# Federated Learning on Molecular Property Prediction using GCNs and Transformers

**Yash Jakhotiya, Neo Neng Kai Nigel, Zach Minot, You Liang Tan**

School of Computer Science
Georgia Institute of Technology
North Ave NW, Atlanta, GA 30332
{ yashjakhotiya, nkai3, zjminot, ytan320 }@gatech.edu

## Abstract

Molecular chemistry has recently been greatly benefited from advances in machine learning (ML). However, commercial usage of ML in chemistry is constrained by data-sharing restrictions in place at most organizations. In this paper, we explore the use of federated learning for molecular property prediction. More precisely, we explore the 'accuracy to the number of clients' and 'accuracy to communication rounds' trade-offs of two state of the art ML models for molecular graphs - namely graph convolutional neural networks (GCNs) and graph transformers. In the process, we also contribute a novel data splitting method based on molecular fingerprints and a lightweight federated learning (FL) setup for these two models. Our experiments and results show that, despite the larger size of graph transformers, they have similar performance trade-offs when compared to GCNs. The dataset split implementation, FL implementation and our analysis code can be found at `https://github.com/yashjakhotiya/Federated-Learning-on-Molecular-Property-Prediction`.

## 1 Introduction

Molecular property prediction has benefited from open scientific data for the past few decades, with many open chemical datasets culminating on past work. Producing such data is expensive, with an expected cost of $30 million to develop and analyse a drug DiMasi et al. [2016]. Thus, biotechnology firms leverage on their own proprietary datasets for data mining. However, smaller biotech startups may not have large datasets, and one possibility to create a larger dataset is to collaborate with other startups too. Yet these collaborators are future competitors after the collaboration, thus sharing datasets openly will lead to a loss of competitive advantage.

To learn on these fragmented datasets without sharing data, FL can be used in this problem to train a model across multiple isolated datasets. However, these biotech firms may specialise in different molecules, resulting in heterogeneous, non independent and identically distributed (i.i.d.) data; running FL in this context introduces accuracy tradeoffs due to the possibility of inconsistent data across each node.

We note that many FL for molecular property prediction models are based on graph convolutional networks (GCNs), as it is the traditional model used for molecular property prediction. However, transformers have also been shown to be promising for this task, with transformer models topping the leaderboard at Stanford's OGB-LSC Hu et al. [2021] competition two years in a row. Is it then possible to use transformers in this FL problem?

Preprint. Under review.

Hence, in this paper, we aim to analyze if we can achieve similar or better accuracy, in terms of AUC-ROC (Area under Receiver Operator Characteristic curve), with regards to the number of clients and communication rounds when using state-of-the-art graph transformers instead of GCNs for non-i.i.d. federated learning for chemistry data.

## 2 Related Work

Neural networks on graphs have been around for some time now, primarily introduced by Gori et al. [2005] and Scarselli et al. [2009]. More recently, Kipf and Welling [2016] has been one of the most cited approaches using a form of convolutional network for graph data. Other works include Hamilton et al. [2017] and Ying et al. [2018]. Velickovic et al. [2017] use attention networks on graphs.

Transformer models have been shown to be state-of-the-art for molecular property prediction. The top-3 models from the Hu et al. [2021]'s PCQM4M-v2 NeurIPS '22 challenge rely on transformer models as their architecture Masters et al. [2022] Wang et al. [2022] Darabi et al. [2022], modifying it to encode graph properties of molecules.

Transformer models have also been used in FL problems outside the chemical domain. Vision transformers have been created for FL and shown to generalize better to non-i.i.d. distributions and converge more quickly with less communication rounds needed Qu et al. [2021].

This is our motivation to use a transformer architecture to the FL molecular property prediction problem and evaluate it's 'accuracy to communication rounds' and 'accuracy to number of clients' trade-offs when compared to graph convolutional networks. In this era of ubiquitous computing, numerous federated learning frameworks have emerged over the years, which include Tensorflow-Federated Google, IBM Federated Ludwig et al. [2020], FedML He et al. [2020]. It is attainable to demonstrate transformer training for chemistry with one of the existing federated learning frameworks.

