Graphon-based Visual Abstraction for Large Multi-Layer Networks

Category: Research

Paper Type: Technique

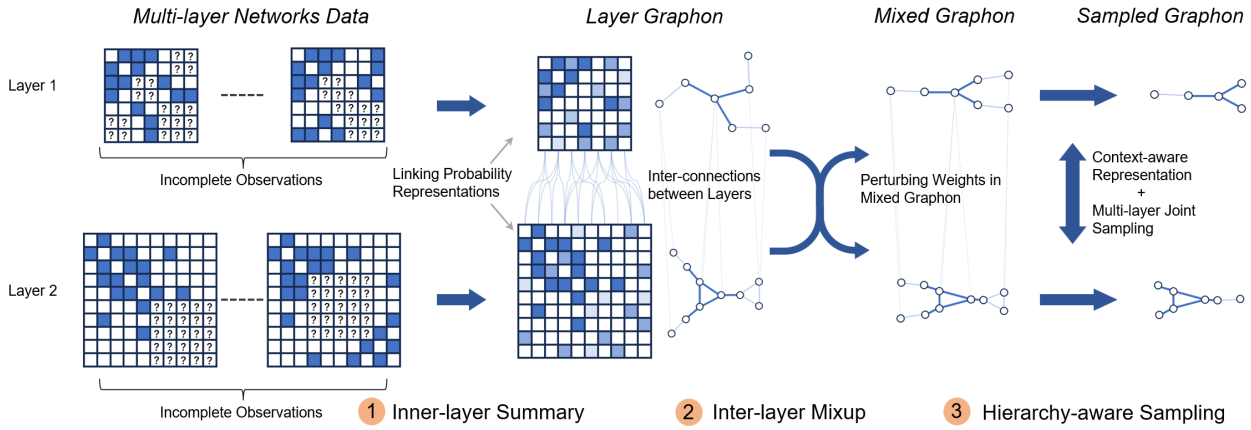


Fig. 1: The workflow of our approach. For a network with multiple layers, (1) we firstly summarize incomplete observations into a graphon-based probabilistic representation for each layer. (2) Then, we mix heterogeneous graphons between layers by propagating inner-layer information through inter-layer connections. (3) Finally, we sample multiple layers jointly in a vectorized space to simplify networks while preserving inner- and inter-layer characteristics. The approach distills abstractions of large multi-layer networks into tractable weighted graphs to provide an overview.

Abstract— Graph visualization techniques provide a foundational framework for offering comprehensive overviews and insights into cloud computing systems, facilitating efficient management and ensuring their availability and reliability. Despite the enhanced computational and storage capabilities of larger-scale cloud computing architectures, they introduce significant challenges to traditional graph-based visualization due to issues of hierarchical heterogeneity, scalability, and data incompleteness. This paper proposes a novel abstraction approach to visualize large multi-layer cloud computing networks. Our method leverages graphons, a probabilistic representation of network layers, to encompass three core steps: an inner-layer summary to identify stable and volatile substructures, an inter-layer mixup for aligning heterogeneous network layers, and a context-aware multi-layer joint sampling technique aimed at reducing network scale while retaining essential topological characteristics. By abstracting complex network data into manageable weighted graphs, with each graph depicting a distinct network layer, our approach renders these intricate systems accessible on standard computing hardware. We validate our methodology through case studies, quantitative experiments and expert evaluations, demonstrating its effectiveness in managing large multi-layer networks.

Index Terms—Multi-Layer Networks, Network Abstraction, Network Visualization, Cloud Network Visualization

1 Introduction

Graph visualization techniques are commonly used in abstracting cloud computing networks, which has emerged as a widespread paradigm for solving complex computational challenges by harnessing a vast array of servers, applications, data, and other computing resources [13, 17, 28, 47]. To identify anomalies, analyze behaviors, and optimize architectures for both high availability and reliability, monitoring of cloud computing systems is a critical necessity for cloud managers [54]. Since the cloud system consists of a great variety of computing devices and physical connections, it is common to monitor the cloud system through graph-based visual analysis systems [21], which depict devices as nodes and their connections as links. Such a node-link diagram facilitates a comprehensive overview and understanding of the architecture, enables the examination of device functionalities and communications, and assists in the identification of areas exhibiting anomalous behavior [2, 3, 34]. Despite the extensive adoption of graph-based monitoring methods and management systems, they often fail to tackle three significant challenges in visualizing multi-layer networks.

C1: Hierarchical Heterogeneity. Cloud computing systems exhibit multi-layered and heterogeneous structures, integrating an array of devices spanning physical hosts, network devices, and virtual services into distinct categories. Consider a three-tier service architecture with a process layer, a microservice layer, and a cloud service layer: inner-layer links denote communication within the same layer (e.g., the flow of information), while inter-layer links represent associative relationships between different layers (e.g., the allocation of physical resources). Efforts to visualize these constructs have yielded various methodologies, ranging from 2D and hybrid 2.5D to full 3D visualizations [14, 26]. In visualizing a multi-layer network, Fig. 2 (a) represents each network layer separately, omitting the connections between layers. While straightforward in application, understanding a node's role within such networks necessitates the consolidation of data from both inner-layer and inter-layer connections. This method may result in a perceptual gap when inspecting between layers due to the absence of inter-layer links. Fig. 2 (b) introduces a 2.5D visualization model where each network layer is positioned on a distinct plane, and connections spanning across these planes (inter-layer links) are explicitly visualized. Moreover, Fig. 2 (c) showcases a fully 2D expression of (b). Although (b) and (c) can offer strategies like link transparency to distinguish different link types and maintain the visibility of all links, they necessitate a substantial expenditure of computational resources for visualization presentation. Critically, they fall short in presenting a unified, aggregated node profile, forcing analysts to manually integrate data and construct these profiles, which is a time-consuming and error-prone task. This highlights the need for an innovative visual-

ization framework capable of handling hierarchical heterogeneity and providing a comprehensive representation of node profiles within cloud computing infrastructures.

C2: Scalability. The task of visualizing multi-layered graphs becomes even more complex when the scale of cloud systems expands. Networks in real scenarios are often dynamic and have a large number of nodes and edges. Operational experts need to quickly determine their functions based on the time-varying characteristics and connections of various components of the network, such as identifying the main linkings of the network based on the duration of the communication. The node-link diagram has a limited space with respect to the increasing number of nodes and links, resulting in visual clutter and occlusion, especially in densely networked areas. Previous works has simplified the large-scale networks while preserving the graph's structural context by using summarization [10] and sampling [4, 36] strategies. Yet, sporadic sampling frames fail to provide a comprehensive overview and can disrupt the original hierarchical structure of the network. Besides, when these methods are applied to multi-layered graphs, they may inadvertently hide nodes based on all link information and ignore the weight of strong links, thereby omitting vital links and resulting in the loss of crucial information and obscuring valuable insights.

C3: Data incompleteness. The frequency and duration of communication between nodes serve as critical indicators to reflect the network connectivity and the relative importance of entities. Based on such information, experts can discern the network's core components through inspection of the consistency and fluctuation of communication patterns. However, capturing these communications can be challenging, hampered by the inherent unpredictability of system behaviors. The scale and complexity of these communications surpass the data capture capabilities of existing monitoring tools, leading to inevitable missing in the collected data. Relying on such data for visualization risks overlooking critical connections. Therefore, there is a need to address the issue of data incompleteness while maintaining analytical precision.

To address the above-mentioned challenges, we propose a novel large multi-layer network abstraction approach to provide an overview of the cloud system. We aim to produce a set of tractable weighted graphs as a simplification of the original complex networks, which are then visualized using node-link diagrams. First, to tackle the issue of incomplete data (**C3**), we leverage graphons [37], a method for representing networks probabilistically, to accurately estimate the communication probability between nodes. This approach involves using a sophisticated singular value thresholding technique to fill in missing information in the communication matrix. The application of the graphon method results in a weighted network that identifies both stable and volatile substructures of the network. Second, we introduce a layer mixup strategy to mitigate the heterogeneity between layers by aligning the structure between adjacent layers (**C1**). Thanks to the probabilistic representation based on graphons, the layer mixup strategy can blend adjacent heterogeneous layers by perturbing their weights. The mixup strategy propagate the inter-layer associations between the visualization of different layers, enabling a comprehensive understanding of both inner-layer and inter-layer connections. Third, we employ a context-aware multi-layer joint sampling technique to reduce the scale of the cloud system network while preserving the hierarchical structure (**C2**). We propose to jointly sample associated nodes from adjacent layers by adaptively adjusting the sampling weight based on the sampling result of the previous layer. Finally, we develop a prototype visualization interface to support efficient management and exploration of large-scale multi-layer networks. We demonstrate the effectiveness of our Graphon-based network abstraction approach on both synthetic and real-world datasets.

Our contributions can be summarized as follows:

- an abstraction method for large multi-layer networks visualization based on graphons, which simplifies complex networks into tractable weighted graphs.
- a layer mixup strategy to propagate inter-connections and overcome the heterogeneity between layers.
- extensive evaluations on both synthetic and real-world datasets to demonstrate the usability and effectiveness of our visual abstraction approach.

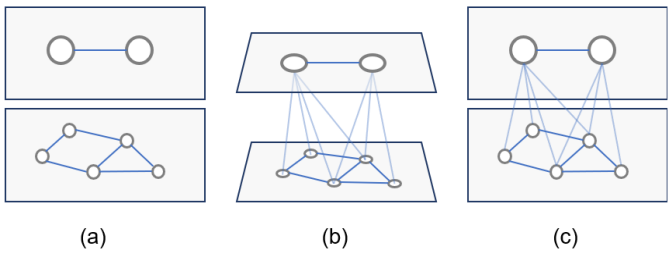


Fig. 2: Three potential visualization types for multi-layer networks. (a) separate visualizations without inter-layer connections. (b) 2.5D visualization. (c) fully 2D expression of (b).

2 Related Work

In this section, we review related work in Network Visualization, Network Simplification and Network Mixing.

2.1 Network Visualization

The goal of network visualization is to represent abstract network structures and connections. A multitude of strategies have been employed to capturing and representing the frequency and complexity of these connections. Notably, linear [56] and matrix-based visual representations [41] offer structured yet compact representations. Despite the learning curve associated with these depictions, they can effectively reveal cluster partitioning and behaviors via advanced techniques like row and column reordering.

