# Federated ensemble learning with graph neural networks for improved local performance

Erika Musterfrau[c], Max Mustermann[c], Georg Expert[c], Julia Expertin[c], Andreas Holzinger[a,b,*]

[a]*Human-Centered AI Lab, Austria*
[b]*Alberta Machine Intelligence Institute, Canada*
[c]*Lab Name, University Name, Address*

**Abstract**

... the idea is very much like federated feature selection ... on isolated datasets which may come from the same or different distribution.

*Keywords:* Graph Neural Networks, Explainable AI, Federated Learning, Ensemble classifier

## 1. Introduction

This is a sample reference [1]

Why is Federated Learning (FL) important [2]?

## 2. Related work

Federated learning is very closely related to client-side machine learning. A still current problem is that interesting approaches in machine learning are limited to large data centres and to very expensive GPUs. Therefore, the idea of using the billions of smartphones after all - which can provide enormous computing power globally - was interesting. This makes it possible to scale enormously at low cost and, as a quasi-waste product, you get good privacy because the data that is only used for the learning process never leaves the client - it remains on the mobile devices. Since privacy is automatically fulfilled, the entire

*Corresponding author
*Email address:* `andreas.holzinger@human-centered.ai` (Andreas Holzinger )

range of personal information on the client device can be used for the learning process and thus results from users can be made available without having to go through any central servers and can actually contribute to an improved user experience. Malle et al. (2017) [3] have presented an architecture for federated learning that consists of personalized graph-based recommendations computed on client devices that collectively create and improve a global knowledge graph. The original idea here is to train in a network with the effect that overfitting highly subjective data will degrade the global model, which is still an interesting approach.

## 3. Approach

Two major points of justification

- No data sharing at all, because the potential high heterogeneity among data sets may lead to a decrease of performance
- No data sharing at all, because shared gradients maybe attacked and decoded to gain insights from secure data
- Instead, knowledge is shared in the form of GNN explanations (encrypted node ids of the locally important subnetworks)

See Figure 1

Each partie has its own data based on which a GNN classifier is trained. A GNN explainer infers the best performing subnetworks, and the corresponding partie shares these with the other parties. Note, that no private data is shared, only the node ids, when necessary encrypted, are communicated. From the shared subnetworks an ensemble classifier is constructed. Each member of that ensemble consists of a GNN classifier trained based on a shared top-ranked subnetwork. Each ensemble classifier does make predictions on an independent test data set via majority voting.

Our hypothesis is that the ensemble classifier based on the shared networks perform better than those from the locally trained models.

Furthermore, a partie might not necessarily have a suboptimal model on its side, but rather highly optimized one for a specific disease subtype. Thus, the subnetwork recommendations shared by the participating parties can also be rejected. As a consequence, each partie still has its own local classifier, which may differ to the ones from the other parties (see Figure 2)

## 4. Results

## 5. Federated GNN

### *5.1. Previous work*

There are 3 ways to federate the GNN training process [4], as seen in figure 3:

1. Graph-level: Each client holds a set of graphs
   Tasks: graph classification/regression
2. Subgraph-level: Each client holds a subgraph of a larger global graph
   Tasks: node classification, link prediction
3. Node-level: Each client holds the ego-networks of one or multiple nodes
   Tasks: node classification, link prediction

One idea of how to implement this is presented in figure 4. The goal is "to improve their GNN models without necessarily revealing their graph datasets" [4]. In other words, the minimization of the loss function $F(\boldsymbol{W})$ of the central GNN model with weights $\boldsymbol{W}$ is considered to be equivalent to the minimization of the losses of each of the local GNN models with the same weights $\boldsymbol{W}$ as the central one, weighted by their relative number of samples. Equation 1 expresses this principle although this weighting scheme is highly debatable, especially if the datasets are imbalanced in general.

$$\underset{\boldsymbol{W}}{min} F(\boldsymbol{W}) = \underset{\boldsymbol{W}}{min} \sum_{k=1}^{K} \frac{N^(k)}{N} f^{(k)}(\boldsymbol{W}) \tag{1}$$

In the FedAvg algorithm [5], the aggregation function for the computation of the central GNN is the average of the parameters:

The code can be found in: `https://github.com/FedML-AI/FedML/tree/master/python/app/fedgraphnn`. The authors also make experiments with different benchmark datasets and non-IID splits thereof (synthesized partitions) and came to the following conclusions.

