weighted graph

Author and editors

Alexander Jung*, Department of Computer Science, Aalto University, Espoo, Finland (ORCID 0000-0001-7538-0990)
Editors: Konstantina Olioumtsevits (ORCID 0000-0003-2697-8659), Ekkehard Schnoor (ORCID 0000-0001-8252-1499), Aalto University, Espoo, Finland
*Corresponding author: alex.jung@aalto.fi

Abstract

A weighted graph is a graph whose edges carry nonnegative numbers, the edge weights, that measure how strongly two nodes are tied. It consists of a finite set of nodes, a set of undirected edges joining pairs of distinct nodes, and a positive weight for each edge; a pair of nodes that is not an edge has weight zero. The weights form the weight matrix $\mathbf{A}$, a symmetric matrix with a zero diagonal that is also the adjacency matrix of the graph, and the sum of a row is the weighted node degree of its node. In machine learning (ML), the nodes are often the data points of a dataset and the weight of an edge is the similarity of two data points, e.g., a Gaussian function of the Euclidean distance between their feature vectors, kept for the k nearest neighbors of each data point. Graph clustering seeks subsets of nodes whose outgoing edges carry little weight. Transfer learning and federated learning (FL) use the weights of a graph of learning tasks as the coefficients of a penalty term that couples the hypotheses of neighboring tasks.

Definition

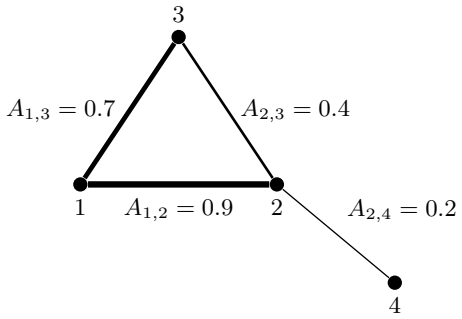
Consider ten days of the year 2024 at Krems an der Donau, every 37th day from 1 January to 29 November, each a data point with a feature vector $\mathbf{x} \in \mathbb{R}^2$ that holds the morning minimum and the maximum daytime temperature of the day. Two days are similar when their feature vectors are close. A drawing records the similarities: each day is a dot, called a node, two similar days are joined by a line, called an edge, and a number on the edge says how similar the two days are. Fig. 2(a) shows this drawing for the ten days, with an edge between two days whenever one of them is among the three days nearest to the other and a number between 0 and 1 that falls off with the Euclidean distance between them. Nodes, edges, and a number on every edge make a weighted graph (von Luxburg, 2007, Sect. 2.1; Newman, 2010).

Formally, a weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consists of a finite set of nodes $\mathcal{V} = \{1, \ldots, n\}$, a set $\mathcal{E}$ of edges, and an edge weight for each edge. Each edge is an unordered pair $\{i, i'\}$ of two distinct nodes, so the graph is an undirected graph without self-loops, and the two nodes of an edge are neighbors of each other. The weight of an edge $\{i, i'\} \in \mathcal{E}$ is a positive number $A_{i,i'} > 0$, and a pair of nodes that is not an edge has weight $A_{i,i'} = 0$ (Chung, 1997, Sect. 1.4). With this convention, the weights determine the edge set, and they are collected in

the weight matrix

$$\mathbf{A} \in \mathbb{R}^{n \times n} \text{ with entries } A_{i,i'} = A_{i',i} \geq 0 \text{ and } A_{i,i} = 0 \text{ for } i, i' \in \mathcal{V}. \qquad (1)$$

The weight matrix (1) is a symmetric matrix with a zero diagonal. It is the adjacency matrix of the graph: the entry in row i and column i' is the weight of the edge between i and i', or zero when there is no edge (Hastie et al., 2009, Sect. 14.5.3; von Luxburg, 2007, Sect. 2.1). An unweighted graph is the special case in which every edge has weight 1, so the entries of its weight matrix (1) are 0 and 1 (Chung, 1997, Sect. 1.4). Fig. 1 shows a weighted graph with four nodes beside its weight matrix.



$$\mathbf{A} = \begin{pmatrix} 0 & 0.9 & 0.7 & 0 \\ 0.9 & 0 & 0.4 & 0.2 \\ 0.7 & 0.4 & 0 & 0 \\ 0 & 0.2 & 0 & 0 \end{pmatrix}$$

Figure 1: A weighted graph with four nodes $\mathcal{V} = \{1, 2, 3, 4\}$ and four edges, each drawn as a line whose width grows with its weight, beside its weight matrix (1). The entry in row 2 and column 4 is the weight 0.2 of the edge $\{2, 4\}$, and the zero in row 1 and column 4 records that the nodes 1 and 4 are not joined by an edge

The sum of the weights of the edges at a node is its weighted node degree,

$$d^{(i)} = \sum_{i' \in \mathcal{V}} A_{i,i'}, \qquad (2)$$

which is the sum of row i of the weight matrix (Chung, 1997, Sect. 1.4; von Luxburg, 2007, Sect. 2.1). For an unweighted graph the node degree (2) counts the neighbors of the node. The Laplacian matrix of the graph is built from the weight matrix and the degrees (2), and its eigenvectors are what spectral clustering computes (von Luxburg, 2007, Sect. 3).

