Scientific Research Publishing

Open Journal of Applied Sciences, 2024, 14, 2985-2995
https://www.scirp.org/journal/ojapps
ISSN Online: 2165-3925
ISSN Print: 2165-3917

Federated Graph Learning Based on Graph Distance Computing

Wei Gao

School of Information Science and Technology, Yunnan Normal University, Kunming, China
Email: gaowei@ynnu.edu.cn

How to cite this paper: Gao, W. (2024) Federated Graph Learning Based on Graph Distance Computing. *Open Journal of Applied Sciences*, **14**, 2985-2995.
https://doi.org/10.4236/ojapps.2024.1411195

Received: October 3, 2024
Accepted: October 29, 2024
Published: November 1, 2024

Abstract

Federated learning is a classic of privacy-preserving learning, which enables collaborative learning without sharing data. Structured data has become the mainstream of current applications, where there is a correlation between samples in the same dataset, which can be represented by a graph. Federated graph learning (FGL) is the integration of structured data into federated learning, assuming that each user has structured graph representation data and uses graph learning models for training. This article proposes an FGL algorithm based on graph distance calculation, which determines the similarity of users in terms of graph distance, then clusters users, and aggregates local models of users in the same cluster. This algorithm can adjust the number of clusters by changing the threshold (the larger the threshold, the fewer clusters and the more users in each cluster, and vice versa), thereby controlling the scope of user collaboration.

Keywords

Federated Learning, Federated Graph Learning, Graph Distance

1. Introduction

As an archetypal framework of privacy-preserving cooperative learning, federated learning (FL) has received great attention and in-depth research, which enables users to cooperatively learn models without exchanging data and thereby protecting users' personal privacy (see [1] and [2]). Specifically, each user possesses their own private data, which is used to learn their local models. After several local learning rounds, users upload their model parameters to the server, which performs aggregation and distributes the global model parameter to all users for the next episode of learning (see Korba *et al.* [3], Bouzabia *et al.* [4] and Venkatesan *et al.* [5]).

The key issues of FL are how to deal with data heterogeneity and how to transfer knowledge between users with similar data. Each user is eager to absorb positive knowledge from other users to promote local model training. Transfer learning has become one of the hot branches of FL research. Selvakanmani *et al.* [6] raised a transfer learning trick integrated into a federated learning framework in privacy-preserving breast cancer classification circumstances. Raja *et al.* [7] proposed a federated transfer learning framework that integrated a pre-trained ResNet-18 for transfer learning with a meticulously designed convolutional neural network. Singh and Bedi [8] proposed a streamlined approach that merged clustering analysis with federated and transfer learning. Qian *et al.* [9] presented a comprehensive review of federated transfer learning in machinery fault diagnosis. However, due to the huge discrepancies between private data, users with huge differences in data structure tend to provide negative knowledge, and absorbing excessive negative knowledge will reduce the efficiency of the local model.

When considering the data structure of machine learning, we have to take structured data or irregular data into account. For regular data structure, we can think of them as a standard set of data or an Excel spreadsheet. While for the structured data or irregular data, it can be represented by a graph, *i.e.* since there are mutual correlations among samples, and hence vertices are used to represent samples and their associated structures are represented by edges. To resolve the limitations stated above, federated graph learning (FGL) is introduced to deal with the structured data involved in FL. In FGL setting, each user possesses a structured dataset presented by a (weighted) graph, users train the local model parameters in terms of graph neural networks (GNNs), graph convolutional neural networks (GCNNs), or other graph learning approaches (see [10], [11] and [12] for more details). The server aggregates the local model parameters, which are submitted by users, and disseminates the global model to each user (see Xie *et al.* [13], Peng *et al.* [14] and Tang *et al.* [15]). However, the existing FGL has defects in calculating user similarity. Specifically, the similarity function is calculated from scalars or vectors, which are used to represent the user's graph-structured data, while the topological structure of the graph is ignored. Although a small number of contributions extract graph features, the highly concentrated graph feature information often distorts the graph topological features.