Machine learning for chemistry has been growing over the years, with FL in chemistry being developed very recently. SpreadGNN He et al. [2022] was introduced as a Federated GCN model for use in chemical contexts, but introduces heterogeneity mainly through dataset sizes and label skew, which is not relevant in a chemical context. FedChem Zhu et al. [2022] is also another model related to our problem, and introduces heterogeneity in a more meaningful way using scaffold splits that rely on the structural properties of molecules. MELLODDY Oldenhof et al. [2022] is a platform jointly organised by industry and academic labs to realise the potential for FL in molecular property prediction, but does not allow for changes to the model architecture used. The data for MELLODDY is not released, and its implementation is too bloated for our problem.

## 3 Data

We intend to use a HIV drug dataset from the MoleculeNet benchmark Wu et al. [2018], which contains 41127 molecules represented as SMILES stringsWeininger [1988] and a binary target of whether the molecule can inhibit HIV replication. This is relevant to our problem as there are a number of startups (ViiV Healthcare, Vir Biotechnology, Sangamo Therapeutics etc.) that are actively involved in this area of research. However, this HIV dataset is designed without FL in mind; that is, it is used as a compilation of different resources to be analysed as a whole. Thus, it has to be artificially split into different segments to represent the dataset of each client for use in our FL problem. Additionally, client datasets should be different enough from each other to mimic the setting where each biotech firm has their own speciality that is reflected in their proprietary dataset.

FedChem's method to create dataset splits uses scaffold splitting to group similar molecules together. Molecules are reduced to their scaffold, or backbone structure, which is usually a carbon ring or chain. A Dirichlet distribution is used to allocate molecules based on their scaffolds to the different client datasets while ensuring that a minimum dataset size is satisfied for each client. FedChem introduces heterogeneity by modifying the alpha value of the Dirichlet distribution, with a lower $\alpha$ leading to a more heterogenous distribution. However, scaffold splits may miss many important information that are excluded when computing the scaffold, such as functional groups. Additionally, scaffolds are difficult to determine when molecules have multiple carbon rings or have more complex structure.

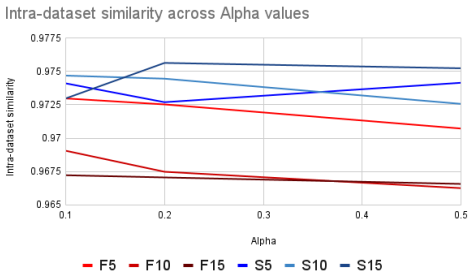


Figure 1: Intra-dataset similarity values using Fingerprint splits (F5, F10, F15) vs Scaffold splits (S5, S10, S15). Numbers in label correspond to number of clients.

Figure 2: tSNE plots for Fingerprint (left) vs Scaffold (right) splits, with each colour corresponding to one client.

### 3.1 Novel Fingerprint Dataset Splits

In this work, we introduce a dataset split based on molecular fingerprints. Molecular fingerprints can be though of as binary embeddings of molecules based on whether certain substructures are present in it. Thus, molecular fingerprints are generalisations of scaffolds, and with a large enough vector space, can include the same structure as scaffolds too. We used ECFP4 fingerprints Rogers and Hahn [2010] in our method with 1024 bits, as it is the standard tool used in the cheminfomatics field.

With these embeddings, Latent Dirichlet Allocation (LDA) can be used to group molecules together. Using LDA also allows the tuning of heterogeneity by adjusting the $\alpha$ as well. To ensure that client datasets also have a minimum size, a greedy approach is used. With the existing LDA allocation, client datasets are sorted in ascending order. For each client $C$, if it does not have enough molecules ($|C| < k$), all unassigned molecules are sorted according to the probability it should be in $C$ and the top-$k$ molecules assigned to $C$. Empirically, this method has allowed us to generate artificial dataset splits without running into any errors or impossible to fulfill scenarios. Notably, we do not assign train/test partitions before running this splitting algorithm, as this should be defined within each client dataset.

