WebShapes: Network Visualization with 3D Shapes

Shengmin Jin
shengmin@data.syr.edu
Data Lab, EECS Department
Syracuse University

Richard Wituszynski
rwituzynski@gmail.com
West Genesee High School
United States

Max Caiello-Gingold
mgingold02@gmail.com
Corcoran High School
United States

Reza Zafarani
reza@data.syr.edu
Data Lab, EECS Department
Syracuse University

ABSTRACT

Network visualization has played a critical role in graph analysis, as it not only presents a big picture of a network but also helps reveal the structural information of a network. The most popular visual representation of networks is the node-link diagram. However, visualizing a large network with the node-link diagram can be challenging due to the difficulty in obtaining an optimal graph layout. To address this challenge, a recent advancement in network representation: *network shape*, allows one to compactly represent a network and its subgraphs with the distribution of their embeddings. Inspired by this research, we have designed a web platform WebShapes that enables researchers and practitioners to visualize their network data as customized 3D shapes (http://b.link/webshapes). Furthermore, we provide a case study on real-world networks to explore the sensitivity of network shapes to different graph sampling, embedding, and fitting methods, and we show examples of understanding networks through their network shapes.

CCS CONCEPTS

• **Human-centered computing** → **Visualization**; **Visualization techniques**; • **Computing methodologies** → *Machine learning*.

KEYWORDS

Network Visualization, Graph Representation

ACM Reference Format:
Shengmin Jin, Richard Wituszynski, Max Caiello-Gingold, and Reza Zafarani. 2020. WebShapes: Network Visualization with 3D Shapes. In *The Thirteenth ACM International Conference on Web Search and Data Mining (WSDM '20), February 3–7, 2020, Houston, TX, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3336191.3371867

1 INTRODUCTION

Networks have become ubiquitous and network analysis has been used in many research fields, from science (e.g. biological networks and computer networks), engineering (e.g. power grids and infrastructure networks) to our daily life (e.g. social networks). Among network analysis techniques, network visualization plays an important role, as a well-designed network visualization can provide various insights on the structure of a network. Network visualization can assist in understanding the transportation flow of a city via its metro map [16], or can help identify economic patterns in a trade network [2]. However, network visualization is challenging, especially when visualizing large graphs. The most popular visual representation of graphs is the *node-link* diagram, which is difficult to draw for large graphs due to natural clutter, crossing, and overdrawing issues [15]. To address such challenges, and inspired by recent advancements in network embedding, we have previously proposed a framework, *network shapes* [6], to represent networks as 3D shapes using their embedding space. In this demonstration, (1) we develop WebShapes, a web platform that provides users the possibility to create various network shapes with different methods and parameters, where users can upload their own networks and download the shape information for further analysis; (2) we provide a case study on some networks as examples of network analysis using network shapes and the WebShapes platform.

The rest of the paper is organized as follows. We first review related work in Section 2. In Section 3, we detail the *network shapes* framework and introduce the WebShapes platform, its functionality, its backend, and the algorithms used. After presenting a network analysis case study in Section 4, we conclude in Section 5.

2 RELATED WORK

Our work has links to the following research areas:

Network Visualization. Network visualization research has been mostly concerned with generating node-link diagrams that are aesthetically pleasing; hence, the focus has been mostly on graph layouts, i.e., the placement of the nodes, to increase graph readability [5]. As a result, visulization tools such as Gephi [1], the Javascript D3 library [3] and Cytoscape [13] are introduced and widely used. These tools often scale well on small to medium-sized graphs, however, can hardly visualize larger networks, i.e. with many thousands or million of nodes.

Network Embedding. Recent network embedding methods, especially node embedding methods, are utilized for graph visualization. A well-known example is the spectral drawing which uses the eigenvectors of the Laplacian matrix to embed nodes, allowing one to identify an optimal graph drawing layout according to specific requirements [8]. Another example is *LINE* [14], used to visualize

WSDM '20, February 3–7, 2020, Houston, TX, USA

ACM ISBN 978-1-4503-6822-3/20/02…$15.00
https://doi.org/10.1145/3336191.3371867

the coauthorship network to aid with clustering authors in the same field. Different from embedding techniques that enhance node-link diagrams, network shapes represent a network with a 3D shape, which approximates the distribution of the embedding space of the network and its subgraphs. Therefore, technically it combines graph sampling, network embedding, and shape fitting techniques. In the next section, we will provide more details on network shapes. Compared with node-link diagrams, network shapes have limited clutter and capture various graph structural properties [6].

