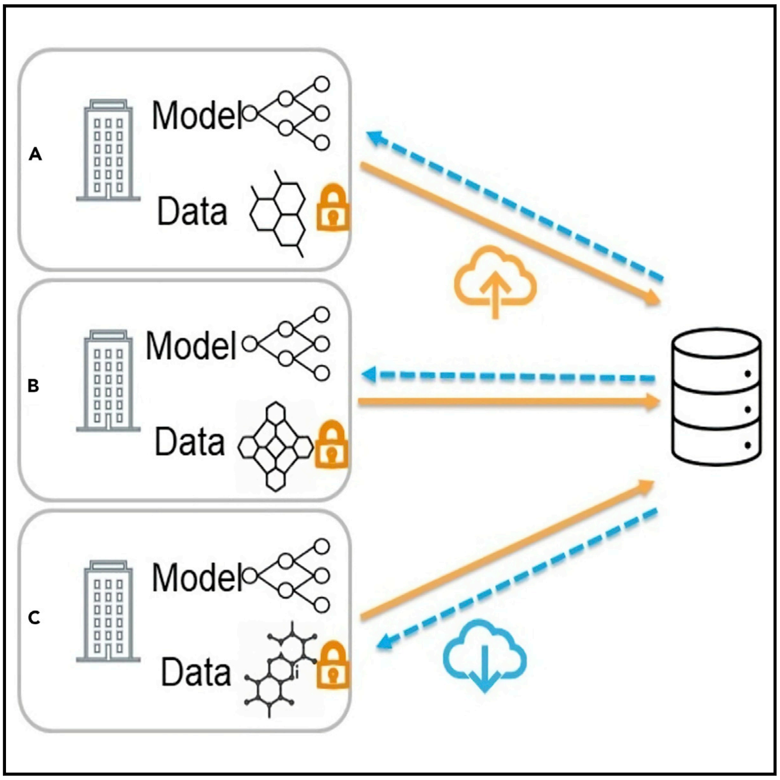
Article

Patterns

Federated learning of molecular properties with graph neural networks in a heterogeneous setting

Graphical abstract



Highlights

- FedChem employs scaffold splitting and LDA for heterogeneous settings
- We propose FLIT(+) algorithms to alleviate the heterogeneity problem
- We conduct experiments to benchmark the proposed and existing methods on FedChem

Authors

Wei Zhu, Jiebo Luo, Andrew D. White

Correspondence

andrew.white@rochester.edu

In brief

This work presented a federated heterogeneous molecular learning benchmark based on MoleculeNet as FedChem. Several federated-learning methods are benchmarked on the proposed suites and show remarkable performance degradation. The authors then demonstrate federated learning by instance reweighting (FLIT) to alleviate the heterogeneity problem. Extensive experiments validate the effectiveness of the proposed methods.

Zhu et al., 2022, Patterns *3*, 100521
June 10, 2022 © 2022 The Authors.
https://doi.org/10.1016/j.patter.2022.100521

CellPress

Patterns

CellPress
OPEN ACCESS

Article

Federated learning of molecular properties with graph neural networks in a heterogeneous setting

Wei Zhu,[1] Jiebo Luo,[1] and Andrew D. White[2,3,*]
[1]Department of Computer Science, University of Rochester, Rochester, NY, USA
[2]Department of Chemical Engineering, University of Rochester, Rochester, NY, USA
[3]Lead contact
*Correspondence: andrew.white@rochester.edu
https://doi.org/10.1016/j.patter.2022.100521

THE BIGGER PICTURE Generating datasets with thousands of molecules for machine learning in chemistry is cost prohibitive due to the high material and/or computational costs. Additionally, chemical data's intrinsic value makes institutions reluctant to contribute to a centralized dataset. Recent studies suggest that deep learning has the potential to accelerate molecule discovery, but there are few large datasets for chemistry. Instead, individual institutions gather their data privately, which leads to under-trained models with poor generalization performance. Even worse, the local models can be biased because institutions often focus on certain regions of chemical space important for their interests and expertise. We propose a federated-learning method with graph neural networks that can treat this heterogeneity and enable accurate federated learning on molecular-property prediction. We propose a heterogeneous federated-learning benchmark and show that our method is state of the art.

1 2 **3** 4 5 **Development/Pre-production:** Data science output has been rolled out/validated across multiple domains/problem

SUMMARY

Chemistry research has both high material and computational costs to conduct experiments. Intuitions are interested in differing classes of molecules, creating heterogeneous data that cannot be easily joined by conventional methods. This work introduces federated heterogeneous molecular learning. Federated learning allows end users to build a global model collaboratively while keeping their training data isolated. We first simulate a heterogeneous federated-learning benchmark (FedChem) by jointly performing scaffold splitting and latent Dirichlet allocation on existing datasets. Our results on FedChem show that significant learning challenges arise when working with heterogeneous molecules across clients. We then propose a method to alleviate the problem: Federated Learning by Instance reweighTing (FLIT(+)). FLIT(+) can align local training across clients. Experiments conducted on FedChem validate the advantages of this method. This work should enable a new type of collaboration for improving artificial intelligence (AI) in chemistry that mitigates concerns about sharing valuable chemical data.

INTRODUCTION

There is an increasing trend to apply machine learning for molecule-property prediction to avoid the expense of experiments or reduce the tremendous computational costs required for accurate quantum-chemical calculations. A large focus has been on applying graph neural networks to predicting molecular properties.[1–6] These works assume a central server that has access to all data. However, such a centralized-learning scenario may not represent how institutions share chemical data. Due to intellectual-property concerns and the intrinsic value of chemical data, it can be difficult for academic labs, national labs, and private institutions to share their molecule datasets.

We propose federated learning to obtain a generalized global model without access to private molecular data. For federated learning, local models are trained with their data on the client

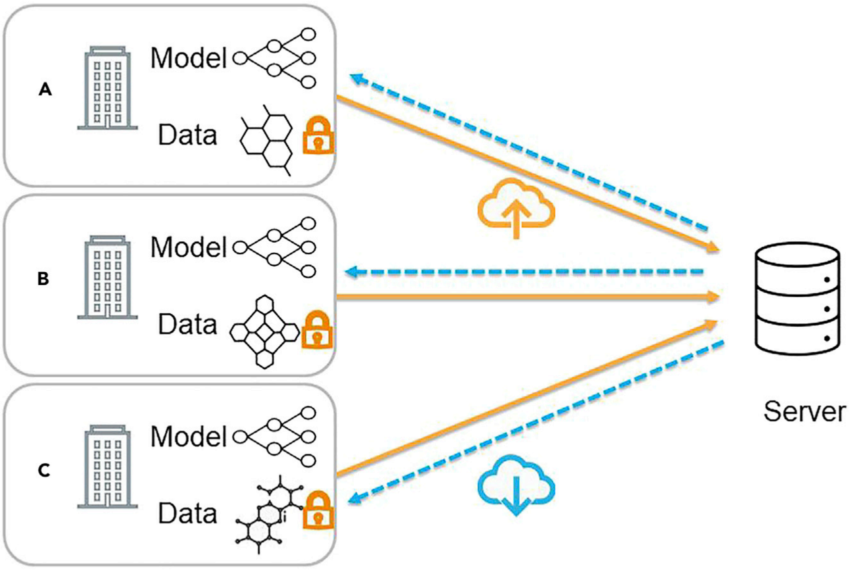


Figure 1. Heterogeneous federated molecular learning where three institutions focus on different types of molecules
The server has no access to training data.