To alleviate the foregoing issue, we propose a novel FGL based on graph distance computing, and it has the following two exclusive advantages:

- Graph distance calculation is completely dependent on the topological structure of the graph, and its calculation results are highly reliable, so that the similarity between user data can be accurately determined.
- By means of graph distance computing, FGL algorithm with dynamic clustering can be designed, where each cluster contains users with similar data structures, and the size of the cluster can be adjusted according to the threshold.

In this work, we proposed a novel FGL algorithm based on graph distance computing, where users are clustered into several clusters: the more clusters there are, the fewer users in each cluster; the fewer clusters there are, the more users in

each cluster. The number of clusters can be adjusted by a threshold. The server aggregates the model parameters of users in the same cluster, thereby achieving collaborative training of similar users.

The subsequent sections are organized as follows. First, a comprehensive survey on FGL is presented in the next section. Then, the settings and problem formalization are described. After that, we present the main algorithm and detailed analysis. Finally, we conclude the paper.

2. Related Works on Federated Graph Learning

The aim of this section is to present a comprehensive survey on FGL in recent years. Chen *et al.* [16] proposed FedGraph for FGL among multiple computing clients, and an intelligent graph sampling algorithm in terms of deep reinforcement learning was also suggested. Wu *et al.* [17] raised a multi-level FGL and self-attention-based personalized indoor localization method to capture the intrinsic features of received signal strength. Zeng *et al.* [18] introduced a feature-contrastive graph federated learning approach to improve the robustness of the federated learning system in graph data. Chang *et al.* [19] determined a graph-based client selection framework for heterogeneity in federated learning. Liu *et al.* [20] studied FGL under a cross-silo setting to deal with network traffic throttling and flow scheduling problems. Zhang *et al.* [21] argued a specificity-aware FGL framework for rs-fMRI analysis and automated brain disorder identification. Zhang *et al.* [22] propose an FGL framework by means of ego-graphs, in which each client trains their local models while also contributing to the training of a global model. Li *et al.* [23] proposed federated graph topology-aware aggregation in terms of topologyaware local smoothing confidence and mixed neighbor features. Chen *et al.* [24] investigated the problem of stealing graph data in the FGL setting. Fu *et al.* [25] devised an FGL approach for privacy-preserving individual-level infection prediction. Kong *et al.* [26] considered a model heterogeneous graph federated learning framework in terms of prototype learning and local data augmentation. Zhong *et al.* [27] designed a lightweight FGL approach for accelerating classification inference in unmanned aerial vehicles-assisted multiaccess edge computing systems. Xie *et al.* [28] proposed a decentralized cross-institutional personalized FGL algorithm for IoT service recommendation. Abdel-Basset *et al.* [29] introduced a digital twin framework for creating a digital replica of the physical slicing-supported network in terms of FGL approach. Tian *et al.* [30] proposed a FGL approach for privacy-preserving cross-domain recommendations to capture users' preferences. Xing *et al.* [31] proposed an FGL framework for threat detection of TAP rules while simultaneously protecting user data and privacy.

The existing approaches of FGL fully reflect the fusion of data and try to use a variable to represent the topological structure of a graph, so as to realize the graph similarity calculation by calculating the gap between variables. These methods cannot reflect the real topological structure gap between graphs, even if the variable describes a certain topological indicator parameter in the graph.

3. Preliminary and Settings

The purpose of this section is to introduce the setting of FGL, and the corresponding problem is formally formulated.

3.1. Federated Graph Learning

Federated graph learning is considered as a distributed training framework for graph learning models (e.g. GNNs, GCNNs), or as a framework for large-scale subgraphs collaborative training. It leverages to cope with the computational and storage restrictions by dealing with large-scale subgraphs in terms of local graph training policies.

