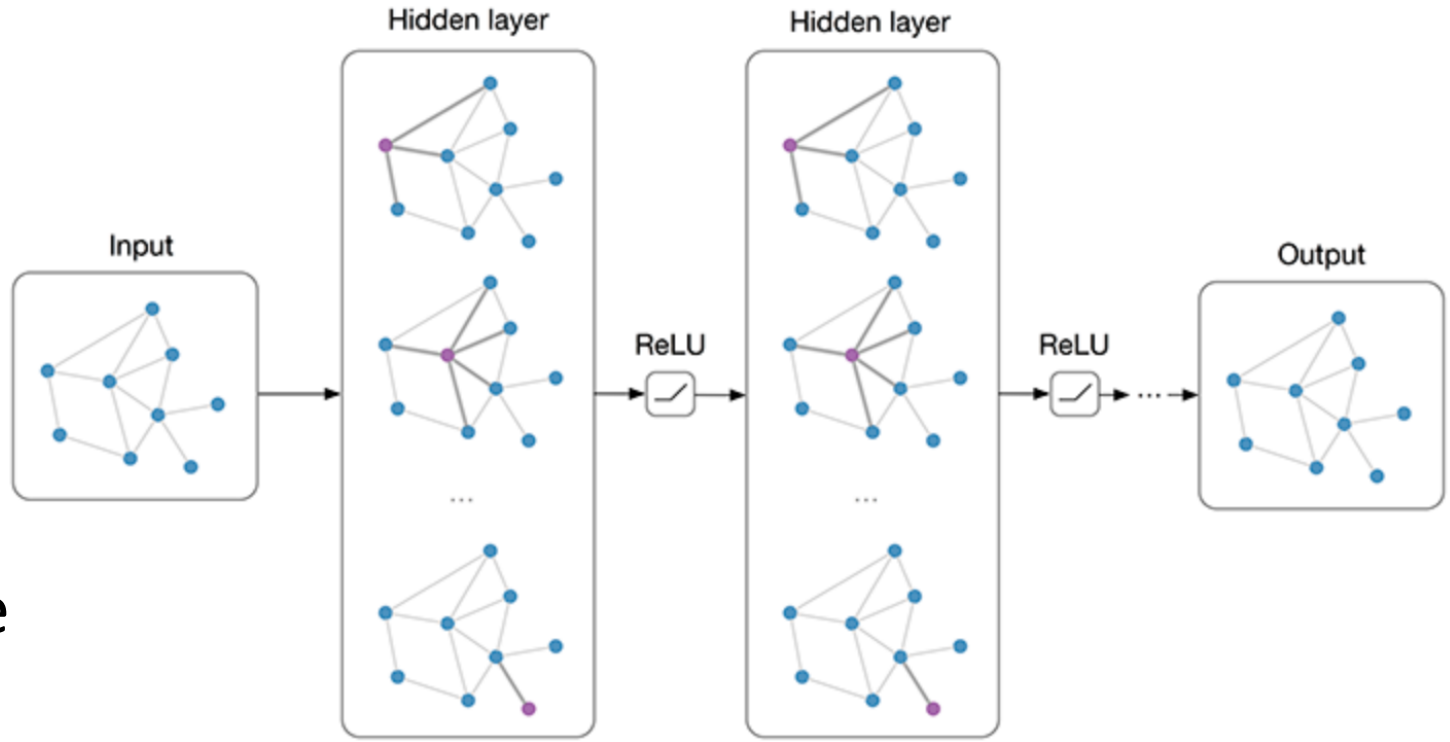


Figure 4: GCN Layers[3]

[3] https://tkipf.github.io/graph-convolutional-networks/

5.1 Few Limitations of this Equation

- Aggregated representation of a node does not include its own features.
- Larger in-degree nodes have large value in the aggregated representation when compared to comparatively smaller in-degree nodes.