3 WebShapes

In this section, we present the background of *Network Shapes* representation, and we introduce the WebShapes platform.

3.1 Background on Network Shapes

We first briefly review the *network shape* framework. A network shape is built in three simple steps [6]: (1) **Sample** many subgraphs from the network. Theoretically, any sampling method can work. To sample systematically, we sample by varying proportions of nodes or edges (e.g., from 0% to 100%) with some fixed step size s. For each proportion, we sample t independently sampled subgraphs, i.e., a total of $t \times 100/s$ subgraphs for one network; (2) **Map** the network and its subgraphs to 3D vectors. To do so, an embedding method is chosen to provide 3D embedding vectors that can capture the properties of the network and its subgraphs. After Step 2, we can represent a network and its subgraphs as a set of 3D points; and (3) **Fit** a 3D shape to the set of 3D vectors obtained in Step 2. Fitting can be done by various shapes, e.g., squares or spheres. The pseudocode for building network shapes is provided in Algorithm 1. Network shapes have been used to capture network similarity, to classify and identify networks, and to validate network-based research [7].

Algorithm 1: Network Shape algorithm

input : an undirected network graph: $G(V, E)$
output : the network shape of G: Shape_G
parameter: Sample : a graph sampling method to sample proportion p of nodes (or edges);
s : sampling proportion step size;
t : number of samples for one proportion;
Embed : a graph embedding technique that can map a network to a 3D point;
Fit : a technique to fit a 3D shape to a set of 3D points;

```
Embedding_points = { };
for ( p = s; p < 100%; p = p + s ) {
    for ( i = 1; i ≤ t; i = i + 1 ) {
        % Sample a subgraph G_p
        G_p = Sample(G, p);
        % Embed G_p to a 3D point
        Embedding_point = Embed(G_p);
        Embedding_points.add(Embedding_point);
    }
}
% Embed the whole graph G
Embedding_point = Embed(G_p);
Embedding_points.add(Embedding_point);
% Fit a shape to the set of points
Shape_G = Fit(Embedding_points);
return Shape_G;
```

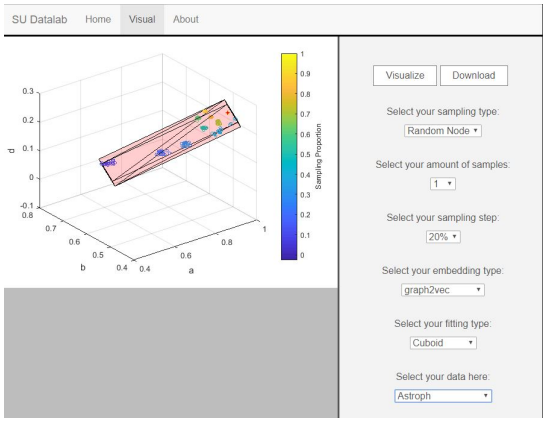


Figure 1: WebShapes Interface

3.2 Web Platform

From Section 3.1, we know that one can select various sampling methods (Step 1), embedding methods (Step 2), and fitting methods (Step 3) to build a network shape. Hence, we developed WebShapes, a web platform for users to create customized network shapes for their need. Using WebShapes, users can upload their network data or choose an existing dataset, select predefined methods and parameters (a sampling method, an embedding method, and a fitting method), and can visualize their network data as a 3D shape. Users can download the information of the network shape for further analysis. Figure 1 is a screenshot of the platform. Next, we will briefly introduce the sampling methods, embedding methods, and fitting methods currently supported by WebShapes.

(1) Three sampling methods:

- *Random Node Sampling*. Selects p% of nodes uniformly at random and outputs the subgraph induced by the selected nodes [10].
- *Random Edge Sampling*. Select p% of edges uniformly at random and outputs the subgraph induced by the selected edges [10].
- *Random Walk Sampling*. Starting from an initial random node performs a random walk to sample nodes. At each step, the random walk restarts from the initial node with probability (=0.15) [10]. The process continues until p% of nodes are sampled.

(2) Two embedding methods:

- *Kronecker Point* [6] is a 3D point (a, b, d) that embeds an undirected network, or any of its subgraphs, obtained using Stochastic Kronecker Graphs [9]. Values a, b and d are in range $[0, 1]$ and are closely related to the network structure. Generally, one can view a network as two groups of nodes and interpret a and d as the proportion of edges within each of the groups, and b as the fraction of edges between the two groups. Hence, we can split the whole embedding space into three regions based on a, b, d values: *Core-Periphery* ($a \geq b \geq d$), *Dual-Core* ($a \geq d \geq b$), and *Random* ($b \geq a \geq d$), and value a represents the *core strength* [6].
- *Graph2vec* views a graph as a document and the rooted subgraphs around each node as words. It extends document embedding neural networks to embed a graph as a vector [12]. For creating network shapes, we set the output dimension to 3.