Setting the aggregation function to a weighted average is a very simple solution. There are other ones such as FedOpt [6], FedAtt [7], Per-FedAvg [8], and pFedME [9] and most of them are compared in [10].

During the first international FeatureCloud hackathon we thought of a solution where a multi-objective optimization problem is solved in the server through f.e. a genetic algorithm. The process develops as follows:

1. The central GNN starts with a particular sets of weights $\boldsymbol{W}$.
   Those weights are distributed to each of the local client's GNN.
2. Each local client's GNN's loss is returned to the server.
3. According to the losses the genetic algorithm tries another set of weights $\boldsymbol{W}'$.
4. The process is repeated until a good (but locally suboptimal) common set of weights is found for all clients.

This idea is as far as privacy is concerned one of the safest because the only entities that are exchanged are the losses [11]. In most of the other cases, the clients upload the encrypted weights to the server; nevertheless, just the exchange of losses and multi-objective optimization is highly time-consuming.

Other solutions that are tailored for GNNs use Bayesian networks for the determination of the near-optimal parameters [12], without necessarily needing a central model and control. Other solutions take advantage of the fact that the architecture of GNNs is modular and implement the message passing for the computation of the embeddings clients separately, and the loss computing part federally [13].

The difference with ensemble models, as described in [14], [15] is that although the local models influence the global/central model, in FL the parameters of the central model influence - to the point of even being adopted as is sometimes - the parameters of the local models. That means that the exchange of information is not one-way and can happen in many steps.

### *5.2. Similar Graph Counterfactual platforms*

There is a plethora of research works that deal with the automatic generation of counterfactual explanations for graphs [16]. Most of them try to generate the counterfactuals automatically, for example with the use of generative models [17], by just concentrating on edge deletion [18] or by a search driven by a Reinforcement Learning (RL) algorithm [19]. The main goal of those methods is to from a provided graph instance to compute another that has a small graph edit distance and sparsity (modification of fewer features as possible) in the smallest possible time. Preliminary research tries to compare several graph counterfactual generation frameworks under many datasets and evaluation metrics [20], and the result shows no clear winner in all metrics. The methods listed and evaluated do not incorporate solutions where the graph counterfactual generation was driven by human decisions. The evaluation of human-in-the-loop search strategies by similar means as the automated ones is one of the long-term goals of our research.

A similar work to ours that deals with explanations of GNNs via targeted topology perturbation is presented in [21]. The platform "GNNViz" belongs also to the category of Human-in-the-Loop Machine Learning Systems (HITL ML) [22] that was created with the goal that also non-expert users will manipulate graphs and thereby influence the GNN's prediction outcome. They show with the benchmark datasets they are using that with the use of this tool label correlations, unfairness and biases can be uncovered and that a partial comparison with the results of GNNExplainer [23] can be made. The main difference with our work is that we allow and encourage much more changes in the graph as node addition and deletion, as well as nodes and edges features addition and deletion, whereas the GNNViz only considers edge addition and deletion. Furthermore, the recalculation of the prediction result seems to be equivalent to our "predict" method, but there is no functionality that corresponds to our "retrain" method.

The researchers differentiate between three main perturbation strategies: "preserve", which encompasses changes in the topology that leave the prediction unchanged, "promote", which contains all perturbations that affect the prediction results (change the predicted class) and "attack", which are the ones that generate counterfactuals. Their work is not constrained as ours on just accepting changes from the user, but with specially designed losses to compute the edges that can be added or deleted in each of those three modes. The platform sorts and suggests those edges to the user, although it is up to him or her to select that edge (exploration is allowed). Furthermore, there are also filters on the dataset that are used for graph selection and rules by which the graphs will be displayed. The researchers put emphasis on explanations of incorrect predictions, compute and recommend the egde(s) that will flip the prediction towards the correct class. It would be interesting to see how experiments with human users work and what the benefits of this platform is in terms of user acceptance, trust, usability, and causability [24]. The source code will be published upon acceptance and it is going to be an API-based web service.