Solving First Problem:

Aggregated representation of a node does not include its own features

Consider a simple GCN Model with the following graph as input:

$$f(\mathbf{H}^{(l)}, \mathbf{A}) = \sigma(\mathbf{A}\mathbf{H}^{(l)}\mathbf{W}^{(l)})$$

- Choose $\mathbf{W}^{(l)}$ such that $f(\mathbf{H}^{(l)}, \mathbf{A}) = \sigma(\mathbf{A}\mathbf{H}^{(l)})$,
- Let σ be an identity function, i.e., $f(\mathbf{H}^{(l)}, \mathbf{A}) = \mathbf{A}\mathbf{H}^{(l)}$

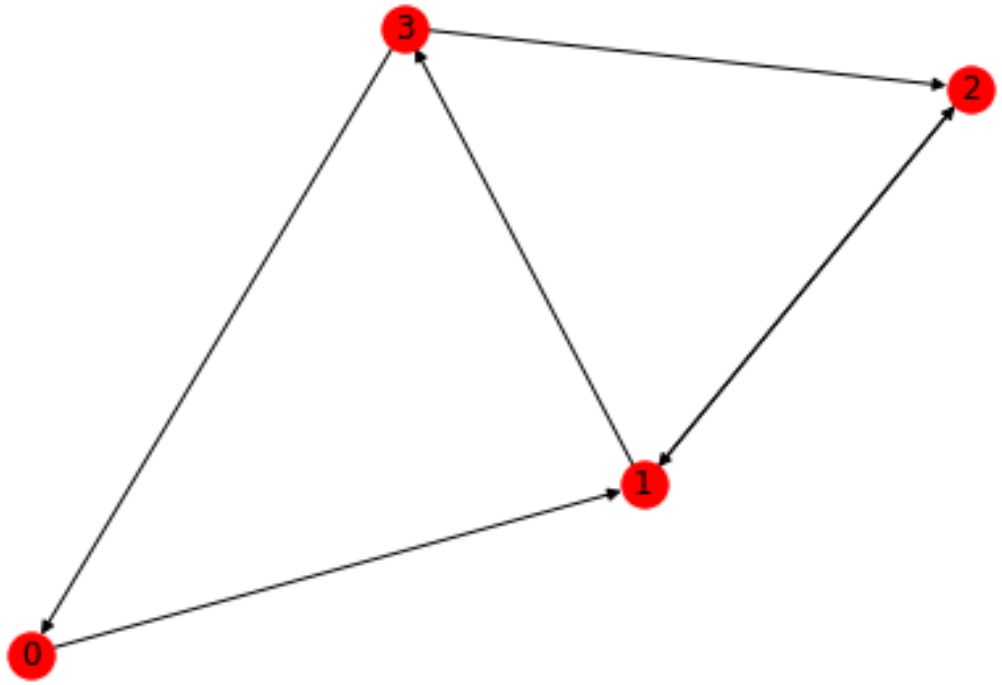


Figure 5: Graph

Adjacency Matrix $\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix}$, Node Feature Matrix $\mathbf{X} = \begin{bmatrix} 0 & 0 \\ 1 & -1 \\ 2 & -2 \\ 3 & -3 \end{bmatrix}$

Matrix Multiplication before using self loops

- $\mathbf{A} \times \mathbf{X} = \begin{bmatrix} 1 & -1 \\ 5 & -5 \\ 1 & -1 \\ 2 & -2 \end{bmatrix}$

Here we can see that the node is not considering its own features.

Matrix Multiplication after using self loops

- $\widehat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$
- $\widehat{\mathbf{A}} \times \mathbf{X} = \begin{bmatrix} 1 & -1 \\ 6 & -6 \\ 3 & -3 \\ 5 & -5 \end{bmatrix}$

Solving Second Problem:

Larger in-degree nodes have large value in the aggregated representation when compared to comparatively smaller in-degree nodes

- Normalize the Adjacency Matrix $\mathbf{A}$
- $\mathrm{f}\left(\mathbf{H}^{(\mathrm{l})}, \mathbf{A}\right) = \sigma(\widehat{\mathbf{D}}^{-\frac{1}{2}}\widehat{\mathbf{A}}\,\widehat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(\mathrm{l})}\mathbf{W}^{(\mathrm{l})})$
- Where, $\widehat{\mathbf{D}}$ is the diagonal node degree matrix of $\widehat{\mathbf{A}}$ and $\widehat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$
- Where $\widehat{\mathbf{D}}^{-\frac{1}{2}}\widehat{\mathbf{A}}\,\widehat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(\mathrm{l})}\mathbf{W}^{(\mathrm{l})} = \begin{bmatrix} 0.70710678 & -0.70710678 \\ 3.12132034 & -3.12132034 \\ 0.5 & -0.5 \\ 1.41421356 & -1.41421356 \end{bmatrix}$

