Mathis EMBIT
mathis.embit@gmail.com

1 Question 1

- The DeepWalk algorithm, by leveraging random walks, tends to capture the local neighborhood structure of the graph. Nodes within the same connected component are likely to have similar or identical neighbors, leading to similar embeddings. In our extreme case, there are only two nodes so their neighborhood in the RW will be very similar and their embeddings will be almost equal. Therefore, you can expect the cosine similarity to be close to 1, indicating that the embeddings of the two nodes are aligned.
- It's difficult to know how an embedding function can lead to negative cosine similarity. Particularly for deepwalk because we use randomness. Noise tends to prevent two embeddings from being opposed (we clearly observe that -1 cosine similarity is rare in the real world). Here we would need to guess or inspect the weights of the skip gram NN. What we can say is that there will be no common neighbors in the nodes discovered by RW starting in a given connected componant, and by a RW starting in another one. Embeddings won't be similar at all, they could be either perpendicular or even opposed. That's why the cosine similarity will tends to be 0.

2 Question 2

Let's compute the time complexity of Deepwalk and Spectral Embedding algorithms.

2.1 Deepwalk algorithm

Algorithm 1 Deepwalk(G, w, d, γ, t)

Require: graph $G(V, E)$
window size w
embedding size d
walks per vertex γ
walk length t
Ensure: matrix of vertex representations $\Phi \in \mathbb{R}^{|V|\times d}$
Initialization: Sample Φ from $\mathcal{U}^{|V|\times d}$ $\longrightarrow \mathcal{O}(|V| \times d)$
Build a binary Tree T from V $\longrightarrow \mathcal{O}(|V|)$
for $i = 0$ to γ **do** $\longrightarrow \mathcal{O}(\gamma)$
    $\mathbb{O} =$ Shuffle(V) $\longrightarrow \mathcal{O}(|V|)$
    for each $v_i \in \mathbb{O}$ **do** $\longrightarrow \mathcal{O}(|V|)$
        $\mathcal{W}_{v_i} = RandomWalk(G, v_i,t)$ $\longrightarrow \mathcal{O}(t)$
        SkipGram(Φ, $\mathcal{W}_{v_i}$, w) $\longrightarrow \mathcal{O}(t \times w \times log_2|V|)$
    end for
end for

Algorithm 2 SkipGram(Φ, $\mathcal{W}_{v_i}$, w)

for each $v_j \in \mathcal{W}_{v_i}$ **do** $\longrightarrow \mathcal{O}(t)$
    for each $u_k \in \mathcal{W}_{v_i}[j - w : j + w]$ **do** $\longrightarrow \mathcal{O}(2w)$
        $J(\Phi) = -\log \Pr(u_k \mid \Phi(v_j))$ $\longrightarrow \mathcal{O}(log_2|V|)$ (hierarchical softmax)
        $\Phi = \Phi - \alpha * \frac{\partial J}{\partial \Phi}$ $\longrightarrow \mathcal{O}(1)$
    end for
end for

We compute that SkipGram time complexity is $\mathcal{O}(t \times w \times log_2|V|)$. We then use this complexity to compute that Deepwalk's one is

$$\begin{aligned} &\mathcal{O}(|V| \times d + |V| + \gamma \times (|V| + |V| \times (t + t \times w \times log_2|V|))) \\ =&\mathcal{O}(|V| \times log_2|V| \times \gamma \times t \times w) \end{aligned}$$

Conclusion: the Deepwalk time complexity is

$$\boxed{\mathcal{C}_{deepwalk} = \mathcal{O}(|V| \times log_2|V| \times \gamma \times t \times w)}$$

2.2 Spectral Embedding algorithm

Algorithm 3 Spectral Clustering

Input: Graph $G = (V, E)$ and parameter k

Output: Clusters $C_1, C_2, ..., C_k$ (i.e., cluster assignments of each node of the graph)