We evaluate our fingerprint split method against scaffold splits to see if there is an improvement in terms of molecules being more appropriately clustered together. We report the similarity of molecules within a client dataset, with the similarity of molecules based on Tanimoto Similarity Bajusz et al. [2015] (akin to Jaccard index). This is calculated as the one minus the mean across all clients of the standard deviation of Tanimoto similarity in a dataset. As can be seen in Figure 1, client datasets are decreasingly homogenous as the alpha value is increased using our fingerprint split, but that is not the case when using scaffold splits. Additionally, using a t-SNE (t-distributed stochastic neighbor embedding) visualisation tool in Figure 2, it can be seen that points with the same colour are more clustered together in the fingerprint split (left) compared to scaffold split (right) where points of the same colour are more diffuse and spread out. Lastly, the silhouette coefficient of the clustering used by the fingerprint split is 0.018 for fingerprint splits compared to -0.007 for scaffold splits, where a negative value indicates that molecules are assigned to the wrong cluster.

Hence, we prepared the dataset splits using our fingerprint split for each client dataset, returning indices of the original dataset that belong to each client. We use the OGB dataset as our reference as it has already generated PyTorch-Geometric graphs for use. The train/test splits were generated individually for each client in a 90/10 proportion and loaded into a PyTorch-Geometric Dataloader.

## 4 Models

To compare the federated learning trade-offs of graph convolutional neural networks (GCNs) and transformers on graph data, we rely on the Hu et al. [2021] baseline as our GCN implementation which ultimately uses Kipf and Welling [2016]'s proposed approach. We also rely on the Graphormer Ying et al. [2021] Shi et al. [2022] as our transformer implementation.

### 4.1 Graph Convolutional Network

The GCN used in our analysis has the following node propagation rule,

$$H^{(\ell+1)} = f(H^{(\ell)}, A) = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(\ell)}W^{(\ell)})$$

where, $\ell$ denotes the layer number, $H^{(0)}$ is the input node feature matrix, $A$ is the adjacency matrix, $\hat{A} = A + I$ to introduce self loops to count a node's current features in its next layer features and $D$ is the diagonal node degree matrix to introduce symmetric normalization, and $W^{(\ell)}$ is the weight matrix at layer $\ell$.

This makes a node's features at layer $\ell$ to be a normalized weighted average of itself and its neighbors passed into a non-linearity function $\sigma(\bullet)$.

### 4.2 Graphormer

We modify the Transformer architecture Vaswani et al. [2017] for graphs by modifying the Transformer's Query-Key product matrix A as,

$$A_{ij} = \frac{(h_i W_Q)(h_j W_K)^T}{\sqrt{d}} + b_{\phi(v_i,v_j)} + c_{ij},$$

where $A_{ij}$ is the $(i,j)$-element of the matrix A, $b_{\phi(v_i,v_j)}$ denotes the spatial encoding and $c_{ij}$ denotes edge encoding as defined in Ying et al. [2021].

We tuned hyperparameters of both of these models (the number of convolution layers, the number of attention heads and layers, learning rates, etc) separately with only one client configured in our federated learning setup 5, and kept these model hyperparameters constant when evaluating tradeoffs with the federated learning hyperparameters like number of clients and communication rounds. Adam Kingma and Ba [2014] is our optimizer of choice for training these models.

## 5 Federated Learning Setup

After appointing GCN as our model baseline to evaluate Graphormer, we intend to run training rounds with both models in a federated learning setting. We have surveyed the existing frameworks for federated learning, with strong contenders such as Tensorflow-Federated Google, FATE Webank-AI, and FLOWER Beutel et al. [2020]. In this project, FedML is selected here as its usage is evidently demonstrated in FedGraphNN He et al. [2021] and FedChem Zhu et al. [2022] benchmarking. Note that our result is FL-framework invariant, and this can be replicated with other federated learning frameworks.

FedML supports different types of federated learning configurations: Cross-device, Cross-silo, Distributed training, and Simulation. In our experimentation, we are interested to evaluate both GCN and Graphormer models in simulation mode. This will provide us with first-hand information on training accuracy vs communication rounds in multiple client processes. With FedML, we can easily switch the same model setup to other system configurations for distributed training on remote clients.