5.2 Dataset

Data Format

```
YNL236W YGL238W
YNL236W YOR355W
YNL236W YJL030W
YNL236W YJL013C
YNL236W YJR034W
YNL236W YKL012W
YNL236W YFR033C
YNL236W YGR046W
YNL236W YGR117C
YGL208W YGL115W
YDR328C YLR399C
YDR328C YFL009W
YDR328C YMR094W
YDR328C YJR090C
YDR328C YIL046W
YDR328C YDR139C
YDR328C YOR057W
```

Figure 6: Yeast Edgelist[4]

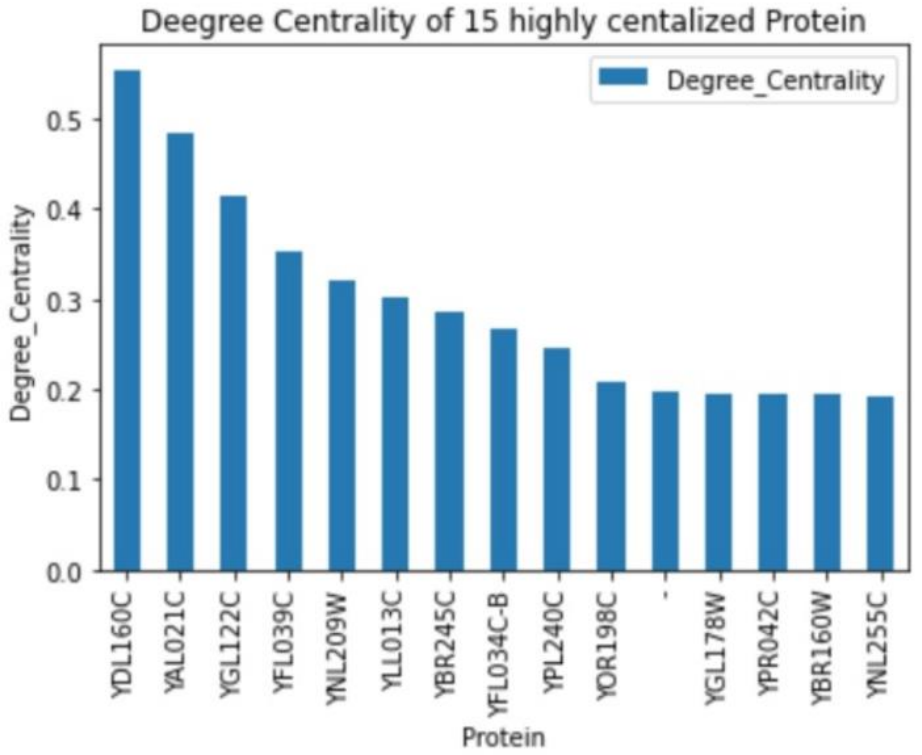


Figure 7: Degree Centrality

Data Statistics

Total Nodes	6526
Total Edges	1062675
Total Nodes after removing invalid node "-"	6525
Total Edges after removing invalid node "-"	1060093
Total Edges after removing diagonal elements	1058408
Total edges in Upper Triangle of Adjacency Matrix	529204

Figure 8: Data Statistics

[4] http://snap.stanford.edu/deepnetbio-ismb/ipynb/

5.3 Data Preprocessing

- We split up our dataset into six categories as.

Training Positive Edges	Training Negative Edges
Test Positive Edges	Test Negative Edges
Validation Positive Edges	Validation Negative Edges

- Positive Edges were determined when there existed an edge between two nodes.
- Negative Edges were determined when there existed no edge between tow nodes.

