Paper Reading Task

node2vec : Scalable Feature Learning for Networks

Manas Mittal

2024121003

More detailed summary can be found in the **GitHub** repo

Brief Overview and Broad Area

- Broad Area :
  - **Graph Representational Learning** automates feature extraction by learning node representations. Ensures **similar nodes** have **similar embeddings***.
  - The notion of similarity generally revolves around the nodes features and labels and the structure of the graph.
- Two major paradigms of looking toward this problem :
  - **Task-specific embeddings :** Example : GNN and other supervised techniques that generate embeddings specific to the downstream task.
  - **Task-agnostic embeddings**: Example: **Node2Vec** generates general-purpose embeddings, independent of the prediction task.

*notion of similarity in the embedding space and the graph should be clearly defined

Challenges and Approaches in Feature Learning for Graphs

Manual Feature Engineering

- Labor-intensive and tailored to specific tasks
- Lacks generalizability across different applications.

Optimization-Based Feature Learning

- **Supervised Learning:** High accuracy but computationally expensive and task-specific.
- **Unsupervised Learning:** Less computationally expensive but lower accuracy due to costly matrix decompositions in large networks.

Neighbourhood-Based Learning Objective

- Focuses on preserving local node neighbourhoods.
- Efficient with **stochastic gradient descent (SGD)**.

