COMP4332 / RMBI4310 Project 2 Group 8

Data Pre-processing:

In the Data Loader, we use networkx.DiGraph to store the network and edges with different weights. In Random Walk Generator, alias sampling is used in order to reduce the time complexity to O(1) with O(n) space.

Algorithm:

For the embedding algorithms, we will use DeepWalk and node2vec, which generate first-order and second-order random walks respectively. AUC scores will then be used for the model evaluation.

Dataset:

	Training data	Validation data	Testing data
Number of nodes	8328	5440	8509
Number of edges	100000	19268	40000

Models:

We have trained Deepwalk and Node2vec algorithms by tuning 5 hyperparameters..

node_dim	[5, 10, 15, 20, 25, 30]
num_walk	[5, 10, 20, 30, 40]
walk_length	[5, 10, 20, 30, 40]
p_val	[0.25, 0.5, 0.75, 1, 1.25]
q_val	[0.25, 0.5, 0.75, 1, 1.25]

Performance:

For each algorithm, after we tried all the hyperparameter combinations, we sorted out the top 20 models with the highest AUC and performed 5-fold validation methods (Appendix 1) in order to ensure the generalization of the model. Here we would show the top 5 models performance in the following table.

Deepwalk:

node_ dim	num_ walk	walk_length	p_val	q_val	AUC with Valid data	5-fold validation AUC	5-fold validation Std
10	30	10	0.75	0.5	0.9338	0.9242	0.001
10	**30**	**10**	**1**	**0.5**	**0.9335**	**0.9242**	**0.0008**
10	30	10	0.75	0.25	0.9335	0.9239	0 0013
10	30	10	0.25	1.25	0.9333	0.9236	0.001
10	40	10	0.75	1.25	0.9332	0.9239	0.0012

According to the above table, after using 5-fold cross validation, the first two models generated the same AUC 0.9242 while the second model had a lower standard deviation. Although the first model had a higher AUC with valid data only, to ensure the model generalization, we chose a more stable model (the second model) as the selected model.

Node2vec:

node_dim	num_walk	walk_length	p_val	q_val	AUC with Valid data	5-fold validation AUC	5-fold validation Std
10	10	20	0.25	1	0.9335	0.9209	0.0015
10	10	20	1.25	1.25	0.9331	0.9220	0.0011
10	5	30	0.25	1.25	0.9329	0.9202	0.001
10	10	20	1	1	0.9329	0.9218	0.0017
10	10	20	0.5	1.25	0.9328	0.9209	0.0013

Evaluation:

Generally, Deepwalk performed better than Node2vec among different parameter values.

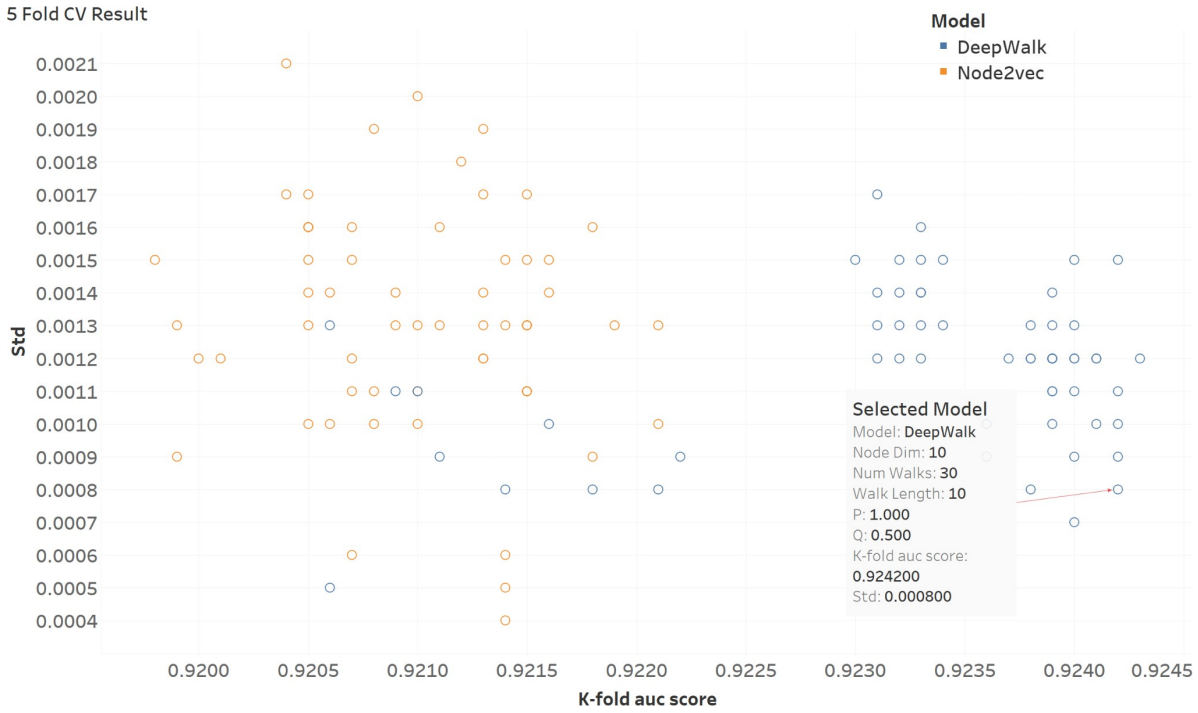
In the Deepwalk model, the accuracy increased from num_walk = 5 to num_walk = 30, which reached the highest accuracy. When node_dim and walk_length = 10, the accuracy of the Deepwalk model reached the highest point and gradually decreased with increasing node_dim and walk_length. (Appendix 2)