Multiple steps are required for users to deploy custom models with personalized configurations with FedML. First, the ML framework agnostic nature of FedML makes it seamless for us to deploy the existing GCN and Graphormer pytorch models to FedML. Subsequently, user is required to implement custom "get" and "set" weights policies and actions with FedML "ClientTrainer" and "ServerAggregator" APIs. Internally, FedML will handle the internal communications between clients and server, and execute the default FedAvg algorithm McMahan et al. [2017]. In our custom implementation setup, we also log the change in accuracy on each training and communication round on both the server and clients, which is helpful for us to derive insights for our evaluation. Lastly, we create our custom "DataLoader" to load our novel fingerprint dataset splits to each client trainers. The detailed algorithm of FedML's training process is shown 1.

Our federated learning setup consists of multiple configuration knobs to adjust during training. These training configurations can be easily modified with a parameter server or a simple YAML configuration file. In our case, we are interested to change the configuration namely: the selection of

**Algorithm 1** FederatedAveraging (FedAvg) Algorithm: K clients; B as batch size; E as local client epoch; $\eta$ is the learning rate

**function** SERVEREXECUTE ▷ Run on server aggregator
  $S_t \leftarrow$ random set of clients selected for training
  **for** each client k $k \in S_t$ in parallel **do**
    $w_{t+1}^k \leftarrow ClientUpdate(k, w_t)$
  **end for**
  $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$ ▷ FedAvg Operation
**end function**

**function** CLIENTUPDATE($k, w$) ▷ Run on Client Trainer $k$
  $B \leftarrow$ (Splits data into batches of size B) ▷ With our Fingerprint Data Split
  **for** each local epoch $i$ from 1 to $E$ **do**
    **for** batch $b \in B$ **do**
      $w \leftarrow w - \eta \nabla l(w; b)$ ▷ GCN or Graphormer Training
    **end for**
  **end for**
  return $w$ to server
**end function**

GCN or Graphormer, the number of clients, communication rounds, and the learning rate. Switching each configuration will provide us with insightful assessments during each training round.

## 6 Results and Analysis

In this section, we intend to showcase the experiment results which to examine our falsifiable hypothesis: evaluate the accuracy versus the number of clients and communication rounds trade-offs for GCN and Graphormer in a Federated learning setting. In addition, after rounds of hyper parameter tuning, we use a learning rate of $1e-5$ for graphormer since it yields the best accuracy

In Figure 3, we trained the GCN (OGB baseline) model and Graphormer on multiple rounds with a varying number of clients (1, 2, 4, 8, 16, clients), and subsequently observed their Maximum AUC-ROC. One consistent observation is the accuracy of the aggregate model and clients tends to decrease when the number of clients increases. This behavior is consistent between both GCN and Graphormer models.

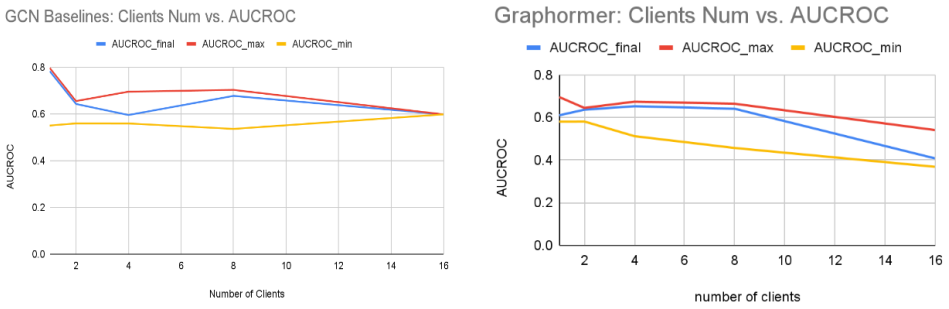


Figure 3: Number of Clients in FL v.s AUC-ROC for GCN and Graphormer models

With respect to communication rounds, in eventual reality the communication rounds did not show much differential between the two types of models. Both models converged within the first few communication rounds, and achieved a similar amount of consistency afterwards. We can see this in the left graph of Figure 4, where both graphs achieve a similar convergence time and consistency of performance afterward.