Let $\mathcal{U}=\{U_1,U_2,\cdots,U_N\}$ be the set of N users, and user U_i possesses a private local dataset, which is represented by a graph $G_i=(V_i,E_i)$. Actually, $V_i=\left\{v_i^1,v_i^2,\cdots,v_i^{|V_i|}\right\}$ is the set of samples of user U_i, while there is an inherent correlation between samples in V_i, and hence edges between samples represent such intrinsic connection. The edge set E_i indicates the topological characteristics of the data structure of user U_i. Without loss of generality, we assume samples can be represented by $v_i^j=\left(\mathbf{x}_i^j,y_i^j\right)$ for $i\in\{1,2,\cdots,N\}$ and $j\in\{1,2,\cdots,|V_i|\}$, where $\mathbf{x}_i^j\in R^d$ and $y_i^j\in R$. In particular, $v_i^j=\mathbf{x}_i^j$ in an unsupervised learning setting.

Prosaically speaking, in the FGL setting, each user U_i trains a local model parameter using a graph-structured dataset G_i and a training framework (GNN, GCNN, etc.), and the server obtains the global model parameter by aggregating the local model parameters submitted by users. FGL currently has a wide range of applications in various fields, such as biomedicine, finance, and industry.

Example 1: Ontology can be seen as a collection of concepts with semantic information. Based on the hierarchical relationship between concepts, ontology can be represented as a directed acyclic graph, where concepts are represented by vertices, and the inherent relationships between concepts are expressed by edges between vertices. Assume that the semantic information of each vertex in the ontology graph can be represented by d-dimensional vector, then an ontology function $f:\mathbb{R}^d\to\mathbb{R}$ (*i.e.* $f:V\to\mathbb{R}$, and here V is the vertex set of ontology graph) is a dimensionality reduction operator that aims to map each vertex to a real number, and similarity between two concepts is measured by the difference between their mapped real numbers, *i.e.* the similarity between v and v' (represents two concepts) is determined by $\left|f(v)-f(v')\right|$.

Suppose we have multiple ontology digraphs and aim to get an ontology function that enables us to map each vertex (in any ontology graph) to a real number. This is the standard problem of ontology mapping, and the similarity between two concepts (in the same ontology graph or two vertices from different ontologies) can be computed in a similar way. In this setting, all ontology digraphs are merged into one large ontology digraph, and each original ontology digraph becomes a

connected component of the merged ontology digraph (see Gao and Chen [32] and Gao *et al.* [33] for more details about ontology similarity measuring and ontology mapping).

Now, imagine that ontology digraphs are distributed among multiple users, each user training ontology mapping function using the local ontology dataset, and each user doesn't expect to share their private ontology data. However, the server aims to obtain the global ontology mapping function, which facilitates to mapping each vertex (it can be from an ontology graph for any user) to a real number. In this case, the ontology mapping problem becomes a standard FGL problem.

3.2. Distance between Graphs

In this subsection, we mainly introduce the computational method for graph distance computing, which was raised by Chowdhury and Mémoli [34].

Suppose that (V_1, w_1) and (V_2, w_2) are two finite graphs such that $w_1 : V_1 \times V_1 \to \mathbb{R}$ and $w_2 : V_2 \times V_2 \to \mathbb{R}$ can be regarded as the weight function on two graphs respectively. It should be noted that we ignore the edge set of the graph here, because for $v, v' \in V_1$ with $vv' \notin E(V_1, w_1)$, then we specify $w_1(v, v') = 0$. A correspondence between V_1 and V_2 is a relation $R \subseteq V_1 \times V_2$. The collection of all correspondences between V_1 and V_2 are denoted by $\mathcal{R}(V_1, V_2)$. The distortion of $R \in \mathcal{R}(V_1, V_2)$ is denoted by:

$$\operatorname{dis}(R) = \sup_{(v_1, v_2), (v_1', v_2') \in R} \left| w_1(v_1, v_1') - w_2(v_2, v_2') \right|. \tag{1}$$

The distance between (V_1, w_1) and (V_2, w_2) is denoted by:

$$d\left((V_1, w_1), (V_2, w_2)\right) = \frac{1}{2} \inf_{R \in \mathcal{R}(V_1, V_2)} \operatorname{dis}(R). \tag{2}$$

If we set $G_1 = (V_1, w_1)$ and $G_2 = (V_2, w_2)$, then $d\left((V_1, w_1), (V_2, w_2)\right)$ can be abbreviated as $d(G_1, G_2)$.

