edge weight

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

An edge weight is a nonnegative number $A_{i,i'}$ attached to the edge $\{i, i'\}$ of a weighted graph that measures how strongly the two nodes it joins are connected. Collected in the weight matrix $\mathbf{A}$, the edge weights of an undirected graph form a symmetric matrix with a zero entry for every pair that is not an edge. In machine learning (ML), the nodes are often the data points of a dataset, and the weight of a pair is the Gaussian similarity of their feature vectors, a decreasing function of their Euclidean distance with a width σ as hyperparameter; a k-nearest-neighbor graph (k-NN graph) keeps only the weights of nearby pairs. Edge weights can also encode a relation given with the data, such as citations between papers or the collaboration between the devices of a federated learning network (FL network). ML methods use the weights through weighted sums over the edges. The quadratic form of the Laplacian matrix is the weighted sum of the squared differences of a node signal, which for the indicator of a cluster is the total weight of the cut edges. The penalty term of convex clustering charges the distance between two cluster centroids in proportion to the weight of their edge. Edge weights are given before training and differ from the weights of an artificial neural network (ANN), which are model parameters.

Definition

Consider eight days of 2024 at Krems an der Donau, every 46th day from 1 January, 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 with similar temperatures are similar data points. One way to record the similarity of every pair of days is a graph. The days are its nodes $i = 1, \ldots, n$, here with $n = 8$, and an edge $\{i, i'\}$ between two days carries a number that says how similar they are: the edge weight $A_{i,i'} \geq 0$. In Fig. 1(a), a thick edge joins 1 January and 18 November, whose temperatures differ by less than two degrees Celsius, while no visible edge joins 1 January and 18 August.

A weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$ consists of a node set $\mathcal{V} = \{1, \ldots, n\}$, an edge set $\mathcal{E}$, and an edge weight $A_{i,i'}$ for every edge $\{i, i'\} \in \mathcal{E}$. The weights of an undirected graph without self-loops are collected in the weight matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$. Its entries satisfy

$$A_{i,i'} = A_{i',i} \geq 0, \quad A_{i,i} = 0, \quad \text{and} \quad A_{i,i'} = 0 \text{ for } \{i, i'\} \notin \mathcal{E}. \tag{1}$$

The first condition in the weight matrix (1) holds because the edge $\{i, i'\}$ of an undirected graph is the edge $\{i', i\}$, so the weight matrix is a symmetric matrix. The third condition makes a zero weight and a missing edge the same thing, so the weight matrix determines the graph and is its adjacency matrix (Chung, 1997, Sect. 1.4; von Luxburg, 2007, Sect. 2). An unweighted graph is the special case in which every edge has weight 1. The weighted node degree of node i is the row sum $\sum_{i'=1}^{n} A_{i,i'}$ of the weight matrix (Hastie et al., 2009, Sect. 14.5.3).

In machine learning (ML), edge weights come from two sources. The first source is the feature vectors of the data points: the m data points of a dataset become the nodes of a similarity graph, with $n = m$, and the weight of a pair is a decreasing function of the distance between their feature vectors. The usual choice is the Gaussian similarity

$$A_{i,i'} := \exp\Big(- \Big\| \mathbf{x}^{(i)} - \mathbf{x}^{(i')} \Big\|_2^2 / (2\sigma^2) \Big) \quad \text{for } i \neq i', \tag{2}$$

with a width $\sigma > 0$ (Hastie et al., 2009, Sect. 14.5.3; von Luxburg, 2007, Sect. 2). The Gaussian weight (2) is 1 for identical feature vectors, $\exp(-1/2) \approx 0.61$ at Euclidean distance σ, and about 0.01 at Euclidean distance 3σ. For the eight days and $\sigma = 5$ degrees, the weight is 0.95 for 1 January and 18 November, whose feature vectors are 1.6 degrees apart, and $6 \cdot 10^{-7}$ for 1 January and 18 August, 26.8 degrees apart. Six of the 28 pairs weigh at least 0.5, and eight weigh below 0.01 (Fig. 1(b)). The width σ is a hyperparameter. With $\sigma = 1$, only two pairs weigh at least 0.01, so the graph falls apart into isolated days. With $\sigma = 50$, every pair weighs at least 0.87, so the weights no longer separate similar from dissimilar days. A width σ in the order of the mean distance of a data point to its k-th nearest neighbor is a common choice (von Luxburg, 2007, Sect. 8).

