# Section 2: Overview of Network Representation Learning (NRL) methods

## Matrix factorization-based

#### GraRep

## Random walk-based

#### Deepwalk

#### Node2vec

#### Node2vec+

#### Struc2vec

#### Ripple2vec

## Neural network-based

#### Large-scale Information Network Embedding (LINE)

#### Structural Deep Network Embedding (SDNE)

#### Variational Graph Auto-Encoders (VGAE)

## Node proximity for signed networks

A definition that is generalized from the second-order node proximity for unsigned networks (e.g. LINE) [1st condition]

Neural network signed network embedding (nSNE) introduced a 2nd condition, that is, if two nodes in a signed network are similar, they not only should satisfy the second-order node proximity but also have similar sign context.

[Demonstrated by tradeoff parameter Beta]

Provides a unified objective function that can preserve both the node and edge pattern of the network

nSNE: Embeddings of nodes and mapping functions learned via back-propagation algorithm

# Section 3: Experiments

## Chosen IAV strains

All experiments in this review was conducted based on four Influenza A Virus (IAV) strains of interest, as listed in Table 1.

Table 1: IAV strains of interest

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Strain** | **Subtype** | **Taxonomy ID** | **Abbreviation** | **Reason** |
| A/Puerto Rico/8/1934 | H1N1 | 211044 | PR8 | Commonly used in lab experiments |
| A/California/04/2009 | H1N1 | 641501 | CA04 | 2009 Pandemic strain |
| A/California/07/2009 | H1N1 | 641809 | CA07 | 2009 Pandemic strain  (Contains an “unusual” amino acid ‘X’ in its NP segment) |
| A/Aichi/2/1968 | H3N2 | 387139 | Aichi | H3N2 subtypes of IAV have cause seasonal epidemics since 1968 \cite{RN13} |

## Datasets

In this paper, unique datasets were constructed using the human-virus PPI (HVPPI) {Yang, 2020 #5} prediction tool (<http://zzdlab.com/hvppi/predict.php>). In HVPPI, an unsupervised sequence embedding approach, doc2vec, was applied to represent protein sequences as rich feature vectors of low dimensionality. Then, these embeddings were used as inputs to train a random forest (RF) classifier to predict human-virus PPIs.

Positive - Host-Pathogen Interaction Database (HPIDB) \cite{RN17, 16} V3.0

* + Contains manually curated host-pathogen interactions and also integrates corresponding molecular interactions from other public protein interaction databases

Negative samples – Dissimilarity-Based Negative Sampling

To reduce training noise, only proteins pairs with interaction probability score above the 0.95 threshold were considered.

**Threshold** **Score range (between 0 and 1)**

0.99 >= 0.375

0.95 >= 0.212 and < 0.375

0.90 >= 0.143 and < 0.212

Negative < 0.143

### Overview of complete network graph

|  |  |  |
| --- | --- | --- |
| Data | Complete | Edgelist1 |
|
| # nodes | 15, 685  (41 IAV, 15, 644 Human) | |
| # nodes involved in interaction | 12, 438  (41 IAV,  12, 397 Human) | 12, 437  (41 IAV,  12, 396 Human) |
| # edges | 91, 217  (3738 0.99,  26, 044 0.95, 61, 435 0.90) | 48, 8822  (2872 0.99,  15, 782 0.95,  30,228 0.90) |
| # non-interacting pairs | 550, 187 | |
| HVPPI Score (MIN) | 0.000 | 0.001 |
| HVPPI Score (MAX) | 0.99 | 0.99 |
| Degree (Average) | 14.6675 | 7.8607 |
| Degree (Minimum) | 1 | 1 |
| Degree (Maximum) | 10, 985  P03496  (PR8, NS1) | 10, 982  P03496  (PR8, NS1) |

1 All experiments conducted were based on this edgelist.

2 As HVPPI is a prediction tool, only edges with consistent prediction scores on 2 runs were considered. **Therefore, only 53.5% of the interacting edges were used to construct the edgelist as input to the different graph embedding models.**

### Edgelist

Review of network representation learning (NRL) methods to learn node embedding from an (unsigned) network

* Weighted network (using HVPPI scores (score ranges from 0 to 1) as the edge weights): node2vec+

[Mention that edge weights were changed to -1 and 1 based on threshold of 0.143]