side and then are aggregated for a global one on the server side without seeing the data. One of the main concerns for federated molecular-property prediction is the heterogeneously distributed client data since institutions focus on specific categories of molecules for their research interests. For example, institutions may wish to collaborate to construct an accurate model of pharmacokinetic clearance time of small molecules like shown in Figure 1. Each institution studies specific drug-like molecules and their variants for their therapeutic targets. Each institution cannot share molecules, but it is beneficial to have a model for clearance time. Trained local models will heavily deviate from each other in this example, and it is thus sub-optimal to directly apply vanilla federated-learning methods, e.g., Federated Average (FedAvg), to aggregate the heterogeneous local models.[7] Although several works are proposed to handle the heterogeneity problem,[8,9] a broader problem is the lack of heterogeneous federated molecular learning benchmarks to judge these methods for chemical data.[5]

This paper first proposes a federated heterogeneous molecular learning benchmark, FedChem. FedChem simulates the heterogeneous settings based on scaffold splitting[10] and latent Dirichlet allocation (LDA).[8] We first adopt scaffold splitting to split the molecules based on their two-dimensional structure, and molecules with similar structures are grouped accordingly.[5] Then, a heterogeneous setting is obtained by applying LDA on the scaffold subgroups, where LDA is a commonly used technique to simulate heterogeneous settings in conventional federated classification tasks.[8,11] We benchmark existing federated-learning methods on the proposed heterogeneous suite FedChem and observer a remarkable performance degradation for the commonly used method FedAvg.[7] We then propose Federated Learning with Instance reweighTing (FLIT) to alleviate the heterogeneity problem by adapting focal loss for federated learning. The motivation of FLIT is that local models will be trained to overfit their data, which, however, do not share the same distribution as the global one. That is, the prediction of local models would be over-confident for certain types of molecules while with high uncertainty for others. FLIT can align client training by adding weights to the uncertain cases by utilizing the local and received global models. As a result, the locally trained models will be more consistent with each other, and the federated-learning performance can be eventually improved. We measure the uncertainty for training samples by the loss values and the prediction consistency among neighbored samples and develop two methods as FLIT and FLIT+ (FLIT(+) being the abbreviation for both). Our experiments on the proposed benchmark FedChem validate the advantages of FLIT(+) over existing federated-learning methods.

Our main contributions are summarized as follows:

1. We propose a federated heterogeneous molecular learning benchmark based on MoleculeNet,[5] termed FedChem. FedChem employs scaffold splitting and LDA to simulate the heterogeneous settings.
2. We propose FLIT(+) algorithms to alleviate the heterogeneity problem. FLIT(+) can align the client training by putting more weights on uncertain samples.
3. We conduct experiments to benchmark the proposed and existing federated-learning methods on FedChem. Comprehensive experiments validate the effectiveness of the proposed methods.

RELATED WORK

Federated learning

Federated learning was proposed by McMahan et al.[7] and has been applied in a wide range of fields including healthcare,[12] biometrics,[13] and natural images and videos.[14,15] As a popular method, FedAvg element wisely aggregates the parameters of local models to obtain a global one.[7] However, recent studies indicate that FedAvg may not handle the heterogeneity problem properly.[8,16] There are two categories of methods developed to alleviate the problem: improvements for server-side aggregation[8,17–24] and client-side regularization methods.[25–30]

Client-side methods can use the local training data and attract increasing attention. Our method also follows this line of research. Federated proximal (FedProx) regularizes the local learning with a proximal term to encourage the updated local model not to deviate significantly from the global model.[29] A similar idea is adopted in personalized federated learning.[26] SCAFFOLD adopts additional control variates to alleviate the gradient dissimilarity across different communication round.[27] Federated model distillation transfers the soft predictions of a shared dataset to reduce the communication cost and regularizes the local training with distillation loss.[18] Federated meta-learning incorporates model agnostic meta-learning (MAML) for local training to improve the generalization ability of local models.[25,31] Robust federated-learning has been studied by several works.[16,20,28] Reference Architecture for Federated Learning Systems (FLRA) adversarially conducts training on clients to make the model robust to affine distribution shifts.[28] Most of the client-side federated-learning methods add a regularization term to restrict the local training process so that the optimized local model would not significantly deviate from the global one.[18,27,29] Consequently, the local models will be more consistent with each other, and the consistency could benefit the server-side aggregation. However, the regularization may also hinder the local optimization and lead to sub-optimal results for local training. Our method does not impose constraints on the local training, and, alternatively, we instance wisely reweight the local training samples to align the local data distribution to the global one inspired by recent work.[32–35]

Heterogeneous federated learning is related to federated domain adaptation (FDA).[36–38] FDA aims to improve the performance for specific target training domains, while general heterogeneous federated learning aims to improve the performance for all training data.

There are several works focusing on federated graph neural networks[9,39–44] and federated molecular-property prediction.[45,46] GraphFL applies MAML to improve the robustness of training.[43] The method in Xie et al.[9] alleviates the heterogeneity problem by group wisely aggregating clients' models. However, existing work does not study federated molecular learning in heterogeneous settings where the clients' datasets are non-independent and identically distributed (IID) in molecular structure and properties.

Deep molecular-property prediction

Graph neural network is commonly adopted for molecular learning.[3–5,47] Message-passing neural network (MPNN) iteratively propagates the vertex features through message-passing layers.[1] SchNet adopts continuous-filter convolution to achieve E(3)-invariant molecular learning.[4] DimeNet and DimeNet++ include directional information when training graph neural network for better performance.[2,48] Other works apply an SO(3) equivariance message-passing layer to predict the properties of molecular data.[49,50] A new structure is proposed by EQNN to efficiently achieve an E(n) equivalent.[3] We employ MPNN[1] and SchNetThe[4] for client-side training in the proposed federated molecular learning framework FedChem, and our framework can seamlessly integrate other models for client-side training, e.g., other graph network networks,[2,3,49] sequence models,[51,52] etc.

RESULTS AND DISCUSSION

Notations and settings

We first briefly describe federated heterogeneous molecular learning (FedChem). We assume that there are L institutions that work on the same tasks with roughly different groups of molecules. That is, the data are distributed heterogeneously across institutions. Each institution develops a neural network for molecular-property prediction.[1,4,5] The neural network trained on their data may suffer from poor generalization ability, and they thus intend to collaborate for a global model without sharing their data with the central server and other participants.