5.4 Model Architecture

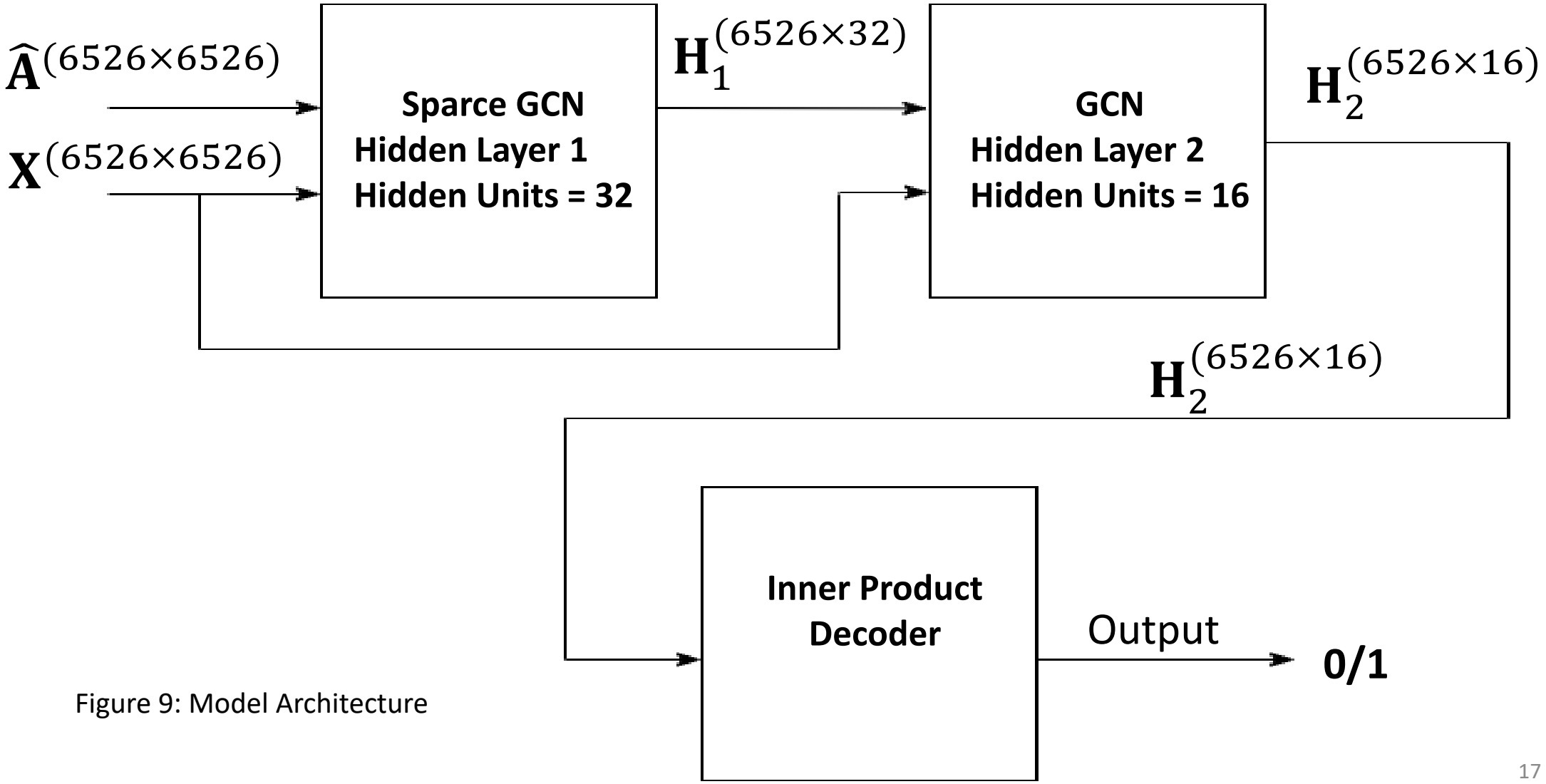


Figure 9: Model Architecture

Inner Product Decoder

$$\mathbf{H}_2 = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,16} \\ a_{2,1} & a_{2,2} & & a_{2,16} \\ \vdots & & \ddots & \vdots \\ a_{6526,1} & a_{6526,2} & \cdots & a_{6526,16} \end{bmatrix}$$

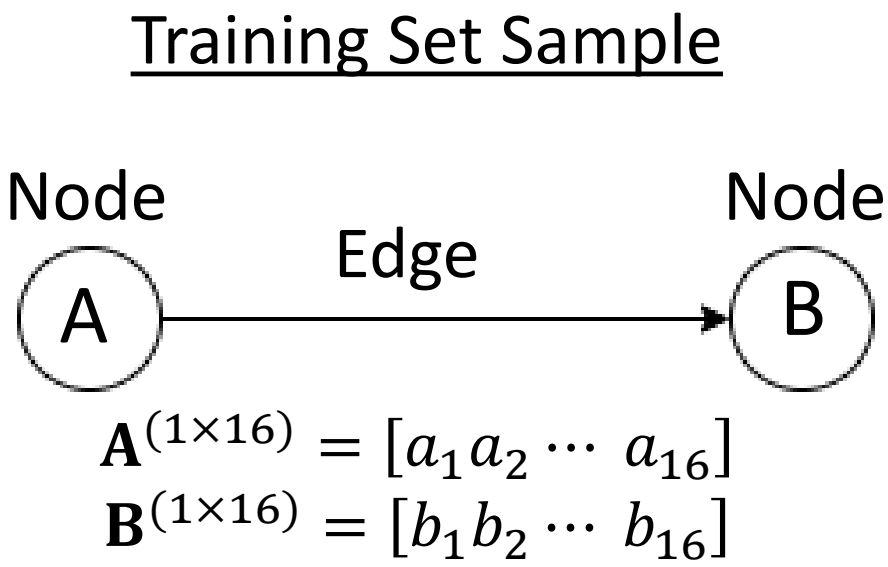


Figure 10: Training Set Sample Nodes

6. Evaluation

Effect of Epochs on Area Under the Curve (AUC)