CLEAR: [17]

GRETEL: [20]

and Survey: [16]

The above-described frameworks are not (yet) federated.

### *5.3. Federated Human-in-the-loop Counterfactual xAI Platform*

The federated Human-in-the-loop Counterfactual xAI Platform belongs to the first category (see section 5.1) where each client and user will interact with one and the same graph dataset in the beginning, but since each person will change the dataset in a different way, eventually the graph datasets will be different in each client. Question: is it non-IID? It is data isolation [13]. It will work as follows:

1. Each user begins with the same test dataset and the same GNN - same architecture and weights since it was trained with the same training set.

The training set is immutable; it does not change by user interactions.

2. Each user interacts on the test set. The test set is the same for all users before any interaction but each user selects its own interactions. There exist ideas on recommending actions that many of the other users also tried. Still, even this will not prevent situations where the resulting datasets, after many additions and deletions will be entirely different.
3. When the aggregation of the central model is going to be triggered is to be defined. The local GNNs change after each "retrain", therefore after every retrain there is an opportunity to change the central model. Nevertheless, since each user decides autonomously when to retrain its own local GNN, the aggregation process is asynchronous.
4. What the aggregation function of shared embeddings will be - from a simple mean to a complex learnable nonlinear function.
5. What the weights could be so that a "consensus" can be reached, where central GNN is sub-optimal to all local datasets, but it is a model that is "good enough" for all.

A simple depiction that does not include the trigger and type of aggregation is in figure 5.

In general, since the action possibilities (node add/remove, edge add/remove, features add/remove) are countable, together with the asynchronous triggers of aggregation, the whole process can be simulated before deployment.

At the moment we are working on the effective storage of each user's GNN models after retrain and a history of interactions. We use the "user_token" to separate the folders of the stored models (and graph datasets) that are created during the interaction with the application. The initial model (as the initial graph dataset) is common to all users. It does not have to be created for each user separately, nor is it loadable/dependable on any "user_token". There is an "init_"folder on the same level as the user folders. The initial GNN models should not be overwritten. The "latest_" folders belong to each user independently. If a user has not had any retrain done, then loading the model can be done directly from the “init_” folder. The folder’s contents will look like in figure 6:

Saving all the “intermediate” graph datasets that are produced by retrain by a user with the datetime is not that straightforward because the datasets can be too big for the server to cope with. Whereas the .pkl files of the GNN models are some hundreds of KB the graph datasets can reach a size of $\sim 100$ MB. The idea is to store a text file that contains only the applied actions of the user (ideally, each one with its timestamp). That would save enormous space resources, but the downside is that for load purposes this file should be loaded, and all actions should be applied in the sequence that is specified in the file - so there will be time costs at load time. Nevertheless, this achieves maximum retraceability.

We assume that at each aggregation step the FL algorithm will use either the latest local model or one of the intermediate.

## 6. Conclusion

## Credit authorship contribution statement

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper. This work does not raise any ethical issues.

## Acknowledgements

The Authors declare that there are no conflicts of interests. This work does not raise any ethical issues. Parts of this work have been funded by the Austrian Science Fund (FWF), Project: P-32554 explainable Artificial Intelligence. Parts of this work have received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 826078 (Feature Cloud). This publication reflects only the authors’ view and the European

Commission is not responsible for any use that may be made of the information it contains.

## References

[1] A. Holzinger, B. Malle, A. Saranti, B. Pfeifer, Towards multi-modal causability with graph neural networks enabling information fusion for explainable ai, Information Fusion 71 (2021) 28–37.

