1 Algorithm

1. Given graph $G = (V, E)$ as input, define its adjacency matrix P where element p_{ij} is the weight (probability) of edge $(i, j) \in E$.

   - p_{ij} is missing for $(i, j) \notin E$.
   - Number of nodes $N = |V|$, number of edges $M = |E|$.
   - $p_{ij} = e^{-\gamma \|x_i - x_j\|_2^2}$ where $x_i, x_j \in \mathbb{R}^D$ are the attribute vectors of nodes i, j.
   - Time complexity $O(MD)$.
   - **Issue**: how do we choose a value of γ?
   - **Issue**: is it better to define $p_{ij} = \frac{1}{2}\left(\frac{x_i^\top x_j}{\|x_i\|\|x_j\|} + 1\right)$ for DeepWalk features, or $p_{ij} = \frac{x_i^\top x_j}{\|x_i\|\|x_j\|}$ for non-negative DeepWalk features?

2. Optimize the following objective function to factorize $P \in \mathbb{R}_+^{N \times N}$ into two non-negative matrices $W \in \mathbb{R}_+^{K \times N}, H \in \mathbb{R}_+^{K \times N}$:

$$\arg\min_{W,H} L = \arg\min_{W,H} \frac{1}{2} \sum_{(i,j) \in E} \left(p_{ij} - w_i^\top h_j\right)^2 + \lambda \left(\sum_{i \in V} \|w_i\|_1 + \sum_{j \in V} \|h_j\|_1 \right)$$

   - i.e. non-negative matrix factorization $P \approx W^\top H$.
   - $K \ll N, K \ll M$ is the number of latent factors.
   - $w_i \in \mathbb{R}^K$ is the i-th column vector of W, so is h_j of H.
   - For those $(i, j) \notin E$, $w_i^\top h_j$ predicts their connection likelihood.
   - Optimization algorithm: stochastic gradient descent (SGD).
     - $\frac{\partial L_{ij}}{\partial w_i} = -(p_{ij} - w_i^\top h_j) h_j + \lambda$.
     - $\frac{\partial L_{ij}}{\partial h_j} = -(p_{ij} - w_i^\top h_j) w_i + \lambda$.
   - Time complexity $O(MK)$ for each epoch.
   - **Issue**: how do we choose a value of K and λ?

3. Let $w_i = \mathbf{0}$ if node i has no out-link; let $h_j = \mathbf{0}$ if node j has no in-link.

   - Since the vectors are never learned (i.e. updated) in SGD.
   - We do not want the noise to influence the later steps.

4. Normalize each row of prediction matrix $W^\top H$.

   - Sum $\sum_{j \in V} w_i^\top h_j = w_i^\top \sum_{j \in V} h_j$ for row i; hence we compute $\alpha = \sum_{j \in V} h_j \in \mathbb{R}^K$, followed by $w_i^\top \alpha$.
   - Let $s = W^\top \alpha \in \mathbb{R}^N$ be the row-sum vector where $s_i = w_i^\top \alpha$.
   - Let Y be the normalized matrix where $s^\top$ divides every row of W; be careful to skip zero division $s_i = 0$ for dangling nodes i.
   - Time complexity $O(NK)$.

5. Run PageRank until convergence:

$$\pi = (1-d)r + d\bigg(H^\top Y\pi + br\bigg)$$

- Reset probability vector $r = \frac{1}{N}\mathbf{1}$.
- We first compute $Y\pi$ to reduce time complexity.
- Dangling node (i.e. zero-out-degree nodes) term br
  - Scalar $b = \sum_{i\in B}\pi_i$ where B is the set of dangling nodes.
- Time complexity $O(NK)$ for each iteration.
- **Issue**: could it achieve better performance if we re-define $p_{ij} = \beta e^{-\gamma\|x_i - x_j\|_2^2} + (1-\beta)\pi_j$ and run the algorithm again?