We propose to apply federated learning to obtain a global model for all participants without access to clients' data. Formally, we denote the overall dataset as $X = \{X^l\}_{l=1}^{L}$, where $X^l = (G^l, y^l) = \{(g_i^l, y_i^l)\}_{i=1}^{N_l}$ is the local dataset owned by the l-th institution/client that may not share the same distribution as the overall data. $g_i^l = (v_i^l, e_i^l)$ is the i-th molecule in graph representation with vertex as v_i^l, edge as e_i^l, and ground-truth label as y_i^l. Ground truth could be either concrete values for regression tasks or categorical values for classification tasks. We utilize a local graph neural network F^l to handle the data for the l-th client, and it is implemented with MPNN[1] or SchNet.[4] To enable the clients to collaborate with each other, we have a central sever that receives and aggregates the uploaded local networks for a global one $F^g = FedAgg(\{F^l\}_{l=1}^{L})$, where F^g is the global model, and $FedAgg(\cdot)$ is the aggregation function, e.g., FedAvg,[7] federated optimzation,[21] federated distillation,[53] "ensemble distillation and model fusion (FedDF),[19] federated matched averaging,[8] etc. Note that the central server contains no training data and also cannot access any local data.

FedChem simulates heterogeneous federated molecular learning with existing datasets, e.g., MoleculeNet.[5] Our method relies on scaffold splitting to group molecules based on their structure (graph). Molecules with similar structures are grouped into a scaffold subset. Scaffold splitting first groups the molecules into scaffold groups and then assign samples from each group to clients according to the unbalanced partition method LDA.[10] We detail the approach to generating heterogeneous settings in the experimental section.

Our method of generating a heterogeneous dataset is different from typical existing methods, which simulate label-distribution shift.[40] For example, Karimireddy et al.[27] and Wang et al.[8] split samples based on class to each client, which makes the label distributions of local datasets on clients inconsistent with the global label distribution. In reality, institutions focus on molecules with similar structures via processes like lead optimization or hit finding.[54] Thus, we typically see structurally heterogeneous molecules on the client side (domain shift), while the label distributions among local clients can be similar. To simulate the structural heterogeneity with existing centralized datasets, we adopt scaffold splitting and do not rely on the ground-truth label. Intuitively, samples from different scaffold subsets are analogous to the samples from different domains for general machine-learning tasks, and molecules (images) within a scaffold subset (domain) share similar structures (style) but show different chemical properties (ground-truth label). We illustrate the scaffold splitting to help readers better understand our heterogeneity simulation method. Moreover, it is non-trivial to generalize existing heterogeneous federated dataset simulation methods to regression and multi-label tasks, while our method can be easily adapted to any problems. We benchmark several existing federated-learning methods on FedChem and observe that the heterogeneity problem brings significant challenges to federated molecular learning.

Algorithm 1. Federated heterogeneous molecule learning (FedChem with FedAvg)

```
Input: # clients $L$, # local updates $T$, # Comms round $C$.
Output: Global Model $F^g$
  Server initialize a global model $F^g$                ▷ Server init.
  while Communication Round $< C$ do
    Server broadcasts $F^g$ to clients
    $F^l \leftarrow F^g$                                ▷ Client init.
    for $l : 1$ to $L$ in parallel do                   ▷ Client Update
      for $t : 1$ to $K$ do                             ▷ Update $F^l$ for $K$ steps
        Sample a minibatch $\{g^l_i, y^l_i\}^B_{i=1} \sim X^l$
        Update local model $F^l$ by gradient descent
      end for
      Client sends updated model $F^l$ to Server
    end for
    Server gets $F^g \leftarrow \sum_{l=1}^{L} \frac{|X^l|}{|X|} F^l$    ▷ Server Update
  end while
```

Federated learning with FedChem

The basic training pipeline for FedChem is briefly introduced as follows: we first initialize a global model F^g at the server side and then for each federated-learning communication round: (1) the server broadcasts global model F^g to clients, (2) clients conduct training in parallel, and specifically, the l-th client is trained with its own data X^l for an updated model as F^l, and (3) the server collects updated local models from clients and then aggregate these models into a global one as $F^g = FedAgg(\{F^i\}^L_{i=1})$. We iteratively perform steps 1–3 for C communication rounds to obtain the final global model. We adopt FedAvg for server-side aggregation throughout the paper, but FedChem can be easily extended to involve other aggregation methods.[8,21] We summarize the training procedure for federated learning with FedChem in Algorithm 1 by taking FedAvg as the aggregation method. Note that the server may select a subset of clients during each communication round for scalability.

Client-side updates

For completeness, we describe typical training steps to update the graph neural network (GNN) model for client-side training. We adopt MPNN set-to-set (MPNNs2s)[1] and SchNet[4] for molecule-level property prediction in our experiments, and other popular models (such as DimeNet,[2] Gin,[55] Graph Convolutional Network [GCN],[56] etc.) can also be unified in FedChem.

Molecule-level GNN usually contains two phases: a message-passing phase and a readout phase.[1,40] The message-passing phase allows the vertex to propagate and collect information from their neighbors through the graph and is usually composed of two steps as message generation and vertex update. Formally, given the l-th client model F^l with T message-passing layers and a sampled graph G^l (we omit the subscript for the sample, i.e., $G^l = G^l_i$), we define the message-passing function M^l_t on the i-th vertex as[1]

$$m^l_{t+1,i} = M^l_t\left(v^l_{t,i}, \left\{v^l_{t,w}, e^l_{t,iw}\right\}_{w \in N(i)}\right), \qquad \text{(Equation 1)}$$

and the vertex update function U^l_t as

$$v^l_{t+1,i} = U^l_t\left(v^l_{t,i}, m^l_{t+1,i}\right), \qquad \text{(Equation 2)}$$

where $v^l_{t,i}$ denotes the representation of the i-th vertex in the t-th layer of G^l, $e^l_{t,iw}$ denotes the edge between the i-th and w-th vertex, and $N(i)$ denotes the set of neighbors for vertex i in graph G^l. $M^l_t(\bullet)$ generates the message $m^l_{t+1,i}$ by aggregating the feature of $v^l_{t,i}$ and its neighbors and also the edges between them. $U^l_t(\bullet)$ updates the i-th vertex by transforming the original features and the received message $m^l_{t+1,i}$. Different GNN models are implemented with different M^l_t and U^l_t. For example, the message function of GCN is defined as $m^l_{t+1,i} = \sum_w \hat{e}^l_{t,iw} FC(v^l_{t,w})$ and $U^l_t = FC(m^l_{t,i})$,[56] where FC is a linear layer and $\hat{e}$ is the Laplacian-regularized adjacency matrix. SchNet implements the message function M^l_t with a continuous filter layer and U^l_t with a vertex(atom)-wise convolutional module.[4] The message-passing phase could aggregate and transform the vertex features for high-level representations.