[2] R. Torkzadehmahani, R. Nasirigerdeh, D. B. Blumenthal, T. Kacprowski, M. List, J. Matschinske, J. Späth, N. K. Wenke, J. Baumbach, Privacy-preserving artificial intelligence techniques in biomedicine, Methods of Information in Medicine.

[3] B. Malle, N. Giuliani, P. Kieseberg, A. Holzinger, The more the merrier - federated learning from local sphere recommendations, in: Machine Learning and Knowledge Extraction, Lecture Notes in Computer Science LNCS 10410, Springer, 2017, pp. 367–374. `doi:10.1007/978-3-319-66808-6_24`.

[4] C. He, K. Balasubramanian, E. Ceyani, C. Yang, H. Xie, L. Sun, L. He, L. Yang, P. S. Yu, Y. Rong, et al., Fedgraphnn: A federated learning benchmark system for graph neural networks, in: ICLR 2021 Workshop on Distributed and Private Machine Learning (DPML), 2021.

[5] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial intelligence and statistics, PMLR, 2017, pp. 1273–1282.

[6] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečnỳ, S. Kumar, H. B. McMahan, Adaptive federated optimization, arXiv preprint arXiv:2003.00295.

[7] S. Ji, S. Pan, G. Long, X. Li, J. Jiang, Z. Huang, Learning private neural language modeling with attentive aggregation, in: 2019 International joint conference on neural networks (IJCNN), IEEE, 2019, pp. 1–8.

[8] A. Fallah, A. Mokhtari, A. Ozdaglar, Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach, Advances in Neural Information Processing Systems 33 (2020) 3557–3568.

[9] C. T Dinh, N. Tran, J. Nguyen, Personalized federated learning with moreau envelopes, Advances in Neural Information Processing Systems 33 (2020) 21394–21405.

[10] C. Wu, F. Wu, L. Lyu, T. Qi, Y. Huang, X. Xie, A federated graph neural network framework for privacy-preserving personalization, Nature Communications 13 (1) (2022) 1–10.

[11] H. Zhu, Y. Jin, Multi-objective evolutionary federated learning, IEEE transactions on neural networks and learning systems 31 (4) (2019) 1310–1322.

[12] A. Lalitha, O. C. Kilinc, T. Javidi, F. Koushanfar, Peer-to-peer federated learning on graphs, arXiv preprint arXiv:1901.11173.

[13] L. Zheng, J. Zhou, C. Chen, B. Wu, L. Wang, B. Zhang, Asfgnn: Automated separated-federated graph neural network, Peer-to-Peer Networking and Applications 14 (3) (2021) 1692–1704.

[14] A. Géron, Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow, " O'Reilly Media, Inc.", 2022.

[15] I. H. Witten, E. Frank, M. A. Hall, C. J. Pal, M. DATA, Practical machine learning tools and techniques, in: Data Mining, Vol. 2, 2005.

[16] M. A. Prado-Romero, B. Prenkaj, G. Stilo, F. Giannotti, A survey on graph counterfactual explanations: Definitions, methods, evaluation, arXiv preprint arXiv:2210.12089.

[17] J. Ma, R. Guo, S. Mishra, A. Zhang, J. Li, Clear: Generative counterfactual explanations on graphs, arXiv preprint arXiv:2210.08443.

[18] A. Lucic, M. A. Ter Hoeve, G. Tolomei, M. De Rijke, F. Silvestri, Cf-gnnexplainer: Counterfactual explanations for graph neural networks, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2022, pp. 4499–4511.

[19] D. Numeroso, D. Bacciu, Meg: Generating molecular counterfactual explanations for deep graph networks, in: 2021 International Joint Conference on Neural Networks (IJCNN), IEEE, 2021, pp. 1–8.

[20] M. A. Prado-Romero, G. Stilo, Gretel: Graph counterfactual explanation evaluation framework, in: Proceedings of the 31st ACM International Conference on Information & Knowledge Management, 2022, pp. 4389–4393.

[21] Y. Sun, A. Valente, S. Liu, D. Wang, Preserve, promote, or attack? gnn explanation via topology perturbation, arXiv preprint arXiv:2103.13944.