Node-link diagrams have emerged as the predominant strategy of network visualization [38] and have an intuitive portrayal of multi-layer network structures. In these diagrams, nodes are positioned on a two-dimensional grid based upon (x, y) coordinates, with edges illustrating the web of inner- and inter-layer connections. In the context of multi-layer networks, these diagrams leveraging color coding to differentiate node types and to highlight the intensity of connections. Through the application of constraint-based layout algorithms, nodes within a layer are carefully arranged to craft a visually coherent hierarchical representation, thus exemplifying the fusion of design and analytical prowess required in advanced network visualization inherently reveals the topology of the network [14, 26]. Building upon techniques, 3D visualizations [18, 52] integrate the z-axis to position nodes within a volumetric space and improve network readability. However, it demands visual analysis skills for adjusting the viewpoint, ensuring a comprehensive observation and understanding of the network's structure. Other researchers have adopted the 2.5D visualization [11] which utilizes a staggered z-axis to place each network layer on a separate plane, thereby facilitating a clear display of the inter-layer connections. This hybrid leverages the user's intuitive perception of depth to effectively convey hierarchical structures, proving particularly advantageous in tasks that require exploration of inter-layer interactions.

Previous research has treated inter-layer connections with equal priority as inner-layer links in visual depictions. However, in the context of cloud computing systems analysis, inter-layer connections frequently hold a lower priority. These are typically considered weak links, whose visibility is not necessary during the analytical phase. Traditional visualization techniques that indiscriminately display all inter-layer connections often result in significant visual clutter. Our approach integrates the notion of inter-layer connection information directly into the weights of inner-layer edges. We adeptly mitigate the visual overload issue, thereby enhancing the clarity and comprehensibility of the network visualization.

2.2 Network Simplification

The objective of Network Simplification is to enhance computational and visualization efficiency by reducing or aggregating nodes and edges while preserving the pivotal graph structure. This paper concentrates

on two critical aspects of Network Simplification: the scaling down of data and the mitigation of hierarchical complexity [27].

A comprehensive review of recent techniques that reduce the scale of network data is offered by the surveys [4,25,36]. Existing studies can be broadly categorized into selection-based and aggregation-based strategies. **Selection-based methods** utilize filtering or sampling techniques to prune the original graph to a representative subset of its nodes, edges, layers, or temporal frames. Innovations in representation learning have furthered this pursuit, enabling context-aware sampling that adeptly preserves the structural hallmarks of the parent network [29,45,58,60]. On the other hand, **aggregation-based methods** concentrate on condensing the graph through partitioning and summarization techniques. For instance, Multiscale Snapshots traces the temporal progression of dynamic graph properties across different scales through temporal hierarchical summarization [6].

To simplify the hierarchical structure, existing works employ node and edge aggregations method to merge elements across layers, reducing the number of inter-layer interactions or layers [27,32]. Projection-based approaches are particularly beneficial in addressing networks with hierarchical heterogeneity by projecting nodes in multiple layers into a singular unified graph representation [49]. The weights assigned to these projections can be meticulously calibrated, taking into account network salience and exclusivity metrics to suit the bespoke analytics exigencies encountered in real-world applications [42,43].

Our work introduces an abstraction approach that integrates both hierarchical and spatial reductions, specifically designed to address the demands posed by large multi-layer networks within cloud systems.

2.3 Graph Mixing

The key of graph mixing involves the fusion of distinct structural characteristics derived from various graphs. This process are often applied to combine different sources of data to enrich the information of limited graph datasets [12]. **Node-level alignment** is one category of methods to achieve network mixing, such as ifMixup [22] and S-Mixup [35]. They mix nodes across the source and target graphs by either randomly generating sequences and interpolating graph adjacency matrices or referring to the node correspondence. **Subgraph synthesis methods** is another category for mixing networks; SubMix [57] and Graph Transplant [44] engage in selecting and combining subgraphs randomly or based on node information. However, these methods are designed to optimize statistical metrics rather than preservation of features discernible through visual analysis.

Graphons, a contraction of "graph function" [1, 37], is a mapping $W : [0,1]^2 \to [0,1]$ from the unit square to the unit interval. It can be interpreted as an adjacency matrix representing a graph with innumerable nodes, where the matrix entries, or function values, denote edge weights [7]. Mathematically, a graphon is the limit of a converging sequence of graphs, capturing the essence of a graph series through edge existence probabilities [37]. In practical applications, a matrix of node linking probability is used to represent a garphon discretely [53]. A notable property of graphons is that conducting linear combination on two graphons blends the characteristics of them, with the result interpreted as a new graphon [7]. Based on this property, G-mixup [24] proposes to mix graph data through the linear combination of graphons extracted from different graph classes.

Our work extends the application of graphon mixing to represent inter-layer connections within inner-layer frameworks. Moreover, our aggregation-based mixup scheme can address the node misalignment challenges posed by layer heterogeneity.

3 Design Goals

This research is origin from the practical requirements of a cloud network system maintenance and management project. In collaboration with six operational experts from a multinational telecommunications corporation (including some co-authors of this paper), we have engaged in discussions and investigations over a year. To help experts gain insights into the cloud system's overview structure, the following design goals are expanded:

G1: Hierarchy Preservation and Intuitive Visualization, which balances users' effort with the benefits of observing hierarchical structures. This involves deemphasizing the weaker inter-layer connections to reduce visual clutter, while still maintaining their presence to ensure the integrity of the multi-layer network's overview and improve cognitive continuity during layer switching.

G2: Layer Compression for Insightful Summarization, highlighting both stable and volatile structures within the network to provide operational experts with a clear understanding of the network's backbone and periphery. This involves creating a representation that can quickly convey the stability and significance of various substructures from incomplete observations of communication relationships.

G3: Spatial Compression for Enhanced Readability, focusing on reducing the scale of large networks to make structural features more discernible. The goal is to present a simplified yet accurate representation of the network that enables the identification of important features such as clusters and bridges while maintaining the coherence of inter-layer relationships.

4 Methodology

Informed by our design goals, we present a three-stage network abstraction method to simplify large-scale multi-layer networks. First, in **Inner-layer Summary** stage, we compress a sequence of observations of inner-layer communications into a graphon-based probability network representation [37]. Then, we introduce a **Inter-layer Mixup** strategy to mix heterogeneous graphons between adjacent layers by propagating inner-layer information through inter-layer connections. Finally, to reduce visual clutter, we propose a **Hierarchy-aware Sampling** strategy to learn a structure-aware semantic space [60] for all nodes and propose a blue noise-based multi-layer joint sampling technique to simplify multi-layer networks.

4.1 Definition

Here, we formalize the data structure for multi-layer networks in cloud computing systems. A multi-layer network can be represented by $G = (\mathscr{L}, \mathscr{R})$ which consists of a set of layers $\mathscr{L} = \{L_1, \ldots, L_n\}$ and inter-layer connections $\mathscr{R} = \{R_1, \ldots, R_{n-1}\}$. Each layer $L_k = (V_k, \mathscr{E}_k)$ comprises a set of nodes V_k and inner-layer links $\mathscr{E}_k = \{E_k{}^1, \ldots, E_k{}^{m_k}\}$ for m_k observations. Each E_k represents an incomplete observation of the inner-layer communication behavior. The inter-layer connection $R_k \subseteq V_k \times V_{k+1}$ between the k-th and $(k+1)$-th layers. We also define two concepts for mathematical simplification. A graphon-based probability representation $GO = (V_k, W_k)$ offers a condensed expression of an inner layer L_k, where V_k denotes the node set and W_k is the weight matrix describing the connection probabilities between nodes. A sampled subset of the graphon $GO = (V_k, W_k)$ is denoted by $GO^s = (V_k^s, W_k^s)$, where $V_k^s \subseteq V_k$ and W_k^s is the remaining weight matrix after selecting the rows and columns corresponding to nodes within V_k^s.

4.2 Inner-layer Summary

We aim to estimate a graphon with a communication probability matrix W from a series of incomplete observations $E_1, \ldots, E_m$ of layer L.

We model the observation process as follows. Assume that W is the ground-truth probability matrix, where each entry $w_{ij} \in [0,1]$ describes the communication probability between a pair of nodes. Since the actual communication behaviors in the network are time-varying, we set each link e_{ij} as a random variable that satisfies $e_{ij} \in \{0,1\}$ and $\mathbb{E}(e_{ij}) = w_{ij}$. However, given the numerous nodes within a cloud computing system, it is impractical to monitor all communications simultaneously, resulting in a small portion of communications being observable and recorded. Suppose that each communication e_{ij} has a probability p of being observed, independently of the others. Therefore, we have the observation e_{ij}^o of the link e_{ij}:

$$e_{ij}^o = \begin{cases} e_{ij}, \text{ with probability } p \\ \text{unobserved, with probability } 1-p \end{cases} \tag{1}$$

which forms an observation sequence as $E_1, \ldots, E_m$.

Our goal is to obtain the optimal estimate of W from m observations. To avoid statistical bias that may be caused by incomplete observations [40], we optimize the following objective function:

$$\mathrm{MSE}(\widehat{W}) = \mathbb{E}[\frac{1}{|V|^2}\sum_{i=1}^{|V|}\sum_{j=1}^{|V|}(\widehat{w}_{ij} - w_{ij})^2], \tag{2}$$

where $\widehat{W}$ is our estimate of W and $\mathbb{E}$ is the expectation of a random variable based on m observations. The objective function is equivalent to an incomplete matrix estimation problem. Due to computational efficiency and implementation simplicity, we employ the Universal Singular Value Thresholding (USVT) algorithm [8] to solve $\widehat{W}$.

4.3 Inter-layer Mixup

Generally, given two graphons $GO^1 = (V,A)$, $GO^2 = (U,B)$ and their connections $R \subseteq V \times U$, we aim to mix the structures of adjacent graphons into $\widetilde{GO}^1 = (V,\widetilde{A})$ and $\widetilde{GO}^2 = (U,\widetilde{B})$ according to inter-layer connections R.

