Graph Representation Learning [1]

Notes[2]

Winter 2021

[1] William L. Hamilton

[2] Authors: Eeshaan Jain

Keyphrases: Graph Representation Learning. Graph Statistics. Graph Embeddings. Graph Neural Networks. Graph Generative Models.

This set of notes begin by introducing graphs, and learning tasks on it. [Introduction will be updated as file is updated.]

Contents

1 Introduction to Graphs and Learning on Graphs

1.1 Basic definitions

Definition 1.1 (*Graph*)

A graph $G = (V, E)$ is defined by a set of nodes V and a set of edges between these nodes. We denote an edge going from $u \in V$ to $v \in V$ as $(u, v) \in E$.

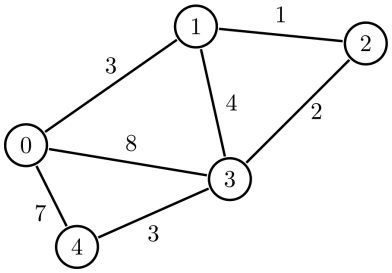

Figure 1: Undirected graph with ordered edges and nodes

Note that definition 1.1 is general, and most of the times we are only concerned with *simple graphs*, which have the following properties:

- For any two nodes in the set V, there exists at most one edge between them.
- There are no self-loops, i.e for any $u \in V$, $(u, u) \notin E$.
- All edges of the graph are *undirected*, i.e for $u, v \in V$, $(u, v) \in E \Leftrightarrow (v, u) \in E$.

Definition 1.2 (*Adjacency Matrix*)

For a simple graph $G = (V, E)$, the adjacency matrix $\mathbf{A} \in \mathbb{R}^{|V| \times |V|}$ is a square matrix with $A_{uv} = 1$ if there is an edge from vertex u to v, and $A_{uv} = 0$ otherwise.

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

The adjacency matrix for Fig 1.

Since we are dealing with simple graphs, some immediate consequences are that

- $\mathbf{A}$ is symmetric, i.e $\mathbf{A}^T = \mathbf{A}$.
- $A_{uu} = 0 \; \forall \; u \in V$, i.e all diagonal elements are zero.

Notice that since we are defining the graph in terms of the matrix $\mathbf{A}$, we need to order the nodes prior to the representation (See Fig 1). An extension to definition 1.2 occurs when we allow the use of weighted edges, and in that case for $u, v \in V$, $A_{uv} \in [0, 1]$.

1.2 Multi-relational Graphs

We have noted three types of edges till now: undirected, directed and weighted. But for graphs, it is possible to have different types of edges. In such cases, we extend the notation to include relation type $\tau \in \mathcal{R}$ and write that $(u, \tau, v) \in E$ along with defining an adjacency matrix $\mathbf{A}_\tau$ for each type. The above class of graphs are called multi-relational, and are overall represented by an adjacency tensor $\mathcal{A} \in \mathbb{R}^{|V| \times |\mathcal{R}| \times |V|}$. Two major subsets of such graphs are:

1. **Heterogeneous graphs:** We can partition the set of nodes into k disjoint sets $V = \bigcup_{i=1}^{k} V_i$, and thus we have types imbued on

nodes. Usually in such graphs, we pose constraints on edges, for example they connect nodes of different types only (*multipartite graphs*).

2. **Multiplex graphs:** We can decompose such graphs into a set of layers (see Fig 2), and each node is in every layer with each layer possessing a unique relation. Edges will be present intra-layer and inter-layer.

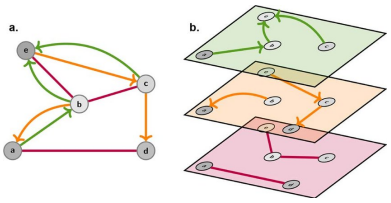

Figure 2: Layered multiplex graph. Source: [1]

1.3 *Information representation*

Usually we have features representing graph-level features - may it be on nodes, edges or the entire graph itself. The most common one is node-level, and we associate an $m-$dimensional feature vector with each node, thus overall having a matrix $\mathbf{X} \in \mathbb{R}^{|V| \times m}$.

Now, let's go over some common graphical learning tasks.

1.4 *Node Classification*

The goal of node classification is to predict a label y_u associated with all nodes $u \in V$ when we have been only given the true labels of a subset of the nodes $V_{train} \subset V$. A few examples in literature of the task are:

1. Interactome protein function classification: [2]
2. Document classification based on citation graphs: [3]

Though it seems similar to supervised classification, there is a difficulty that the nodes in a graph are not *i.i.d.* and thus we need to model dependencies between data points. This is exactly what node classification takes a benefit of through

1. **Homophily:**

References

[1] T. Gaudelet, B. Day, A. Jamasb, J. Soman, C. Regep, G. Liu, J. Hayter, R. Vickers, C. Roberts, J. Tang, D. Roblin, T. Blundell, M. Bronstein, and J. Taylor-King, "Utilizing graph machine learning within drug discovery and development," *Briefings in Bioinformatics*, vol. 22, 05 2021.

[2] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *CoRR*, vol. abs/1706.02216, 2017. [Online]. Available: http://arxiv.org/abs/1706.02216

[3] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *CoRR*, vol. abs/1609.02907, 2016. [Online]. Available: http://arxiv.org/abs/1609.02907