[22] A. Holzinger, Interactive machine learning for health informatics: when do we need the human-in-the-loop?, Brain Informatics 3 (2) (2016) 119–131.

[23] R. Ying, D. Bourgeois, J. You, M. Zitnik, J. Leskovec, Gnnexplainer: Generating explanations for graph neural networks, Advances in neural information processing systems 32 (2019) 9240.

[24] A. Holzinger, G. Langs, H. Denk, K. Zatloukal, H. Müller, Causability and explainability of artificial intelligence in medicine, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 9 (4) (2019) e1312.

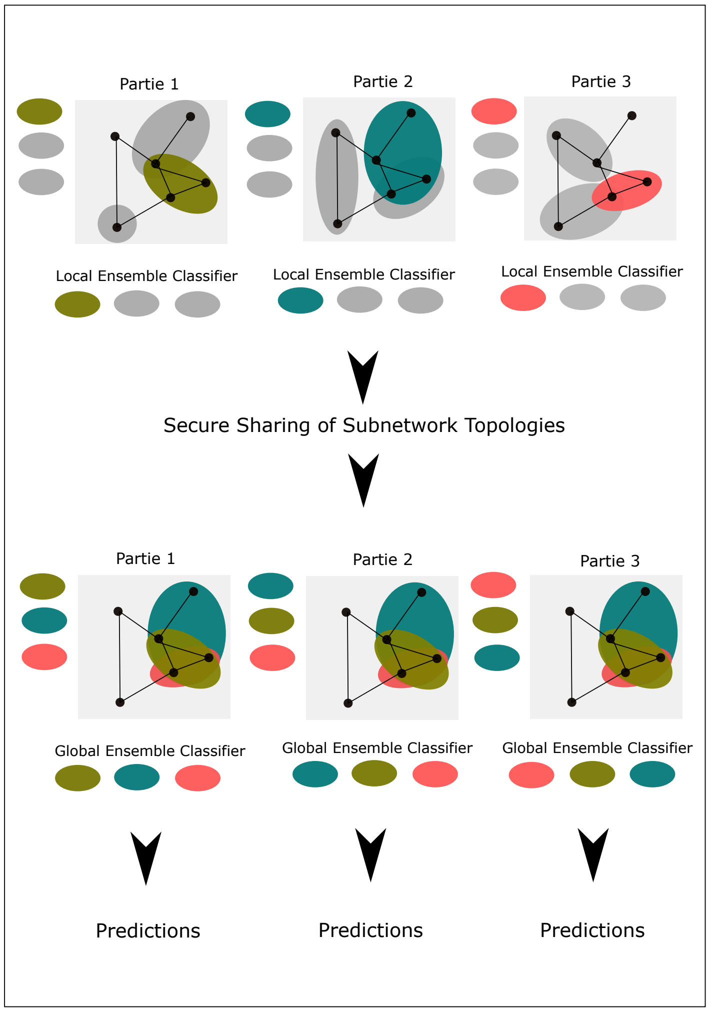

Partie 1
Partie 2
Partie 3
Local Ensemble Classifier
Local Ensemble Classifier
Local Ensemble Classifier
Secure Sharing of Subnetwork Topologies
Partie 1
Partie 2
Partie 3
Global Ensemble Classifier
Global Ensemble Classifier
Global Ensemble Classifier
Predictions
Predictions
Predictions

Figure 1: Each partie has its own locally stored patient data. The network topology is visible for all participants. Informations of locally important subnetworks are shared with the other parties while the patients raw data itself remains strictly local and secure. The figure illustrates the case where all subnetworks are 'accepted' by all parties, resulting in a shared global model.