After T message-passing layers, we adopt a readout function R^l to aggregate the vertex representations for graph level representation as

$$h^l = R^l\left(v^l_{T,i} \middle| i \in G^l\right). \qquad \text{(Equation 3)}$$

R^l should be permutation invariant and can be implemented with either a simple sum pooling or a learnable neural network. The graph-level representation h^l is further used to obtain an estimation $\hat{y}^l = F^l(G^l)$ for the ground-truth molecular property y^l.

FEDERATED LEARNING BY INSTANCE REWEIGHTING FLIT(+)

According to our experiments on the proposed heterogeneous federated-learning benchmark FedChem, heterogeneity brings significant difficulties to federated molecular learning. This

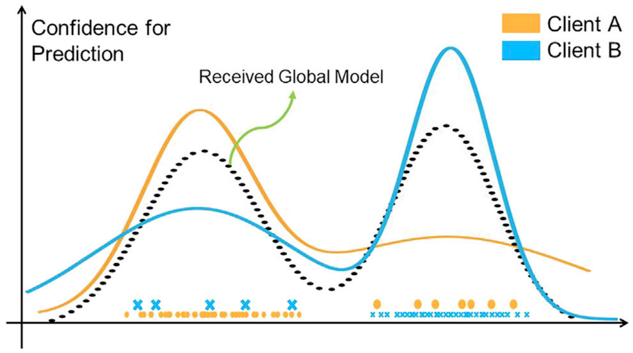


Figure 2. Illustration for the motivation of FLIT

We assume two clients as *A* and *B*, and the local data on these clients do not share the same distribution as the global one. Local models trained on biased local data will overfit the majority groups of data and underfit others. FLIT measures each sample's prediction confidence and puts more weight on the uncertain data. As a result, the local data distribution will be better aligned to the global one, and the trained local models will also be more consistent with each other.

section proposes a method to alleviate the heterogeneity problem, namely FLIT. FLIT adapts the formulation of focal loss for federated learning by involving a global model in local training objectives and can align the local training across clients by focusing on uncertain samples.[32,35] We illustrate the motivation of FLIT in Figure 2.

Learning to reweight training samples is widely used in curriculum learning,[57] hard-sample mining,[32] domain generalization,[35,58,59] debiasing,[60] model calibration,[34] adversarial defense,[61] etc. Our method is closely related to focal loss[32] and worst-case optimization.[35] Mukhoti et al. point out that focal loss could make the objective value aligned with the prediction confidence.[34] GroupDRO improves the model generalization ability by assigning more weights for groups with the worst performance.[35]

FLIT relies on an instance-reweighting framework to improve the federated molecular-property prediction in a heterogeneous setting. The basic observation of FLIT is that, under the heterogeneous settings, the local model will be trained to overfit the small-scaled data at hand. Therefore, the local model will be over-confident for the majority groups of local training samples and may perform poorly and even worse than the received global model on the rare molecules at the client-side. As a result, the local models trained on different clients will significantly deviate from each other, and the inconsistency remarkably degrades the performance of the global model F^g, which is aggregated from the local models in a data-free manner.[8] The sub-optimal performance of FedAvg is wildly admitted by existing studies.[8,27] FLIT puts more weight on samples with low prediction confidence by utilizing the local and global models to alleviate the problem. FLIT explores two different ways to define the prediction confidence, i.e., the loss value (FLIT), and also augmented with prediction consistency among the neighbors (FLIT+). By focusing on the identified uncertain samples, FLIT(+) makes local training more consistent across clients and eventually leads to better federated-learning performance.

Federated learning by instance reweighting

By jointly using the local model F^l and global model F^g, FLIT reweights training samples to align the biased local data distribution to the global one. Eventually, the local models across clients will be well aligned for better performance.

Given a molecule $x^l = (g^l, y^l)$ sampled from the dataset of the l-th client X^l, the original focal loss for binary classification tasks is defined as[32]

$$\mathcal{L}_{focal}(x^l) = -(1-\hat{y}_t^l)^\gamma \log(\hat{y}_t^l), \qquad \text{(Equation 4)}$$

where $\hat{y}_t^l$ is defined based on the prediction of molecule $\hat{y}^l = F^l(g^l)$ as

$$\hat{y}_t^l = \begin{cases} \hat{y}^l & \text{if } y^l = 1 \\ 1-\hat{y}^l & \text{otherwise.} \end{cases}$$

By substituting the binary cross entropy loss $\mathcal{L}(\hat{y}^l, y^l) = -\log(\hat{y}_t^l)$ into Equation 4, we have

$$\mathcal{L}_{focal}(x^l) = (1-\exp(-\mathcal{L}(\hat{y}^l, y^l)))^\gamma \mathcal{L}(\hat{y}^l, y^l). \qquad \text{(Equation 5)}$$

A generalized formulation for instance-reweighting can then be obtained as

$$\mathcal{L}_{FLIT}(x^l) = (1-\exp(-\omega(x^l, F^l, F^g)))^\gamma \mathcal{L}(\hat{y}^l, y^l), \qquad \text{(Equation 6)}$$

where $\omega(x^l, F^l, F^g)$ is a non-negative function that indicates the uncertainty of training samples and is defined by jointly utilizing the local model F^l and global model F^g as

$$\omega(x^l, F^l, F^g) = \varphi(x^l, F^l) + \max(\varphi(x^l, F^l) - \varphi(x^l, F^g), 0), \qquad \text{(Equation 7)}$$

where $\varphi(x, F)$ indicates the prediction uncertainty of x with the model F. Equation 7 puts more weights on samples if the updated local model is less confident than the global model. We note that $\omega(x^l, F^l, F^g)$ can take other types of formulation, and we implement it with Equation 7 for simplicity. Moreover, for FLIT, we follow the focal loss and define $\varphi(\cdot)$ as the loss value,[32] i.e.,

$$\varphi(x^l, F) = \mathcal{L}(\hat{y}^l, y^l). \qquad \text{(Equation 8)}$$

We substitute Equation 8 into Equations 7 and 6, and the resulted method is termed FLIT. Compared with the vanilla focal loss, FLIT integrates the global model F^g into the local training, which turns out to benefit the federated learning according to our experiments.