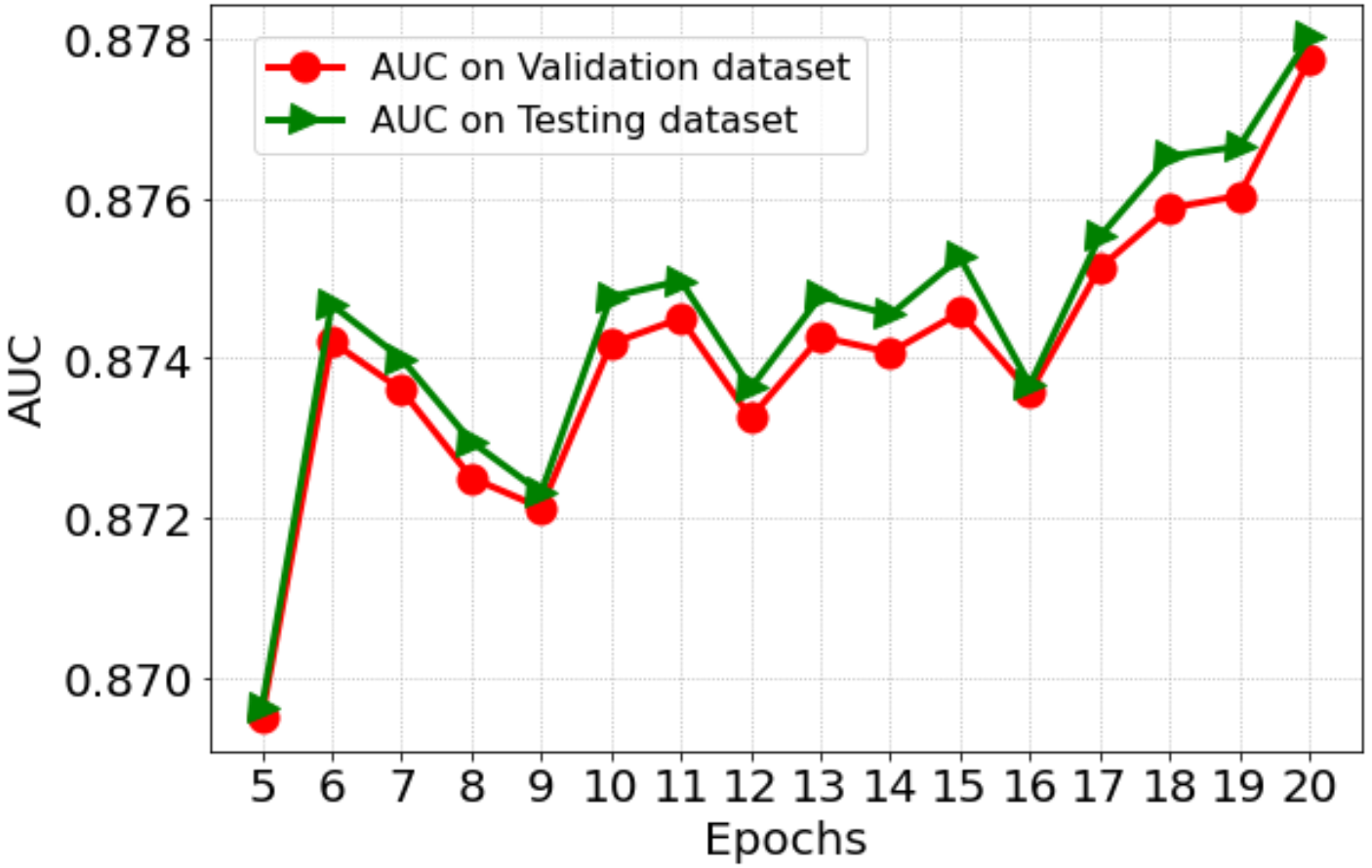