* Signed network (-1 and 1): Neural network signed network embedding (nSNE) {Song, 2018 #50}

### Overview of constructed datasets

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dataset |  | | | Link Prediction  (Unbalanced dataset)4 | |
|  | Train1 | Test 1  (Network Reconstruction)1 | Test 2  (Experimentally Verified) | Train | Test |
| # nodes involved in interaction | 1098  (19 IAV,  1079 Human) | 376  (16 IAV,  360 Human) | 346  (14 IAV.  332 Human) | 6522  (30 IAV,  6492 Human) | 2681  (28 IAV,  2653 Human) |
| # Positive samples (i.e., interacting pairs) | 4014 | 446 | 4462 | 38, 101 | 4234 |
| # Negative samples (i.e., non-interacting pairs) | 4014 | 446 | 4463 | 152, 404 | 38, 106 |
| HVPPI Score  (MIN) | 0.000 | 0.001 | 0.286 | 0.004 | 0.004 |
| HVPPI Score  (MAX) | 0.99 | 0.931 | 0.988 | 0.845 | 0.743 |

**1 Edges are found in edgelist.**

2 Positive samples from HPIDB 3.0 \cite{}. These samples were checked against HVPPI probability scores and identified to also be interacting.

3 Negative samples from negatome 2.0 \cite{} database. Human-Human protein interactions only, involving 482 Human Protein Nodes.

4 All remaining **interacting** protein pairs not in edgelist made up the positive samples in this dataset. [91, 217 – 48, 882 = 42, 335 (Split into 90% Training + Validation, 10% Test)]

Ratio (Positive: Negative)

Training + Validation 1 (20%): 4 (80%) – Randomly sampled 4 times as many negative samples

Test 1 (10%): 9 (90%) – Randomly sampled 9 times as many negative samples

#### Reason for unbalanced dataset

To (further) demonstrate robustness of different (graph embedding/NRL) methods on incomplete interaction network.

1. [Does not need all edges to achieve “good quality” graph embeddings]. (coverage)
2. Real-world situation {Dunham, 2021 #43}

Hubs, biasing problem

Single protein appears far more times in the positive dataset

#### Reason for experimentally verified dataset

## 3.3 Experiments

### Models

#### Feedforward neural network (FNN)

The softmax FNN classifier has three hidden layers containing 128, 64 and 32 neurons in each layer respectively. The default number of epochs is 200 and learning rate was set to be 0.001 with Adam optimizer {Kingma, 2014 #54}. Dropout and early stopping were adopted during training to avoid overfitting. Dropout rate is 0.5.

#### Skip-GNN {Huang, 2020 #6}

Use same code and hyperparameters as original paper

Experiment using different NRL methods to train input features (embedding) for the original graph

### Hyper-parameter Tuning

Standard hyperparameters

Workers 8

Window size 10

Iteration (number of epochs in SGD?) ??

Dimensions (standardized to 128, except VGAE which depends on the number of hidden units)

Learning rate?

Sensitive hyper-parameters which were mentioned to be important by their authors and in the general guidelines provided by {Yue, 2020 #3} were carefully tuned via grid search. Table 2 shows the meanings and selected hyper-parameters for each NRL method. The effect of these hyper-parameters on each method is shown in *Supplementary material*

Table 2: Optimized hyper-parameters (\*based on network reconstruction dataset)

|  |  |  |  |
| --- | --- | --- | --- |
| **NRL Method** | **Definition** | **Chosen hyper-parameters** | |
| **FNN** | **Skip-GNN** |
| Matrix factorization-based | | |  |
| GraRep | k-steps = number of transition steps | k-steps = 2 | k-steps = 1 |
| Random walk-based | | |  |
| deepwalk | Num\_walks = number of walks per vertex  Walk\_length =  [node2vec, node2vec+]  p = return parameter1  q = in-out parameter2 | num\_walks = 16, walk\_length = 32 | num\_walks = 128, walk\_length = 8 |
| Node2vec | num\_walks = 8, walk\_length = 32  p = 0.25 , q= 0.5 | num\_walks = 8, walk\_length = 64  p = 0.5 , q= 0.25 |
| Node2vec+ | num\_walks = 8, walk\_length = 32  p = 0.25 , q= 0.5 |  |
| Struc2vec | num\_walks = 128, walk\_length = 16 | num\_walks = 64, walk\_length = 64 |
| Ripple2vec | num\_walks = 8, walk\_length = 8 |  |
| Neural network-based | | |  |
| LINE |  | 1st + 2nd order proximity, epochs = 10 |  |
| SDNE | α = balances the weight of 1st and 2nd-order proximities between nodes  β = Modulates the reconstruction weight of non-zero elements in the graph | α = 0, β = 10 | α= 0.3, β=10 |
| VGAE |  | hidden1 = 256, hidden2 = 128 | hidden1=16, hidden2=8 |
| Node proximity for signed networks | | |  |
| nSNE |  | K = 128, β = 0.005 | N/A |