Despite the G-Mixup [24] method can linearly combine two graphons into a mixed structure by $\widetilde{A} = (1-\lambda)A + \lambda B$ and $\widetilde{B} = \lambda A + (1-\lambda)B$, G-Mixup assumes that two graphons having isomorphic layers. In other words, G-Mixup requires these graphons to have the same number of nodes, and exist a one-to-one map between the nodes of two graphons. We seek to mix heterogeneous graphons through aggregation by extending the application of G-Mixup to heterogeneous networks. For the (i,j)-th entry a_{ij} in A, we can identify its corresponding nodes v_i and v_j in V, and their associated nodes U_{v_i} and U_{v_j} in U. We average the weights of links between U_{v_i} and U_{v_j} and mix them into A_{ij}. Formally, we have:

$$\hat{b}_{ij} = \frac{1}{|U_{v_i}||U_{v_j}|}\sum_{u_i \in U_{v_i}}\sum_{u_j \in U_{v_j}} B(u_i,u_j) \tag{3}$$

$$\tilde{a}_{ij} = (1-\lambda)a_{ij} + \lambda\hat{b}_{ij} \tag{4}$$

where λ is the mixing coefficient controlling the contribution of adjacent layers. Furthermore, we can mix multiple graphons by calculating $\hat{b}_{ij}, \hat{c}_{ij}, \ldots$, and combining them linearly with a_{ij}.

4.4 Hierarchy-aware Sampling

To preserve the network structures, we perform hierarchy-aware sampling on the multi-layer weighted network in two phases: we first embed the weighted network into a vectorized representation, which compresses the structural and neighboring information of the network into an Euclidean space. Then, we sample the network in the representation space while maintaining the hierarchical structure.

4.4.1 Context-aware Representation of Weighted Networks

Numerous network representation methodologies have been introduced [20,31,46,59], yet only a limited number of studies concentrate on weighted networks. In the context of a weighted network, the significance of a node is greatly influenced by its neighbors and the respective weights of the connections between them. For instance, central nodes tend to establish connections with other nodes through links of higher weights. Consequently, an effective network representation must consider both the structural topology and the link weights.

We employ the recent unsupervised network representation learning method SUGRL [39] as a fundamental model to learn the embedding vectors. Specifically, we employ the unweighted adjacency matrix to capture the topology information, and summarize a weight list for each node as its feature to depict structural similarity. By inputting the adjacency matrix and node features into SUGRL [39], we are able to generate a high-dimensional vector that accurately represents each node. Additionally, to address the challenges posed by high-dimensional space, which makes sampling time-intensive, we apply t-Distributed Stochastic Neighbor Embedding (t-SNE) [51] with automatic parameter tuning [23] to compress these representative vectors into a two-dimensional space, thereby facilitating more efficient sampling.

4.4.2 Multi-layer Joint Sampling

In the process of sampling, it is crucial to address both the structural characteristics of nodes within each layer and the interactions between layers. Sampling layers in isolation can potentially break the hierarchical structure. Imagine a scenario where numerous nodes are connected to a marginal node in the adjacent layer. However, the marginal status may reduce its likelihood of being selected during sampling, resulting in a loss of valuable hierarchy information.

To mitigate these issues, our hierarchy-aware sampling strategy takes into account both inner-layer and inter-layer requirements. **Inner-layer Requirements**: Despite nodes being positioned in the vector space according to their topological and structural similarities (i.e., nodes with akin characteristics are positioned closer to each other), certain attributes like connectivity might not be inherently preserved by importance sampling [60]. Therefore, we enhance the sampling strategy from two aspects. (*R1*) *Preserving backbone*. To preserve the network backbone, we determine the sampling probability as the maximum weight $w_{\max}$ among the edges connected to each node. (*R2*) *Maintaining connectivity*. The sampling strategy should maintain the network as a connected graph. Thus we sample the nodes that do not affect connectivity by checking whether the removal of the node would lead to a disconnected graph. **Inter-layer Requirements**: (*R3*) *Retaining Hierarchy*. To preserve information across layers, we employ an alternating sampling strategy across multiple layers, dynamically adjusting the sampling probabilities based on the outcomes from the preceding layer. This ensures that each node, along with its corresponding nodes in adjacent layers, is either jointly sampled or excluded.

As shown in Fig. 3, the sampling process is as follows.

Step 1: We start with the first layer. We employ the blue-noise sampling technique [55] to pick a node based on its weights $w_{\max}$ (R1) among the nodes that affect connectivity (R2). We generate a Poisson disk [33] with a radius of r around the sampled node, mark all nodes inside this disk as "inactive", and mark nodes within twice the disk radius as active.

Step 2: For nodes in the next layer that have a connection to the nodes we sampled in the previous layer, we increase their chance of being selected (R3). The chance increases depending on how many connections they have to previously selected nodes, calculated with $\alpha^n w_{\max}$, where n is the number of these connections and $\alpha > 1$ controls the consistency of sampling across different layers. After selecting a node while keeping the network connected, we mark its neighbor nodes as "inactive" or "active" just like in Step 1. We repeat this process until initial nodes have been sampled for each layer.

Step 3: We sample another node from active nodes in the first layer using the same method as in Step 1. At this step, we adaptively adjust the disk radius based on how densely packed the nodes are around our selected node: $r_{new} = \frac{r}{density_{node}}$, where $density_{node}$ is the density at the node calculated by Kernel Density Estimation (KDE) [9].

Step 4: We then proceed to the next layer and select one node from the "active" using the method from Step 2. We adjust the disk radius and update nodes to be "inactive" and "active" just like before. This step is repeated, layer by layer, until we mark all nodes as "inactive."

Finally, we obtain a thorough and balanced sampling of nodes of a multi-layer weight network, taking into account both the connection within layers and the hierarchy across layers.

5 Evaluation

In this section, we demonstrate how our proposed network abstraction and visualization approach enable the depiction and exploration of large multi-layer networks. The data are derived from experts' activities in analyzing a real-world cloud computing system, which have been anonymized to ensure privacy.

5.1 Visual Interface

Firstly, to facilitate the evaluations, we have applied our abstraction methodology to multi-layer networks and developed a prototype interface. The interface showcases data information, displays intermediate and final results, and provides interactive network exploration. In

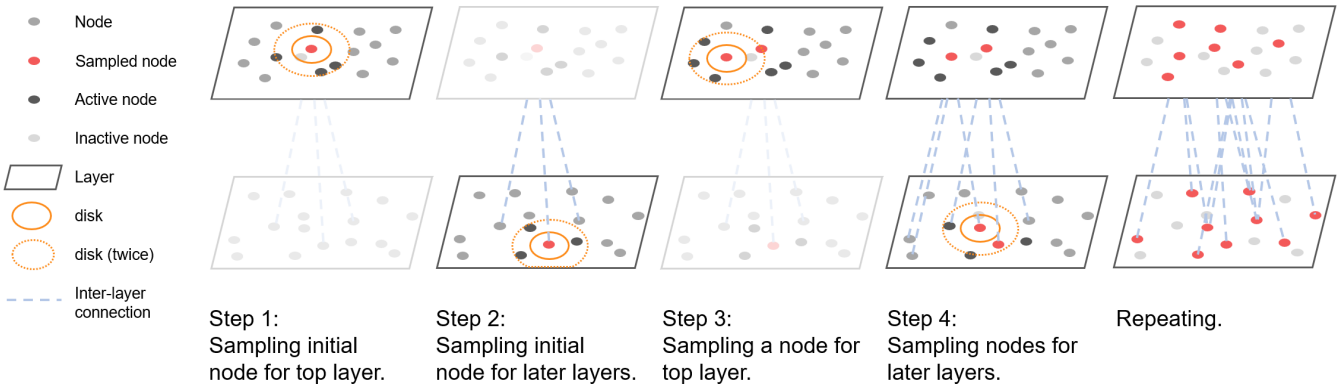


Fig. 3: The proposed multi-layer joint sampling method. In each layer, the nodes are sampled based on link weights, network connectivity, and sampling outcomes of the preceding layer. We employ a blue-noise sampling with an adaptive radius to sample nodes in a representation space.

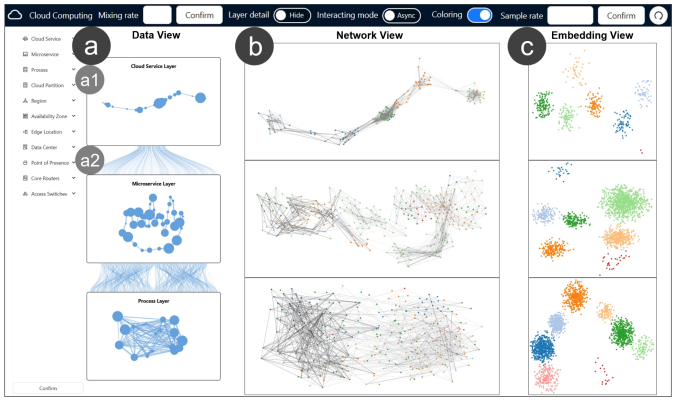


Fig. 4: Our prototype interface. (a) **Data view** with overviews of original networks, including cards (a1) containing inner-layer information and connections (a2) presenting inter-layer information. (b) **Network view** depicting node-link diagrams for layers. (c) **Embedding view** to show representation learning outcomes through scatter plots.

particular, as illustrated in Fig. 4, this interface comprises three key components: a Data View, offering a distinct overview of the original network; a Network View, which presents a node-link diagram for displaying the mixed and sampled network; and an Embedding View, showing the vector representations of nodes, thereby facilitating efficient node clustering.

5.1.1 Data View

To provide an overview of the original network data, our interface includes a card list where each card representing the nodes within a specific layer as shown in Fig. 4(a). Alongside this, a layer list enables users to manually add and manage layers of interest within the card list. On each layer card, essential information including the count of nodes and edges is displayed. We also implemented a high-level aggregated view, or "thumbnail," for each card. Specifically, we extract the top 5% of links that hold the highest probabilities of connection as determined by the graphon of each layer. Leveraging the representation learning method described in Sec. 4.4.1, we project the nodes corresponding to these links into a two-dimensional space. By utilizing the SNAP (Nodes on Attributes and Pairwise edges) algorithm [50], we aggregate the backbone graph into a supergraph and the aggregated graph is then rendered using a force-directed layout, as demonstrated in Fig. 4(a1). Finally, as shown in Fig. 4 (a2), for two adjacent layers in the network, we randomly sample 5% of nodes from each and illustrate their associated inter-layer connections using Bézier curves.