1: Let A be the adjacency matrix of the graph. $\longrightarrow \mathcal{O}(|V|)$

2: Compute the Laplacian matrix $L_{rw} = I - D^{-1}A$, where D is the diagonal degree matrix. $\longrightarrow \mathcal{O}(|V|^2)$

3: Apply eigenvalue decomposition to L_{rw} and compute the eigenvectors corresponding to d smallest eigenvalues. Let $U = [u_1|u_2|...|u_d] \in \mathbb{R}^{m\times d}$ be the matrix containing these eigenvectors as columns. $\longrightarrow \mathcal{O}(|V|^3)$

4: For $i = 1, \ldots, m$, let $y_i \in \mathbb{R}^d$ be the vector corresponding to the i-th row of U. Apply k-means to the points $\{y_i\}_{i=1}^m$ (i.e., the rows of U) and find clusters $C_1, C_2, ..., C_k$.

Conclusion: the Spectral Embedding time complexity is

$$\boxed{\mathcal{C}_{spectral} = \mathcal{O}(|V|^3)}$$

3 Question 3

- Single Layer GNN: The node's embedding implicitly created by the weight matrix of the hidden layer will only depend on its neighbors, which will lead to less robust information propagation and processing. The node's own information will be absent of its embedding.

- Two-Layer GNN: The node's information will be forgotten during the first hidden layer but will partially come back during the second hidden layer through its neighbors embeddings. The node's own information won't be lost but will come and go, which may lead to instability.

4 Question 4

The adjacency matrix of the star graph is $A_{S_4} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$.

Then we compute $\tilde{A}_{S_4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$ and $\tilde{D}_{S_4} = \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}$.

We get $\tilde{D}_{S_4}^{-\frac{1}{2}} = \begin{bmatrix} 1/2 & 0 & 0 & 0 \\ 0 & 1/\sqrt{2} & 0 & 0 \\ 0 & 0 & 1/\sqrt{2} & 0 \\ 0 & 0 & 0 & 1/\sqrt{2} \end{bmatrix}$

We then have $\hat{A}_{S_4} = \tilde{D}_{S_4}^{-\frac{1}{2}} \times \begin{bmatrix} 1/2 & 1/\sqrt{2} & 1/\sqrt{2} & 1/\sqrt{2} \\ 1/2 & 1/\sqrt{2} & 0 & 0 \\ 1/2 & 0 & 1/\sqrt{2} & 0 \\ 1/2 & 0 & 0 & 1/\sqrt{2} \end{bmatrix} = \begin{bmatrix} 1/4 & 1/\sqrt{2} & 1/2\sqrt{2} & 1/2\sqrt{2} \\ 1/2\sqrt{2} & 1/2 & 0 & 0 \\ 1/2\sqrt{2} & 0 & 1/2 & 0 \\ 1/2\sqrt{2} & 0 & 0 & 1/2 \end{bmatrix}$.

And finally, $ReLU(\hat{A}XW^0) = ReLU\left(\begin{bmatrix} \frac{1+3\sqrt{2}}{4} \\ \frac{2+\sqrt{2}}{4} \\ \frac{2+\sqrt{2}}{4} \\ \frac{2+\sqrt{2}}{4} \end{bmatrix} \times \begin{bmatrix} 0.5 & -0.2 \end{bmatrix}\right)$

So,

$$Z^0 = \begin{bmatrix} \frac{1+3\sqrt{2}}{8} & 0 \\ \frac{2+2\sqrt{2}}{8} & 0 \\ \frac{2+2\sqrt{2}}{8} & 0 \\ \frac{2+2\sqrt{2}}{8} & 0 \end{bmatrix}$$

Let's compute Z^1: $ReLU(\hat{A}Z^0W^1) = ReLU\left(\begin{bmatrix} \frac{1+3\sqrt{2}}{8} & 0 \\ \frac{2+2\sqrt{2}}{8} & 0 \\ \frac{2+2\sqrt{2}}{8} & 0 \\ \frac{2+2\sqrt{2}}{8} & 0 \end{bmatrix} \times \begin{bmatrix} 0.3 & -0.4 & 0.8 & 0.5 \\ -1.1 & 0.6 & -0.1 & 0.7 \end{bmatrix}\right)$