FLIT+

An alternative way to define $\varphi(\cdot)$ for sample x^l is the prediction discrepancy between the sample and its neighbors.[62] Intuitively, the larger the discrepancy is, the less confident the model is for predicting the sample. To measure the prediction discrepancy for the neighborhoods, we aim to search for the data pairs with largest prediction discrepancy in the neighborhoods. Since directly searching for the exact neighbor is computationally expensive and is implausible with the local biased dataset, we alternatively adopt adversarial neighbor inspired by virtual

Algorithm 2. FLIT(+) for *l*-th client updates

```
Input: $F^g, X^l = \{(g_i^l, y_i^l)\}_{i=1}^{N_l}, \gamma$.
Output: $F^l$
    Save $\varphi(g_i^l, F^g)$ Equation 8 or $\varphi_+(g_i^l, F^g)$ Equation 10
    $F^l \leftarrow F^g$                                        ▷ Init. $F^l$
    for $t$ : 1 to $K$ do                                       ▷ Train on the $l$-th Client
        Sample a minibatch $\{g_i^l, y_i^l\}_{i=1}^{B}$
        Calculate $\varphi(g_i^l, F^l)$ by Equation 8 (or $\varphi_+(g_i^l, F^l)$ by Equation 10)
        Obtain $\omega_{(+)}(x^l, F^l, F^g)$ by Equation 7
        $\omega(G_i^l, F^l, F^g) \leftarrow \frac{\omega(G_i^l, F^l, F^g)}{\overline{\omega}(G_i^l, F^l, F^g)}$        ▷ Normalize $\omega_{(+)}$
        Update $F^l$ by optimizing Equation 6 (or Equation 11)
        $\overline{\omega}_{(+)} \leftarrow \beta\overline{\omega}_{(+)} + (1-\beta)\frac{1}{B}\sum_i \omega_{(+)}$        ▷ Update moving average $\overline{\omega}_{(+)}$
    end for
    Client sends updated model $F^l$ to Server
```

adversarial training (VAT).[33] Adversarial neighbors are similar to x^l in terms of the input g^l but has the most different prediction.[62] Concretely, we measure the discrepancy by adversarial learning with a given model F for x^l as[33]

$$\Delta(x^l, F) = D\left(F(g^l), F(g^l + \xi r_{adv})\right) \qquad \text{(Equation 9)}$$

$$\textit{where } r_{adv} = \underset{r;\|r\| \le \varepsilon}{\arg\max} D\left(F(g^l), F(g^l + r)\right),$$

where $\varepsilon = 0.0001$ is a small positive value, $\xi = 2.5$ is the step size, $D(\cdot)$ can be Kullback-Leibler (KL) divergence for classification or Euclidean distance for regression.[33] Equation 9 measures the discrepancy between predictions of the molecule with graph g^l and its virtual adversarial neighbor $g^l + \xi r_{adv}$. Equation 9 generates a virtual adversarial neighbor $g^l + \xi r_{adv}$ that is similar to g^l (since ε is small) but with the most different prediction. We optimize r on the positions for QM9 and vertex features for other datasets. We omit detail steps for optimizing Equation 9, and please refer to Miyato et al.[33] for details. We jointly use the loss value and the discrepancy defined in Equation 9 and obtain

$$\varphi_+(x^l, F) = \mathcal{L}(\hat{y}^l, y^l) + \lambda\Delta(x^l, F)) \qquad \text{(Equation 10)}$$

where λ is a hyperparameter. By substituting the formulation φ_+ into Equation 7, we obtain $\omega_+(x^l, F^l, F^g)$ to measure the uncertainty of the training samples, and accordingly, we obtain FLIT+ by optimizing the objective as

$$\mathcal{L}_{FLIT+}(x^l) = \left(1 - \exp\left(-\omega_+(x^l, F^l, F^g)\right)\right)^{\gamma}\left(\mathcal{L}(\hat{y}^l, y^l) + \Delta(x^l, F^l)\right). \qquad \text{(Equation 11)}$$

Including $\Delta(x^l, F^l))$ in the training objective is essential to make the neighborhood prediction consistency a valid uncertainty measurement. Moreover, in experiments, we notice that federated learning can benefit from the virtual adversarial training alone, i.e., setting $\gamma = 0$. This should be attributed to the fact that virtual adversarial training could improve the generalization ability of the local model and can be regarded as another way to align the local training implicitly. Detailed results and analysis can be found in the experimental procedures.

We use FLIT(+) to denote both FLIT and FLIT+. We summarize FLIT(+) for client update in Algorithm 2.

Implementation details

Since the scale of $\omega_{(+)}$ may vary significantly especially for regression tasks, it is not proper to directly apply Equations 6 and 11 for general tasks. We propose to normalize the $\omega_{(+)}(\cdot)$ by its moving average as $\omega_{(+)} \leftarrow \frac{\omega_{(+)}}{\overline{\omega}_{(+)}}$, where

$$\overline{\omega}_{(+)} \leftarrow \beta\overline{\omega}_{(+)} + (1-\beta)\frac{1}{B}\sum_i \omega_{(+)} \qquad \text{(Equation 12)}$$

is the moving average and B is the size of minibatch; β is set as 0.8 in this paper.

Moreover, we note that the prediction $F^g(g^l)$ and the discrepancy $\Delta(x^l, F^g)$ for the received global model only need to be calculated once per communication round and thus will not bring much computational cost.

EXPERIMENTAL PROCEDURES

Resource availability

Lead contact

Any further information, questions, or requests should be sent to A. White (andrew.white@rochester.edu).

Materials availability

Our study did not involve any physical materials.

Data and code availability

All used data are publicly available. For reproducibility, our code is available at https://github.com/ur-whitelab/fedchem.git. The code has also been deposited at Zenodo under https://doi.org/10.5281/zenodo.6485682.

Datasets

We conducted experiments on a total of nine datasets retrieved from MoleculeNet[5] for molecular-property prediction, including four regression datasets (FreeSolv, Lipophilicity, ESOL, and QM9) and five classification datasets (Tox21, SIDER, ClinTox, BBBP, and BACE). We follow the prediction tasks in Wu et al.[5] and summarize the statistics for all datasets in Table 1.

Compared methods

To justify the proposed benchmark FedChem, we compare our results with MoleculeNet (MolNet) for centralized training.[5] To validate the effectiveness of FLIT(+), we compare FLIT(+) with FedAvg,[7] FedProx,[29] and MOON.[63] Moreover, we also implement two variants of FLIT(+) as FedAvg with focal loss for

Table 1. Statistics of datasets

Dataset	#Compounds	#tasks	Task type	Metric
FreeSolv	642	1	Reg.	RMSE
Lipophilicity	4,200	1	Reg.	RMSE
ESOL	1,128	1	Reg.	RMSE
QM9	133,885	12	Reg.	MAE
Tox21	7,831	12	Cls.	ROC-AUC
SIDER	1,427	27	Cls.	ROC-AUC
ClinTox	1,478	2	Cls.	ROC-AUC
BBBP	2,039	1	Cls.	ROC-AUC
BACE	1,213	1	Cls.	ROC-AUC

Reg., regression; Cls., classification; RMSE, root-mean-square error; MAE, mean absolute error; ROC-AUC, receiver operating characteristic-area under the curve.

client training (federated focal [FedFocal]) and FedAvg with VAT for client training (FedVAT). We describe the compared methods as follows:

1. FedAvg[7] simply element wisely aggregates the local models to a global one.
2. FedProx[29] regularizes local training to alleviate the heterogeneity problem.
3. MOON[63] applies contrastive learning for federated learning to correct the local training.
4. FedFocal is proposed in this paper and is a variant of FLIT. FedFocal applies focal loss Equation 4 to local training and adopts FedAvg for server update. FedFocal is proposed to validate the effectiveness of involving the global model into local training as FLIT.
5. FedVAT is also proposed in this paper and is a variant of FLIT+. FedVAT jointly optimizes Equation 9 and original training loss for client training and adopts FedAvg for server update. Compared with FLIT+, FedVAT does not use an instance-reweighting training strategy.
6. FLIT is proposed in this paper and is described in Algorithm 2.
7. FLIT+ is proposed in this paper. Compared with FLIT, FLIT+ jointly uses loss values and the discrepancy between nearby samples to measure the uncertainty of samples as described in Equation 10 and adopts Equation 11 as the learning objective.