5.1.2 Network View

The Network View (Fig. 4 (b)) presents an integrated visualization of the mixed and sampled network layers. The Fruchterman-Reingold layout algorithm [19] is employed to arrange nodes in each layer, where the weights of link from the mixed graphon denotes the degree of attraction between nodes. This results in a node-link diagram for every layer, where nodes are positioned and connected based on these calculated attractions. The interface allows users to adjust the mixing coefficient, enabling a transition between mixed and unmixed links. The visualization leverages both grayscale color gradation and line thickness to represent the link weights, offering a dual encoding scheme for increased clarity. Users are permitted to navigate through the network via panning, rotating, mirroring, and zooming. Any adjustments made are automatically synchronized across all layers, ensuring a cohesive viewing experience. Sampling is enabled by default to manage the network's complexity. Users have ability to modify the sampling radius or disable sampling for individual layers as needed. Additionally, the interface displays statistics (e.g., the count of nodes, links, and clusters) before and after sampling to aid users in refining their sampling strategies, allowing for exploration of the network's structure.

5.1.3 Embedding View

To gain insights into node roles and the overall network architecture, we compute the vector representations of nodes via representation learning methods, which are then visualized in the Embedding View (Fig. 4 (c)). Within this view, nodes from each layer are depicted as points in a scatter plot, offering a spatial metaphor for their relationships. Selecting nodes within any layer triggers an synchronous highlighting of those nodes across all layers. Furthermore, this view supports clustering analysis; users can cluster nodes directly within the Embedding View. The resulting clusters are colored according to users requirements.

5.2 Usage Scenarios

5.2.1 Examination of Networks Structure

We explored a typical three-tier service architecture comprised of cloud services, microservices, and process entities, containing approximately 500, 2,000, and 2,400 nodes respectively. The original network's structure is depicted in Fig. 4 (a), showcasing distinct geometric arrangements for each tier: the cloud service layer adopts a linear configuration, the microservice layer organizes into a ring formation, and the process layer displays clustering tendencies. Furthermore, the visualization reveals that cloud service nodes establish connections with multiple nodes within the microservices tier. In addition, the connections between the microservices and process entities are divided into two distinct subsets.

Following the application of our techniques, the simplified networks are visually shown in Fig. 5 (a-c), with approximately 100, 300, and 300 nodes in three layers respectively. Experts observed pronounced dark-colored communication backbones within the microservice layer, noting a transformation from a ring to a linear layout (Fig. 5 (b)). This

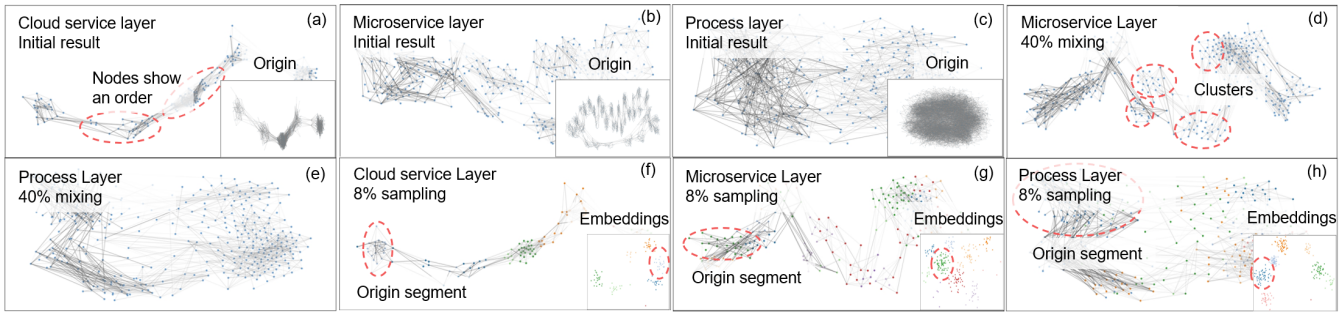


Fig. 5: Visual abstractions for a three-tier service architecture, containing cloud service layer, microservice layer, and process layer. (a-c) are computed under the default setting. (d-e) are obtained by increasing the mixing coefficient. (f-h) are generated by decreasing the sampling rate, with nodes colored according to the clustering results in the embedding view.

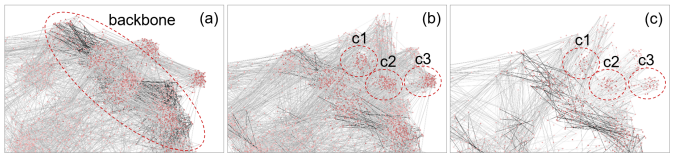


Fig. 6: Abstraction results generated by our approach when locating faults in a subnet through mixing a router network. (a-c) demonstrate abstractions of the subnet after layer summary, mixing, and sampling, respectively.

change was ascribed to the fact that the ring represents two redundant pathways connecting the same origin and destination. Hence, the abstraction process converted this arrangement to a linear form, aligning better with operative insights. The experts also found an ordered pattern among cloud service nodes as illustrated in Fig. 5 (a) which indicates the communication sequence among services by refer to the characteristics in microservice layer. Then, the experts focused on the process layer and recognized a distinct separation within clusters (Fig. 5 (c)). The experts believed this delineation was originated from the bipartite connections between the processes and microservices. To further investigate this speculation, experts adjusted the mixing coefficient to 40%. As shown in Fig. 5 (d-e), microservice nodes exhibited a more pronounced separation, which revealed a dual-route structure, with nodes aggregating into clusters that collectively support various microservices.

Finally, the experts decreased the sampling rate to 8% and performed clustering within the Embedding View, with the results presented in Fig. 5 (f-h). They pinpointed an aligned region across all three layers situated at one extremity of the communication route, encompassing numerous vital communication backbones in both the microservice and process layers. From this observation, the experts deduced that this region represents the route's origin, where a significant concentration of uncompleted communication tasks may be causing bottlenecks. As a result of this inference, the experts marked this particular region as an anomaly for further exploration and potential remediation.

5.2.2 Identification of Anomalous Behaviors

In this scenario, the experts concentrated on pinpointing the causes of failures occurring within a specific subnet, consisting of nearly 30,000 servers. During the investigative process, they observed a router network, whose communication links are influenced by geographic closeness. These routers then connect to servers within the subnet based on the physical network topology. Visual representations in Fig. 6 (a-c) depict the network after applying a 50% mixing coefficient and a 5% sampling rate. Within these visualizations, the intensity of node's color indicates the frequency of failures, thereby allowing for a rapid visual assessment of problematic areas within the subnet.

The investigation began with the experts identifying areas of failure close to a key communication backbone, which serves as a hub connecting various clusters, three of which exhibited failures at a high frequency. The experts suggested that these malfunctioning clusters negatively impacted the overall performance of the backbone. In the visualization using traditional methods, these problematic clusters were occluded by the backbone, as illustrated in Fig. 6 (a). However, the employment of mixing with the router layer and sampling strategy repositioned these clusters, making them distinctly visible in the visual representation (Fig. 6 (b-c)). Detailed analysis revealed that clusters c1 and c2 were directly connected to the backbone, serving as major communication conduits. Conversely, cluster c3 connected to the backbone indirectly, with its communications passing through c1 and c2. This setup highlighted the crucial role of c1 and c2 in handling and transmitting communications from multiple clusters to the backbone. Disruptions in cluster c3, therefore, had a ripple effect, causing disturbances in both c1 and c2. Moreover, the experts uncovered a significant risk factor: the lack of alternative routing paths near the backbone significantly increases its risk of failure. The absence of backup routes meant that any issue in the network's primary or indirect communication lines could lead to extensive delays or blockages, stressing the importance of redundant routes in a network infrastructure.

5.3 Comparative Experiments

To evaluate the effectiveness of each component of our approach, we conducted comparative experiments by modifying specific modules and quantitatively analyzing the effects before and after modification.

5.3.1 Metrics

We first define the following metrics to measure the effectiveness of each method in multiple aspects:

- **Backbone Structure Preservation (BSP)**: We assess whether the main communication structures within a layer are fully preserved. This metric calculates the proportion of backbone structure edges retained in the sampling results. Higher values indicate better preservation.
- **Cluster Disparity (CD)**: Clusters within the network are patterns of interest to operational experts. We calculate the cluster distribution histogram of the original network (across all observations) and the simplified network, measuring the preservation of clusters by the disparity between them [60] (i.e., the Jensen–Shannon divergence [16]). Lower values indicate smaller changes, suggesting that the transformation does not significantly disrupt the original network characteristics.
- **Inter-layer Connection Reflection (ICR)**: This metric measures whether the characteristics of other layers are expressed in the current layer. Specifically, we draw both the original and mixed (unsampled) networks within the same spatial scope to ensure that the Euclidean distances between nodes have the same meaning. For each layout, we calculate the average distances between the nodes to the corresponding cluster of the adjacent layer, where each cluster produces a distance. We compute the ratio of the

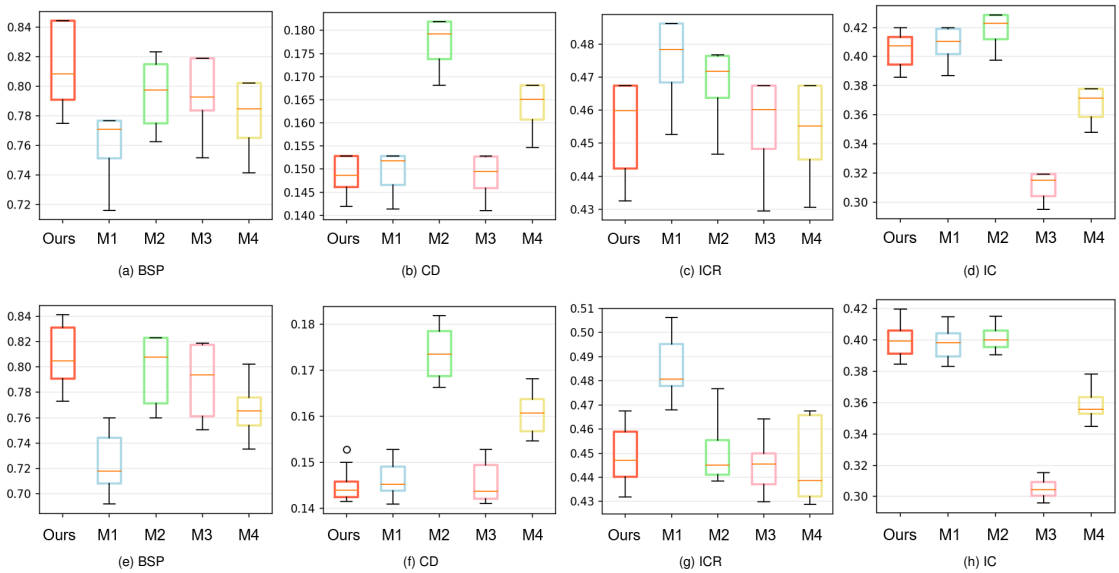


Fig. 7: Experiment results shown in boxplots. Each plot corresponds to a metric. For metrics BSP and IC, achieving higher values is advantageous, while for metrics CD and ICR, lower values mean higher performance. (a)-(d) are experimented on synthetic datasets and (e)-(h) are experimented on real-world datasets.

distances after and before the transformation for each cluster and average the ratios as the metric. To eliminate randomness, multiple layouts with random initialization are computed and averaged. Lower values indicate a higher degree of reflection.