The above computing trick is also applicable to unweighted graphs. For any two vertices in the graph, the weight is set to 1 if they are adjacent, and the weight is set to zero if they are not adjacent.

3.3. GraphSAGE

GraphSAGE is a classic graph convolutional neural network algorithm, and its name SAGE is abbreviated as Sample and AggreGatE. It argued that convolution is a combination of sampling and information aggregation, thus its core idea is to divide convolution into two steps: sampling and aggregation. The aggregation function is independent of the input order, i.e. the vertices in the neighborhood do not need to be sorted. The specific steps can be divided into the following three steps:

1) Get the domain vertex in terms of sampling.

2) Aggregate the information of the neighboring vertices by means of aggregation

function and obtain the embedding of the target vertex.

3) Use the information aggregated on the vertex to predict the label of the vertex/graph.

GraphSAGE uses uniform sampling to sample fixed neighborhood vertices, that is, for a certain vertex, uniform sampling is performed on its first-order neighboring vertices to construct a neighborhood with a fixed number of vertices. Furthermore, GraphSAGE has several kinds of aggregation approaches:

- Mean aggregator:

$$\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W}\cdot \mathrm{MEAN}\left(\left\{\mathbf{h}_v^{k-1}\right\}\cup\left\{\mathbf{h}_u^{k-1}, \forall u\in N(v)\right\}\right)\right).$$

- LSTM aggregator: Use LSTM to encode the features of neighbors. The order of neighbors is ignored here, that is, they are randomly shuffled and input into LSTM.
- Pooling aggregator:

$$\mathrm{AGGREGATE}_k^{\mathrm{pool}} = \max\left\{\sigma\left(\mathbf{W}_{\mathrm{pool}}\mathbf{h}_{u_i}^k + \mathbf{b}\right)\forall u_i \in N(v)\right\}.$$

In the above formulations, k represents the number of depth or layers of the neighborhood; the weight matrix $\mathbf{W}$ (or presented by W^k) denotes the convolution kernel parameters of the k-th layer, which is learnable; σ is the nonlinear parameters, which are usually taken as ReLU function, and sometimes sigmoid function; $\mathrm{AGGREGATE}_k$ represents the aggregation function selected for the k-th layer; h_v^k represents the signal or feature of vertex v on the k-th layer; the information of $\mathbf{h}_v^{k-1}\}$ and $\{\mathbf{h}_u^{k-1}$ is combined using CONCAT.

Note: 1) The sampling domain obtained by GraphSAGE is different in each iteration, which is the biggest difference from GNN. GNN obtains a fixed domain for each vertex through random walks and does not change, but the vertex domain of GraphSAGE is obtained through sampling, and sampling is required once in each iteration, so the domain of each vertex changes. 2) The second type of LSTM aggregation is related to the input order, but a special treatment trick is applied here.

The differences between GraphSAGE and GNN can be summarized by the following three points:

1) In GNN (and traditional CNN), the order of neighborhood vertices needs to be determined. However, in GraphSAGE, the vertices in the graph convolution neighborhood do not need to be sorted (the aggregation function is independent of the vertex order).

2) In GNN, each vertex in the neighborhood has different convolution kernel parameters. In GraphSAGE, all vertices in the neighborhood share the same convolution kernel parameters.

3) In terms of neighborhood selection method, GNN constructs the neighborhood by the probability size of random walks, and GraphSAGE constructs the neighborhood by uniform sampling.

