MLG:
Machine Learning with Graphs (*Strojno učenje z grafi*)

Assignment: Homework 1

Submission time: ~ 1:00 **and date:** 15. 10. 2021

Submission Fill in and include this cover sheet with each of your assignments. It is an honor code violation to write down the wrong date and/or time. Assignments are due at 9:00am and should be submitted through Gradescope and eUcilnica. Students should check Piazza for submission details.

Late Periods Each student will have a total of *two* free late periods. *One late period expires the morning on the day before the next class.* (Assignments are usually due on Fridays, which means the first late period expires on the following Tuesday at 9:00am.) Once these late periods are exhausted, any assignments turned in late will be penalized 50% per late period. However, no assignment will be accepted more than *one* late period after its due date.

Honor Code We strongly encourage students to form study groups. Students may discuss and work on assignments in groups. However, each student must write down their solutions independently, i.e., each student must understand the solution well enough in order to reconstruct it by him/herself. Students should clearly mention the names of all the other students who were part of their study group. Using code or solutions obtained from the web (GitHub, Google, previous years etc.) is considered an honor code violation. We check all submissions for plagiarism. We take the honor code very seriously and expect students to do the same.

Name: Sara Bizjak

Email: sarabizjak97@gmail.com **SID:** 27202020

Study group: Maruša Oražem, Vid Stropnik

I acknowledge and accept the Honor Code.

(Signed) Sara Bizjak, l. r.

Machine Learning with Graphs

CS224W Homework 1

Sara Bizjak | 27202020

October 2021

1 Link Analysis

We have the following users and their teleport sets:

- A: {1, 2, 3},
- B: {3, 4, 5},
- C: {1, 4, 5},
- D: {1}.

Therefore, their teleport vectors can be expressed as:

- $t_A = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, 0, 0]$
- $t_B = [0, 0, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$
- $t_C = [\frac{1}{3}, 0, 0, \frac{1}{3}, \frac{1}{3}]$
- $t_D = [1, 0, 0, 0, 0]$

We are computing the personalized PageRank vectors for the following users assuming fixed teleport parameter β.
Note that from $\beta t_1 + (1-\beta)t_2 = t_3$ follows $\beta v_1 + (1-\beta)v_2 = v_3$.

1.1 Presonalized PrageRank I

User Eloise, denoted as E, with teleport set $\{2\}$ and teleport vector $t_E = [0, 1, 0, 0, 0]$.
We want to express t_E as a linear combination of t_A, t_B, t_C, t_D. It follows

$$t_E = [0, 1, 0, 0, 0] = \alpha \cdot \left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, 0, 0\right] + \beta \cdot \left[0, 0, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right] + \gamma \cdot \left[\frac{1}{3}, 0, 0, \frac{1}{3}, \frac{1}{3}\right] + \delta \cdot [1, 0, 0, 0, 0]$$

$$
\begin{aligned}
0 &= \frac{1}{3}\alpha \qquad\quad + \frac{1}{3}\gamma + \delta \\
1 &= \frac{1}{3}\alpha \\
0 &= \frac{1}{3}\alpha + \frac{1}{3}\beta \\
0 &= \qquad\quad \frac{1}{3}\beta + \frac{1}{3}\gamma \\
0 &= \qquad\quad \frac{1}{3}\beta + \frac{1}{3}\gamma \\
&\Rightarrow \alpha = 3, \beta = -3, \gamma = 3, \delta = -2
\end{aligned}
$$

so

$$t_E = 3 \cdot t_A - 3 \cdot t_B + 3 \cdot t_C - 2 \cdot t_D$$

and

$$v_E = 3 \cdot v_A - 3 \cdot v_B + 3 \cdot v_C - 2 \cdot v_D.$$

1.2

User Felicity, denoted as F, with teleport set $\{5\}$ and teleport vector $t_F = [0, 0, 0, 0, 1]$.
We want to express t_F as a linear combination of t_A, t_B, t_C, t_D.
Intuitively, it seems impossible to do that, since the last two coordinates of teleport vectors are always present together and have the same values, so we cannot sum 4^{th} coordinate to 0 and 5^{th} to 1. Formally,