1 Value <1 encourages returning back to previous node, and value >1 discourages

2 Value <1 encourages walks to go outwards, and value >1 encourage walks within a localized neighbourhood \cite{RN40}

p=1 and q=1, is a special case of node2vec, where sampling approach in deepwalk can be seen \cite{RN7}

Choosing mapping function to compute edges

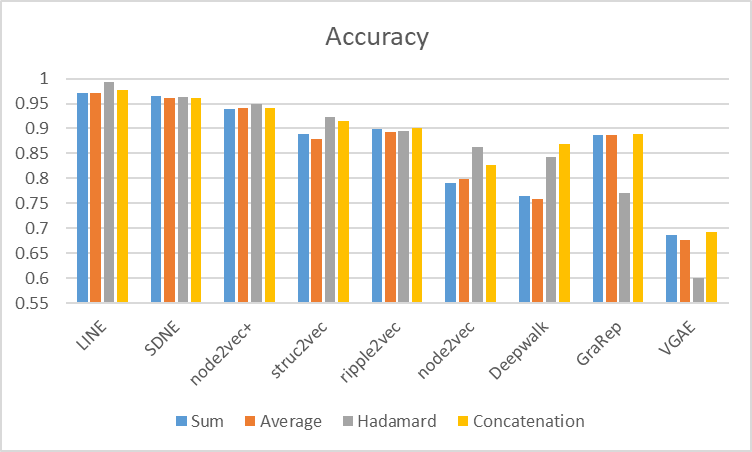
* Only applicable to FNN
* Skip-GNN uses concatenation by default

Reason for choosing edges {Song, 2018 #50}

Problem with converting node to edge embeddings

*Table 8: Element-wise operators for computation of representation for pairs of nodes, where u and v represent the source and target nodes respectively.*

|  |  |
| --- | --- |
| **Operator** | **Definition** |
| Sum |  |
| Average |  |
| Hadamard |  |
| Concatenation |  |



*Figure 1: Comparison of element-wise operators used across NRL methods on the network reconstruction dataset*

For five out of the nine methods, hadamard product is highly stable and consistently yeild the best performance.

In contrast, for GraRep and VGAE, hadamard product was the worst performing, while sum consistently yield the best performance.

For ripple2vec and Deepwalk, results obtained using hadamard and concatenation operators are comparable. Therefore, hadamard was selected as the number of dimensions is half that of concatenation.

## 3.4 Results

All results presented in this section were based on 5-fold cross validation on the training set.

### Network Reconstruction dataset

Table 3: FNN

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | Accuracy | Sensitivity | Specificity | Precision | F1-Score | ROC-AUC | PR-AUC |
| Matrix factorization-based | | | | | | | |
| GraRep | 0.8863 | 0.9215 | 0.8511 | 0.8613 | 0.8903 | 0.9559 | 0.9523 |
| Random walk-based | | | | | | | |
| deepwalk | 0.8419 | 0.8502 | 0.8336 | 0.8365 | 0.8432 | 0.9005 | 0.8815 |
| node2vec | 0.8626 | 0.8964 | 0.8287 | 0.8394 | 0.8668 | 0.9124 | 0.8718 |
| node2vec+ | 0.9493 | 0.9713 | 0.9274 | 0.9305 | 0.9504 | 0.9872 | 0.986 |
| struc2vec | 0.9226 | 0.9283 | 0.917 | 0.918 | 0.9231 | 0.9714 | 0.968 |
| ripple2vec | 0.8942 | 0.9067 | 0.8816 | 0.8847 | 0.8955 | 0.9502 | 0.9441 |
| Neural network-based | | | | | | | |
| LINE | 0.9922 | 0.9978 | 0.9865 | 0.9867 | 0.9922 | 0.9994 | 0.9992 |
| SDNE | 0.9623 | 0.9883 | 0.9363 | 0.9396 | 0.9633 | 0.9912 | 0.9897 |
| VGAE | 0.6868 | 0.7229 | 0.6507 | 0.675 | 0.6972 | 0.7287 | 0.6782 |
| Node proximity for signed networks | | | | | | | |
| nSNE | 0.9964 | 0.9969 | 0.996 | 0.996 | 0.9964 | 0.9999 | 0.9999 |