We perform grid search on the excluded validation set for hyperparameter tuning and model selection. For FedProx, we search the hyperparameter μ from $[0.001, 0.01, 0.1, 1, 10]$. For MOON, we search the hyperparameter from $[0.1,\ 1, 5, 10]$. We search γ used for instance reweighting for FLIT(+) and FedFocal from $[0.5, 1, 2]$ and search λ from $[0.01, 0.1, 1]$ for FLIT+. FedVAT adopts a hyperparameter to balance VAT loss and primary loss, which is searched from $[0.01, 0.1, 1]$. We report results on the testing set by the model with the best performance on the validation set.

Main results

The experimental results on regression and classification datasets are shown in Tables 2 and 3, respectively. We draw several points according to the results. First, comparing our centralized training results (denoted as FedChem) with MolNet,[5] we obtain competitive results by using MPNNs2s[1] and SchNet.[4] Specifically, we obtain a significant performance gain by adopting SchNet for QM9 dataset.[4] Second, comparing the performance of FedAvg with different α for each dataset, we can conclude that the heterogeneity settings introduced by FedChem indeed lead to performance degradation for 7 out of 9 datasets (i.e., FreeSolv, ESOL, QM9, Tox21, ClinTox, BBBP, and BACE). FedAvg shows stable performance for Lipophilicity and SIDER. The reason may be that we do not consider the relation between scaffold subgroups in our current settings, and the resulted clients' datasets are rather homogeneous. Third, we observe a significant performance gain for most datasets by comparing heterogeneous federated-learning methods with FedAvg. For example, the proposed FLIP+ achieves a 0.543 improvement with $\alpha = 0.1$ and 0.162 improvement with $\alpha = 1$ for FreeSolv. The results suggest the necessity to mitigate the heterogeneity when conducting federated learning and validate the effectiveness of the proposed FLIT(+). However, we also observe that the performance improvements of our methods are rather marginal for several datasets. The reasons may be attributed to the fact that our current scaffold splitting may not lead to heterogeneous datasets. We will continue our work for a better method to simulate the heterogeneity problem for federated molecular-property prediction.

Table 2. Performance for federated molecular regression

		Centralized training		Federated learning						
Dataset	α	MolNet[a]	FedChem[a]$_{\text{ours}}$	FedAvg	FedProx	MOON	FedFocal$_{\text{ours}}$	FedVAT$_{\text{ours}}$	FLIT$_{\text{ours}}$	FLIT+$_{\text{ours}}$
FreeSolv ⇓	0.1	1.40	1.430	1.771	1.693	1.376	1.686	1.371	1.634	1.228[d]
	0.5			1.445	1.376	1.423	1.322	1.299	1.366	1.127[d]
	1			1.223	1.216	1.469	1.294	1.150	1.277	1.061[d]
Lipophilicity ⇓	0.1	0.655	0.6290	0.6361[d]	0.6403	0.6426	0.6403	0.6556	0.6563	0.6392
	0.5			0.6306	0.6365	0.6339	0.6351	0.6333	0.6368	0.6270[d]
	1			0.6505	0.6474	0.6442	0.6461	0.6488	0.6443	0.6403[d]
ESOL ⇓	0.1	0.97	0.6570	0.8016	0.7702	0.7537[d]	0.8022	0.7776	0.7788	0.7642
	0.5			0.7524	0.7382	0.7258	0.7708	0.7243	0.7426	0.7119[d]
	1			0.7056	0.6828	0.6751	0.6822	0.7253	0.6705[d]	0.6998
QM9 ⇓	0.1	0.0479[b]	0.0890[c]	0.5889	0.6036	0.5817	0.6164	0.5606	0.5713	0.5356[d]
	0.5			0.5906	0.5751	0.5707	0.6059	0.5656	0.5658	0.5222[d]
	1			0.5786	0.5691	0.5808	0.5822	0.5602	0.5621	0.5282[d]

⇓, ⇑ indicate if lower or higher numbers are better.

[a] Results were obtained with centralized training.

[b] Results were retrieved from Klicpera et al.[2] with a seperate SchNet for each task.

[c] Results were obtained by a single multitask network. Smaller α of LDA generates more extreme heterogeneous scenario. FedFocal and FedVAT are proposed in this paper as the variants of FLIT(+).

[d] Best federated-learning results.

Table 3. Performance for federated molecular classification

		Centralized training		Federated learning						
Dataset	α	MolNet[a]	FedChem[a]$_{ours}$	FedAvg	FedProx	MOON	FedFocal$_{ours}$	FedVAT$_{ours}$	FLIT$_{ours}$	FLIT+$_{ours}$
Tox21 ⇑	0.1	0.829	0.8182	0.7705	0.7732	0.7331	0.7696	0.7733	0.7711	0.7802[b]
	0.5			0.7811	0.7774	0.7461	0.7812	0.7787	0.7825	0.7870[b]
	1			0.7770	0.7775	0.7457	0.7881[b]	0.7706	0.7748	0.7806
SIDER ⇑	0.1	0.638	0.6260	0.6029	0.6056[b]	0.5885	0.6016	0.6027	0.6035	0.6038
	0.5			0.6011	0.5931	0.5966	0.6086	0.5981	0.6096	0.6146[b]
	1			0.6011	0.6023	0.5901	0.6003	0.6053	0.6072	0.6174[b]
ClinTox ⇑	0.1	0.832	0.8903	0.7491	0.7540	0.7892[b]	0.7789[b]	0.7581	0.7761	0.7775
	0.5			0.7521	0.7423	0.7917[b]	0.7770	0.7614	0.7888[b]	0.7852
	1			0.7784	0.7791	0.8001	0.8036[b]	0.7743	0.7849	0.7993
BBBP ⇑	0.1	0.690	0.8674	0.8361	0.8610	0.8737[b]	0.8550	0.8673[b]	0.8666	0.8663
	0.5			0.8594	0.8879[b]	0.8865	0.8726	0.8641	0.8671	0.8774
	1			0.8453	0.8557[b]	0.8487	0.8378	0.8386	0.8515	0.8515
BACE ⇑	0.1	0.806	0.8834	0.8203	0.8328	0.8373	0.8253	0.8166	0.8242	0.8467[b]
	0.5			0.8212	0.8398	0.8285	0.8332	0.8417	0.8516	0.8667[b]
	1			0.8486	0.8408	0.8561	0.8497	0.8578[b]	0.8497	0.8561

⇓, ⇑ indicate if lower or higher numbers are better.
[a]Results are obtained with centralized training.
[b]Best federated-learning results.