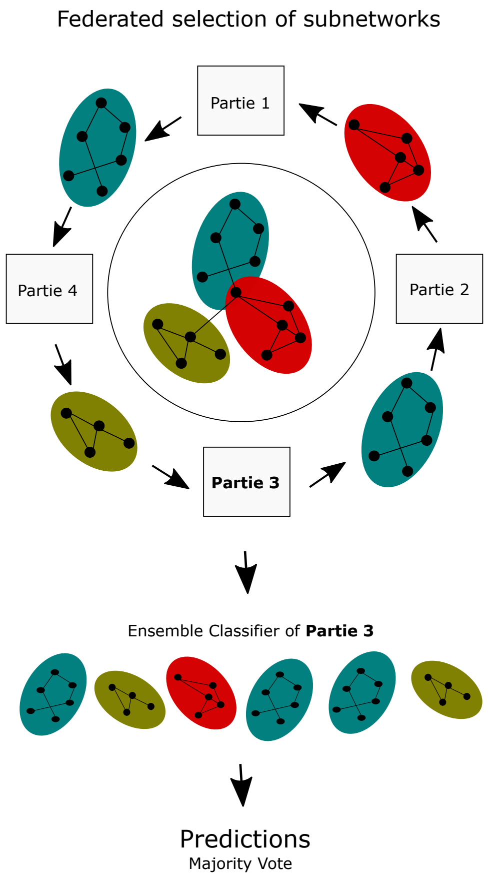
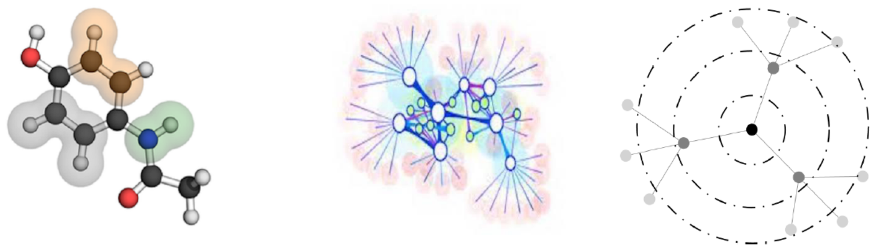

Federated selection of subnetworks
Partie 1
Partie 2
Partie 3
Partie 4
Ensemble Classifier of Partie 3
Predictions
Majority Vote

Figure 2: Recommandations of subnetworks might be rejected by a partie, depending on its accuracy on the local test data set. The used ensemble classifier thus might differ slightly compared to the ensemble models used by the other parties. An example is shown for partie 3.



(a) Graph-level FL (b) Subgraph-level FL (c) Node-level FL

Figure 3: The three possible ways to federate the GNN training process, as far as dataset federation is concerned.

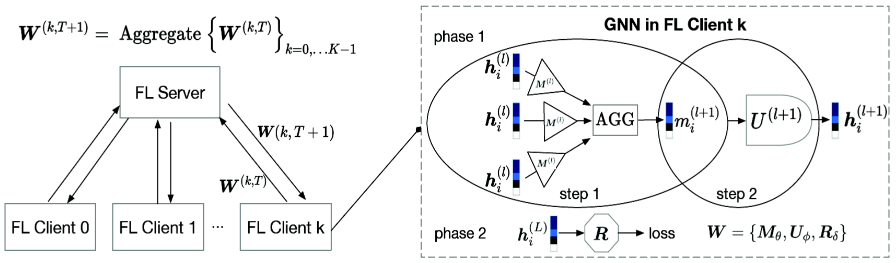


Figure 4: The main idea behind FedGraphNN [4]. On the right side of the image, what happens in each client is the continuous aggregation and combination of neighbouring embeddings for the computation of adequate embeddings for each node and edge until convergence.