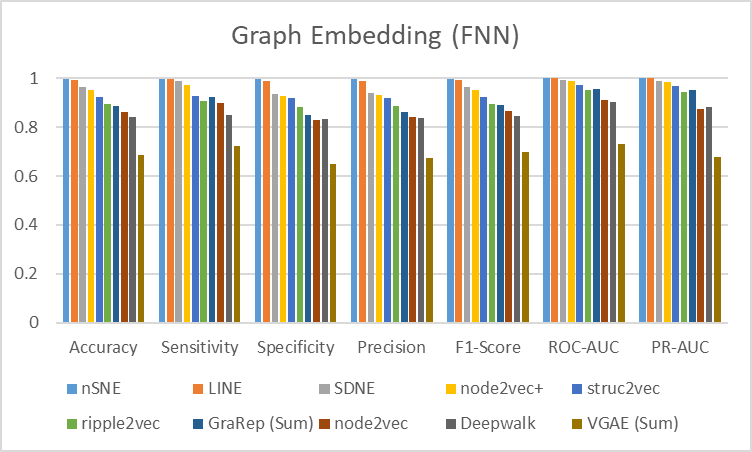


Table 4: Skip-GNN

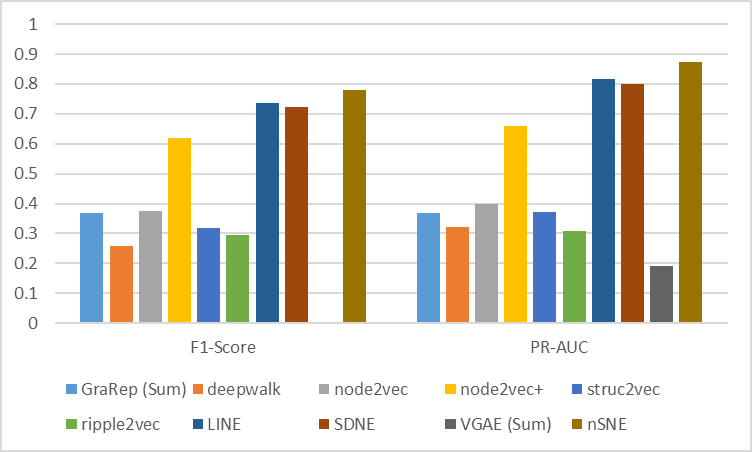
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | Accuracy | Sensitivity | Specificity | Precision | F1-Score | ROC-AUC | PR-AUC |
| Matrix factorization-based | | | | | | | |
| GraRep | 0.9318 | 0.8999 | 0.9639 | 0.962 | 0.9292 | 0.9875 | 0.9867 |
| Random walk-based | | | | | | | |
| deepwalk | 0.9393 | 0.9194 | 0.9589 | 0.9575 | 0.9377 | 0.9844 | 0.9838 |
| node2vec | 0.9435 | 0.9205 | 0.967 | 0.966 | 0.9423 | 0.989 | 0.9861 |
| node2vec+ |  |  |  |  |  |  |  |
| struc2vec | 0.9487 | 0.9276 | 0.9692 | 0.968 | 0.9471 | 0.9887 | 0.9876 |
| ripple2vec |  |  |  |  |  |  |  |
| Neural network-based | | | | | | | |
| LINE | 0.937 | 0.9118 | 0.9619 | 0.9602 | 0.9351 | 0.9877 | 0.9868 |
| SDNE | 0.9435 | 0.9241 | 0.9627 | 0.962 | 0.9424 | 0.989 | 0.9884 |
| VGAE |  |  |  |  |  |  |  |