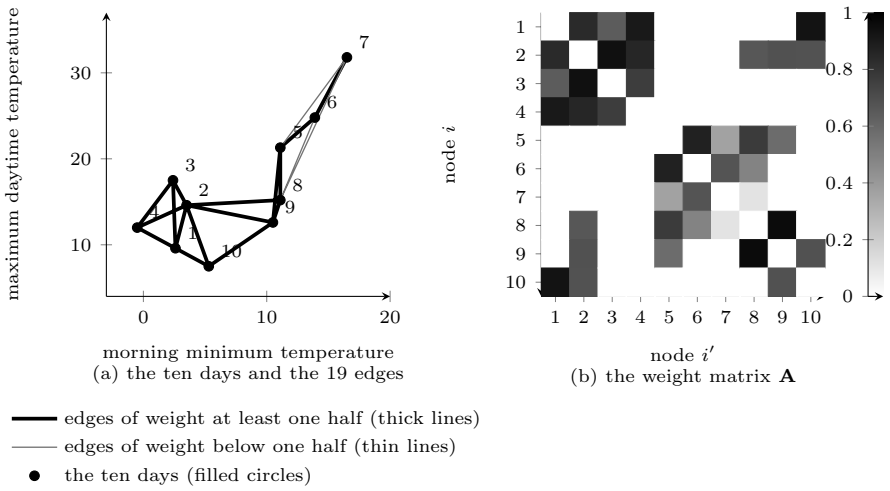
What a weight means is fixed by the application, and two conventions are in use. In a road network, the nodes are intersections, the edges are road segments, and the weight of an edge is the length of the segment or the time needed to travel it (Newman, 2010). A large weight then means far apart, and the shortest route between two intersections is the path whose weights add up to the smallest value. In machine learning (ML), the weight of an edge usually measures similarity: a large weight means close together. A method that reads a similarity as a length, or a length as a similarity, inverts every conclusion

it draws, so the convention must be stated. The graph in Fig. 2(a) uses the similarity convention.

The main use of a weighted graph in ML is the similarity graph of a dataset $\mathcal{D} = \{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$. Its nodes are the data points, $\mathcal{V} = \{1, \dots, m\}$ with node r standing for $\mathbf{x}^{(r)}$, and the weight of an edge measures the similarity of its two data points. For feature vectors in $\mathbb{R}^d$, a common choice is the Gaussian weight

$$A_{r,r'} = \exp\Big(- \Big\| \mathbf{x}^{(r)} - \mathbf{x}^{(r')} \Big\|_2^2 / (2\sigma^2) \Big), \tag{3}$$

which is 1 for two identical feature vectors and falls off with their Euclidean distance at a rate set by the width $\sigma > 0$ (Hastie et al., 2009, Sect. 14.5.3; von Luxburg, 2007, Sect. 2.2; Shalev-Shwartz and Ben-David, 2014, Sect. 22.3). Giving the Gaussian weight (3) to every pair of data points produces a fully connected graph with $m(m-1)/2$ edges, whose weight matrix has no zero off the diagonal. A sparse alternative is the k-nearest-neighbor graph (k-NN graph): an edge joins r and r' when one of the two data points is among the k nearest data points of the other, it carries the Gaussian weight (3), and every other pair has weight zero (von Luxburg, 2007, Sect. 2.2). The width σ and the number k of neighbors are hyperparameters of the construction; a usual choice for σ is the mean distance of a data point to its k-th nearest neighbor (von Luxburg, 2007, Sect. 8.1). For the ten days at Krems and $k = 3$, this choice gives $\sigma = 8.27$ degrees, and 19 of the 45 pairs of days are edges. Their weights range from 0.11, between 10 August and 16 September, to 0.95, between 16 September and 23 October, and the weighted degrees (2) from 1.13, on 10 August, to 4.61, on 7 February. Fig. 2(b) shows the weight matrix.