More studies and applications on GraphSAGE can be referred to Zhang *et al.*

[35], Mirthika *et al.* [36], Liu *et al.* [37], El Alaoui *et al.* [38] and Seon *et al.* [39].

4. Proposed Algorithm Description

The procedure of our main algorithm is presented in Algorithm 1.

Algorithm 1 Graph distance based FGL

Input: Set of user $\mathcal{U} = \{U_1, U_2, \cdots, U_N\}$, $G_i = (V_i, E_i)$ as private data for each user U_i, total training rounds T

Output: Global model for each cluster

for each user in $\mathcal{U}$ in parallel **do**
    Generate weighted graph structure $G_i^{'} = (V_i^{'}, w_i)$ where $V_i^{'}$ doesn't contain any data information (remove $(\mathbf{x}, y)$ for each sample)
    Submit $G_i^{'}$ to server
end for
Compute the graph distances in light of $G_i^{'}$
$t = 0$
while $t < T$ **do**
    for each user in $\mathcal{U}$ in parallel **do**
        Local training by means of GraphSAGE
        Submit the local model parameter to sever
    end for
    Expert determines the threshold M_t
    Obtain the clusters $\mathcal{C}_t = \{C_t^1, C_t^2, \cdots, C_t^{n_t}\}$ according to the current threshold M_t, where n_t is the number of clusters in round t
    for each cluster in $\mathcal{C}_t$ in parallel **do**
        Sever aggregate the local model parameters for users in cluster
        Distribute the global model parameter to each user in cluster
    end for
    $t \leftarrow t + 1$
end while

It can be seen from Algorithm 1 that the graph distance is applied to user selection in view of threshold-based user clustering, and the threshold value is not given by input in advance, but determined by experts (decision makers) according to graph distance results. The threshold M_t can be artificially adjusted, and hence the cluster number n_t and combination of users among clusters may have clear distinctions among various time slots. Generally speaking, in the initial stage, the value of M_t can be selected to be larger, so that the number of clusters is smaller and each user can collaborate with more other users. As learning progresses, users tend to collaborate with more similar ones, so the value of M_t can be selected to be smaller in the later stage, which leads to a larger number of clusters and a smaller number of users in each cluster.

It is worth mentioning that Algorithm 1 is applicable to the scenario of mobile edge computing. Each mobile user has its own structured data, which represented by a graph, and their location changes in real time. Each user needs to determine the server closest to it in each round and upload the model parameter to the closest server. In this case, the threshold calculation can be changed to the distance calculation between users, or the cluster calculation centered on the servers. At

this time, the number of clusters is fixed, that is, the number of servers.

5. Conclusion and Discussion

In this paper, we propose a new algorithm for FGL, all users are divided into several clusters according to graph distance, and users in the same cluster aggregate local model parameters, thereby achieving the purpose of knowledge interaction among users with similar graph structures. In practical application, we found that the proposed algorithm has some flaws. One salient issue lies in that the algorithm is only sensitive to graph structure and insensitive to labels and data. We noticed that even two graphs with the same structure have significant differences in their data and different categories of labels. It is difficult to achieve mutual fusion for graph data with significant differences in data categories, even though their graph structures may be very similar.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Chen, Y., Liang, L. and Gao, W. (2023) Non-Trust Detection of Decentralized Federated Learning Based on Historical Gradient. *Engineering Applications of Artificial Intelligence*, **120**, Article ID: 105888. https://doi.org/10.1016/j.engappai.2023.105888

[2] Chen, Y., Liang, L. and Gao, W. (2023) DFedSN: Decentralized Federated Learning Based on Heterogeneous Data in Social Networks. *World Wide Web*, **26**, 2545-2568. https://doi.org/10.1007/s11280-023-01152-4