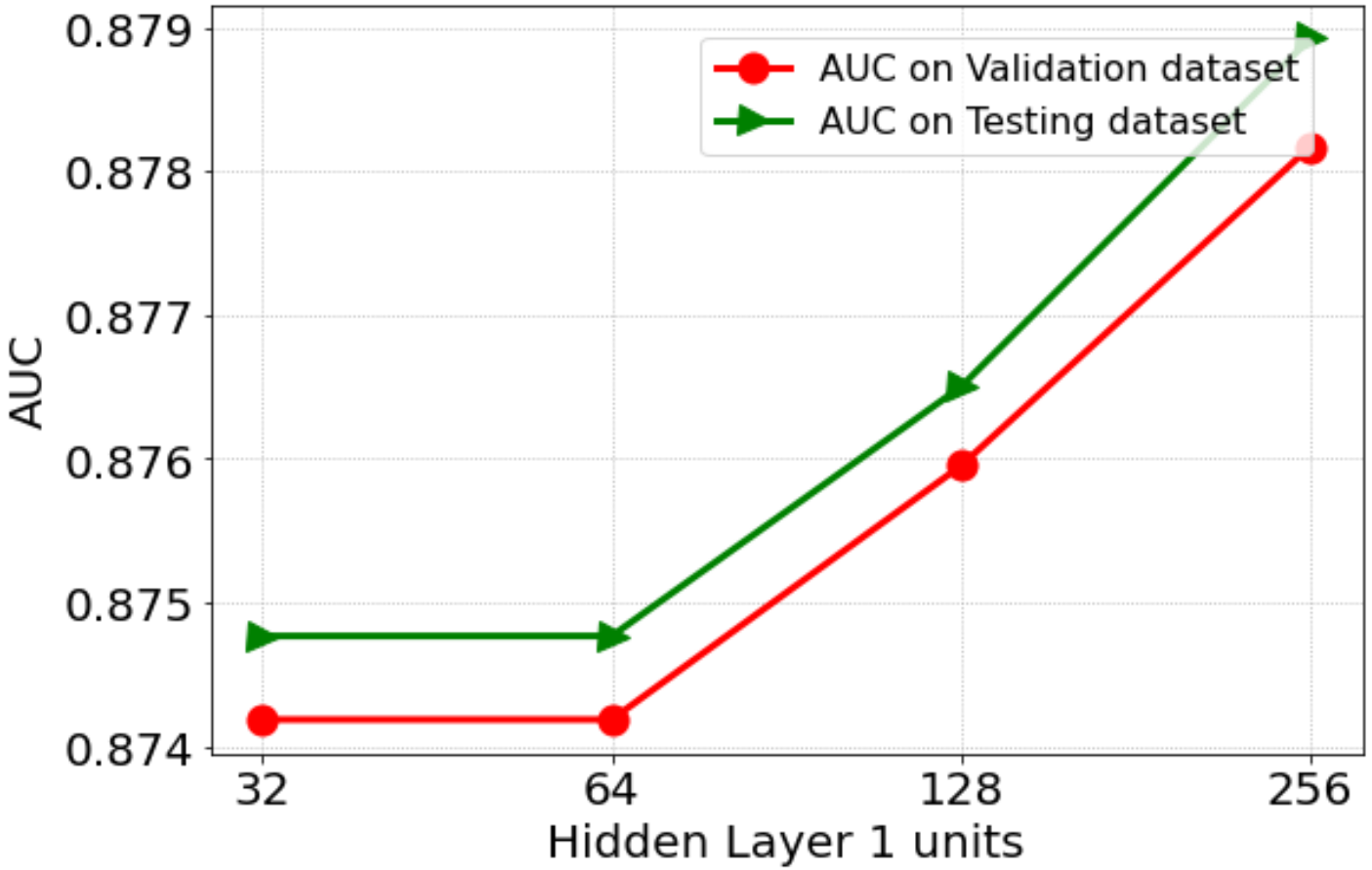