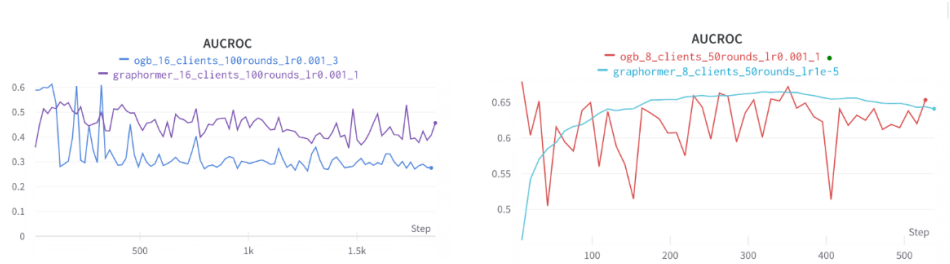


Figure 4: Similar consistency across communication rounds for both GCN and Graphormer. With 16 clients (Left), 8 clients (Right).

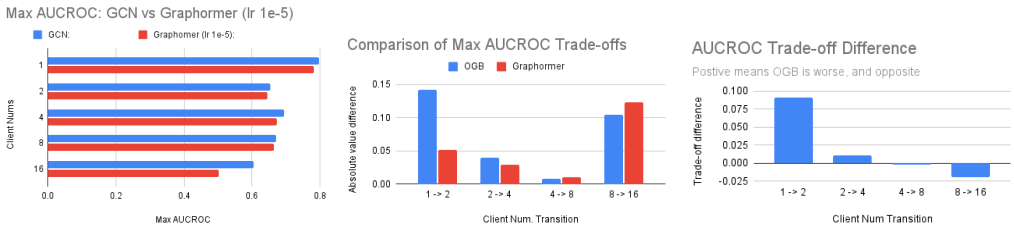


Figure 5: Bar graphs showing from left-to-right the direct comparison, trade-offs, and trade-off difference between the baseline GCN and the Graphormer as we increase the number of clients. Primarily, the right graph shows that the Graphormer achieves similar, and in earlier cases better trade-off results compared to the baseline GCN.

The most important aspect of our analysis are the trade-offs of performance as we increase client numbers (i.e., as we 'federate' more). In the right-most graph of Figure 5, we see that the graph transformer model actually has a lower trade-off in the earlier transitions of amount of clients, while the GCN has a very slight advantage in the later stages. However, in the later stages, we also want to note that the data was becoming sparse between clients, so a differential AUC-ROC of 0.02 is not enough to be substantial. Therefore, from this, we can say that the graph transformer model for this setup has at least a similar or better trade-off of performance when compared to the GCN.

## 7 Technical Challenges and Discussions

### 7.1 Technical Challenges

One of the main technical challenges with this project was the resource constraint. We initially had a much larger scope for this project, but due to the eventual lack of resources and time we needed to cut down on the scope much further. In specifics, we were wanting to test the federation of the models on the PCQM4Mv2 dataset and potentially more, but had to settle for the HIV dataset due to training times. Similarly, more resources and time could also mean we test more configurations of the two models. Unfortunately, this extension is left as future work.

Another issue we came across was poor documentation and inability to reproduce some baselines. Certain tools used, such as FedML, had difficult 'documentation-by-example' systems that perpetuated unclear execution throughout the program. We deemed this as unacceptable for a project intended on being extended by other users, so we needed to take some time and effort understanding what exactly was going on within the our test bench, document the program, and remove extraneous code. This documentation eventually derived the test bench contribution. On the other hand, FedChem, a related work we initially intended to use, was in fact not reproducible despite our best efforts. This led to a medium loss of time and effort designated for these contributions.

### 7.2 Limitations

Our fingerprint splitting method is not truly representative of proprietary datasets belonging to biotech firms. This is because we are modifying an original, centralised dataset, and thus our client datasets are generated to use all molecules even if they do not fit in any client dataset. It is possible that this method works best for a set number of clients where a clear split can occur, but as we needed a comparison of the models across different client sizes, this would introduce a weakness into our methodology.

We assumed within this project that these models were fairly representative of their specific type of neural network (i.e., GCNs and graph transformers). We did so due to these model's specificity toward the dataset tested at the time. However, this may not exactly be the case, and certain design choices have a chance lead to different end conclusions. A divisive factor between models could be complexity, which instead could mean that increasing, for example, embedding size on the GCN could lead to different results. Increasing epochs per communication round could also lead to massive overfitting from both models, and it is not known where this would lead. On the other hand, model aggregation methods other than FedAvg may have an affect on how well a model performs in federated learning settings as well.