\* nSNE was not tested on Skip-GNN

### Unbalanced dataset

Table 5: FNN

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | Accuracy | Sensitivity | Specificity | Precision | F1-Score | ROC-AUC | PR-AUC |
| Matrix factorization-based | | | | | | | |
| GraRep | 0.8849 | 0.3377 | 0.9457 | 0.4134 | 0.3681 | 0.8368 | 0.3687 |
| Random walk-based | | | | | | | |
| deepwalk | 0.9076 | 0.1687 | 0.9897 | 0.6614 | 0.2584 | 0.7065 | 0.3203 |
| node2vec | 0.9098 | 0.2722 | 0.9806 | 0.6106 | 0.3762 | 0.7779 | 0.3981 |
| node2vec+ | 0.9187 | 0.6594 | 0.9475 | 0.5829 | 0.6187 | 0.9369 | **0.6609** |
| struc2vec | 0.8975 | 0.2406 | 0.9705 | 0.4764 | 0.3194 | 0.8309 | 0.3723 |
| ripple2vec | 0.8864 | 0.239 | 0.9583 | 0.3893 | 0.296 | 0.7979 | 0.3087 |
| Neural network-based | | | | | | | |
| LINE | 0.9418 | 0.8117 | 0.9562 | 0.6743 | 0.7361 | 0.9743 | **0.817** |
| SDNE | 0.94 | 0.7871 | 0.957 | 0.671 | 0.7241 | 0.9675 | **0.8001** |
| VGAE | 0.9 | 0 | 1 | 0 | 0 | 0.6881 | 0.1909 |
| Node proximity for signed networks | | | | | | | |
| nSNE | 0.9522 | 0.8504 | 0.9635 | 0.7216 | 0.7806 | 0.9825 | **0.874** |



### Experimentally verified

Table 6: FNN

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | Accuracy | Sensitivity | Specificity | Precision | F1-Score | ROC-AUC | PR-AUC |
| Random walk-based | | | | | | | |
| node2vec+ | 0.5906 | 0.961 | 0.2202 | 0.553 | 0.7017 | 0.802 | 0.7783 |
| Neural network-based | | | | | | | |
| LINE | 0.7648 | 0.9574 | 0.5722 | 0.6937 | 0.8038 | 0.8747 | 0.8159 |
| SDNE | 0.9821 | 0.9843 | 0.9798 | 0.9799 | 0.9821 | 0.9931 | 0.9812 |
| Node proximity for signed networks | | | | | | | |
| nSNE | 0.9388 | 0.9538 | 0.9238 | 0.9346 | 0.9415 | 0.983 | 0.9891 |

Table 7: Protein sequence embedding

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | **Dim** | Accuracy | Sensitivity | Specificity | Precision | F1-Score | ROC-AUC | PR-AUC |
| AAC | 20 | 0.6742 | 0.8928 | 0.4556 | 0.6212 | 0.7327 | 0.7211 | 0.6404 |
| DPC | 400 | 0.6599 | 0.9018 | 0.4179 | 0.61 | 0.7267 | 0.7477 | 0.672 |
| C/T/D | 273 | 0.5 | 0.6 | 0.4 | 0.3 | 0.4 | 0.5061 | 0.5076 |
| QSOrder | 100 | 0.6733 | 0.8753 | 0.4713 | 0.6249 | 0.7281 | 0.7074 | 0.6189 |
| APAAC | 80 | 0.6778 | 0.8874 | 0.4682 | 0.6253 | 0.7336 | 0.7184 | 0.6338 |
| CT | 343 | 0.6251 | 0.8906 | 0.3596 | 0.5822 | 0.7038 | 0.7363 | 0.6793 |
| NMBroto | 240 | 0.7744 | 0.8529 | **0.696** | 0.7404 | 0.7905 | 0.8233 | 0.7593 |

[Provide brief description of protein sequence embedding methods]

## 3.5 Graph Visualization

### UMAP

[All Edges]

[2 by 5 image, arranged according to accuracy]