The Gaussian weight (2) is positive for every pair, so every pair is an edge and the weight matrix has no zero off its diagonal. A sparse graph is obtained from the k-nearest-neighbor graph (k-NN graph). The pair $\{i, i'\}$ is an edge if $\mathbf{x}^{(i')}$ is among the k nearest neighbors of $\mathbf{x}^{(i)}$ or $\mathbf{x}^{(i)}$ among those of $\mathbf{x}^{(i')}$. An edge keeps the Gaussian weight (2), and every other weight is set to zero (Hastie et al., 2009, Sect. 14.5.3; von Luxburg, 2007, Sect. 2). For the eight days and $k = 2$, ten of the 28 pairs are edges, and the graph is connected. The k-NN graph is not a threshold on the weights. It keeps 18 May and 18 August at weight 0.07, since 18 May is one of the two days nearest to 18 August. It drops 3 October and 18 November at weight 0.54, since each of the two has two closer days. Fig. 1(b) marks the kept pairs by filled circles and the dropped pairs by open squares. The k-NN graph therefore connects data points in sparse and in dense regions of the feature space alike, which is why it is the recommended first choice (von Luxburg, 2007, Sect. 8).

The second source is a relation given with the data rather than computed from feature vectors: citations between scientific papers, friendships between the users of a social network, links between web pages, or the number of measurements that two weather stations share. In an image, the pixels are the nodes, and the weight of two pixels falls with the difference of their brightness

and with their distance in the image, a construction used in image segmentation by normalized cuts (Shi and Malik, 2000). In a federated learning network (FL network), the nodes are devices, and the edge weight measures the extent of their collaboration, e.g., the capacity of the communication link between them or the statistical similarity of their local datasets. In transfer learning, the nodes are learning tasks, and the weight couples the tasks of nearby weather stations more strongly than those of distant ones.

The edges of an artificial neural network (ANN) also carry weights, and those are a different object: an ANN weight is a model parameter, learned by training, and it can be negative. The edge weights of a weighted graph are nonnegative and are given before training starts; they belong to the data, not to the model.

Edge weights enter ML methods through weighted sums over the edges. A node signal $\mathbf{u} = (u_1, \ldots, u_n)^\top \in \mathbb{R}^n$ assigns one number to every node, such as a label or a prediction. The Laplacian matrix $\mathbf{L}^{(\mathcal{G})}$ of the graph satisfies

$$\mathbf{u}^\top \mathbf{L}^{(\mathcal{G})} \mathbf{u} = \sum_{\{i,i'\} \in \mathcal{E}} A_{i,i'} \big(u_i - u_{i'}\big)^2, \tag{3}$$

the Laplacian quadratic form (von Luxburg, 2007, Prop. 1). It is a weighted sum of squared differences. A heavy edge charges a difference between its two nodes more than a light edge, and the sum is zero for a signal that is constant on every connected component of the graph. For the indicator of a subset $\mathcal{C} \subseteq \mathcal{V}$ of the nodes, $u_i = 1$ for $i \in \mathcal{C}$ and $u_i = 0$ otherwise, the quadratic form (3) is the total weight of the edges between $\mathcal{C}$ and the rest of the graph, the weight of the cut. For the eight days, the five days with a maximum temperature below 20 degrees cut edges of total weight 1.09, which is 15 percent of the total weight 7.44 of all pairs. Splitting the days into the first four and the last four of the year cuts 4.97, or 67 percent. Graph clustering methods seek clusters that cut little weight; spectral clustering approximately minimizes the cut weight divided by the size of the cluster through eigenvectors of $\mathbf{L}^{(\mathcal{G})}$ (von Luxburg, 2007, Sect. 5).

Semi-supervised learning (SSL) on a graph seeks a node signal that agrees with the labels of the few labeled nodes and makes the quadratic form (3) small (Zhu et al., 2003). The weights then carry the labels from labeled to unlabeled nodes. They decide to which data points outside the training set a learned hypothesis is extended, which is the question of generalization. The generalized total variation (GTV) replaces the squared difference in the quadratic form (3) by a measure of the discrepancy between the local models of the two nodes. Generalized total variation minimization (GTVMin) adds it to the local training errors, as in federated learning (FL) and transfer learning. The network Lasso is the case of the unsquared distance between the model parameters of the two local models (Hallac et al., 2015).