Node2vec lies here

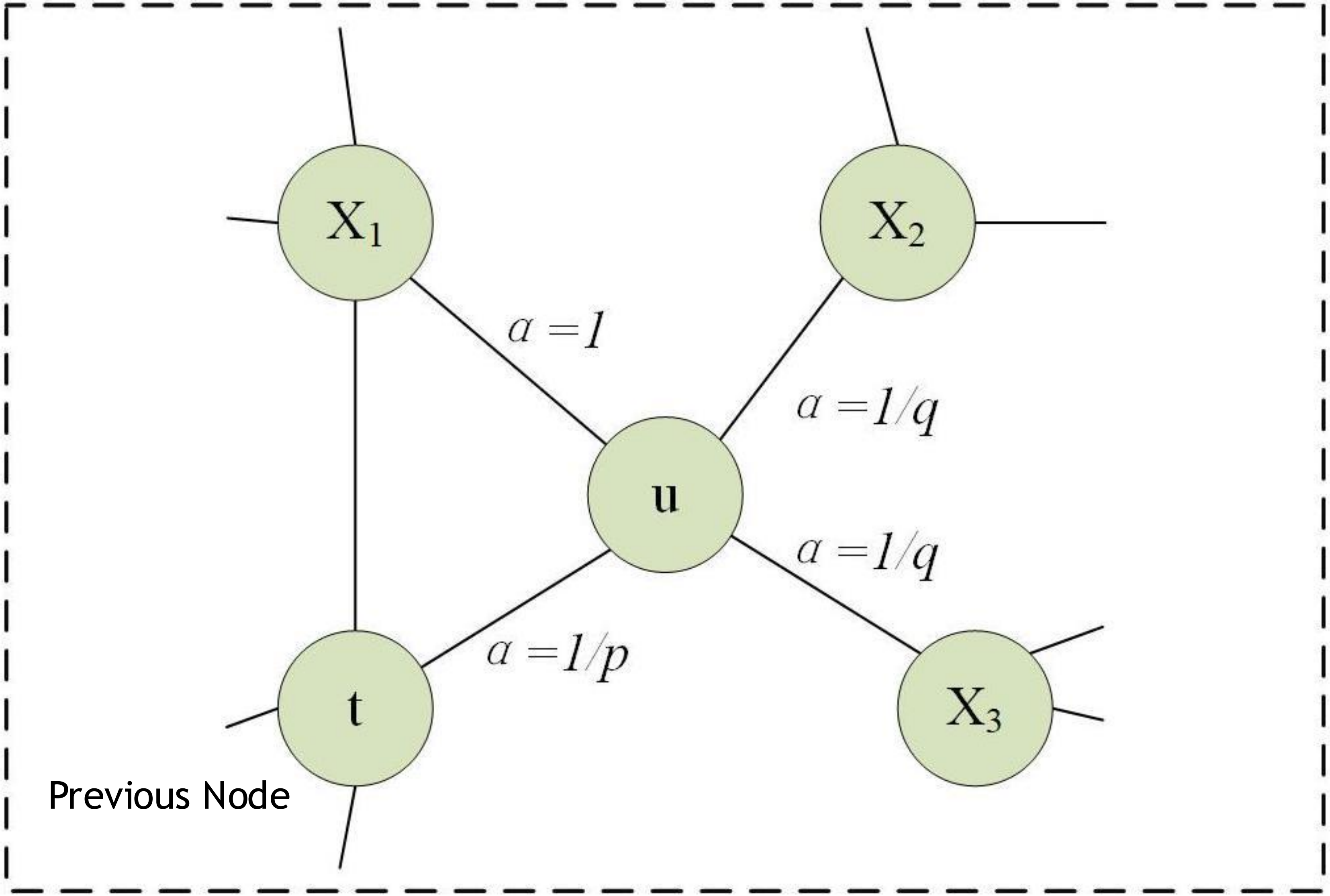
node2vec approach

- **Flexible neighbourhood definition** to overcome rigid prior methods.
- **2nd order biased Random walk strategy.**
- 2 different notion of similarity :
  - **Homophily**: Groups nodes from the same community closely.
  - **Structural Similarity**: Embeds nodes with similar roles together.
- Hyperparameters (p , q)* :
  - Control the balance between homophily and structural similarity.
- Other hyperparameters :
  - Number of walks
  - Walk length
  - Dimension d and Context window size (for the SkipGram model)

*these parameters can be learnt in a semi-supervised setting

Understanding the hyperparameter

- **p (Return Parameter):**
  - Governs the likelihood of revisiting the previous node.
  - $p > \max(q, 1) \rightarrow$ More **exploratory** walks (avoids backtracking).
  - $p < \min(q, 1) \rightarrow$ Less **exploratory** walks (more backtracking) and more local.
- **q (In out parameter):**
  - Defines the tendency to explore local vs. distant nodes.
  - $q > 1 \rightarrow$ More local (BFS like behaviour)
  - $q < 1 \rightarrow$ More outward exploration
- Why BFS captures structural similarity ?
  - Nodes with similar roles can be identified by examining their immediate neighbours, making BFS effective for capturing local structural relationships.
- Why DFS captures homophily ?
  - In DFS, the sampled nodes more accurately reflect a macro-view of the neighbourhood which is essential in inferring communities based on homophily.



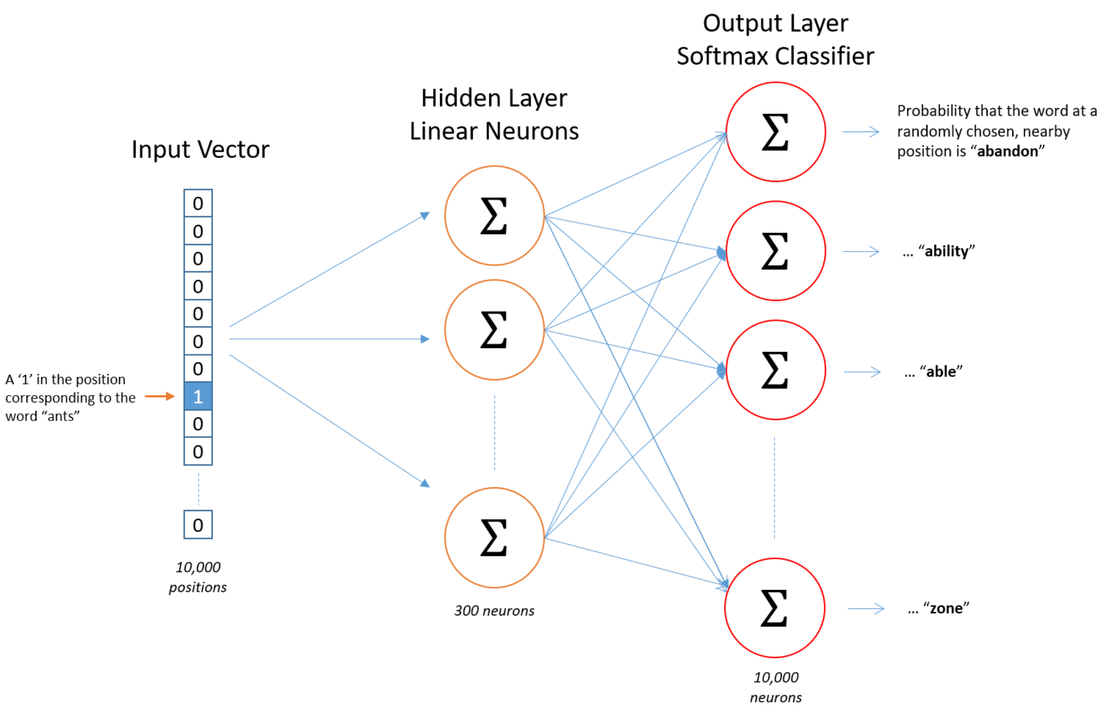
Understanding the RANDOM WALK STRATEGY used by Node2vec