**Network Reconstruction Problem**

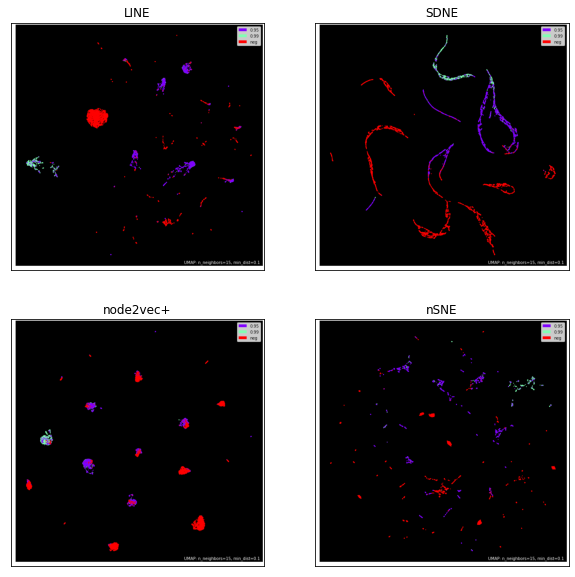


Figure 2(a): Plot of edges used in the network reconstruction (and experimentally verified) training set

*Full caption: Plot of edges used in the network reconstruction (and experimentally verified) training set, split into 3 classes. Green, purple and red data points denotes the interacting positive samples within 0.99 threshold, 0.95 threshold and the non-interacting negative samples respectively.*

These 4 methods were able to clearly distinguish highly interacting protein pairs (in green) from remaining protein pairs (with lower interaction probability) and the negative pairs.

However, only SDNE performance (w/ tune hyperparamters) remained consistent, even on other datasets (without having to retune hyperparameters).

Due to Homogeneity, completeness and v-measure score?

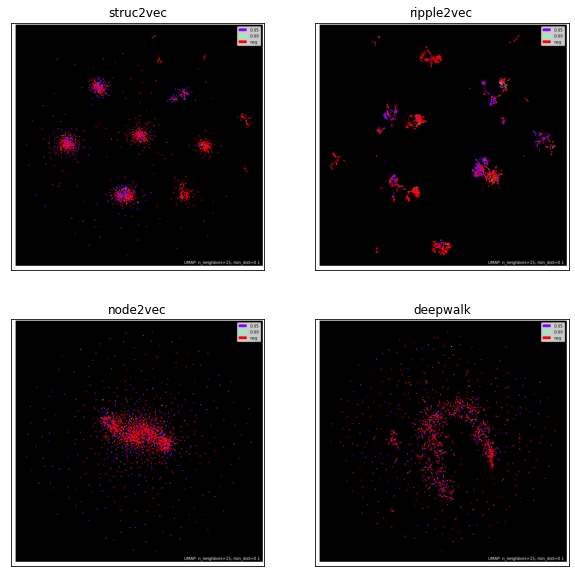


Figure 2(b): Plot of edges used in the network reconstruction (and experimentally verified) training set

[Random-walk methods]

For node2vec and deepwalk, there is only 1 distinct cluster, with no clear separation between the positive and negative protein pairs.

Although clusters were formed, the positive and negative pairs overlap and are not able to distinguish the highly interacting protein pairs.

Methods that did not perform well in the network reconstruction problem was “discarded”.

A good network embedding method should ensure that the learned embeddings can preserve the original network structure {Wang, 2016 #10} \cite{RN10}.

Therefore, in subsequent experiments, (i.e., unbalanced dataset and experimentally verified), we only tested on these 4 methods (i.e., LINE, SDNE, node2vec+ and nSNE).

### Clustering evaluation

* **Metrics**

1. Homogeneity, completeness and V-measure {Rosenberg, 2007 #53}

Homogeneity: Each cluster contains only members of a single class

Completeness: All members of a given class are assigned to the same cluster

V-measure: Harmonic mean of homogeneity and completeness

Β is set as default

1. Fowlkes-Mallows score

Geometric mean of the pairwise precision and recall

1. Adjusted rand index

Function that measures the similarity of two assignments

Takes into account that random chance will cause some (data points) to occupy the same clusters

1. Adjusted mutual information score

Measures the agreement of two assignments (split between clusters and split between the ground truth class labels)