(3) Three fitting methods:

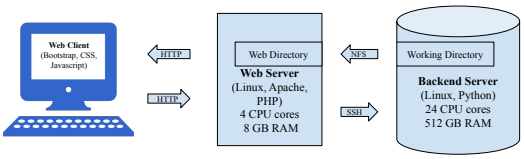


Figure 2: Platform Architecture

- *Convex Hull* yields the smallest convex set that contains all 3D points given [4]. To save a convex hull, we save its boundary.
- *Cuboid* computes the minimal box (with right angles) around the 3D points. To save the cuboid, we save the 8 extreme points.
- *Sphere* uses the arithmetic mean of the points as the center and the longest distance from the points to the center as the radius to fit a sphere. To save it, we save the center and the radius.

After WebShapes creates a network shape, users can download the figure and the files including boundary points (Convex Hull), extreme points (Cuboid) or the center and the radius (Sphere).

Architecture. Here, we briefly introduce the software architecture of the Web platform. The WebShapes platform is composed of three components: the Web client (i.e., front-end), the Web server and the back-end server. The Web client is basically the webpages designed by the bootstrap framework using CSS and Javascript. Users can use the Web client to interact with the Web server through HTTP calls. The Web server uses the Linux stack, Apache Web server, and PHP. When the server receives a request to create a network shape, it will send a remote call to a strong back-end server. The back-end server creates a network shape and saves the figure file and related information of the shape in a directory, which is shared with the Web server using the Network File System (NFS). Figure 2 illustrates the architecture and the servers configuration.

4 CASE STUDY

Experimental Setup. In the case study, we aim to use WebShapes to explore the sensitivity of network shapes to various methods and to derive insights from network shapes. To simplify our case study, we choose the following as the default methods for creating a network shape: (1) Sampling: use *Random Edge Sampling*; set sampling step size 20%, i.e., $s = 20\%$; for each sampling proportion we generate 5 independently sampled subgraphs, i.e., $t = 5$. (2) Embedding: use *Kronecker Point*. (3) Fitting: use *Convex Hull*. For each step, when we compare methods and parameters, we maintain the default configuration of other steps. We use nine real-world networks from three different categories. Table 1 provides the statistics.

Social Networks: In total, we include three social networks.

(1) *Hyves* [17]: the most popular social networking site in the Netherlands with mainly Dutch visitors.
(2) *MySpace* [18]: a social network having a significant influence on pop culture and music.

Table 1: Dataset Statistics

Type	Network	$\|V\| = n$	$\|E\| = m$	Average Degree	Clustering Coefficient
Social Networks	Hyves	1,402,673	2,777,419	3.960	0.0448
	MySpace	854,498	5,635,296	13.190	0.0433
	YouTube	1,134,890	2,987,624	5.265	0.0808
Collaboration Networks	Astro-Ph	18,772	198,050	21.100	0.6306
	Cond-Mat	23,133	93,439	8.078	0.6334
	Hep-Th	9,877	25,973	5.259	0.4714
Road Networks	Road-CA	1,965,206	2,766,607	2.816	0.0464
	Road-PA	1,088,092	1,541,898	2.834	0.0465
	Road-TX	1,379,917	1,921,660	2.785	0.0470

Table 2: Shape Volumes for Sampling Methods ($\times 10^{-4}$)

Type	Network	Random Node	Random Edge	Random Walk
Social Networks	Hyves	1.39	0.30	0.20
	MySpace	0.76	0.09	0.13
	YouTube	1.80	0.18	0.39
Collaboration Networks	Astro-Ph	1.55	4.22	0 (reduced to 2D)
	Cond-Mat	9.31	1.22	1.36
	Hep-Th	5.11	0.80	0.48
Road Networks	Road-CA	4.65	0.17	0.57
	Road-PA	3.65	3.83	0.26
	Road-tx	1.00	1.08	0.24

(3) *YouTube* [11]: a video-sharing site with a social network.

Collaboration Networks: We have three collaboration networks.