Figure 11: Effect of Epochs on Area Under the Curve (AUC)

- We find that area under curve rises for both validation and testing set when epoch is increased from 5 to 6.
- However, AUC gets zig-zag behaviour for the epochs between range 6 to 16.
- There is again a significant rise in AUC for the epochs increased after 16.

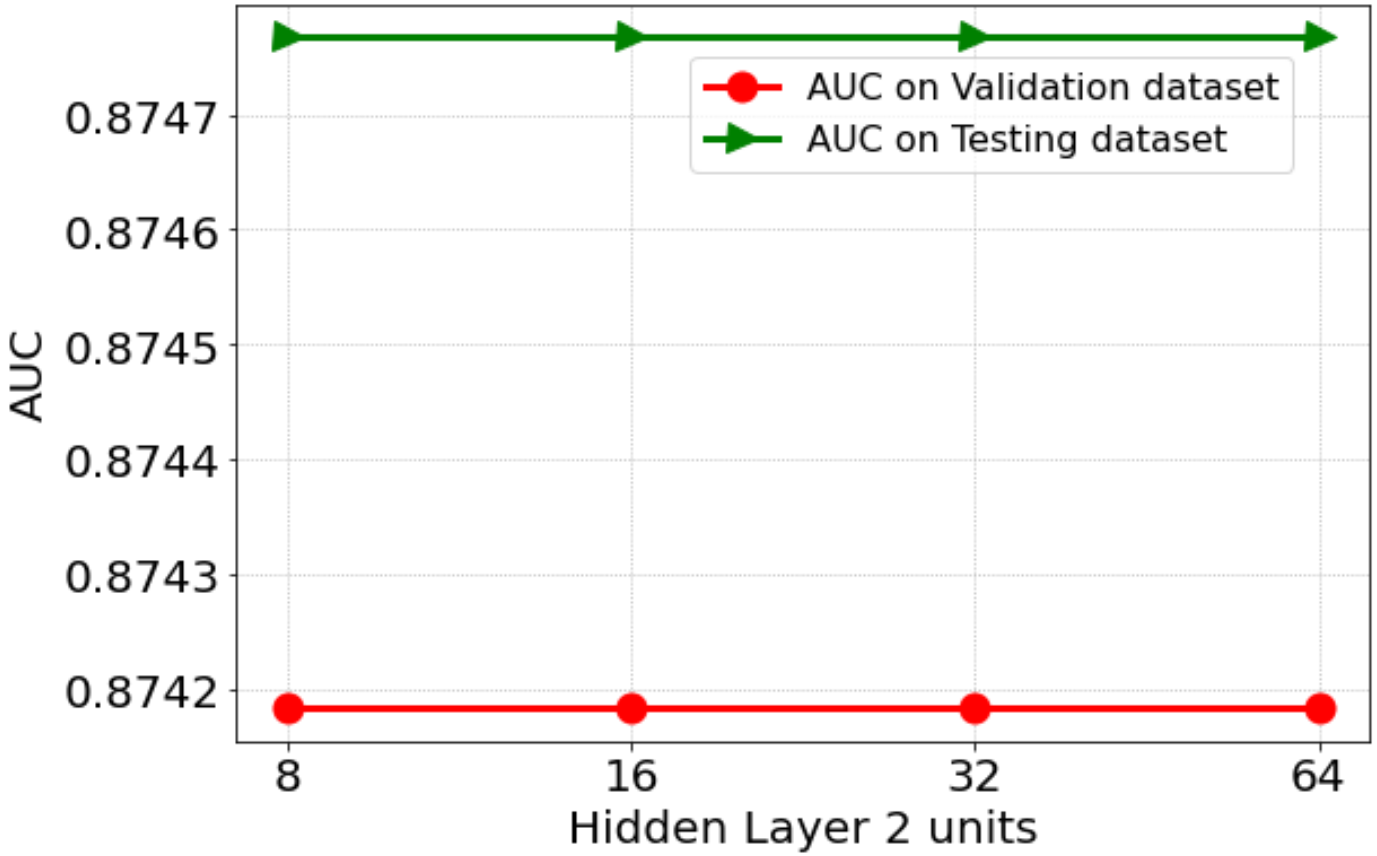
Effect of Number of units in hidden layer 1 on Area Under the Curve (AUC)



- We find that the area under the curve for both testing set and validation set at first remains constant till the number of units of the hidden layer 1 is 64.
- However it increases with the increase in the number of units of the hidden layer.