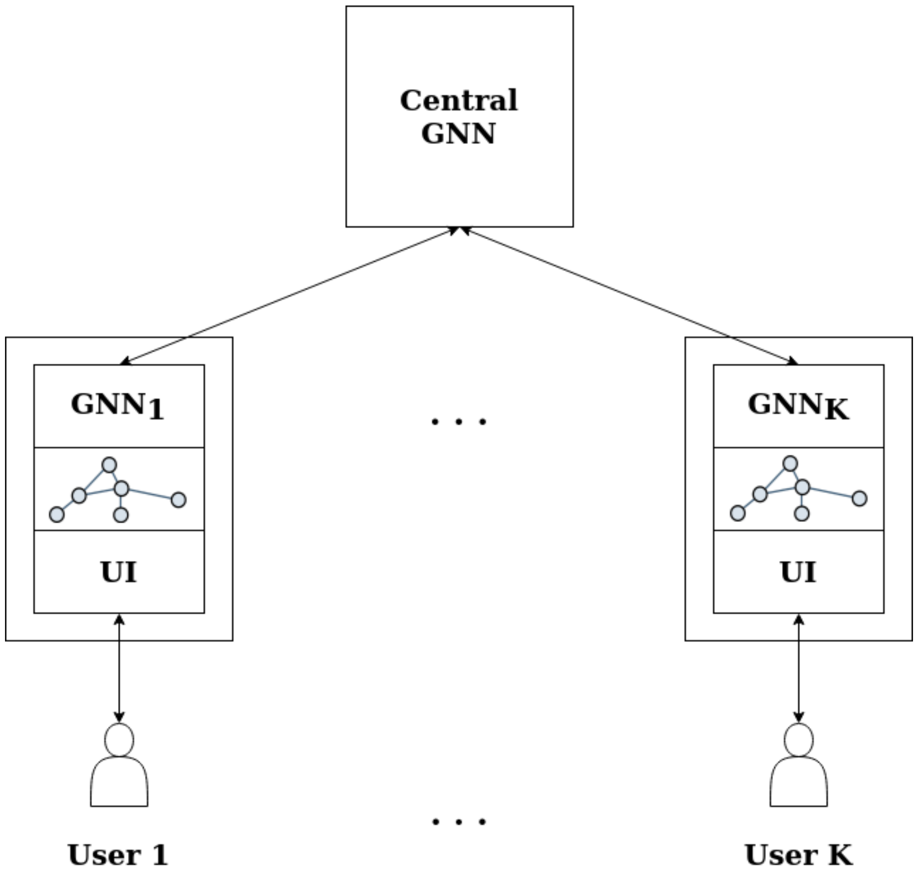
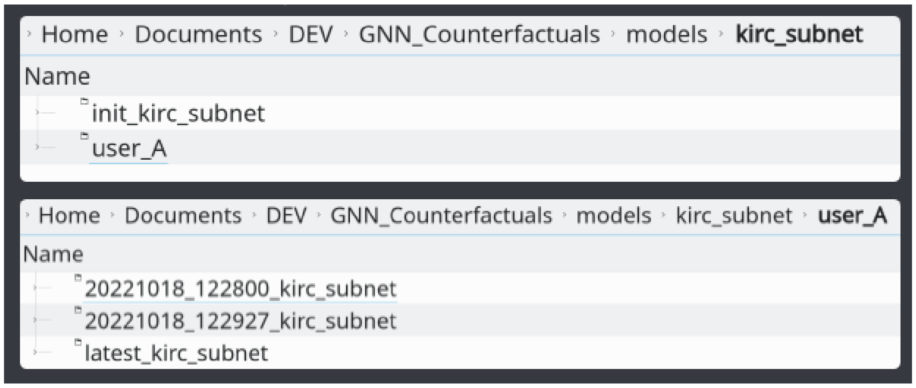


Figure 5: Federated Human-in-the-loop Counterfactual xAI Platform. Each user has a local dataset, and local GNN and interacts with the same UI. The central GNN is created by the transferred components of the local GNNs, which can be weights and/or embeddings.



Home › Documents › DEV › GNN_Counterfactuals › models › **kirc_subnet**

Name

init_kirc_subnet

user_A

Home › Documents › DEV › GNN_Counterfactuals › models › kirc_subnet › **user_A**

Name

20221018_122800_kirc_subnet

20221018_122927_kirc_subnet

latest_kirc_subnet

Figure 6: The hierarchy of folders containing the models for each user before and after each retrain. The date and time of the end of retrain are in Universal Time Coordinated (UTC) (so that we eliminate any potential problems with different time zones) and are used in the name of the folder to help sort.