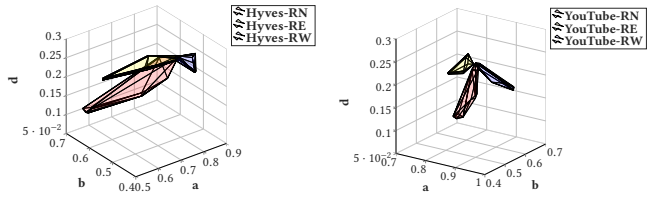
(4) *Astro-Ph* [11]: Astro physics.
(5) *Cond-Mat* [11]: Condense matter physics.
(6) *Hep-Th* [11]: High energy physics theory.

Road Networks: We include three road networks. In road networks, nodes are intersections/endpoints and undirected edges are the roads connecting these intersections/road endpoints.

(7) *Road-CA* [11]: the road network of California.
(8) *Road-PA* [11]: the road network of Pennsylvania.
(9) *Road-TX* [11]: the road network of Texas.

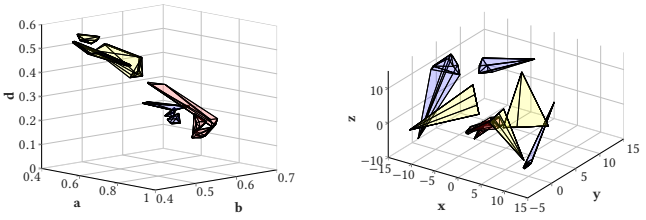
4.1 Analysis

Sampling methods. Table 2 lists the volume of the network shapes created using different sampling methods. The result shows that shapes using Random Node sampling are in general large, while shapes using Random Walk sampling are generally small, and shapes using Random Edge sampling vary in volumes for different networks. As the volume captures the variance of the Kronecker points of the network and its subgraphs, it indicates that Random Node sampling generates samples with more variance, while Random Walk sampling generates more similar subgraphs. Figure 3 provides the network shapes of Hyves and YouTube obtained using different sampling methods. It turns out that the three shapes intersect at the Kronecker point of the whole graph, and the Kronecker points of subgraphs are distributed in different directions. We can connect the observation to the sampling strategies: (1) The blue shapes have a much larger a value than others, as the Random Walk sampling (restarting with some probability from its initial node) prefers visiting the initial node and its near neighbors, i.e., 1-hop or 2-hop neighbors, which forms a dense core of the sample; (2) Though the yellow shapes and the red shapes are both in the core-periphery region ($a \geq b \geq d$), the yellow shapes are above the red ones (having greater d value). This can be explained by the fact that Random Edge sampling has a slight bias towards high degree nodes and it generates denser samples than Random Node sampling does. Hence, for subgraphs sampled by Random Edge sampling, the group of periphery nodes have more internal connections.



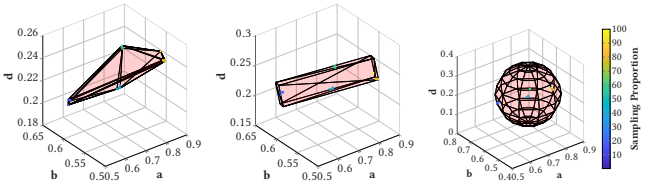
(a) Hyves **(b) YouTube**

Figure 3: Comparison of Different Sampling Methods. Red: Random Node; Yellow: Random Edge; Blue: Random Walk



(a) Kronecker Point (b) graph2vec

Figure 4: Comparison of Different Embedding Methods. Red: Collaboration Networks; Yellow: Road Networks; Blue: Social Networks



(a) Convex Hull (b) Cuboid (c) Sphere

Figure 5: Comparison of Different Fitting Methods

Embedding methods. Figure 4 provides the network shapes of the nine networks using Kronecker Point and *graph2vec* respectively. We have colored the shapes based on their network category. We observe (1) For Kronecker Point, as values a, b and d are all between 0 and 1, the whole embedding space is a $1 \times 1 \times 1$ cube; For *graph2vec*, the embedding values have no such bound; (2) When using Kronecker Point, the networks from the same category exhibit a clustering phenomenon. Moreover, social networks and collaboration networks are close while road networks are relatively far from others. This observation can be explained by previous results [6], i.e., most social networks and collaboration networks exhibit the core-periphery structure and these two categories have overlaps as they both involve human social behavior, but road networks often exhibit the dual-core structure. However, when using *graph2vec*, networks from different categories seem to form an onion-like structure: the collaboration networks are in the inside layer, social networks in the outside layer, and road networks are in the middle.

Fitting methods. Based on the definition of our three fitting methods, we know that for the same set of points, $ConvexHull \subseteq Cuboid \subseteq Sphere$. Figure 5 provides an example of the network shapes of Hyves. We list the volume of the shapes using different fitting methods in Table 3. In general, we find that compared with other methods, convex hull is very compact, while sphere is very loose. Cuboid is in the middle, with 3 times volume of convex hull. As the process of building a network shape involves the graph sampling step, a good approximation of the embedding space of a network and its subgraphs should be both accurate and robust to sampling variance. Considering the trade-off, we believe convex hull and cuboid perform better in this case.