$$t_F = [0, 0, 0, 0, 1] = \alpha \cdot \left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, 0, 0\right] + \beta \cdot \left[0, 0, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right] + \gamma \cdot \left[\frac{1}{3}, 0, 0, \frac{1}{3}, \frac{1}{3}\right] + \delta \cdot [1, 0, 0, 0, 0]$$

$$
\begin{aligned}
0 &= \frac{1}{3}\alpha \qquad\quad + \frac{1}{3}\gamma + \delta \\
0 &= \frac{1}{3}\alpha \\
0 &= \frac{1}{3}\alpha + \frac{1}{3}\beta \\
0 &= \qquad\quad \frac{1}{3}\beta + \frac{1}{3}\gamma \\
1 &= \qquad\quad \frac{1}{3}\beta + \frac{1}{3}\gamma \\
&\Rightarrow \text{from the last two equations: } 0 = 1 \text{which is contradiction}
\end{aligned}
$$

Since t_F cannot be written as linear combination of t_A, t_B, t_C, t_D, it also follows that v_F cannot be expressed with v_A, v_B, v_C, v_D.

1.3

User Glynnis, denoted as G, with teleport set $\{1, 2, 3, 4, 5\}$ and teleport vector $t_G = [0.1, 0.2, 0.3, 0.2, 0.2]$. We want to express t_G as a linear combination of t_A, t_B, t_C, t_D.

$$t_G = [0.1, 0.2, 0.3, 0.2, 0.2] = \alpha \cdot \left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, 0, 0\right] + \beta \cdot \left[0, 0, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right] + \gamma \cdot \left[\frac{1}{3}, 0, 0, \frac{1}{3}, \frac{1}{3}\right] + \delta \cdot [1, 0, 0, 0, 0]$$

$$\begin{aligned}
\frac{1}{10} = \frac{3}{30} &= \frac{10}{30}\alpha \qquad\qquad + \frac{10}{30}\gamma + \delta \\
\frac{2}{10} = \frac{6}{30} &= \frac{10}{30}\alpha \\
\frac{3}{10} = \frac{9}{30} &= \frac{10}{30}\alpha + \frac{10}{30}\beta \\
\frac{2}{10} = \frac{6}{30} &= \qquad \frac{10}{30}\beta + \frac{10}{30}\gamma \\
\frac{2}{10} = \frac{6}{30} &= \qquad \frac{10}{30}\beta + \frac{10}{30}\gamma \\
\Rightarrow \alpha &= \frac{6}{10}, \beta = \frac{3}{10}, \gamma = \frac{3}{10}, \delta = -\frac{6}{30}
\end{aligned}$$

so

$$t_G = \frac{3}{5} \cdot t_A + \frac{3}{10} \cdot t_B + \frac{3}{10} \cdot t_C - \frac{1}{5} \cdot t_D$$

and

$$v_G = \frac{3}{5} \cdot v_A + \frac{3}{10} \cdot v_B + \frac{3}{10} \cdot v_C - \frac{1}{5} \cdot v_D.$$

1.4

Suppose that we have already computed the personalized PageRank vectors (denoted as V) of a set of users. Then the set af all personalized PageRank vectors that we can compute from V without accessing the web graph are all linear combinations of vectors from V.

1.5

Here we prove that $r = Ar$ is equivalent to $r = \beta Mr + \frac{1-\beta}{N}1$.
Since $A = \beta M + \frac{1-\beta}{N}11^T$, it follows

$$r = \beta Mr + \frac{1-\beta}{N}11^T r$$

Also,

$$11^T = \begin{pmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \dots & 1 \end{pmatrix} \quad \text{and} \quad r = \begin{pmatrix} v_1 \\ \vdots \\ v_N \end{pmatrix}$$

so

$$11^T \cdot r = \begin{pmatrix} v_1 + v_2 + \ldots + v_N \\ \vdots \\ v_1 + v_2 + \ldots + v_N \end{pmatrix}$$

Assuming r is normalized $\left(\sum_{i=1}^{N} r_i = 1\right)$, it follows

$$11^T r = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$$

and

$$\begin{aligned} r &= Ar \\ &= \beta M r + \frac{1-\beta}{N} 11^T r \\ &= \beta M r + \frac{1-\beta}{N} 1 \end{aligned}$$

thus the condition is proved.