node2vec algorithm

- **Generating random walks :** Simulate the fixed length biased random walks over the graph by selecting random nodes. Normalize the probabilities before sampling a random variable and taking next step. (This normalization of probability can happen in the pre-processing step itself)
- **Using Random Walk to optimize the Skip-Gram optimization objective :** We will now consider these random walks as text and will try to learn the similarities of nodes that come together on a lot of random walks.
- **Getting edge features :** We can use the correspondi nodes participating in the edge to get the edge features.

Detour : Skip-Gram

- **Aim :** To predict the context word in a sliding window based on the centre word.
- **Working :** Input to the model will be one-hot encoding vector of the centre word. The weights of the model will try to predict the remaining word which will then be corrected in a supervised manner by the actual words.



Credit : http://www.mccormickml.com

Detour : Optimization objective behind node2vec

- In order to optimize this function, we first assume conditional independence. Then, Pr(node | u) can be expressed using the SoftMax probability function.

$$\max_f \sum_{u \in V} \log Pr(N_S(u) | f(u)).$$

- This will lead to the following optimization objective :

$$\max_f \sum_{u \in V} \left[-\log Z_u + \sum_{n_i \in N_S(u)} f(n_i) \cdot f(u) \right].$$

Detour : Optimization objective behind node2vec

- Optimizing this function can be very expensive because of the O(|V|^2) complexity of calculating the denominator.
- We then do **negative sampling** to approximate the denominator leading to linear time complexity.
- After the approximation we use SGD* to optimize the function.

* Descent of negative of max likelihood function

Experiments and Evaluations

- **Evaluation Tasks:**
  - **Multi-label Node Classification**: Assigning multiple labels per node.
  - **Link Prediction**: Predicting edges between node pairs.
- **Comparison with Baselines:**
  - Evaluated against **LINE, Spectral Clustering, and Deep-Walk**.
  - **Node2Vec outperforms all baselines** using **macro F1 score**.
- **Parameter Sensitivity (BlogCatalog Dataset):**
  - **Macro F1 score decreases** with higher **p** and **q**.
  - **Improves** with increasing **dimension, number of walks, walk length, and context window**, before saturating.
- **Scalability:**
  - **Linear scaling** with the number of nodes.
- **Perturbation Analysis (BlogCatalog Dataset):**
  - **Missing Edges**: Performance (**Macro F1 score**) declines **linearly** with a small slope.
  - **Noisy Edges**: Initial sharp drop in performance, then a slower decline.

Strengths of the paper

- **Intuitive & Practical:** Simple, easy to understand, and implement.
- **Scalability**: Uses **parallelizable algorithms** for efficient execution.
- **Performance**: Outperforms **state-of-the-art (SOTA)** methods.
- **Cross-Domain Innovation**: Integrates **NLP techniques (e.g., Skip-Gram)** into graph learning.
- **Flexibility**: Scalable and adaptable for various applications.
- **Detailed Parameter Analysis :** variation of different parameter of the BlogCatalog dataset.
- **Robust to MISSING and NOISY edges**

Limitations of the paper

- **Limited Scalability:**
  - Embedding large graphs is computationally expensive.
  - High storage requirements as each node's embedding is learned separately.
- **Inability to Handle Dynamic Graphs:**
  - Learns embeddings **once**; cannot accommodate new nodes on the fly.
- **Ignores Node Features:**
  - Focuses only on graph **structure**, unlike Graph Neural Networks (GNNs).
  - Not suitable for **end-to-end learning** with feature-rich nodes.
- **No Comparison with GNNs:**
  - GNNs are powerful and scalable but are **not benchmarked** in the paper.
- **Limited Insight into Edge Features:**
  - Lacks empirical evaluation of how edge attributes impact embeddings.