Pattern Recognition Letters 100 (2017) 174–180

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

# Drawing clustered graphs by preserving neighborhoods

Eli Parviainen*, Jari Saramäki

*Department of Computer Science, Aalto University School of Science, PO Box 15400, FI-00076 Aalto, Finland*

ARTICLE INFO

*Article history:*
Received 31 March 2017

*Keywords:*
Graph drawing
Stochastic neighbor embedding
Clustered network
Random walk
Similarity

ABSTRACT

Weighted graphs with presumed cluster structure are challenging to many existing graph drawing methods, even though ways of visualizing such graphs would be much needed in complex networks research. In the field of dimension reduction, t-distibuted stochastic neighbor embedding (t-SNE) has proven successful in visualizing clustered data. Here, we extend t-SNE into graph-SNE (GSNE). Our method builds on the sensitivity of random walks to cluster structure in graphs. We use random walks to define a neighborhood probability that realizes the properties behind the success of t-SNE in visualizing clustered data sets: Gaussian-like behavior of neighborhood probabilities, adaptation to local edge density, and an adjustable granularity scale. We show that GSNE correctly visualizes artificial graphs where ground-truth cluster structure is known. Using real-world networks, we show that GSNE is able to produce meaningful visualizations that display plausible cluster structure which is not captured by state-of-the-art visualization methods.



## 1. Introduction

Modern graph drawing methods allow successful visualization of meshes of millions of nodes, almost any kinds of trees, and to a certain extent, large real-world complex networks. The latter come with a bewildering range of inhomogeneities, from broad distributions of node degrees to diverse structural patterns involving groups of nodes, such as cluster structure (also called community structure [1]). Especially cluster structure is important for a number of reasons. It often reflects the functional organization of the network, with different clusters having different functional roles. As a result, clusters represent network organization at a coarse level. It is known to have strong effects on dynamical processes taking place on networks [1]. Because of the importance of this structure, there are numerous methods for detecting and characterizing clusters (see, *e.g.* [1] for a review, and [2] for a comparison against benchmarks). However, there is a lack of methods for visualizing large networks with cluster structure. Such methods would be very useful *e.g.* in data exploration before applying cluster detection methods, whose runtimes are typically long and whose parameters may require adjustment to the problem at hand.

Neighborhood-based dimension reduction methods have been highly successful in visualizing clustered vector data. A notable example is t-SNE, t-distributed stochastic neighbor embedding [3], which followed the older SNE [4]. T-SNE emphasizes local over global, which helps clusters stand out. Neighborhood sizes are adjusted to local density, so that both sparse and dense regions become well visible. A scale parameter enables work on data sets which have interesting structure on several length scales. In this paper, we define a pairwise similarity that realizes these same desirable properties for graphs, and apply the resulting method, graph-SNE (GSNE), to many different visualization tasks. Unlike related similarities, the density-adaptive similarity of GSNE makes the internal structure of clusters visible while retaining a good overall view of the graph. GSNE shows ground-truth clusters of benchmark graphs better than the comparison methods (although in some cases the differences are subtle), and performs on par with or better than competitors on real-life networks of different sizes.

Section 2 gives background on earlier work on related topics. Definition of graph-SNE is presented in Section 3, followed by visualization experiments in Section 4. We discuss some limitations and finish with conclusions in Section 5.

## 2. Earlier work

In this section, we summarize some main lines of graph drawing and use of random walks for characterizing node similarity, and introduce the dimension reduction method t-SNE that our work builds on. Detailed reviews and comparisons are available elsewhere on graph drawing in general [5], on drawing large graphs [6–8], and on the relationship of graphs to random walks [9,10]. Readers wishing more background on dimension reduction are referred to [11].

* Corresponding author.
*E-mail address:* eli.parviainen@iki.fi (E. Parviainen).

https://doi.org/10.1016/j.patrec.2017.10.035
0167-8655/

### 2.1. Graph drawing

Early graph drawing methods emphasized the aesthetic character of visualization. Aesthetic criteria, such as uniform spread of nodes, uniform lengths of edges, and minimal edge crossings, can be explicitly encoded into cost functions [12]. Force-based methods [13,14] tend to produce aesthetically pleasing drawings of small graphs, although aesthetics is not explicitly considered. Methods based on eigendecomposition of graph Laplacians [15,16] have an aesthetic interpretation: they place each node in the barycenter of its neighbors. These classical methods have served as building blocks for more recent methods for large graphs [17,18].

The ever-increasing sizes of graphs to be visualized have all but outdated some aesthetic criteria, resulting in a need to emphasize interpretability [19] alongside aesthetic goals. The earliest methods with a clear interpretation were those that preserve shortest-path distances [20,21], and the idea has been rehashed for larger graphs [22,23]. On a more modern note, the LinLog family of energy functions [19,24] is designed to produce drawings with interpretable clusters.

### 2.2. Similarity metrics on random walks

Path-based measures are known to emphasize clusters in drawings [25] – two nodes are considered similar if there are many paths connecting them. Random walks naturally capture the net effect of multiple paths, and they have been used in many distance/similarity metrics as well as methods for detecting clusters in graphs [26,27]. The commute time distance — the average time it takes to travel from node $r$ to node $c$ and back — has been used for clustering [28] and for graph embeddings [29]. The same distance is motivated by interpreting the graph as an electrical circuit, where the edge weights represent conductances. The total resistance between two nodes is then known as the resistance distance [30] or linear network distance [25]. In the context of t-SNE, random walks have been used to obtain a reduced data set which can be said to capture the structure of the whole data set [3].