2 Relational Classification I

After the first iteration:

- $P(Y_1 = +) = \frac{1}{2}(\frac{1}{2} + \frac{1}{2}) = \frac{1}{2}$
- $P(Y_2 = +) = \frac{1}{4}(\frac{1}{2} + \frac{1}{2} + \frac{1}{2} + 0) = \frac{3}{8}$
- $P(Y_3 = +) = \frac{1}{3}(\frac{1}{2} + \frac{3}{8} + 1) = \frac{5}{8}$
- $P(Y_4 = +) = \frac{1}{4}(\frac{3}{8} + 0 + \frac{1}{2} + 1) = \frac{15}{32}$
- $P(Y_8 = +) = \frac{1}{2}(\frac{15}{32} + \frac{1}{2}) = \frac{31}{64}$
- $P(Y_9 = +) = \frac{1}{2}(\frac{31}{64} + 0) = \frac{31}{128}$

After the second iteration:

- $P(Y_1 = +) = \frac{1}{2}(\frac{3}{8} + \frac{5}{8}) = \frac{1}{2} = 0.5$

- $P(Y_2 = +) = \frac{1}{4}(\frac{1}{2} + \frac{5}{8} + \frac{15}{32} + 0) = \frac{51}{128} = 0.4$
- $P(Y_3 = +) = \frac{1}{3}(\frac{1}{2} + \frac{51}{128} + 1) = \frac{81}{128} = 0.63$
- $P(Y_4 = +) = \frac{1}{4}(\frac{51}{128} + 0 + \frac{31}{64} + 1) = \frac{241}{512} = 0.47$
- $P(Y_8 = +) = \frac{1}{2}(\frac{241}{512} + \frac{31}{128}) = \frac{365}{1024} = 0.36$
- $P(Y_9 = +) = \frac{1}{2}(\frac{365}{1024} + 0) = \frac{365}{2048} = 0.18$

2.1

$P(Y_3 = +) = \frac{81}{128} = 0.63$

2.2

$P(Y_4 = +) = \frac{241}{512} = 0.47$

2.3

$P(Y_8 = +) = \frac{365}{1024} = 0.36$

2.4

Nodes that belong to class "-" after the second iteration are:
1,2,4,5,8,9.

3 Relation Classification II

3.1 Bootstrap Phase

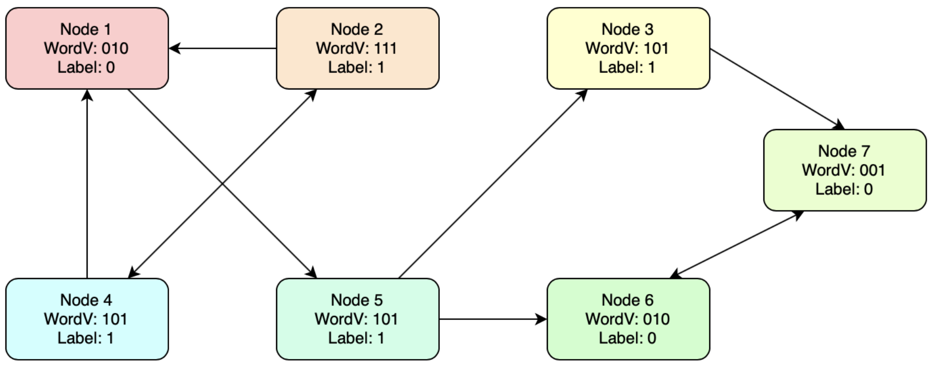

Figure 1: Condition after bootstrap phase.