- **Inter-layer Consistency (IC)**: IC measures the preservation degree of the inter-layer neighborhood (i.e., connected nodes within adjacent layers). For each sampled node, we calculate the ratio of the sampled inter-layer neighbors to the original neighbors. Higher values indicate better consistency across layers in sampling results.

BSP and CD quantify the preservation of inner-layer characteristics, while ICR and IC describe the inter-layer preserving.

5.3.2 Datasets

We utilized both synthetic datasets and anonymized real-world data for our Comparative Experiments.

Synthetic Datasets: Each synthetic dataset is composed of: a source graph, a target graph, and links. We generate graphs with 20,000 nodes arranged in four distinct structure patterns: linear, circular, clustered, and linear-clustered. Connections were randomly established, linking each source node with one to three target nodes. Nodes were categorized based on their clustering group and links can be categorized as: (1) Group-Preserving links: where nodes from the same group in the source graph link exclusively with nodes from a single group in the target graph, and (2) Group-Disrupting links: allowing nodes from a group in the source graph to connect with any node in the target graph. We designated 1%-5% of these connections as 'backbone' structures, applying resampling probabilities of 0.9 for backbone and 0.2 for non-backbone connections, and incorporating a 50% chance of connections being unobserved. This method yielded 2,000 observations per edge, culminating in the creation of 32 datasets. The parameters used to synthesize these datasets facilitated the extraction of precise backbone structures and clusters, serving as a ground truth for evaluation.

Real-World Datasets: We incorporated a comprehensive dataset obtained from experts, encompassing seven layers of information (three of which are extensively analyzed in Sec. 5.2.1). This dataset contains 60,000 nodes and 800,000 inter-layer connections, with observations on communications ranging from 1,000 to 10,000 per layer. Each pair of adjacent layers, acting as source and target with links, was treated as a separate dataset, resulting in a total of 12 distinct datasets for examination.

5.3.3 Results

Note that currently existing methods are tailored to solve only one phase of the problem we address in this paper. To enable comparative analysis, we replaced individual modules within the framework of our approach with four existing methods. As detailed in Tab. 1, we implement existing methods M1 to M3 in place of specific modules of our system. Additionally, method M4 was used to examine the impact of resequencing the last two modules in our approach.

Table 1: Methods employed in the comparative experiments.

Method	Description
M1	Replacing the inner-layer summary module with observation frequency.
M2	Replacing the heterogeneous layer mixup module with the alignment method of S-mixup [35].
M3	Replacing the sampling module with a recent sota single-layer sampling method proposed in [58] that well preserves clustering structures.
M4	Swaping the order of the mixup module and sampling module.

We conducted the comparative experiments on each dataset using a mixing coefficient, λ, set at 50% and a sampling rate of 10%. The outcomes were then evaluated based on four performance metrics as shown in Fig. 7. It's noteworthy that for the ICR, since it is assessed using the full, unsampled graph, methods M3 and M4 yield results comparable to our approach. From the result, we observed:

1. Utilizing frequency as the basis for inner-layer summary led to a reduction in both backbone preservation and the accurate representation of inter-layer connections. This effect was more pronounced in the real-world data compared to the synthetic data.
2. M2 demonstrated improved layer consistency; however, this was at the expense of diluting both backbone and cluster structures inherent in the original graph.
3. Our method outperformed M3 in maintaining inter-layer consistency. Moreover, although M3 effectively preserved cluster characteristics, it resulted in a diminished backbone structure.

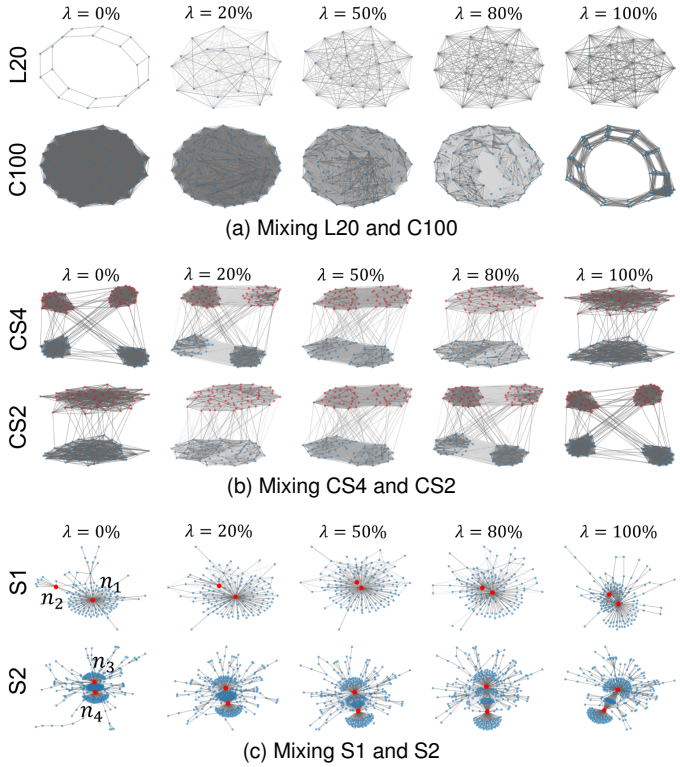


Fig. 8: Three sets of mixing results. The edges of the generated graphs are displayed and the grayscale of the edges encode the weights, where darker edges have higher weights.

4. Altering the sequence of the mixup and sampling processes negatively impacted both inner-layer preservation and inter-layer consistency. This was because the early sampling phase possibly eliminate edges that are vital for the subsequent mixup procedure.

5.4 Evaluating the Inter-layer Mixup

As the mixup plays a significant role in our approach, we designed a series of experiments to evaluate the proposed graphon-based layer mixup method. In the experiments, we regard the graph adjacency matrices as the graphons (also known as associated graphons [15, 37]) to apply the mixup method.

5.4.1 The Underlying Principles of Mixing

We investigated how the mixup affects the layout results. For this purpose, we selected three sets of data for our experiments. The first set contains a circular ladder graph (L20) with 20 nodes and a complete graph (C100) with 100 nodes. Connections between the two graphs were uniformly generated. Specifically, for each L20 node, 5 C100 nodes were randomly selected for connection (without overlap). Each graph serves as the source, with the other as the target. We apply our mixing method under different mixing coefficients to obtain the results. As shown in Fig. 8 (a), each row represents an original graph and each column represents a mixing coefficient λ. When $\lambda = 0$, it indicates no mixing occurs, thus presenting the original graph. In the results, the generated weighted graphs were laid out using the spring layout. For ease of comparison, we display the edges of the generated weighted graphs with the edge weights encoded in the grayscale of the edge color, where darker edges have higher values (the results below are presented in the same manner). We observed that as the mixing coefficient increases, the fully connected edges in L20 gradually become apparent and the layout tends towards a complete graph. Meanwhile, the ladder-shaped edges in C100 are preserved with other edges gradually disappearing, varying the layout accordingly.

The second set of data contains graphs CS4 and CS2, which have four and two clusters, respectively. Both graphs have 100 nodes. We establish one-to-one connections between the nodes of the top two clusters in CS4 and the nodes of the top cluster in CS2, and similarly for the bottom clusters. The colors (red or blue) in Fig. 8 (b) show the correspondence. After mixing, the nodes of the same color in CS4 gradually merge into one cluster, while the original clusters in CS2 split. Eventually, at $\lambda = 100\%$, CS4 and CS2 present the initial layout of each other.

The third set of data includes two REDDIT social network graphs, S1 and S2 [5]. S1 has a single-center structure, while S2 contains two central nodes. We sort the nodes of both graphs by degree respectively and establish a one-to-one correspondence for the top one hundred nodes based on their rankings. For example, the center node n_1 and the sub-center node n_2 in S1 are connected to the two central nodes n_3 and n_4 in S2, respectively (marked in red in Fig. 8 (c)). We observed that mixing adds new edges between n_1 and n_2. The weights of these edges increase as the mixing deepens, eventually drawing n_1 and n_2 closer together and making n_2 a new center. Meanwhile, in S2, one of the original centers gradually degenerates to the periphery.

The above results demonstrate the mechanisms of mixing under conditions of similar or dissimilar graph structures and random or ordered node associations. Upon mixing, the structure of the target graph is integrated into the source through the addition of new edges, and the weights of edges are modified based on the mixing coefficient λ. This process ultimately results in a mixed graph that embodies characteristics of both graphs.

5.4.2 Varying the Inter-connections and the Layout Methods

Next, we varied the inter-connections and analyze the mixing results to demonstrate the effects of mixup. We present the results of mixing a striped-shaped graph with two types of graphs at a 40% coefficient: one featuring linearly connected clusters and another with randomly connected clusters. The striped graph functions differentiatedly with varying inter-connections, e.g., functioning as a line or as clusters, which presenting different layout results. Please refer to the supplementary materials for more details.