(a) the ten days and the 19 edges (b) the weight matrix $\mathbf{A}$

Figure 2: (a) Ten of the 366 days of 2024 at Krems, every 37th day, numbered 1 to 10 in order of date (node 1 is 1 January, node 10 is 29 November), as the nodes of their 3-nearest-neighbor graph, drawn in the plane of the morning minimum and the maximum daytime temperature: an edge of weight at least 0.5 is a thick line, an edge of weight below 0.5 a thin line. (b) The weight matrix (1) of the graph: the darker the cell in row i and column i', the larger the weight $A_{i,i'}$, and a white cell is a pair of days that is not an edge. Data generated by `pythondemos/weightedgraph.py`

A similarity graph turns clustering into a question about the weights. A cluster is a subset $\mathcal{C} \subseteq \mathcal{V}$ of nodes whose inner edges carry large weights and whose outgoing edges carry small ones. The total weight of the edges that leave the subset,

$$\sum_{i \in \mathcal{C}} \sum_{i' \in \mathcal{V} \setminus \mathcal{C}} A_{i,i'}, \tag{4}$$

is the cut of $\mathcal{C}$. Graph clustering methods such as spectral clustering choose the clusters to make their cuts (4) small relative to their sizes (von Luxburg, 2007, Sect. 5; Shalev-Shwartz and Ben-David, 2014, Sect. 22.3.1; Shi and Malik, 2000). For the ten days at Krems, take the seven days with a maximum temperature below 18 degrees as the subset $\mathcal{C}$. Its cut (4) is 1.92 out of a total edge weight of 13.25: the four edges that leave it are 21 percent of the edges and carry 14.5 percent of the weight. On all 366 days of 2024 and with $k = 10$, the graph has 2209 edges. The edges between the 186 cold and the 180 warm days carry 3.7 percent of the total weight, and no edge joins a day of January or February with a day of July or August. On a photograph cut into patches, the same construction with the mean color of a patch as its feature vector gives a weighted graph whose clusters are the regions of an image segmentation.

A weighted graph also enters empirical risk minimization (ERM) through a penalty term. In transfer learning and in federated learning (FL), the nodes are

learning tasks, each with its own local dataset and a local hypothesis $h^{(i)}$, and the weight $A_{i,i'}$ states how similar two tasks are. Generalized total variation minimization (GTVMin) minimizes the sum of the local training errors plus the generalized total variation (GTV)

$$\sum_{\{i,i'\}\in\mathcal{E}} A_{i,i'} d^{(i,i')}, \tag{5}$$

with $d^{(i,i')}$ a measure of the discrepancy between $h^{(i)}$ and $h^{(i')}$. The GTV (5) couples the tasks: an edge with a large weight pulls the two hypotheses at its nodes together, and a zero weight leaves them independent. The network Lasso is the special case with the Euclidean distance between the local model parameters as the discrepancy (Hallac et al., 2015). Adding a penalty term is one of the three routes that regularization distinguishes, and the weighted graph is where this penalty term is specified. On a similarity graph of labeled and unlabeled data points, the penalty term (5) with the squared difference $\big(h(\mathbf{x}^{(r)}) - h(\mathbf{x}^{(r')})\big)^2$ as the discrepancy is the Laplacian regularizer of semi-supervised learning (SSL) (Belkin et al., 2006, Sect. 2.2). It asks a hypothesis to take similar values on two data points joined by an edge of large weight, which is how the unlabeled data points shape the hypothesis. A graph neural network (GNN) uses the weights for a third purpose, to combine the feature vectors of the neighbors of each node into a new feature vector of the node.

Cross References

graph, undirected graph, edge weight, adjacency matrix, Laplacian matrix, node degree, neighbor, similarity graph, k-NN graph, graph clustering, spectral clustering, clustering, transfer learning, GTV, GTVMin, SSL, GNN, matrix

References

1. Belkin et al. (2006). Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. J. Mach. Lear. Research, 7, pp. 2399–2434.

2. Chung (1997). Spectral Graph Theory. American Mathematical Society, Providence, RI, USA.

3. von Luxburg (2007). A tutorial on spectral clustering. Statist. Comput., 17(4), pp. 395–416.

4. Hallac et al. (2015). Network Lasso: Clustering and Optimization in Large Graphs. In: Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, pp. 387–396.

5. Newman (2010). Networks: An Introduction. Oxford Univ. Press, New York, NY, USA.

6. Shalev-Shwartz and Ben-David (2014). Understanding Machine Learning: From Theory to Algorithms. Cambridge Univ. Press, New York, NY, USA.

7. Shi and Malik (2000). Normalized Cuts and Image Segmentation. IEEE Trans. Pattern Anal. Mach. Intell., 22(8), pp. 888–905.

8. Hastie et al. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd ed., Springer Science+Business Media, New York, NY, USA.