AIM 5056-41: Homework 1

2020712702 Gahyung Kim (김가형)

Wikipedia dataset is is non-directed graph dataset consist of node features and its labels. Since a single node represents a single article, it can have multiple classes. The downloaded file was POS.mat and I used scipy.io library to load it. And I used Node2Vec code from the repository the professor provides, and the official NetMF code from its repository to solve problem number 2. (https://github.com/xptree/NetMF)

1. Parameter Sensitivity of node2vec

- Method to choose the parameter and the justification of your selection

To select best length of walk parameter, we need evaluation metrics to measure the prediction performance of the model depending on the parameter. The metric I used is the same method that the author of NetMF used, a micro F1-score and macro F1-score. A macro-averaged F1 score is a mean value of the per-class F1 scores, and a metric of micro-average F1-score is same as accuracy. The equations are showed below.

$$Macro-Precision = \frac{Pecision1 + Precision2}{2}$$

$$Macro-Recall = \frac{Recall1 + Recall2}{2}$$

$$Macro-F-Score = 2.\frac{Macro-Precision\,.\,Macro-Recall}{Macro-Precision + Macro-Recall}$$

$$Micro-Precision = \frac{TruePositives1 + TruePositives2}{TruePositives1 + FalsePositives1 + TruePositives2 + FalsePositives2}$$

$$Micro-Recall = \frac{TruePositives1 + TruePositives2}{TruePositives1 + FalseNegatives1 + TruePositives2 + FalseNegatives2}$$

$$Micro-F-Score = 2.\frac{Micro-Precision\,.\,Micro-Recall}{Micro-Precision + Micro-Recall}$$

When there is a class imbalance problem in datasets, Micro-average F1-score could be a better metrics to use.

- Result of the parameter selection with analysis

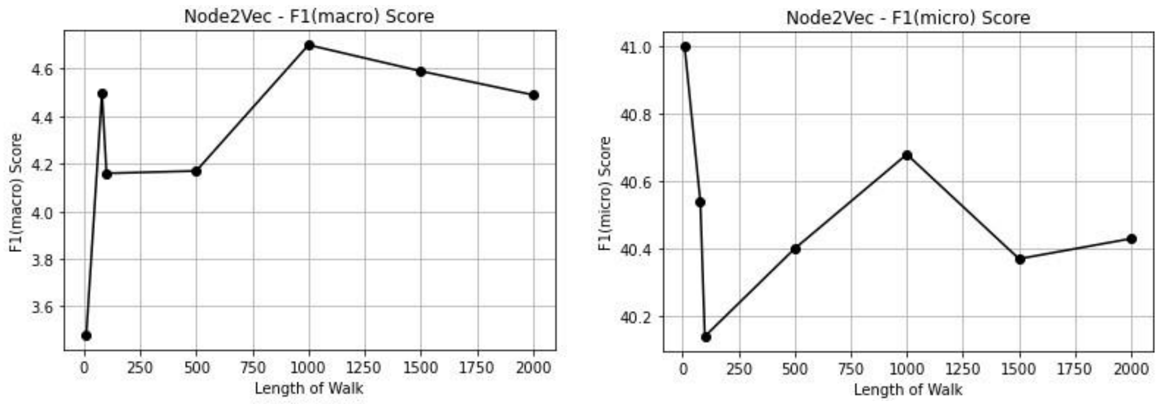

Figure 1. First Experiment Result

The default parameter of "Length of Walk" is 80. I first choose 10, 100, 150, 500, 1000, 1500, 2000 as candidates for the parameter because I want to see the overall movement of F1-scores and then select the section that shows the best result. The upper Figure 1 is the result of this attempt. As you can see that 1000 lengths of walk have the highest scores in both micro- and macro-F1 score. Even though 10 lengths of walk have the highest micro-average F1 score, it has the lowest macro-F1 score. Thus, 1000 lengths of walk seem to be the more promising choice.

After the first experiment, I subdivide parameters around 1000. I choose 90, 1000, 1100, 1200, 1300, 1400 as next candidates for the parameter. The result of second experiment is as followed.