Moreover, the proposed instance-reweighting methods (FedFocal, FLIT, and FLIT+) outperform the regularization-based methods FedProx and MOON. The proposed FLIT additionally utilizes the global model and performs better than its counterpart FedFocal. For example, FLIT improves FedFocal from 0.8022 to 0.7788 with $\alpha = 0.1$ and from 0.7708 to 0.7426 with $\alpha = 0.5$ for ESOL. Lastly, FLIT+ further improves the performance of FLIT by measuring the uncertainty with loss values and discrepancy between neighbors. We also observe that FedVAT can benefit federated learning by encouraging locality smoothness for better generalization performance. By incorporating VAT[33] into the FLIT framework, FLIT+ achieves the best overall performance. FLIT(+) has more consistent results across different settings of α compared with its counterparts, indicating the effectiveness of FLIT+ for dealing with heterogeneity problems (see Figure 3).

Sensitivity analysis for federated learning

This section studies the influence of the number of clients and communication rounds on the federated-learning performance. For simplicity, we conduct experiments on ESOL, ClinTox, and BACE. The results of the different number of maximum communication rounds are shown in Figure 4. We vary the maximum communication round from $\{15, 30, 50\}$ while fixing the total local steps. We find that increasing the frequency of communication can benefit federated learning, although it also leads to increased transfer costs. The performance with different numbers of clients is shown in Figure 4. We vary the number of clients within $\{4, 5, 6\}$ since a large number of clients would lead to small local datasets, making training infeasible. We find that the performance of federated learning usually decreases (ESOL and BACE) or is stable (ClinTox) as the client number increases. This indicates that small-scale local training data degrade the federated-learning performance.

Settings for heterogeneous FedChem

For all datasets except QM9, we first randomly split the dataset into 80% for training, 10% for validation, and 10% for testing following Wu et al.[5] QM9 is partitioned into 110,000 samples for training, 10,000 samples for validation, and the remaining for testing following.[4] To simulate the heterogeneous settings for federated learning, we first perform scaffold splitting[10] to partition the training data into subgroups. Then, we assign the molecules of each subgroup to clients by LDA.[8,11] We control the degree of heterogeneity by tuning α for LDA.[8] Smaller α leads to more severe heterogeneity, and we vary α from $\{0.1, 0.5, 1\}$. Moreover, we deliberately balance the number of molecules for

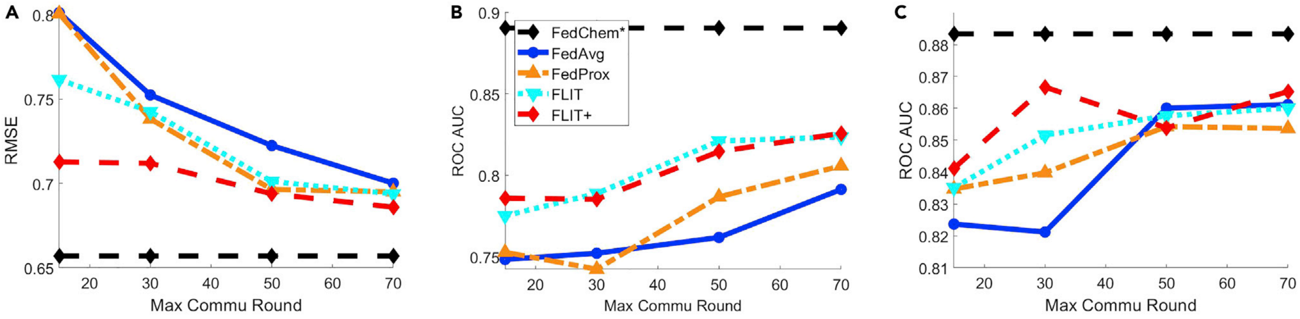


Figure 3. Performance of baseline and our methods with varying communication rounds
Asterisk (*) denotes that the results are obtained with centralized training. We find our method has a strong advantage with a few communication rounds.

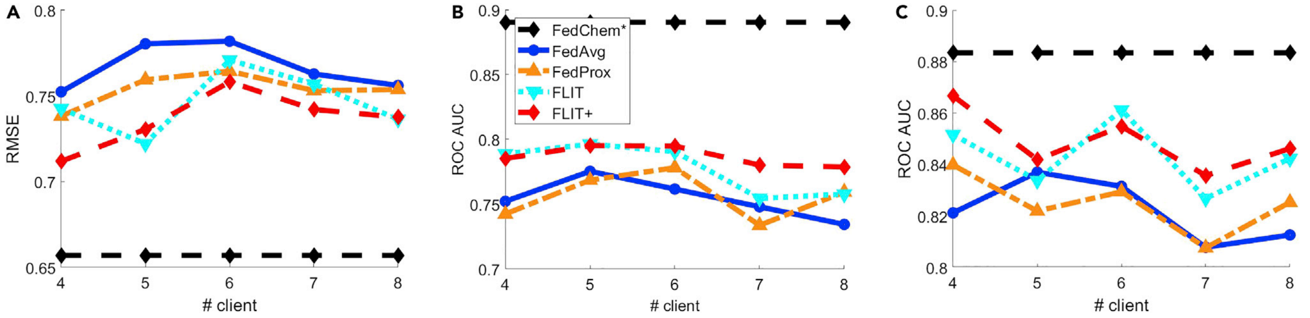


Figure 4. Performance of baseline and our methods with different number of clients

See Figure 3 for color legend. The small-scale local training data reduce federated-learning performance for all methods.

each client following He et al.[11] to control for the effect of the example number on performance.[22]

As for federated-learning settings, we set the default communication rounds *C* to 30 and the default number of clients to four for all datasets except for QM9, which is set to eight.

For client training, we set the batch size to 64 and use Adam[64] with a learning rate of 1 × 10^{-4} and a weight decay of 1 × 10^{-5}. For all datasets except QM9, we simulate four clients and train the local model for 10,000 local steps. QM9 has eight clients, and we train the model for 100,000 local steps. We conduct federated learning with the FedML framework.[11]

For all datasets except QM9, we use MPNNs2s implemented by Deep Graph Library.[65] MPNNs2s has three message-passing layers and three set2sets layers. The hidden feature of the edge is 16, and the output feature of the vertex is 64. We perform three set2set steps. For QM9, we adopt SchNet with six interaction layers[4] and implement SchNet by PyTorch-Geometric.[66] The number of hidden channels and filters of SchNet is 128, and the number of Gaussian is set as 50 for continuous filter layers. We implement a multitask network for datasets with multi-objectives. We run experiments on a server with eight NVIDIA RTX 2080 Ti graphics cards.