[3] Korba, A.A., Boualouache, A. and Ghamri-Doudane, Y. (2024) Zero-X: A Blockchain-Enabled Open-Set Federated Learning Framework for Zero-Day Attack Detection in IoV. *IEEE Transactions on Vehicular Technology*, **73**, 12399-12414. https://doi.org/10.1109/tvt.2024.3385916

[4] Bouzabia, H., Meftah, A. and Kaddoum, G. (2024) Federated Learning-Enabled Smart Jammer Detection in Terrestrial and Non-Terrestrial Heterogeneous Joint Sensing and Communication Networks. *IEEE Communications Letters*, **28**, 2026-2030. https://doi.org/10.1109/lcomm.2024.3428973

[5] Venkatesan, C., Jeevanantham, S. and Rebekka, B. (2024) Energy-Aware Federated Learning for AQI Pollutants Forecasting in Edge Networks. *IEEE Transactions on Network Science and Engineering*, **11**, 4146-4157. https://doi.org/10.1109/tnse.2024.3398795

[6] Selvakanmani, S., Devi, G.D., Rekha, V. and Jeyalakshmi, J. (2024) Privacy-Preserving Breast Cancer Classification: A Federated Transfer Learning Approach. *Journal of Imaging Informatics in Medicine*, **37**, 1488-1504. https://doi.org/10.1007/s10278-024-01035-8

[7] Raja, L., Sakthi, G., Vimalnath, S. and Subramaniam, G. (2024) An Improved Federated Transfer Learning Model for Intrusion Detection in Edge Computing Empowered Wireless Sensor Networks. *Concurrency and Computation*: *Practice and Experience*, **36**, e8259. https://doi.org/10.1002/cpe.8259

[8] Singh, G. and Bedi, J. (2024) A Federated and Transfer Learning Based Approach for Households Load Forecasting. *Knowledge-Based Systems*, **299**, Article ID: 111967.

https://doi.org/10.1016/j.knosys.2024.111967

[9] Qian, Q., Zhang, B., Li, C., Mao, Y. and Qin, Y. (2025) Federated Transfer Learning for Machinery Fault Diagnosis: A Comprehensive Review of Technique and Application. *Mechanical Systems and Signal Processing*, **223**, Article ID: 111837. https://doi.org/10.1016/j.ymssp.2024.111837

[10] Xiong, J., Zhai, X., Zhang, Z. and Gao, F. (2023) Typos Correction in Overseas Chinese Learning Based on Chinese Character Semantic Knowledge Graph. *Journal of Data Analysis and Information Processing*, **11**, 200-216. https://doi.org/10.4236/jdaip.2023.112011

[11] Zhang, B. and Si, H. (2023) Learning Path Planning Based on Knowledge Graph on MOOC Platform. *Open Journal of Social Sciences*, **11**, 457-465. https://doi.org/10.4236/jss.2023.113033

[12] Yu, Y. (2022) Research on the Application of Knowledge Graph in Constructing Ecological Chain of Supply of Lifelong Learning Resource Base. *Open Access Library Journal*, **9**, e9255. https://doi.org/10.4236/oalib.1109255

[13] Xie, Z., Huang, Y., Yu, D., Parizi, R.M., Zheng, Y. and Pang, J. (2023) FedEE: A Federated Graph Learning Solution for Extended Enterprise Collaboration. *IEEE Transactions on Industrial Informatics*, **19**, 8061-8071. https://doi.org/10.1109/tii.2022.3216238

[14] Peng, L., Wang, N., Dvornek, N., Zhu, X. and Li, X. (2023) Fedni: Federated Graph Learning with Network Inpainting for Population-Based Disease Prediction. *IEEE Transactions on Medical Imaging*, **42**, 2032-2043. https://doi.org/10.1109/tmi.2022.3188728

[15] Tang, T., Han, Z., Cai, Z., Yu, S., Zhou, X., Oseni, T., *et al.* (2024) Personalized Federated Graph Learning on Non-IID Electronic Health Records. *IEEE Transactions on Neural Networks and Learning Systems*, **35**, 11843-11856. https://doi.org/10.1109/tnnls.2024.3370297