5 CONCLUSION

In this demonstration, we present WebShapes, a web platform implementing the *network shape* framework which allows users to visualize their network data as 3D shapes with different methods. We have a case study on networks from different categories. We demonstrate that the properties of different graph sampling methods can be connected to the network shape; with different embedding methods networks have individual network shapes; convex hull and cuboid provide a better approximation of the embedding space of a network from the view of the accuracy and robustness trade-off. As the network shape framework is flexible, we can integrate more new graph sampling methods, embedding methods and fitting methods in WebShapes in the future. We also plan to provide interfaces for users to implement their own graph sampling, network embedding or fitting methods in the future.

Table 3: Shape Volumes for Different Fitting Methods ($\times 10^{-4}$)

Type	Network	Convex Hull	Cuboid	Sphere
Social Networks	Hyves	0.30	1.15	673
	MySpace	0.09	0.30	189
	YouTube	0.18	0.61	344
Collaboration Networks	Astro-Ph	4.22	13.00	2398
	Cond-Mat	1.22	5.06	2246
	Hep-Th	0.80	2.18	1977
Road Networks	Road-CA	0.17	0.51	158
	Road-PA	3.83	13.00	3756
	Road-tx	1.08	3.59	2879

REFERENCES

[1] Mathieu Bastian, Sebastien Heymann, Mathieu Jacomy, et al. 2009. Gephi: an open source software for exploring and manipulating networks. *Icwsm* 8 (2009), 361–362.

[2] Vladimir Batagelj, Andrej Mrvar, and Matjaž Zaveršnik. 1999. Partitioning approach to visualization of large graphs. In *International Symposium on Graph Drawing*. Springer, 90–97.

[3] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. 2011. D^3 data-driven documents. *IEEE transactions on visualization and computer graphics* 17, 12 (2011), 2301–2309.

[4] Mark De Berg, Marc Van Kreveld, Mark Overmars, and Otfried Cheong Schwarzkopf. 2000. Computational geometry. In *Computational geometry*. Springer, 1–17.

[5] Josep Díaz, Jordi Petit, and Maria Serna. 2002. A survey of graph layout problems. *ACM Computing Surveys (CSUR)* 34, 3 (2002), 313–356.

[6] Shengmin Jin and Reza Zafarani. 2018. Representing Networks with 3D Shapes. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 177–186.

[7] Shengmin Jin and Reza Zafarani. 2019. Network Identification and Authentication. In *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE.

[8] Yehuda Koren. 2005. Drawing graphs by eigenvectors: theory and practice. *Computers & Mathematics with Applications* 49, 11-12 (2005), 1867–1888.

[9] Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. 2010. Kronecker graphs: An approach to modeling networks. *JMLR* 11, Feb (2010), 985–1042.

[10] Jure Leskovec and Christos Faloutsos. 2006. Sampling from large graphs. In *Proceedings of the SIGKDD*. ACM, 631–636.

[11] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[12] Annamalai Narayanan, Mahinthan Chandramohan, Rajasekar Venkatesan, Lihui Chen, Yang Liu, and Shantanu Jaiswal. 2017. graph2vec: Learning distributed representations of graphs. *arXiv preprint arXiv:1707.05005* (2017).

[13] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S Baliga, Jonathan T Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. 2003. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome research* 13, 11 (2003), 2498–2504.

[14] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. 1067–1077.

[15] Tatiana Von Landesberger, Arjan Kuijper, Tobias Schreck, Jörn Kohlhammer, Jarke J van Wijk, J-D Fekete, and Dieter W Fellner. 2011. Visual analysis of large graphs: state-of-the-art and future research challenges. In *Computer graphics forum*, Vol. 30. Wiley Online Library, 1719–1749.

[16] Yu-Shuen Wang and Ming-Te Chi. 2011. Focus+ context metro maps. *IEEE Transactions on Visualization and Computer Graphics* 17, 12 (2011), 2528–2535.

[17] R. Zafarani and H. Liu. 2009. Social Computing Data Repository at ASU. http://socialcomputing.asu.edu

[18] Yutao Zhang, Jie Tang, Zhilin Yang, Jian Pei, and Philip S Yu. 2015. Cosnet: Connecting heterogeneous social networks with local and global consistency. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1485–1494.