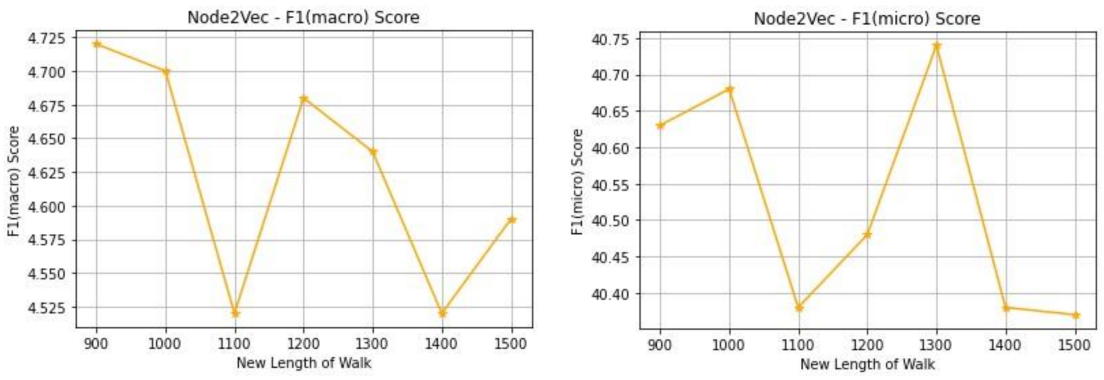

Figure 2 Second Experiment Result

110- lengths of walk clearly shows the worst F1-score in both cases. And in Macro-average F1-score 900 lengths of walk showed the highest result, whereas in micro-average F1-score, 1300 lengths of walk have the highest score. Since the best parameter in both graphs are different, I decided to add micro-average F1-score and macro-average F1-score for each length of walk, and the parameter with the highest sum value can be concluded as the best one in both measures. The overall result of my experiment is given in Table 1. As you can see, 1000 and 1300 have the same sum value, so I concluded to choose both as the best hyper-parameters.

Length of walk	10	80	100	500	900	1000	1100	1200	1300	1400	1500	2000
F1_score(micro)	**41**	40.54	40.14	40.4	40.63	40.68	40.38	40.48	40.74	40.38	40.37	40.43
F1_score(macro)	3.48	4.5	4.16	4.17	**4.72**	4.7	4.52	4.68	4.64	4.52	4.59	4.49
Sum	44.48	45.04	44.3	44.57	45.35	**45.38**	44.9	45.16	**45.38**	44.9	44.96	44.92

Table 1. The Overall Result

2. Theoretical approximation

- Method to measure their similarity

The method I choose to measure the similarity between NetMF and Node2vec embeddings is Cosine Similarity, Euclidean Distance and Manhattan Distance. Since embeddings are vectors with numbers, those three methods are an appropriate choice to measure the similarity between two embeddings. I used sklearn.metrics.pairwise library to implement these three metrics.

- Result with proper analysis

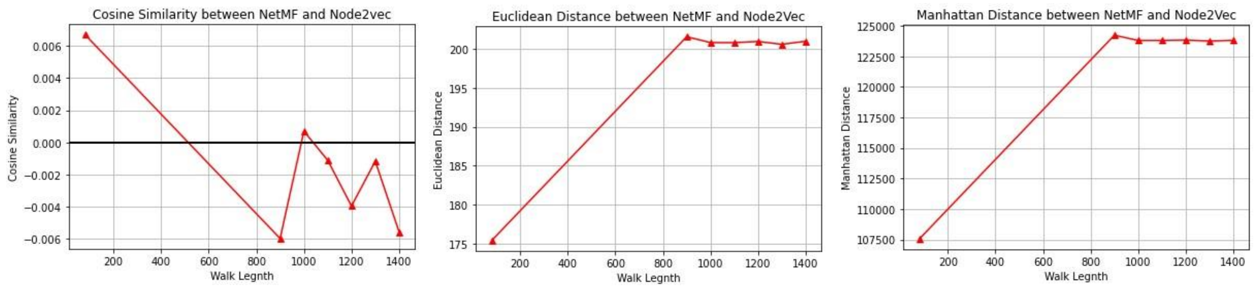

Figure 3. Cosine Similarity, Euclidean Distance, and Manhattan Distance between NetMF and Node2Vec in different length of walks

Here I tested 7 different length of walk parameters: 80(which was the default value in Node2Vec), 900, 1000, 1100, 1200, 1300, 1400. Since cosine value of zero angle is one, the closer the cosine similarity value is to one, the more similar the two embeddings are. And zero value means the embeddings are not similar. And in case of Euclidean distance and Manhattan Distance, smaller the values are the closer the two embeddings are. As you can see in Figure 3, all of three graph shows the same result: embeddings from 80 lengths of walk and the

NetMF embedding are the highest similarity among others.