Finally, we experimented with how the force-directed layout method affects the mixup results. The Fruchterman-Reingold layout algorithm [19] and the Kamada-Kawai layout algorithm [30], which both consider edge weights as forces, were applied in the experiments using different initial coordinates. We experimented three initialization schemes: random layout, circular layout (nodes are evenly distributed on a circle), and multipartite layout (treating each cluster as a disjoint subset, sequentially connecting them, and using a multipartite graph layout). Three striped graphs were used as experimental data, which have been mixed with a linear cluster graph. Overall, both layout algorithms were capable of expressing the mixed graph structure that is implicitly represented in the edge weights. Regarding initialization, randomly chosen coordinates can impair the layout results, while circular and multipartite initializations can produce comparable outcomes. Please refer to the supplementary materials for more details.

5.5 Expert Reviews

We invited four experts (E1-E4) for the evaluation of our approach. The experts are affiliated with a global telecommunications corporation and possess over seven years of experience in the operations and maintenance of cloud computing networks. E1 and E2, in particular, collaborated with us to outline two case studies. At the outset of each interview, we introduced our proposed method for network visualization (we omitted the introductory phase for E1 and E2, as they were already acquainted with it). The presentation was structured around the three modules of our method, each explained in terms of its functionality. Utilizing our system, we then showcased the two case studies (as shown in Sec. 5.2), granting the experts hands-on experience with our system for network analysis. This interactive phase allowed the experts to pose questions and share their insights freely. Each session was conducted over two hours, after which we compiled and summarized the feedback received from the experts.

5.5.1 Effectiveness of Our Approach

The experts endorsed our method, which integrates inter-layer connections within inner-layer visualizations. They specifically emphasized the efficiency gains compared to the 2.5D analysis tools they were accustomed to, which made case analysis notably more time-consuming. For instance, in analyzing a three-tier service architecture, the separated layer layout necessitated extensive examination of inter-layer connections to correlate structures across layers. E1 mentioned, "*Clusters of different layers were matched up. In my past work, I had to observe the lines between layers myself to match the clusters and manually adjust their positions, which was both tiring and time-consuming.*" E3 agreed with our decision to exclude inter-layer connections, "*I often need to switch the visibility of inter-layer connections on and off to avoid them interfering with my observation of communication relationships.*" E4 highlighted the accessibility improvements our method offers, "*In the past, only those familiar with the setup of services could quickly find the origin point of communication pathway.*" E2 echoed this sentiment, appreciating the clarity with which node sequences, once conceptual in the mind, could now be visually tracked. E4 also suggested that layer mixing enhanced the intuitiveness of the method, "*When I switch continuously from the cloud service layer to the underlying layers, I can observe the alignment between the layers. This matches my original intuition of multi-layer networks, making me feel like I'm viewing a three-dimensional network from above.*" Similarly, E3 recognized that the results were easy to understand, "*I don't need to spend much effort to adapt to the 2D presentation and can connect it back to the original data.*"

The experts recognized the effectiveness of the method in revealing network anomalies and assisting in fault localization. E1 and E2 observed that the method effectively highlights the network's backbone, which serves as a crucial reference point for operational and maintenance activities. They noted that the backbone not only captures the primary focus of them but also aligns with their existing knowledge, facilitating a rapid comprehension of the network's structure. E3 and E4 shared insights into how the method has benefitted them in addressing operational and maintenance challenges. E3 stated, "*Using this approach, I'm able to swiftly discern the physical connections among malfunctioning devices,*" while E4 said, "*It enables me to quickly identify viable alternative routes.*" E3 pointed out the method's utility in enhancing cross-layer analysis, "*The system's ability to visually align different layers allows for a more effective representation of cross-layer algorithm outcomes.*" E4 agreed with our approach of establishing layers' probability representations, "*We also complete the connection matrix through algorithms, but usually only for a few observations and not generalized to the layer's graphon. Therefore, there is always a lack of a comprehensive and accurate summary of the layers.*"

5.5.2 Suggestions

While the experts acknowledged the method's effectiveness and practicality, they also offered constructive feedback for the visualization system. Firstly, E3 and E4 recommended enhancing the system by incorporating additional data interfaces. This enhancement enables the visualization of broader node attributes within the network view, thereby supporting a wider range of operation and maintenance activities. In response, we expanded the system's capabilities by introducing new data interfaces. Attributes are now visually represented through variations in color, outline, and size of both nodes and edges. Based on this feature, we showcased the second case (as shown in Sec. 5.2.2).

Secondly, the experts expressed interest in the spatial variations of nodes before and after the mixing (notably E3). To address this interest, we are developing an animation feature that visually depicts these positional changes. We plan to add this feature to future versions of the system.

6 Discussion

Efficiency: We analyze the efficiency of our approach using the three-tier service architecture detailed in Sec. 5.2.1. The execution times for each component are presented in Tab. 2, based on tests conducted on a laptop equipped with an Intel i5 CPU. These results reveal that computing the inner-layer summary from the observations is the most time-consuming task. However, the execution times for layer mixup, graph representation learning, and sampling are within acceptable limits, indicating that our approach can significantly reduce the hardware demands for analyzing cloud networks. For practical applications, it is advisable to pre-compute the graphons for frequently used network layers offline, thereby minimizing online computation time. This strategy, while efficient, may require consideration of additional factors such as storage space and the need for periodic updates.

Table 2: Execution time (seconds) of each component of our approach. CSL, ML, and PL denote cloud service layer, microservice layer, and process layer in the three-tier service architecture, respectively.

Layer	Inner-layer Summary	Inter-layer Mixup	Representation Learning	Multi-layer Joint Sampling
CSL	31.24	3.24	1.60	19.21
ML	85.35	18.79	3.11	
PL	72.28	17.62	6.09	

Layout: Our current approach employs general force-directed layout algorithms, such as the Fruchterman-Reingold layout algorithm [19], to determine node positions in node-link diagrams. Our experiments show that these algorithms, even after adjusting the weights of the links, can achieve satisfactory visual outcomes. However, we recognize that force-directed models may converge to local optima, leading to layouts with numerous crossings. Such an issue can compromise our visual abstraction technique, as it depends on users' ability to discern the network structure clearly in the visualization to develop their mental models. Moreover, link crossings can obscure the weights of links, causing a loss of crucial information, including the network's backbone structure. To overcome these challenges, we are exploring the development of specialized network layout algorithms designed to more effectively reveal the network's structural information and prevent the backbone from being obscured. Additionally, we plan to incorporate probabilistic graph layout techniques [48] to demonstrate the variability in node positions before and after adjustments.

Robustness: In our experiments, we evaluated the proposed mixup method across a range of graph types, including real-world social networks, as illustrated in Sec. 5.4. However, we noted that when mixing two graphs with significant disparities in their node and edge counts, the mixup effect tends to be less pronounced on one of the graphs. This observation is further elaborated in the experiment on varying inter-connections available in our supplementary materials. In such cases, it becomes necessary to adjust the mixing coefficient for the graph with weak effects, suggesting that the 'optimal' mixing coefficient may be contingent upon the size of the graph. In the future, we plan to develop an approach that automatically adjusts the mixing coefficient based on the graph's characteristics to achieve the desired mixup effect.

7 Conclusion

In this paper, we propose an abstraction approach for large multi-layer cloud computing networks. Specifically, our approach concentrates on addressing three challenges, namely hierarchical heterogeneity, scalability, and data incompleteness. We leverage graphons to summarize the incomplete communication observations for networks, which estimates a linking probability representation for each layer. Based on the graphons, we mix heterogeneous layers to propagate inner-layer information through inter-layer connections, simplifying the hierarchy. We also learn vectorized representations of nodes and conduct hierarchy-aware sampling to reduce network scale while preserving significant characteristics. The effectiveness of our approach is demonstrated through case studies, quantitative comparisons, and expert reviews.

References

[1] E. M. Airoldi, T. B. Costa, and S. H. Chan. Stochastic blockmodel approximation of a graphon: Theory and consistent estimation. In *Proceedings of*

Advances in Neural Information Processing Systems, pp. 692–700, 2013. 3

[2] L. Akoglu, D. H. Chau, U. Kang, D. Koutra, and C. Faloutsos. Opavion: Mining and visualization in large graphs. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 717–720, 2012. doi: 10.1145/2213836.2213941 1

[3] R. Angles, A. Hogan, O. Lassila, C. Rojas, D. Schwabe, P. Szekely, and D. Vrgoč. Multilayer graphs: a unified data model for graph databases. In *Proceedings of the 5th ACM SIGMOD Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA)*, pp. 1–6, 2022. doi: 10.1145/3534540.3534696 1

[4] A. Bonifati, S. Dumbrava, and H. Kondylakis. Graph summarization. *arXiv preprint arXiv:2004.14794*, 2020. doi: 10.48550/arXiv.2004.14794 2, 3

[5] C. Cai and Y. Wang. A simple yet effective baseline for non-attributed graph classification. *arXiv preprint arXiv:1811.03508*, 2018. 8

[6] E. Cakmak, U. Schlegel, D. Jäckle, D. Keim, and T. Schreck. Multiscale snapshots: Visual analysis of temporal summaries in dynamic graphs. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):517–527, 2020. doi: 10.1109/TVCG.2020.3030398 3

[7] Y. Cao, J. Xu, C. Yang, J. Wang, Y. Zhang, C. Wang, L. Chen, and Y. Yang. When to pre-train graph neural networks? from data generation perspective! In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 142–153, 2023. doi: 10.1145/3580305.3599548 3

[8] S. CHATTERJEE. Matrix estimation by universal singular value thresholding. *The Annals of Statistics*, 43(1):177–214, 2015. doi: 10.1214/14-AOS1272 4

[9] Y.-C. Chen. A tutorial on kernel density estimation and recent advances. *Biostatistics & Epidemiology*, 1(1):161–187, 2017. doi: 10.1080/24709360.2017.1396742 4

[10] D. Chu, F. Zhang, W. Zhang, Y. Zhang, and X. Lin. Graph summarization: Compactness meets efficiency. *Proceedings of the ACM on Management of Data (SIGMOD)*, 2(3):1–26, 2024. doi: 10.1145/3654943 2