Thus,

$$\boxed{Z^1 = \begin{bmatrix} \frac{1+3\sqrt{2}}{8} \times 0.3 & 0 & \frac{1+3\sqrt{2}}{8} \times 0.8 & \frac{1+3\sqrt{2}}{8} \times 0.5 \\ \frac{2+2\sqrt{2}}{8} \times 0.3 & 0 & \frac{2+2\sqrt{2}}{8} \times 0.8 & \frac{2+2\sqrt{2}}{8} \times 0.5 \\ \frac{2+2\sqrt{2}}{8} \times 0.3 & 0 & \frac{2+2\sqrt{2}}{8} \times 0.8 & \frac{2+2\sqrt{2}}{8} \times 0.5 \\ \frac{2+2\sqrt{2}}{8} \times 0.3 & 0 & \frac{2+2\sqrt{2}}{8} \times 0.8 & \frac{2+2\sqrt{2}}{8} \times 0.5 \end{bmatrix}}$$

Let's proceed the same for C_4:

$$A_{C_4} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

$$\tilde{A}_{C_4} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

$$\tilde{D}_{C_4} = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 3 \end{bmatrix}$$

$$\hat{A}_{C_4} = \begin{bmatrix} 1/3 & 1/3 & 0 & 1/3 \\ 1/3 & 1/3 & 1/3 & 0 \\ 0 & 1/3 & 1/3 & 1/3 \\ 1/3 & 0 & 1/3 & 1/3 \end{bmatrix}$$

$$ReLU(\hat{A}XW^0) = ReLU\left(\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \times \begin{bmatrix} 0.5 & -0.2 \end{bmatrix}\right)$$

So we get,

$$Z^0 = \begin{bmatrix} 0.5 & 0 \\ 0.5 & 0 \\ 0.5 & 0 \\ 0.5 & 0 \end{bmatrix}$$

Finally,

$$ReLU(\hat{A}Z^0W^1) = ReLU\left(\begin{bmatrix} 0.5 & 0 \\ 0.5 & 0 \\ 0.5 & 0 \\ 0.5 & 0 \end{bmatrix} \times \begin{bmatrix} 0.3 & -0.4 & 0.8 & 0.5 \\ -1.1 & 0.6 & -0.1 & 0.7 \end{bmatrix}\right)$$

and,

$$\boxed{Z^1 = \begin{bmatrix} 0.5 \times 0.3 & 0 & 0.5 \times 0.8 & 0.5 \times 0.5 \\ 0.5 \times 0.3 & 0 & 0.5 \times 0.8 & 0.5 \times 0.5 \\ 0.5 \times 0.3 & 0 & 0.5 \times 0.8 & 0.5 \times 0.5 \\ 0.5 \times 0.3 & 0 & 0.5 \times 0.8 & 0.5 \times 0.5 \end{bmatrix} = \begin{bmatrix} 0.15 & 0 & 0.4 & 0.25 \\ 0.15 & 0 & 0.4 & 0.25 \\ 0.15 & 0 & 0.4 & 0.25 \\ 0.15 & 0 & 0.4 & 0.25 \end{bmatrix}}$$

- We observe that for S_4, the first row of Z^1 is different from the 3 others which are identical. The explanation is that each row is the embedding of a node. Thus the first row corresponds to the embedding of the central node and the 3 others correspond to the 3 degree-1 node's embeddings. The GNN successfully captures the structure of the given graph.

- The second example, with C_4, also shows that GNN performs well at capturing the graph structure. The 4 rows of Z^1 are identical. That's completely logic because the 4 nodes of C_4 all have the same role in the graph.