### 2.3. Dimension reduction with t-SNE

T-SNE [3] tries to match the probability of nodes $r$ and $c$ being neighbors in the drawing ($q_{rc}$) to their probability of being neighbors in the data ($p_{rc}$, Euclidean distance of points $r$ and $c$ modified by a Gaussian centered at point $r$, normalized over all neighbors). We will define $p_{rc}$ for GSNE in Section 3. The $q_{rc}$ are defined with normalized Student-t-kernels as

$$t_{rc} = (1 + \|\mathbf{y}_r - \mathbf{y}_c\|^2)^{-1}, \quad t_{rr} = 0, \tag{1}$$

$$q_{rc} = \frac{t_{rc}}{\sum_{i,j \in V} t_{ij}}, \tag{2}$$

where $V$ is the set of nodes, $\mathbf{y}_i$ are the coordinates of nodes in the drawing, and $\|\cdots\|$ is the Euclidean distance. $\mathbf{y}_i$ are found by minimizing (to a local optimum) the Kullback–Leibler divergence

$$D_{\mathrm{KL}}(p \,|\, q) = \sum_{r,c} p_{rc} \log \frac{p_{rc}}{q_{rc}}.$$

### 2.4. Sparse t-SNE for large graphs

Both time and memory requirements of t-SNE scale quadratically with $|V|$, so for large graphs we need an approximation.

Sparse SNE [31] was developed for vector data, but uses a graph as its internal representation, so it is straightforward to modify it for graphs. Other large-scale t-SNE variants have been created with tree-codes [32,33]. The random-walk similarity of GSNE could be used with the tree-code methods or even with the exact t-SNE. For a comparison between tree-code and graph-based techniques (for vector data) see [31].

Minimization of the Kullback–Leibler divergence via its gradient leads to a system of attractive and repulsive forces [33]

$$\frac{\partial \mathrm{C}_{SNE}}{\partial \mathbf{y}_r} = \mathrm{A}_r - \mathrm{R}_r, \tag{3}$$

$$\mathrm{A}_r = 4 \sum_{c \in V} p_{rc} t_{rc} (\mathbf{y}_r - \mathbf{y}_c), \tag{4}$$

$$\mathrm{R}_r = 4 \sum_{c \in V} q_{rc} t_{rc} (\mathbf{y}_r - \mathbf{y}_c). \tag{5}$$

Sparse SNE replaces the full system of forces with a smaller number of weighted forces [31]. Two approximations are made when evaluating the gradient (3). The attractive forces 4 are evaluated exactly, but only for a small set $\mathcal{L}$ of node pairs

$$\mathrm{A}_r \approx 4 \sum_{\substack{c \text{ s.t.} \\ (r,c) \in \mathcal{L}}} p_{rc} t_{rc} (\mathbf{y}_r - \mathbf{y}_c). \tag{6}$$

The pairs include but are not limited to edges of the graph. $\mathcal{L}$ is determined with help of a random walk, and will be defined in Section 3.5.

The repulsive forces are exact for node pairs in $\mathcal{L}$. Another, uniformly randomly sampled set $\mathcal{G}$ is used to approximate the rest of the repulsions. This leads to replacing (5) with

$$\mathrm{R}_r \approx \frac{4}{\tilde{z}} \sum_{\substack{c \text{ s.t.} \\ (r,c) \in \mathcal{L}}} t_{rc}^2 (\mathbf{y}_r - \mathbf{y}_c) + \frac{4}{\tilde{z}} \sum_{\substack{c \text{ s.t.} \\ (r,c) \in \mathcal{G}}} \left[ b I_{rc} + t_{rc}^2 \right] (\mathbf{y}_r - \mathbf{y}_c). \tag{7}$$

The term $bI_{rc}$ results from associating each node pair of $\mathcal{G}$ with an area-to-point interaction, to be used instead of several point-to-point interactions. $I_{rc}$ is the expected force, which a point from inside a $\mathbf{y}_c$-centered sphere exerts on the point $\mathbf{y}_r$. The expectation is weighted by an estimated number $b = (|V| - L - 1)/G - 1$ of all points which the sphere represents. Here, $L$ and $G$ are the numbers of $\mathcal{L}$- and $\mathcal{G}$-linked neighbors per one node. The normalization factor $\tilde{z} \approx \sum_{i,j \in V} t_{ij}$ is approximated with the same idea as the gradient. Formulas for $b$, $I_{rc}$ and $\tilde{z}$ are derived in detail in [31].

## 3. Graph-SNE (GSNE)

To make t-SNE work for graphs, we need the neighborhood probabilities $p_{rc}$, and the node sets $\mathcal{L}$ and $\mathcal{G}$. The probability of node $c$ being a neighbor of node $r$ is defined as the probability of being in node $c$ after taking $s_r$ random walk steps from node $r$. This idea resembles commute times [29], but avoids the costly eigendecomposition needed therein.

We work on undirected, weighted graphs with no self-loops (unweighted graphs can simply be considered as a special case of weighted graphs where all edges have unit weights). The edge weights $w_{rc}$ must be interpretable as *similarities*: they are nonnegative, and the smaller the value, the weaker the relationship between nodes. We scale edge weights to transition probabilities of a Markov chain, collected into a transition matrix $\mathbf{M}$. With $\mathbf{v}_i$ a column vector with a 1 in position $i$ and zeros elsewhere, the probability of ending up in $c$ from $r$ with $s_r$ steps is $\mathbf{v}_r^{\mathrm{T}} \mathbf{M}^{s_r} \mathbf{v}_c$. Since the drawing algorithm needs symmetric probabilities, we must consider both directions,

$$p_{rc} = \left( \mathbf{v}_r^{\mathrm{T}} \mathbf{M}^{s_r} \mathbf{v}_c + \mathbf{v}_c^{\mathrm{T}} \mathbf{M}^{s_c} \mathbf{v}_r \right) / 2. \tag{8}$$