Normalized against chance

K-means algorithm was ran on the embedded edge vectors to cluster all edges in edgelist into 4 classes (positive samples split into 0.99, 0.95, 0.99 thresholds and negative samples).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Method** | Homogeneity | Completeness | V-measure | Fowlkes-Mallows score | Adjusted rand score | Adjusted Mutual Information score |
| Matrix factorization-based | | | | | | |
| GraRep |  |  |  |  |  |  |
| Random walk-based | | | | | | |
| deepwalk | 0.0000 | 0.0000 | 0.0000 | 0.4391 | 0.0002 | 0.0000 |
| node2vec | 0.0029 | 0.0014 | 0.0019 | 0.4436 | 0.0091 | 0.0019 |
| node2vec+ | 0.0178 | 0.0227 | 0.0199 | **0.6898** | -0.0907 | 0.0199 |
| struc2vec | 0.0162 | 0.0074 | 0.0102 | 0.4369 | 0.0114 | 0.0102 |
| ripple2vec | 0.0154 | 0.0072 | 0.0098 | 0.4372 | -0.0035 | 0.0098 |
| Neural network-based | | | | | | |
| LINE | 0.1039 | 0.0552 | 0.0721 | 0.4585 | -0.0993 | 0.0721 |
| SDNE | 0.3909 | 0.2554 | 0.309 | 0.7649 | 0.4221 | 0.309 |
| VGAE |  |  |  |  |  |  |
| Node proximity for signed networks | | | | | | |
| nSNE | 0.6555 | 0.3121 | 0.4229 | 0.6032 | 0.239 | 0.4229 |

LINE low specificity on experimentally verified negative dataset

# Section 4: Discussion

Comment on overall performance of graph embedding and its performance against protein embedding

## Visualizations (UMAP)

and evaluation metrics to support discussion of graph embedding methodologies

* Clustering PPI networks
* Proteins interacting with each other often participates in the same biological processes or can be associated with specific biological functions being strongly related {Tornow, 2003 #52} (i.e., more similar, shorter distance between proteins)
* Segregate data points into groups such that data points placed in the same group are more similar to each other than to those in other groups

## Discussion points

* Comparison of Feed-forward neural network VS Skip-GNN (Network Reconstruction Dataset)
* Why I decided to “abandon” Skip-GNN for unbalanced dataset

Method chosen does not affect performance of Skip-GNN

Therefore, paper decided to use node2vec

* Choosing mapping function (node 🡪 edge embeddings)
* Node2vec VS node2vec +
* Weights would help the random walk to focus more on the relevant nodes in the graph
* LINE (1st order proximity VS 2nd and 1st + 2nd order proximity)

In biological networks, similarity between nodes that do not directly interact has proven incredibly useful in the last decade across a variety of interaction networks. {Huang, 2020 #6}

* Including reason for SDNE to perform better
* Neural network methods (i.e., LINE, SDNE and nSNE which uses a MLP model) able to perform better on link prediction task, even when dataset is unbalanced (Looking at PR-AUC scores)
* Better clustering evaluation scores (Homogeneity) = Better performance on a “new” unseen dataset (i.e., experimentally verified)

[SDNE & nSNE performs better (accuracy) than LINE]

* From **Specificity** scores on the experimentally verified dataset, SDNE and nSNE have lower false positive rate (0 predicted as 1). Even though testing was conducted on human-human interactions instead of iav-human interactions
* Node2vec+ and LINE able to identify the experimentally verified positive samples (high sensitivity score), however not the human-human negative samples (high number of false positives, leading to low specificity and precision scores)

##### Choosing edges

GraRep & VGAE

##### Higher (2nd) – order proximity

###### LINE

(Importance of having 2nd order proximity) + results

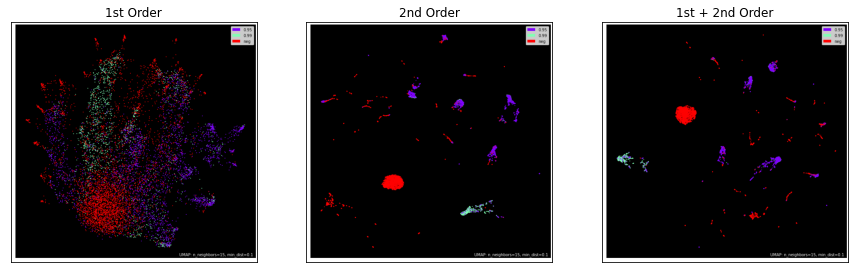


Figure 3: LINE:- UMAP –Network reconstruction dataset

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Accuracy | Sensitivity | Specificity | Precision | F1-Score | ROC-AUC | PR-AUC |
| 1st Order | 0.8345 | 0.8561 | 0.813 | 0.8247 | 0.8395 | 0.9079 | 0.895 |
| 2nd Order | 0.9883 | 0.9906 | 0.9861 | 0.9862 | 0.9884 | 0.9991 | 0.9991 |
| 1st + 2nd Order | 0.9922 | 0.9978 | 0.9865 | 0.9867 | 0.9922 | 0.9994 | 0.9992 |

\* Number of epochs was set to 10 for all.