Convex clustering uses the edge weights in its objective function. Every data point $\mathbf{x}^{(i)}$ gets its own cluster centroid $\mathbf{w}^{(i)}$, and the cluster centroids minimize

$$\sum_{i=1}^{n} \Big\| \mathbf{x}^{(i)} - \mathbf{w}^{(i)} \Big\|_2^2 + \alpha \sum_{\{i,i'\} \in \mathcal{E}} A_{i,i'} \Big\| \mathbf{w}^{(i)} - \mathbf{w}^{(i')} \Big\|_2, \tag{4}$$

with a regularization parameter $\alpha \geq 0$ (Hocking et al., 2011; Sun et al., 2021, Eq. (1)). The first sum in the objective function (4) is the clustering error of the cluster centroids, the training error that empirical risk minimization (ERM) minimizes; the second sum is a penalty term, so the method regularizes by loss penalization, one of the three routes that regularization distinguishes. The weights decide which pairs of cluster centroids are pulled together. A pair with weight zero is never charged, and a heavy pair is charged in proportion to its weight. For α large enough, the cluster centroids of a heavy pair coincide, which puts the two data points into one cluster. For a hard clustering in which every data point of a cluster shares one cluster centroid, the penalty term is the distance between the two cluster centroids times the cut weight. For the cold and the warm days above, it is 17.0, the distance 15.6 between their cluster centroids times the cut weight 1.09.

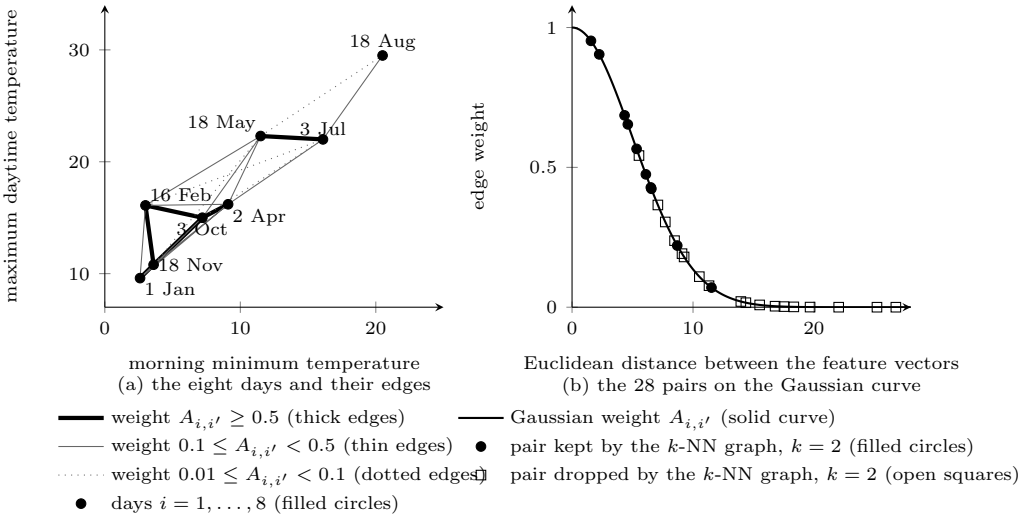


Figure 1: (a) Eight of the 366 days of 2024 at Krems, every 46th day from 1 January (filled circles), with the morning minimum temperature and the maximum daytime temperature of a day as its feature vector. Every pair of days is an edge with the Gaussian weight (2) for $\sigma = 5$ degrees: thick edges weigh at least 0.5, thin edges between 0.1 and 0.5, dotted edges between 0.01 and 0.1, and the eight pairs below 0.01 are not drawn. (b) The 28 pairs on the Gaussian curve (solid curve) of the weight against the Euclidean distance between the feature vectors: filled circles are the ten pairs kept by the k-NN graph with $k = 2$, open squares the pairs it drops. Data generated by `pythondemos/edgeweight.py`

Cross References

weighted graph, graph, undirected graph, adjacency matrix, Laplacian matrix, node degree, similarity graph, k-NN graph, neighbor, graph clustering, spec-

tral clustering, convex clustering, clustering, clustering error, GTV, GTVMin, network Lasso, FL network, transfer learning, SSL, matrix, ANN

References

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

2. Hocking et al. (2011). Clusterpath: An Algorithm for Clustering using Convex Fusion Penalties. In: Proc. of the 28th International Conference on Machine Learning,.

3. Sun et al. (2021). Convex Clustering: Model, Theoretical Guarantee and Efficient Algorithm. J. Mach. Learn. Res., 22(9), pp. 1–32.

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

5. 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.

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

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

8. Zhu et al. (2003). Semi-supervised learning using gaussian fields and harmonic functions. In: ICML, pp. 912–919.