Figure 12: Effect of Number of units in hidden layer 1 on Area Under the Curve (AUC)

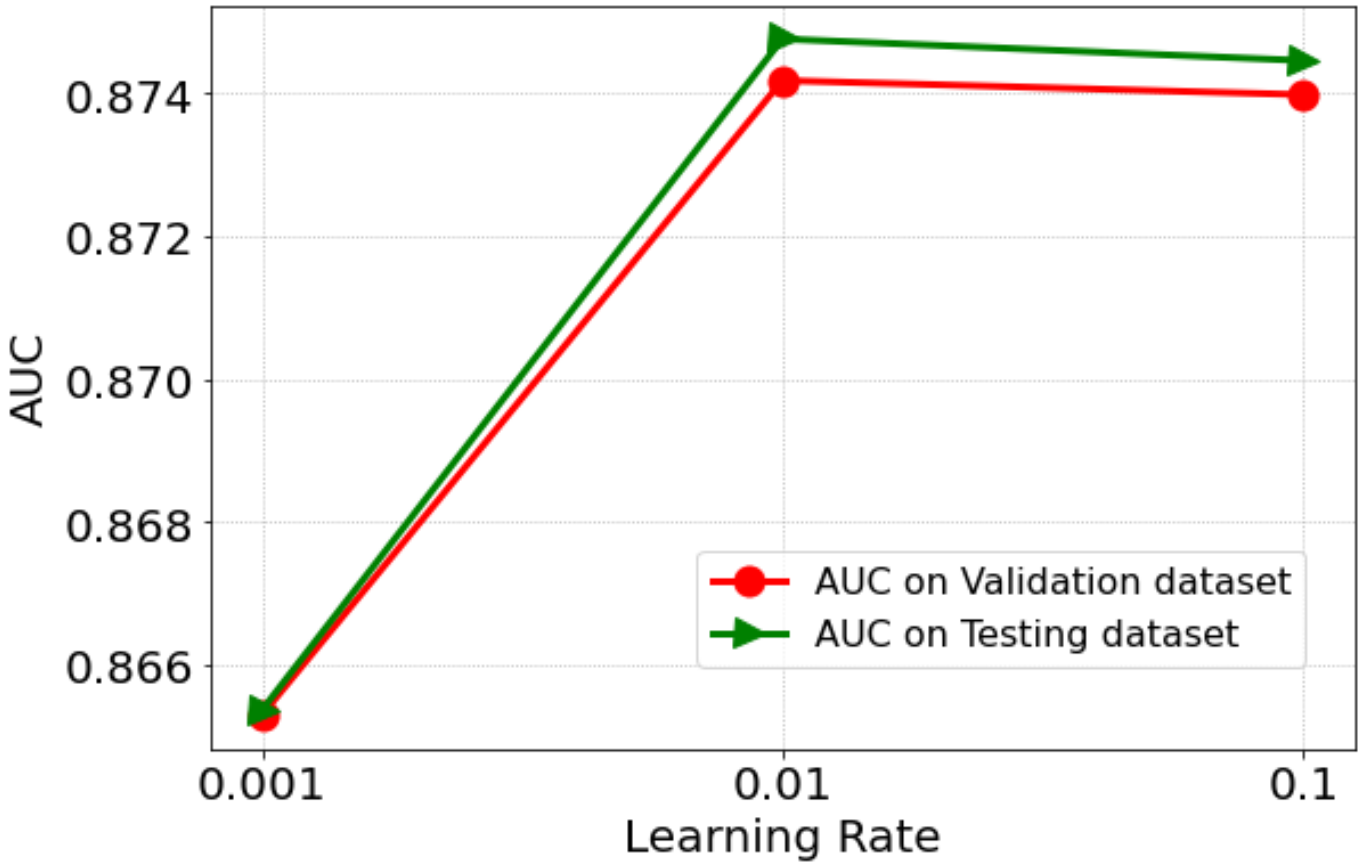
Effect of Number of units in hidden layer 2 on Area Under the Curve (AUC)



- We find that the area under the curve for both testing and validation set remains constant as the number of units in hidden layer 2 increases.

Figure 13: Effect of Number of units in hidden layer 2 on Area Under the Curve (AUC)

Effect of Learning Rate on Area Under the Curve (AUC)



- We find that the AUC for both the testing set and the validation set increases linearly as we increase the learning rate from 0.001 to 0.01 but after 0.01 it remains constant.

Figure 14: Effect of Learning Rate on Area Under the Curve (AUC)

List of Figures

References

[1] Girvan, M., & Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America*, *99*(12), 7821–7826. https://doi.org/10.1073/pnas.122653799

[2] Xue Li, Lifeng Yang, Xiaopan Zhang, and Xiong Jiao. 2019. Prediction of Protein-Protein Interactions Based on Domain Computational and mathematical methods in medicine2019 (2019). https://www.hindawi.com/journals/cmmm/2019/5238406/

[3] Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. https://arxiv.org/abs/cs.LG/1609.02907