3.2 Iteration 1

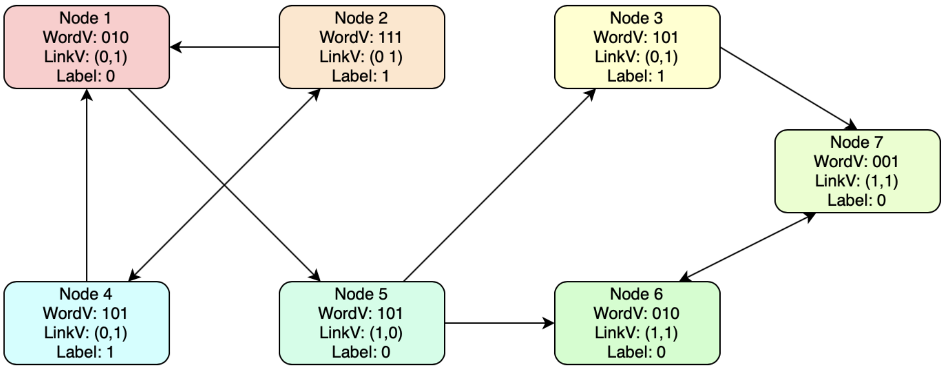

Figure 2: Condition after first iteration.

3.3 Iteration 2

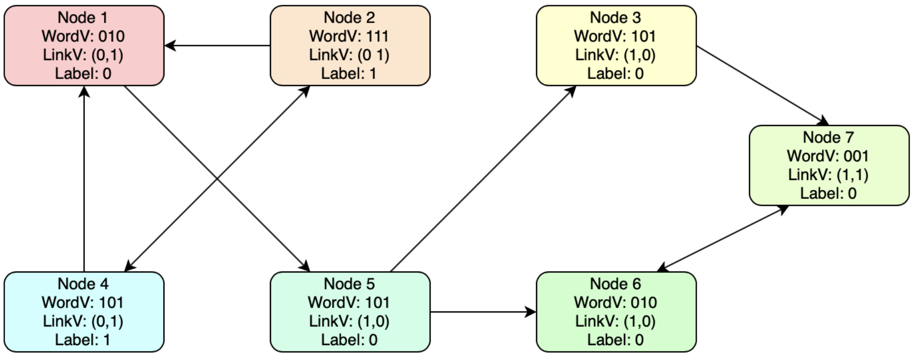

Figure 3: Condition after second iteration.

3.4 Convergence

If we do one more iteration, all the labels remain the same, so the algorithm converges after three iterations.

4 GNN Expressiveness

4.1 Effect of Depth on Expressiveness

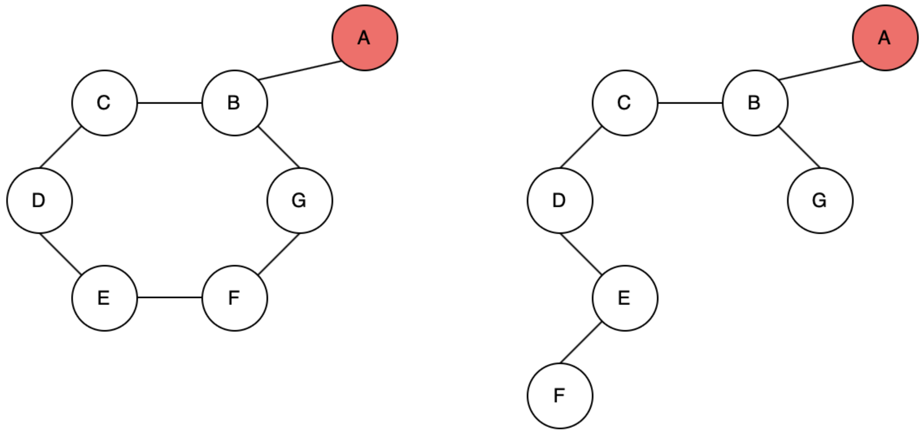

Figure 4: The two given graphs with denoted nodes.

We now run the GNN to compute node embeddings for the two red nodes.

Figure 5: Node embeddings for the red node after first and second layer of message passing are the same, the two red nodes are not distinguished yet.