[11] M. De Domenico, M. A. Porter, and A. Arenas. Muxviz: a tool for multilayer analysis and visualization of networks. *Journal of Complex Networks*, 3(2):159–176, 2015. doi: 10.1093/COMNET/CNU038 2

[12] K. Ding, Z. Xu, H. Tong, and H. Liu. Data augmentation for deep graph learning: A survey. *ACM SIGKDD Explorations Newsletter*, 24(2):61–77, 2022. doi: 10.1145/3575637.3575646 3

[13] J. Du, C. Jiang, A. Benslimane, S. Guo, and Y. Ren. Sdn-based resource allocation in edge and cloud computing systems: An evolutionary stackelberg differential game approach. *IEEE/ACM Transactions on Networking*, 30(4):1613–1628, 2022. doi: 10.1109/TNET.2022.3152150 1

[14] C. Ducruet. Multilayer dynamics of complex spatial networks: The case of global maritime flows (1977–2008). *Journal of Transport Geography*, 60:47–58, 2017. doi: 10.1016/j.jtrangeo.2017.02.007 1, 2

[15] A. Dunyak and P. E. Caines. Graphon field tracking games with discrete time q-noise. In *Proceedings of the 62nd IEEE Conference on Decision and Control*, pp. 8194–8199, 2023. doi: 10.1109/CDC49753.2023.10383780 8

[16] D. M. Endres and J. E. Schindelin. A new metric for probability distributions. *IEEE Transactions on Information Theory*, 49(7):1858–1860, 2003. doi: 10.1109/TIT.2003.813506 6

[17] F. Faghih, T. Ziegler, Z. István, and C. Binnig. Smartnics in the cloud: The why, what and how of in-network processing for data-intensive applications. In *Companion of the International Conference on Management of Data*, pp. 556–560, 2024. doi: 10.1145/3626246.3654690 1

[18] S. P. Feyer, B. Pinaud, S. Kobourov, N. Brich, M. Krone, A. Kerren, M. Behrisch, F. Schreiber, and K. Klein. 2d, 2.5 d, or 3d? an exploratory study on multilayer network visualisations in virtual reality. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):469–479, 2024. doi: 10.1109/TVCG.2023.3327402 2

[19] T. M. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11):1129–1164, 1991. doi: 10.1002/SPE.4380211102 5, 8, 9

[20] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 855–864, 2016. doi: 10.1145/2939672.2939754 4

[21] V. T. Guimaraes, C. M. D. S. Freitas, R. Sadre, L. M. R. Tarouco, and L. Z. Granville. A survey on information visualization for network and service management. *IEEE Communications Surveys & Tutorials*, 18(1):285–323, 2015. doi: 10.1109/COMST.2015.2450538 1

[22] H. Guo and Y. Mao. ifmixup: Interpolating graph pair to regularize graph classification. *arXiv preprint arXiv:2110.09344*, 2021. doi: 10.48550/arXiv.2110.09344 3

[23] A. Häkkinen, J. Koiranen, J. Casado, K. Kaipio, O. Lehtonen, E. Petrucci, J. Hynninen, S. Hietanen, O. Carpen, L. Pasquini, et al. qsne: quadratic rate t-sne optimizer with automatic parameter tuning for large datasets. *Bioinformatics*, 36(20):5086–5092, 2020. doi: 10.1093/BIOINFORMATICS/BTAA637 4

[24] X. Han, Z. Jiang, N. Liu, and X. Hu. G-mixup: Graph data augmentation for graph classification. In *Proceedings of International Conference on Machine Learning*, pp. 8230–8248, 2022. 3, 4

[25] M. Hashemi, S. Gong, J. Ni, W. Fan, B. A. Prakash, and W. Jin. A comprehensive survey on graph reduction: Sparsification, coarsening, and condensation. *arXiv preprint arXiv:2402.03358*, 2024. doi: 10.48550/ARXIV.2402.03358 3

[26] J. Hoffswell, A. Borning, and J. Heer. Setcola: High-level constraints for graph layout. *Computer Graphics Forum*, 37(3):537–548, 2018. doi: 10.1111/CGF.13440 1, 2

[27] R. Interdonato, M. Magnani, D. Perna, A. Tagarelli, and D. Vega. Multilayer network simplification: approaches, models and methods. *Computer Science Review*, 36:100246, 2020. doi: 10.1016/J.COSREV.2020.100246 3

[28] Y. Jadeja and K. Modi. Cloud computing-concepts, architecture and challenges. In *Proceedings of the International Conference on Computing, Electronics and Electrical Technologies*, pp. 877–880, 2012. doi: 10.1109/ICCEET.2012.6203873 1

[29] B. Jiao, X. Lu, J. Xia, B. B. Gupta, L. Bao, and Q. Zhou. Hierarchical sampling for the visualization of large scale-free graphs. *IEEE Transactions on Visualization and Computer Graphics*, 2022. doi: 10.1109/TVCG.2022.3201567 3

[30] T. Kamada, S. Kawai, et al. An algorithm for drawing general undirected graphs. *Information Processing Letters*, 31(1):7–15, 1989. doi: 10.1016/0020-0190(89)90102-6 8

[31] S. Khoshraftar and A. An. A survey on graph representation learning methods. *ACM Transactions on Intelligent Systems and Technology*, 15(1):1–55, 2024. doi: 10.1145/3633518 4

[32] J. Kim and J.-G. Lee. Community detection in multi-layer graphs: A survey. *ACM SIGMOD Record*, 44(3):37–48, 2015. doi: 10.1145/2854006.2854013 3

[33] A. Lagae and P. Dutré. A comparison of methods for generating poisson disk distributions. *Computer Graphics Forum*, 27(1):114–129, 2008. doi: 10.1111/J.1467-8659.2007.01100.X 4

[34] J. Leskovec. Beyond nodes and edges: multiresolution algorithms for network data. In *Proceedings of the 1st ACM SIGMOD Workshop on Network Data Analytics*, pp. 1–1, 2016. doi: 10.1145/2980523.2980525 1

[35] H. Ling, Z. Jiang, M. Liu, S. Ji, and N. Zou. Graph mixup with soft alignments. In *Proceedings of International Conference on Machine Learning*, pp. 21335–21349, 2023. 3, 7

[36] Y. Liu, T. Safavi, A. Dighe, and D. Koutra. Graph summarization methods and applications: A survey. *ACM Computing Surveys*, 51(3):1–34, 2018. doi: 10.1145/3186727 2, 3

[37] L. Lovász. Large networks and graph limits. *Computer Science Review*, 10:35–46, 2013. doi: 10.1016/J.COSREV.2013.09.001 2, 3, 8

[38] F. McGee, M. Ghoniem, G. Melançon, B. Otjacques, and B. Pinaud. The state of the art in multilayer network visualization. *Computer Graphics Forum*, 38(6):125–149, 2019. doi: 10.1111/CGF.13610 2

[39] Y. Mo, L. Peng, J. Xu, X. Shi, and X. Zhu. Simple unsupervised graph representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 7797–7805, 2022. doi: 10.1609/AAAI.V36I7.20748 4

[40] D. A. Newman. Missing data: Five practical guidelines. *Organizational Research Methods*, 17(4):372–411, 2014. doi: 10.1177/1094428114548590 4

[41] M. Okoe, R. Jianu, and S. Kobourov. Node-link or adjacency matrices: Old question, new insights. *IEEE Transactions on Visualization and Computer Graphics*, 25(10):2940–2952, 2019. doi: 10.1109/TVCG.2018.2865940 2

[42] T. Opsahl. Triadic closure in two-mode networks: Redefining the global and local clustering coefficients. *Social Networks*, 35(2):159–167, 2013. doi: 10.1016/J.SOCNET.2011.07.001 3

[43] B. Padrón, M. Nogales, and A. Traveset. Alternative approaches of transforming bimodal into unimodal mutualistic networks. the usefulness of preserving weighted information. *Basic and Applied Ecology*, 12(8):713–

721, 2011. doi: 10.1016/j.baae.2011.09.004 3

[44] J. Park, H. Shim, and E. Yang. Graph transplant: Node saliency-guided graph mixup with local structure preservation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 7966–7974, 2022. doi: 10.1609/AAAI.V36I7.20767 3

[45] Y. Peng, X. Fan, R. Chen, Z. Yu, S. Liu, Y. Chen, Y. Zhao, and F. Zhou. Visual abstraction of dynamic network via improved multi-class blue noise sampling. *Frontiers of Computer Science*, 17(1):171701, 2023. doi: 10.1007/S11704-021-0609-0 3

[46] L. F. Ribeiro, P. H. Saverese, and D. R. Figueiredo. struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 385–394, 2017. doi: 10.1145/3097983.3098061 4

[47] B. P. Rimal, E. Choi, and I. Lumb. A taxonomy and survey of cloud computing systems. In *Proceedings of Fifth International Joint Conference on INC, IMS and IDC*, pp. 44–51, 2009. doi: 10.1109/NCM.2009.218 1

[48] C. Schulz, A. Nocaj, J. Goertler, O. Deussen, U. Brandes, and D. Weiskopf. Probabilistic graph layout for uncertain network visualization. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):531–540, 2016. doi: 10.1109/TVCG.2016.2598919 9

[49] C. Seierstad and T. Opsahl. For the few not the many? the effects of affirmative action on presence, prominence, and social capital of women directors in norway. *Scandinavian Journal of Management*, 27(1):44–54, 2011. doi: 10.1016/j.scaman.2010.10.002 3

[50] Y. Tian, R. A. Hankins, and J. M. Patel. Efficient aggregation for graph summarization. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 567–580, 2008. doi: 10.1145/1376616.1376675 5

[51] L. Van der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(11), 2008. 4

[52] C. Ware and P. Mitchell. Visualizing graphs in three dimensions. *ACM Transactions on Applied Perception*, 5(1):1–15, 2008. doi: 10.1145/1279640.1279642 2

[53] H. Xu, D. Luo, L. Carin, and H. Zha. Learning graphons via structured gromov-wasserstein barycenters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 10505–10513, 2021. doi: 10.1609/AAAI.V35I12.17257 3