For the p value, the Deepwalk model generally performed better with p = 0.75 to 1 while the Node2vec model performed better with p = 0.25 to 0.5. For the q value, the Deepwalk model generally performed better with q = 0.5 while the Node2vec model performed better with q = 1. (Appendix 3)

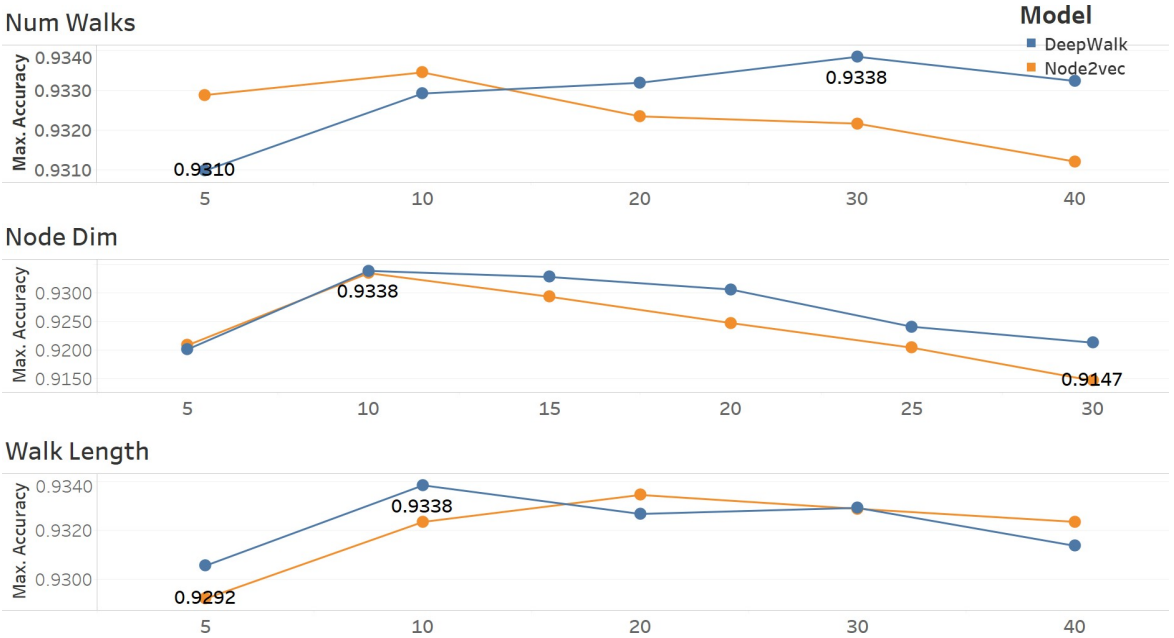
Therefore, we would choose the Deepwalk model with node_dim = 10, num_walk = 30, walk_length = 10, p_val = 1, q_val = 0.5 and AUC = 0.9335 as our final model to predict the testing data.

Appendix

1. Cross Validation Result



2. Performance of different num_walks, node_dim and walk_length values



3. Performance of different p and q values

P and Q

Q	Model / P: DeepWalk 0.25	DeepWalk 0.5	DeepWalk 0.75	DeepWalk 1	DeepWalk 1.25	Node2vec 0.25	Node2vec 0.5	Node2vec 0.75	Node2vec 1	Node2vec 1.25
0.25	0.9330	0.9329	0.9335	0.9331	0.9331	0.9317	0.9322	0.9323	0.9318	0.9311
0.5	0.9332	0.9328	0.9338	0.9335	0.9328	0.9315	0.9319	0.9318	0.9327	0.9324
0.75	0.9328	0.9326	0.9331	0.9332	0.9330	0.9324	0.9325	0.9325	0.9324	0.9318
1	0.9330	0.9332	0.9328	0.9326	0.9332	0.9335	0.9328	0.9326	0.9329	0.9326
1.25	0.9333	0.9327	0.9332	0.9330	0.9331	0.9329	0.9328	0.9320	0.9320	0.9331