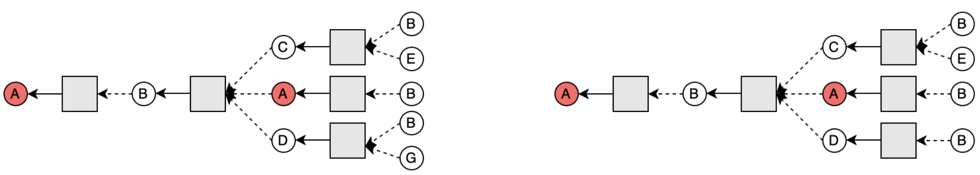

Figure 6: After third layer of message passing, the red nodes are distinguished, since having different GNN embeddings.

To conclude, three layers of message passing are needed so that the red two nodes can be distinguished (i. e., have different GNN embeddings).

4.2 Random Walk Matrix

i Assume that the current distribution is $r = \{0, 0, 1, 0\}$ and after the random walk, the distribution is $M \times r$. The transition matrix M, where each row of M corresponds with the node id in the graph, is then

$$M = \begin{pmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{3} \\ \frac{1}{2} & 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{2} & 1 & 0 \end{pmatrix}$$

ii Calculation of limiting distribution $r = [\alpha, \beta, \gamma, \delta]^T$, where $\alpha + \beta + \gamma + \delta = 1$ is

$$\begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \end{pmatrix} = \begin{pmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{3} \\ \frac{1}{2} & 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{2} & 1 & 0 \end{pmatrix} \times \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \end{pmatrix}$$

$$\begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \end{pmatrix} = \begin{pmatrix} \frac{1}{2}\beta + \frac{1}{3}\delta \\ \frac{1}{2}\alpha + \frac{1}{3}\delta \\ \frac{1}{3}\delta \\ \frac{1}{2}\alpha + \frac{1}{2}\beta + \gamma \end{pmatrix}$$

$$\begin{aligned} \text{first + second + third row: } \Rightarrow \alpha &= \frac{1}{2}\beta + \gamma \\ \beta &= \frac{1}{2}\alpha + \gamma \quad \Rightarrow \beta = \alpha \\ \Rightarrow = \gamma &= \frac{1}{2}\alpha \\ \Rightarrow \delta &= \frac{3}{2}\alpha \end{aligned}$$

following

$$\begin{aligned} 1 &= \alpha + \beta + \gamma + \delta \\ 1 &= \alpha + \alpha + \frac{1}{2}\alpha + \frac{3}{2}\alpha = 4\alpha \quad \Rightarrow \alpha = \frac{1}{4} = 0.25 \\ \Rightarrow \beta &= \frac{1}{4} = 0.25, \gamma = \frac{1}{8} = 0.125, \delta = \frac{3}{8} = 0.375 \end{aligned}$$

thus the rounded limiting distribution equals

$$r = (0.25, 0.25, 0.13, 0.38)$$

4.3 Relation to Random Walk (i)

The transition matrix of the random walk is

$$P = D^{-1} \cdot A.$$

4.4 Relation to Random Walk (ii)

If we add a skip connection in aggregation from Q4.3, the new corresponding transition matrix is

$$P = \frac{1}{2} \cdot I + \frac{1}{2} \cdot D^{-1} \cdot A.$$

4.5 Over-Smoothing Effect

We want to prove that Markov chain is aperiodic and irreducible.

- *Markov Chain is irreducible.* Since the graph is connected, we can move around the graph arbitrarily, meaning that from every node we can get to every other.
- *Markov chain is aperiodic.* It follows from the fact that the graph has no bipartite components.

Then, by the Markov Convergence Theorem, the node embedding $h_i^{(l)}$ will converge as $l \to \infty$, more precisely, it will converge to h_i for node i.