Due to our limited time and resources, we decided to limit our experiments to a small number of independent variables, and all other hyperparameters were set to given defaults. This limits our ability to find the best possible performing model for each type, in the form of hyperparameter search or limited guided Network Architecture Search. Therefore, our experiments may not find the best representation of the model for comparison. We also used the same heterogeneous splitting indices for each test run; doing so increases the ability to reproduce our work, but may result in hidden results from using other heterogeneous splits. Finally, we also only train these models with one dataset. There is the potential that these results could be specific to this dataset only, and more effort is needed to test models' ability to perform in a federated learning set up overall.

### 7.3 Future Work

The main projection of future work involve address the limitations discussed in the previous section, including hyperparameter search, different types of models, more datasets to train and test with, and turning more independent variables will allow extensive exhaustion of the trade-off space along federated learning of graph neural networks. Moreover, this leads to the idea of a metric or a set of metrics comparing two models in the form of trade-offs in the federated learning space. What defines the fact that one model may perform in a federated learning setup better than another model?

Another potential study is to deploy the GCN and Graphormer in a non-simulation environment. More external factors will bring in more insightful observations, for instance, potentially higher communication latency of the Graphormer model from remote Clients. Furthermore, as FedML poses some limitations on lack of documentation and large-scale support, it would be beneficial to switch to other federated learning frameworks, such as Tensorflow-federated which supports large-scale deployment with better analytical tools.

## 8 Conclusion

Due to a lack of current research, we explored the trade-off space of federated learning within the molecular chemistry machine learning arena. We chose to compare two models, a graph convolutional network and a graph transformer network, in hopes of understanding more about the models' interaction with heterogeneous datasets. From this, we needed to develop a better way to introduce heterogeneity into a dataset and run the training/testing simulations for the models. This lead to the construction of a novel chemical fingerprint split for introducing heterogeneity into molecular datasets and a lightweight molecular dataset federated learning setup. Through testing, we found that we can achieve similar performance trade-offs with number of clients and communication rounds when using state-of-the-art graph transformers instead of graph convolutional neural networks for non-i.i.d. federated learning for chemistry.

## Acknowledgments and Disclosure of Funding

We would like to thank Alexey Tumanov, Glenn Matlin, and Monish Ramadoss, the teaching staff of the SysML course at Georgia Institute of Technology, for their support and guidance throughout our work in this contribution.

## References

Dávid Bajusz, Anita Rácz, and Károly Héberger. Why is Tanimoto index an appropriate choice for fingerprint-based similarity calculations? *Journal of Cheminformatics*, 7(1):20, December 2015. ISSN 1758-2946. doi: 10.1186/s13321-015-0069-3.

Daniel J Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, Pedro PB de Gusmão, and Nicholas D Lane. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*, 2020.

Sajad Darabi, Shayan Fazeli, Jiwei Liu, Alexandre Milesi, Pawel Morkisz, Jean-François Puget, and Gilberto Titericz. Heterogenous ensemble of models for molecular property prediction, 2022. URL `https://arxiv.org/abs/2211.11035`.

Joseph A. DiMasi, Henry G. Grabowski, and Ronald W. Hansen. Innovation in the pharmaceutical industry: New estimates of R&D costs. *Journal of Health Economics*, 47:20–33, May 2016. ISSN 01676296. doi: 10.1016/j.jhealeco.2016.01.012.

Google. Tensorboard-federated. URL `https://www.tensorflow.org/federated`.

Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, 2:729–734 vol. 2, 2005.

William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NIPS*, 2017.

Chaoyang He, Songze Li, Jinhyun So, Xiao Zeng, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, et al. Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518*, 2020.

Chaoyang He, Keshav Balasubramanian, Emir Ceyani, Carl Yang, Han Xie, Lichao Sun, Lifang He, Liangwei Yang, Philip S Yu, Yu Rong, et al. Fedgraphnn: A federated learning system and benchmark for graph neural networks. *arXiv preprint arXiv:2104.07145*, 2021.

Chaoyang He, Emir Ceyani, Keshav Balasubramanian, Murali Annavaram, and Salman Avestimehr. SpreadGNN: Decentralized multi-task federated learning for graph neural networks on molecular data. 2022.

Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. Ogb-lsc: A large-scale challenge for machine learning on graphs, 2021. URL `https://arxiv.org/abs/2103.09430`.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

Thomas Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ArXiv*, abs/1609.02907, 2016.

Heiko Ludwig, Nathalie Baracaldo, Gegi Thomas, Yi Zhou, Ali Anwar, Shashank Rajamoni, Yuya Ong, Jayaram Radhakrishnan, Ashish Verma, Mathieu Sinn, et al. Ibm federated learning: an enterprise framework white paper v0. 1. *arXiv preprint arXiv:2007.10987*, 2020.

Dominic Masters, Josef Dean, Kerstin Klaser, Zhiyi Li, Sam Maddrell-Mander, Adam Sanders, Hatem Helal, Deniz Beker, Ladislav Rampášek, and Dominique Beaini. Gps++: An optimised hybrid mpnn/transformer for molecular property prediction, 2022. URL `https://arxiv.org/abs/2212.02229`.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

Martijn Oldenhof, Gergely Ács, Balázs Pejó, Ansgar Schuffenhauer, Nicholas Holway, Noé Sturm, Arne Dieckmann, Oliver Fortmeier, Eric Boniface, Clément Mayer, Arnaud Gohier, Peter Schmidtke, Ritsuya Niwayama, Dieter Kopecky, Lewis Mervin, Prakash Chandra Rathi, Lukas Friedrich, András Formanek, Peter Antal, Jordon Rahaman, Adam Zalewski, Ezron Oluoch, Manuel Stößel, Michal Vančo, David Endico, Fabien Gelus, Thaïs de Boisfossé, Adrien Darbier, Ashley Nicollet, Matthieu Blottière, Maria Telenczuk, Van Tien Nguyen, Thibaud Martinez, Camille Boillet, Kelvin Moutet, Alexandre Picosson, Aurélien Gasser, Inal Djafar, Ádám Arany, Jaak Simm, Yves Moreau, Ola Engkvist, Hugo Ceulemans, Camille Marini, and Mathieu Galtier. Industry-Scale Orchestrated Federated Learning for Drug Discovery, October 2022.

Liangqiong Qu, Yuyin Zhou, Paul Pu Liang, Yingda Xia, Feifei Wang, Li Fei-Fei, Ehsan Adeli, and Daniel L. Rubin. Rethinking architecture design for tackling data heterogeneity in federated learning. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10051–10061, 2021.

David Rogers and Mathew Hahn. Extended-Connectivity Fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, May 2010. ISSN 1549-9596, 1549-960X. doi: 10.1021/ci100050t.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20:61–80, 2009.

Yu Shi, Shuxin Zheng, Guolin Ke, Yifei Shen, Jiacheng You, Jiyan He, Shengjie Luo, Chang Liu, Di He, and Tie-Yan Liu. Benchmarking graphormer on large-scale molecular modeling datasets. *arXiv preprint arXiv:2203.04810*, 2022. URL `https://arxiv.org/abs/2203.04810`.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *ArXiv*, abs/1706.03762, 2017.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio', and Yoshua Bengio. Graph attention networks. *ArXiv*, abs/1710.10903, 2017.

Yusong Wang, Shaoning Li, Tong Wang, Zun Wang, Xinheng He, Bin Shao, and Tie-Yan Liu. An ensemble of visnet, transformer-m, and pretraining models for molecular property prediction in ogb large-scale challenge @ neurips 2022, 2022. URL `https://arxiv.org/abs/2211.12791`.

Webank-AI. Fate (federated ai technology enabler). URL `https://github.com/FederatedAI/FATE`.

David Weininger. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Modeling*, 28(1):31–36, February 1988. ISSN 1549-9596. doi: 10.1021/ci00057a005.

Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay Pande. MoleculeNet: A Benchmark for Molecular Machine Learning, October 2018.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. URL `https://openreview.net/forum?id=OeWooOxFwDa`.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.

Wei Zhu, Jiebo Luo, and Andrew D. White. Federated learning of molecular properties with graph neural networks in a heterogeneous setting. *Patterns*, 3(6):100521, June 2022. ISSN 26663899. doi: 10.1016/j.patter.2022.100521.