##### Preserving edge attributes

###### Edge weights: Node2vec VS node2vec+

[Insert Plots] – network reconstruction and unbalanced dataset

How having edges weights instead of binary (1/0) improve performance

In the network reconstruction dataset, accuracy improved slightly by 8%

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Accuracy | Sensitivity | Specificity | Precision | F1-Score | ROC-AUC | PR-AUC |
| Network Reconstruction Dataset | | | | | | | |
| node2vec | **0.8626** | 0.8964 | 0.8287 | 0.8394 | 0.8668 | 0.9124 | 0.8718 |
| node2vec + | **0.9493** | 0.9713 | 0.9274 | 0.9305 | 0.9504 | 0.9872 | 0.986 |
| Unbalanced Dataset | | | | | | | |
| node2vec | 0.9098 | 0.2722 | 0.9806 | 0.6106 | **0.3762** | 0.7779 | **0.3981** |
| node2vec + | 0.9187 | 0.6594 | 0.9475 | 0.5829 | **0.6187** | 0.9369 | **0.6609** |

###### Signed network: nSNE

Using -1 to represent a negative edge instead of 0

signed VS unsigned network

Satisfying the 2nd condition (If two nodes in a signed network are similar, they should satisfy the condition of having similar sign context) {Song, 2018 #50}

When β = 1, only the first condition for node proximity of signed networks was used (i.e., they only satisfy the second-order node proximity)

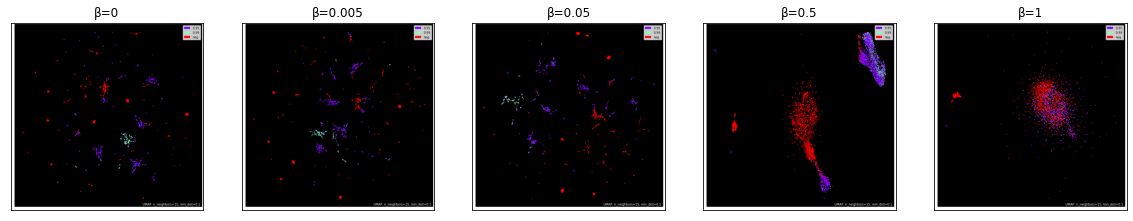


Figure 4: placeholder

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| β | Accuracy | Sensitivity | Specificity | Precision | F1-Score | ROC-AUC | PR-AUC |
| 0 | 0.9957 | 0.9964 | 0.9951 | 0.9951 | 0.9957 | 0.9999 | 0.9999 |
| 0.005 | 0.9964 | 0.9969 | 0.996 | 0.996 | 0.9964 | 0.9999 | 0.9999 |
| 0.05 | 0.9942 | 0.9955 | 0.9928 | 0.9929 | 0.9942 | 0.9999 | 0.9999 |
| 0.5 | 0.9791 | 0.9865 | 0.9717 | 0.9722 | 0.9793 | 0.9984 | 0.9984 |
| 1 | 0.8652 | 0.8664 | 0.8641 | 0.8646 | 0.8654 | 0.9348 | 0.9244 |

## Limitations

Ultimately, HVPPI is a prediction tool. Therefore, it is not 100% accurate.

With reference to (Test 2 Experimentally verified), there exists protein pairs verified by experimental techniques to be interacting but indicated as non-interacting by HVPPI. (i.e., below score of 0.143)

In our dataset, 176 of such samples were included, which hvppi scores ranging from 0.023 to 0.141.

\*\* Need to combine computational method with high-throughput experimental techniques for even more reliable results {Sarkar, 2019 #51}

# Section 5: Future work

* This work can be extended to include other IAV strains.
* Probability instead of binary

# Section 6: Conclusion