4.6 Learning BFS with GNN

Update rule for the i-th node for the GNN can be written as

$$h_i^{(l+1)} = \max_{j \in N_i} \left(h_i^{(l)}, h_j^{(l)} \right)$$

5 Node Embedding and its relation to matrix factorization

5.1 Simple matrix factorization

The decoder here is the dot product

$$z_i^T \cdot z_j$$

5.2 Alternate matrix factorization

The objective function for the corresponding matrix factorization would be

$$\min_{Z,W} ||A - Z^T W Z||_2$$

5.3 Relation to eigendecomposition

The matrix W has to be diagonal. The values on its diagonal would be eigen values and matrix Z would have eigen vectors in its columns.

5.4 Multi-hop node similarity

We know that in the (u, v)-th place in matrix A^k is the number of different paths of length k between nodes u and v. So if we look at elements of the matrix

$$A_p = \sum_{i=1}^{k} A^i$$

we will get the number of paths of length at most k between any of the two nodes in a graph. However, if two nodes are silimar when connected by at least one path of length at most k, we do not need to count how many such paths exist, but it is enough to construct a "classification" matrix, just using 0 and 1 to denote wether there exist a path of length at most k or not. Let us denote such matrix with A_{opt}. Therefore, we are solving the matrix factorization problem for matrix A_{opt} instead of A_p (I guess even with A_p would be possible, but A_{opt} is optimized). So,

$$\min_Z ||A_{opt} - Z^T Z||_2$$

5.5 Limitation of node2vec (i)

On the figure below, we have two cliques and a "bridge/path" between them. On each clique, all nodes are connected and will, therefore, have similar embedding (a node from left clique will have similar embeding than other nodes in left clique and similar for the right one). However, if we are moving around left clique on the graph, there will be a very small probability that our random walk will lead us to reach the right clique. To sum up, even if the strusture of the left and right clique is the same, the nodes on the left cliques will have different (= not similar) embeddings than the nodes on the right clique.

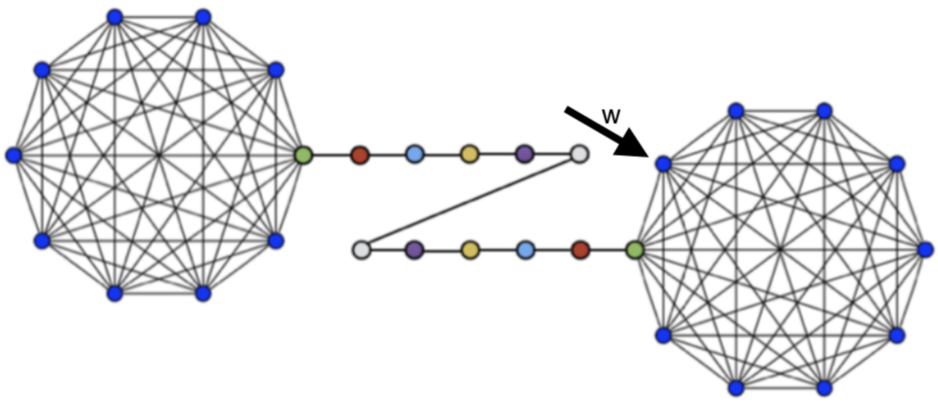

Figure 7: Given graph.

5.6 Limitation of node2vec (ii)

Assume we are at a position of node w in the graph on the figure above. Taking one step further with `node2vec` algorithm, we can only reach the neighbors, that is, nodes in the same clique in which w belongs (right one), however, with `struct2vec` algorithm we can reach all nodes, since all nodes are connected.

5.7 Limitation of node2vec (iii)

To consider different g_k's during the random walk means that we change the distance k. With changing k we can get a better approximation of the graph and a better insight on which nodes are similar.

5.8 Limitation of node2vec (iv)

In contrast to the `node2vec`, with `struct2vec` the nodes in both two cliques will have similar embeddings, since here we can reach all nodes, not only neighbors, so the random walk has more probability to go from one clique to another. Even more, since the algorithm captures the similarity in the structure between nodes, the nodes with the same colors will have similar embeddings.

References

[1] Transition matrix of random walk: `resources.mpi-inf.mpg.de/departments/d1/teaching/ws11/SGT/Lecture5.pdf`