[16] Chen, F., Li, P., Miyazaki, T. and Wu, C. (2022) Fedgraph: Federated Graph Learning with Intelligent Sampling. *IEEE Transactions on Parallel and Distributed Systems*, **33**, 1775-1786. https://doi.org/10.1109/tpds.2021.3125565

[17] Wu, Z., Wu, X. and Long, Y. (2022) Multi-Level Federated Graph Learning and Self-Attention Based Personalized Wi-Fi Indoor Fingerprint Localization. *IEEE Communications Letters*, **26**, 1794-1798. https://doi.org/10.1109/lcomm.2022.3159504

[18] Zeng, X., Zhou, T., Bao, Z., Zhao, H., Chen, L., Wang, X., *et al.* (2023) Feature-Contrastive Graph Federated Learning: Responsible AI in Graph Information Analysis. *IEEE Transactions on Computational Social Systems*, **10**, 2938-2948. https://doi.org/10.1109/tcss.2022.3230987

[19] Chang, T., Li, L., Wu, M., Yu, W., Wang, X. and Xu, C. (2023) Graphcs: Graph-Based Client Selection for Heterogeneity in Federated Learning. *Journal of Parallel and Distributed Computing*, **177**, 131-143. https://doi.org/10.1016/j.jpdc.2023.03.003

[20] Liu, T., Li, P., Gu, Y. and Su, Z. (2023) S-Glint: Secure Federated Graph Learning with Traffic Throttling and Flow Scheduling. *IEEE Transactions on Green Communications and Networking*, **7**, 894-903. https://doi.org/10.1109/tgcn.2022.3187149

[21] Zhang, J., Wang, Q., Wang, X., Qiao, L. and Liu, M. (2024) Preserving Specificity in Federated Graph Learning for fMRI-Based Neurological Disorder Identification. *Neural Networks*, **169**, 584-596. https://doi.org/10.1016/j.neunet.2023.11.004

[22] Zhang, T., Mai, C., Chang, Y., Chen, C., Shu, L. and Zheng, Z. (2023) FedEgo: Privacy-Preserving Personalized Federated Graph Learning with Ego-Graphs. *ACM Transactions on Knowledge Discovery from Data*, **18**, 1-27. https://doi.org/10.1145/3624017

[23] Li, X., Wu, Z., Zhang, W., Zhu, Y., Li, R. and Wang, G. (2023) FedGTA: Topology-Aware Averaging for Federated Graph Learning. *Proceedings of the VLDB Endowment*, **17**, 41-50. https://doi.org/10.14778/3617838.3617842

[24] Chen, J., Ma, M., Ma, H., Zheng, H. and Zhang, J. (2024) An Empirical Evaluation of the Data Leakage in Federated Graph Learning. *IEEE Transactions on Network Science and Engineering*, **11**, 1605-1618. https://doi.org/10.1109/tnse.2023.3326359

[25] Fu, W., Wang, H., Gao, C., Liu, G., Li, Y. and Jiang, T. (2024) Privacy-Preserving Individual-Level COVID-19 Infection Prediction via Federated Graph Learning. *ACM Transactions on Information Systems*, **42**, 1-29. https://doi.org/10.1145/3633202

[26] Kong, X., Yuan, H., Shen, G., Zhou, H., Liu, W. and Yang, Y. (2024) Mitigating Data Imbalance and Generating Better Prototypes in Heterogeneous Federated Graph Learning. *Knowledge-Based Systems*, **296**, Article ID: 111876. https://doi.org/10.1016/j.knosys.2024.111876

[27] Zhong, L., Chen, Z., Cheng, H. and Li, J. (2024) Lightweight Federated Graph Learning for Accelerating Classification Inference in UAV-Assisted MEC Systems. *IEEE Internet of Things Journal*, **11**, 21180-21190. https://doi.org/10.1109/jiot.2024.3365675