5 Visual results from task 4, 5 and 13

5.1 Task 4

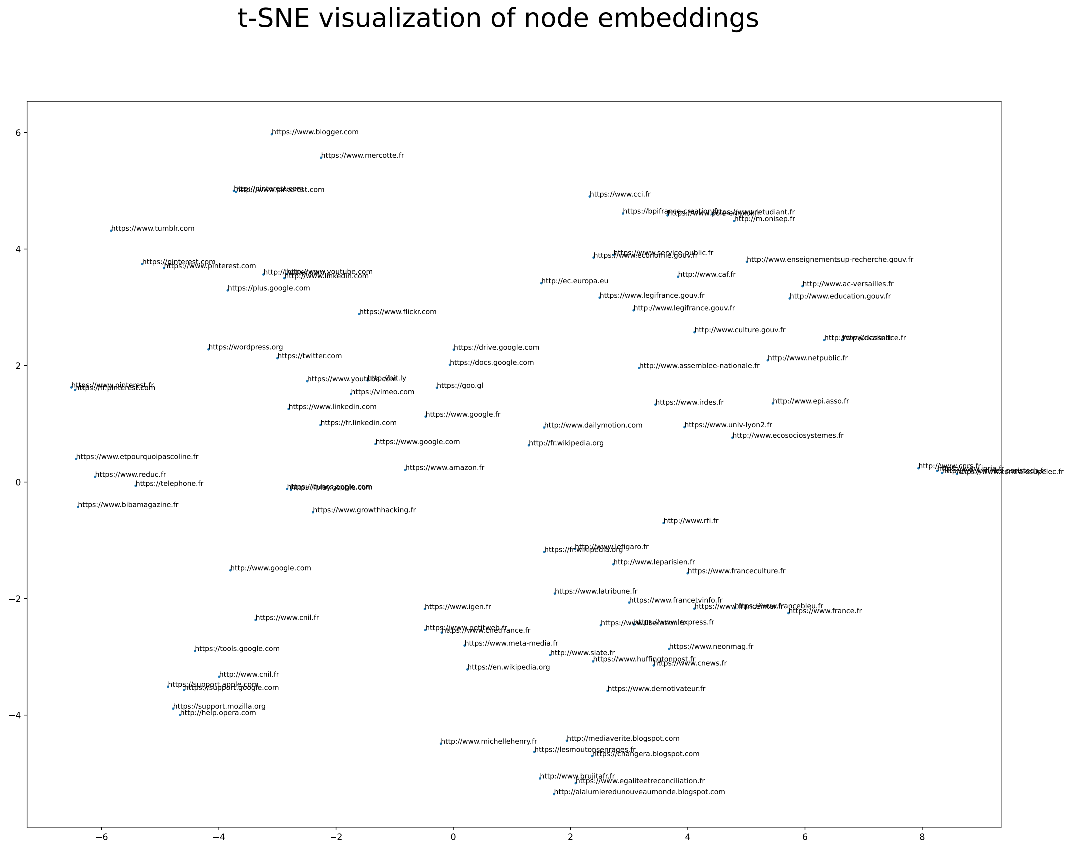

Figure 1: t-SNE on the french web dataset.

5.2 Task 5

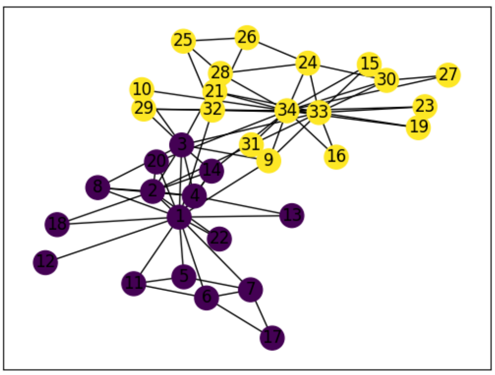

Figure 2: The 2 classes of the karate dataset network.

5.3 Task 13

Figure 3: t-SNE on Cora dataset.

References