[54] K. Xu, Y. Wang, L. Yang, Y. Wang, B. Qiao, S. Qin, Y. Xu, H. Zhang, and H. Qu. Clouddet: Interactive visual analysis of anomalous performances in cloud computing systems. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):1107–1117, 2019. doi: 10.1016/J.NEUCOM.2022.04.110 1

[55] D.-M. Yan, J.-W. Guo, B. Wang, X.-P. Zhang, and P. Wonka. A survey of blue-noise sampling and its applications. *Journal of Computer Science and Technology*, 30(3):439–452, 2015. doi: 10.1007/S11390-015-1535-04

[56] H. Yang, K. Tang, X. Liu, L. Xiao, R. Xu, and S. Kumara. A user-centred approach to information visualisation in nano-health. *International Journal of Bioinformatics Research and Applications*, 12(2):95–115, 2016. doi: 10.1504/IJBRA.2016.077122 2

[57] J. Yoo, S. Shim, and U. Kang. Model-agnostic augmentation for accurate graph classification. In *Proceedings of the ACM Web Conference*, pp. 1281–1291, 2022. doi: 10.1145/3485447.3512175 3

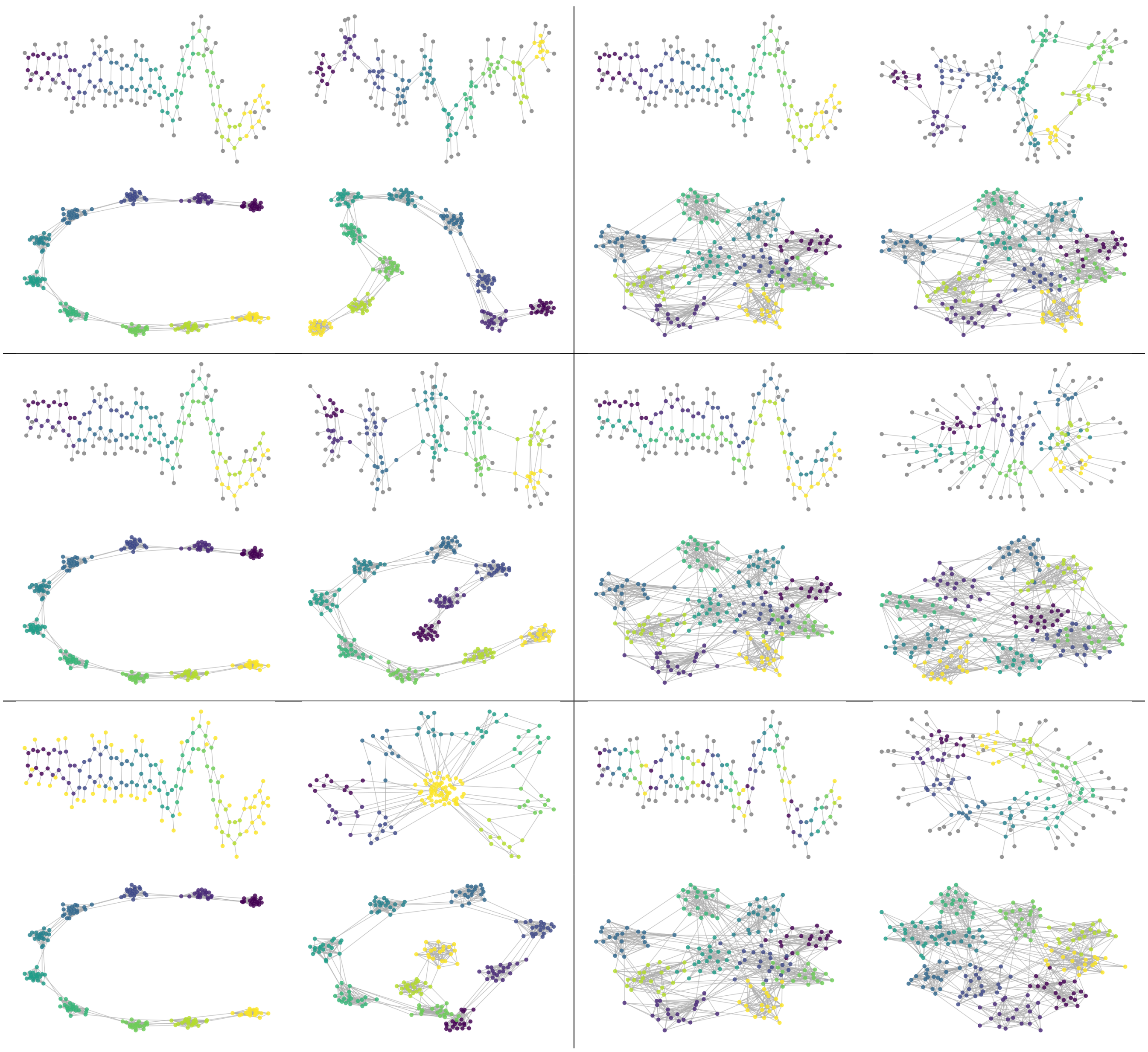
[58] J. Zhang, H. Chen, D. Yu, Y. Pei, and Y. Deng. Cluster-preserving sampling algorithm for large-scale graphs. *Science China Information Sciences*, 66(1):112103, 2023. doi: 10.1007/S11432-021-3370-4 3, 7

[59] W. Zhang, X. Miao, Y. Shao, J. Jiang, L. Chen, O. Ruas, and B. Cui. Reliable data distillation on graph convolutional network. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 1399–1414, 2020. doi: 10.1145/3318464.3389706 4

[60] Z. Zhou, C. Shi, X. Shen, L. Cai, H. Wang, Y. Liu, Y. Zhao, and W. Chen. Context-aware sampling of large networks via graph representation learning. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1709–1719, 2020. doi: 10.1109/TVCG.2020.3030440 3, 4, 6

Table 3: We varied the inter-connections and analyze the mixing results to demonstrate the effects of mixup. We present the results of mixing a striped-shaped graph with two types of graphs at a 40% coefficient: one featuring linearly connected clusters (as depicted on the left side of the table) and another with randomly connected clusters (illustrated on the right side). The node colors signify the inter-connections between nodes. To simplify comparisons, we maintained consistent coloration for the nodes in the two cluster graphs and only altered the node color of the striped graph. All mixed graphs are laid out through force-directed model with edge weights as forces. In addition, we draw the original edges rather than the mixed edges to clarify variations. In the first row, the striped graph functions as a line to connect with the cluster graph. Conversely, in the second row, the two lines of the striped graph separately connect with the cluster graph, which widens the distance between them in the mixing results. In the third row, nodes with distinct structural features are connected, resulting in mixed outcomes significantly different from the original layouts.

Original graph	Mixed graph	Original graph	Mixed graph



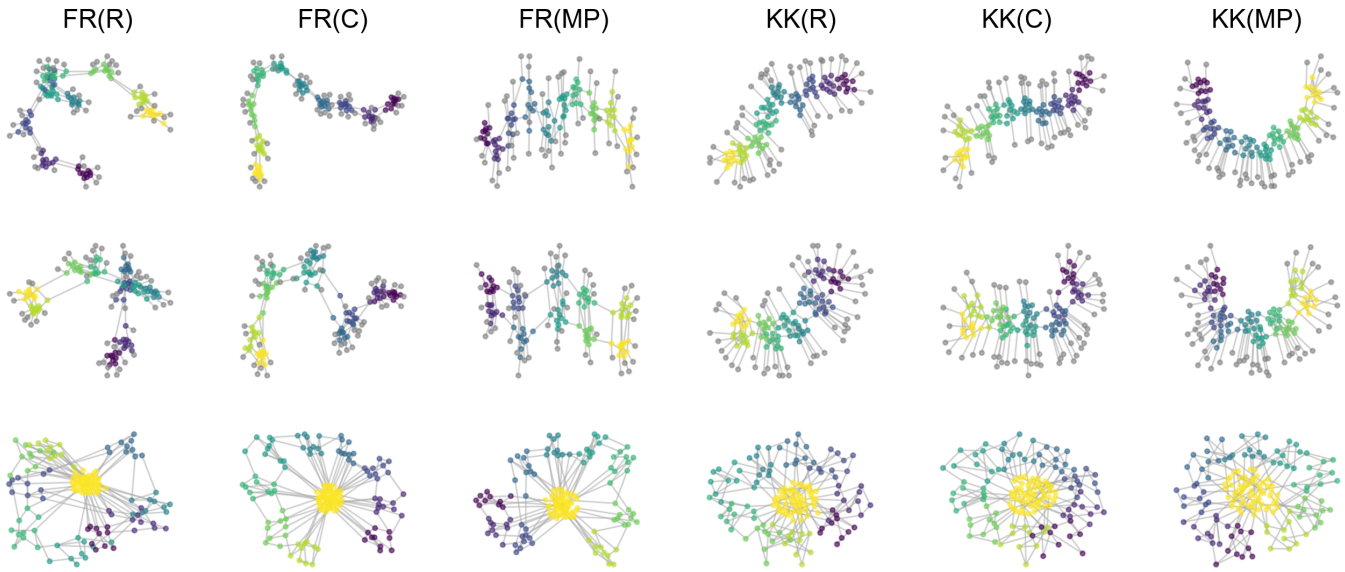


Fig. 9: We experimented with how the force-directed layout method affects the mixup results. The Fruchterman-Reingold layout algorithm and the Kamada-Kawai layout algorithm, which both consider edge weights as forces, were applied in the experiments using different initial coordinates. We experimented three initialization schemes: random layout, circular layout (nodes are evenly distributed on a circle), and multipartite layout (treating each cluster as a disjoint subset, sequentially connecting them, and using a multipartite graph layout). Three striped graphs were used as experimental data, which have been mixed with a graph featuring linearly connected clusters (the same as the mixed results on the left half of Tab. 3), with node colors preserved to illustrate clustering relationships. As shown in the results, each row contains six layouts of a mixed graph. FR and KK stand for Fruchterman-Reingold layout algorithm and Kamada-Kawai layout algorithm, respectively. The R, C, and MP in the brackets stand for using a random, circular, or multipartite initialization for the force-directed model, respectively. Overall, both layout algorithms were capable of expressing the mixed graph structure that is implicitly represented in the edge weights. Regarding initialization, randomly chosen coordinates can impair the layout results, while circular and multipartite initializations can produce comparable outcomes.