[28] Xie, B., Hu, C., Huang, H., Yu, J. and Xia, H. (2024) DCI-PFGL: Decentralized Cross-Institutional Personalized Federated Graph Learning for IoT Service Recommendation. *IEEE Internet of Things Journal*, **11**, 13837-13850. https://doi.org/10.1109/jiot.2023.3340880

[29] Abdel-Basset, M., Hawash, H., Sallam, K.M., Elgendi, I. and Munasinghe, K. (2023) Digital Twin for Optimization of Slicing-Enabled Communication Networks: A Federated Graph Learning Approach. *IEEE Communications Magazine*, **61**, 100-106. https://doi.org/10.1109/mcom.003.2200609

[30] Tian, C., Xie, Y., Chen, X., Li, Y. and Zhao, X. (2024) Privacy-Preserving Cross-Domain Recommendation with Federated Graph Learning. *ACM Transactions on Information Systems*, **42**, 1-29. https://doi.org/10.1145/3653448

[31] Xing, Y., Hu, L., Du, X., Shen, Z., Hu, J. and Wang, F. (2024) A Privacy-Preserving Federated Graph Learning Framework for Threat Detection in IoT Trigger-Action Programming. *Expert Systems with Applications*, **255**, Article ID: 124724. https://doi.org/10.1016/j.eswa.2024.124724

[32] Gao, W. and Chen, Y. (2020) Approximation Analysis of Ontology Learning Algorithm in Linear Combination Setting. *Journal of Cloud Computing*, **9**, Article No. 29. https://doi.org/10.1186/s13677-020-00173-y

[33] Gao, W., Zhang, Y., Guirao, J.L.G. and Chen, Y. (2018) A Discrete Dynamics Approach to Sparse Calculation and Applied in Ontology Science. *Journal of Difference Equations and Applications*, **25**, 1239-1254. https://doi.org/10.1080/10236198.2018.1551383

[34] Chowdhury, S. and Mémoli, F. (2022) Distances and Isomorphism between Networks: Stability and Convergence of Network Invariants. *Journal of Applied and Computational Topology*, **7**, 243-361. https://doi.org/10.1007/s41468-022-00105-6

[35] Zhang, T., Shan, H. and Little, M.A. (2022) Causal GraphSAGE: A Robust Graph Method for Classification Based on Causal Sampling. *Pattern Recognition*, **128**, Article ID: 108696. https://doi.org/10.1016/j.patcog.2022.108696

[36] Mirthika, G. L. S. and Sivakumar, B. and Hemalatha, S. (2024) Data-Driven Drug Treatment: Enhancing Clinical Decision-Making with SalpPSO-Optimized GraphSAGE. *Computer Methods in Biomechanics and Biomedical Engineering*, 1-23.

[37] Liu, J., Ong, G.P. and Chen, X. (2022) GraphSAGE-Based Traffic Speed Forecasting for Segment Network with Sparse Data. *IEEE Transactions on Intelligent Transportation Systems*, **23**, 1755-1766. https://doi.org/10.1109/tits.2020.3026025

[38] El Alaoui, D., Riffi, J., Sabri, A., Aghoutane, B., Yahyaouy, A. and Tairi, H. (2022) Deep GraphSAGE-Based Recommendation System: Jumping Knowledge Connections with Ordinal Aggregation Network. *Neural Computing and Applications*, **34**, 11679-11690. https://doi.org/10.1007/s00521-022-07059-x

[39] Seon, J., Lee, S., Sun, Y.G., Kim, S.H., Kim, D.I. and Kim, J.Y. (2023) GraphSAGE with Contrastive Encoder for Efficient Fault Diagnosis in Industrial IoT Systems. *ICT Express*, **9**, 1226-1232. https://doi.org/10.1016/j.icte.2023.07.012