Conclusions

Chemistry can be a challenging domain for deep learning because of the computational and materials cost per training example. For example, each row in the Tox21 dataset costs about \$50–\$300 million USD.[67] Therefore, contributing data to a public dataset may be impossible for institutions due to the intrinsic value of the data. Federated learning is a way to build global models while preventing the dissemination of chemical data. We propose a benchmark called FedChem for heterogeneous chemical data, which mimics how chemical data distributes among institutions. FedChem is composed of regression- and classification-learning scenarios from the existing MoleculeNet dataset and utilizes scaffold splitting and LDA to assign molecules with different structures to different clients. FedChem can be tuned to generate scenarios with different degrees of heterogeneity. Given that existing federated-learning methods perform poorly on FedChem, we propose an instance reweighting framework called FLIT(+), inspired by focal loss, to align the training process across clients. We show that FLIT(+) is robust to different tasks and datasets with extensive experiments. One possible future direction is to develop personalized federated learning for FedChem.[46] Moreover, since our current heterogeneous simulation method may not lead to severe structural heterogeneity problems in some cases, we will explore other approaches for more heterogeneous settings.

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.patter.2022.100521.

ACKNOWLEDGMENTS

The research reported in this work was supported by the National Institute of General Medical Sciences of the National Institutes of Health under award number R35GM137966.

AUTHOR CONTRIBUTIONS

W.Z., A.D.W., and J.L. conceptualized the study; W.Z. developed the methodology; W.Z. performed the formal analysis; W.Z. wrote the manuscript; J.L. and A.D.W. reviewed and edited the manuscript; A.D.W. and J. L. supervised the study and acquired funding.

DECLARATION OF INTERESTS

The authors declare no competing interests.

Received: January 4, 2022
Revised: April 15, 2022
Accepted: May 6, 2022
Published: June 2, 2022

REFERENCES

1. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., and Dahl, G.E. (2017). Neural message passing for quantum chemistry. In International conference on machine learning (PMLR), pp. 1263–1272.
2. Klicpera, J., Groß, J., and Günnemann, S. (2020b). Directional message passing for molecular graphs. In International Conference on Learning Representations. https://doi.org/10.48550/arXiv.2003.03123.
3. Satorras, V.G., Hoogeboom, E., and Welling, M. (2021). E(n) equivariant graph neural networks. In Proceedings of the 38th International Conference on Machine Learning, pp. 9323–9332. PMLR volume 139.
4. Schütt, K.T., Sauceda, H.E., Kindermans, P.-J., Tkatchenko, A., and Müller, K.-R. (2018). Schnet–a deep learning architecture for molecules and materials. J. Chem. Phys. *148*, 241722.
5. Wu, Z., Ramsundar, B., Feinberg, E.N., Gomes, J., Geniesse, C., Pappu, A.S., Leswing, K., and Pande, V. (2018). Moleculenet: a benchmark for molecular machine learning. Chem. Sci. *9*, 513–530.
6. Yang, Z., Chakraborty, M., and White, A.D. (2021). Predicting Chemical Shifts with Graph Neural Networks. Chemical science *12*, 10802–10809.
7. McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B.A. (2017). Communication-efficient learning of deep networks from decentralized data. In Artificial Intelligence and Statistics (PMLR), pp. 1273–1282.
8. Wang, H., Yurochkin, M., Sun, Y., Papailiopoulos, D.S., and Khazaeni, Y. (2020b). Federated learning with matched averaging. In 8th International Conference on Learning Representations (OpenReview.net). https://doi.org/10.48550/arXiv.2002.06440.

9. Xie, H., Ma, J., Xiong, L., and Yang, C. (2021). Federated graph classification over non-iid graphs. Preprint at arXiv. https://doi.org/10.48550/arXiv.2106.13423.

10. Bemis, G.W., and Murcko, M.A. (1996). The properties of known drugs. 1. molecular frameworks. J. Med. Chem. *39*, 2887–2893.

11. He, C., Li, S., So, J., Zhang, M., Wang, H., Wang, X., Vepakomma, P., Singh, A., Qiu, H., Shen, L., et al. (2020). Fedml: A research library and benchmark for federated machine learning. Preprint at arXiv. https://doi.org/10.48550/arXiv.2007.13518.

12. Chen, Y., Qin, X., Wang, J., Yu, C., and Gao, W. (2020). Fedhealth: a federated transfer learning framework for wearable healthcare. IEEE Intell. Syst. *35*, 83–93.

13. Aggarwal, D., Zhou, J., and Jain, A.K. (2021). Fedface: collaborative learning of face recognition model. In International IEEE Joint Conference on Biometrics (IEEE), pp. 1–8.

14. Deng, Y., Han, T., and Ansari, N. (2020). Fedvision: federated video analytics with edge computing. IEEE Open Journal of the Computer Society *1*, 62–72.

15. Yang, Q., Liu, Y., Chen, T., and Tong, Y. (2019). Federated machine learning: concept and applications. ACM Transactions on Intelligent Systems and Technology (TIST) *10*, 1–19.

16. Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. (2020). On the convergence of fedavg on non-iid data. In 8th International Conference on Learning Representations (OpenReview.net). https://doi.org/10.48550/arXiv.1907.02189.

17. Chen, H., and Chao, W. (2021). Fedbe: making bayesian model ensemble applicable to federated learning. In 9th International Conference on Learning Representations (OpenReview.net). https://doi.org/10.48550/arXiv.2009.01974.

18. Li, D., and Wang, J. (2019). Fedmd: heterogenous federated learning via model distillation. Preprint at arXiv. https://doi.org/10.48550/arXiv.1910.03581.

19. Lin, T., Kong, L., Stich, S.U., and Jaggi, M. (2020). Ensemble distillation for robust model fusion in federated learning. In Advances in Neural Information Processing Systems, *33*, pp. 2351–2363.

20. Mohri, M., Sivek, G., and Suresh, A.T. (2019). Agnostic federated learning. In International Conference on Machine Learning (PMLR), pp. 4615–4625.

21. Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečnỳ, J., Kumar, S., and McMahan, H.B. (2020). Adaptive federated optimization. Preprint at arXiv. https://doi.org/10.48550/arXiv.2003.00295.

22. Wang, J., Liu, Q., Liang, H., Joshi, G., and Poor, H.V. (2020c). Tackling the objective inconsistency problem in heterogeneous federated optimization. In Advances in Neural Information Processing Systems. https://doi.org/10.48550/arXiv.2007.07481.

23. Yurochkin, M., Agarwal, M., Ghosh, S., Greenewald, K., Hoang, N., and Khazaeni, Y. (2019). Bayesian nonparametric federated learning of neural networks. In International Conference on Machine Learning (PMLR), pp. 7252–7261.

24. Zhu, Z., Hong, J., and Zhou, J. (2021). Data-free knowledge distillation for heterogeneous federated learning. In Proceedings of the 38th International Conference on Machine Learning, *139* (PMLR), pp. 12878–12889.

25. Chen, F., Luo, M., Dong, Z., Li, Z., and He, X. (2018). Federated meta-learning with fast convergence and efficient communication. Preprint at arXiv. https://doi.org/10.48550/arXiv.1802.07876.

26. Dinh, C.T., Tran, N.H., and Nguyen, T.D. (2020). Personalized federated learning with moreau envelopes. In Advances in Neural Information Processing Systems, pp. 21394–21405.

27. Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A.T. (2020). Scaffold: stochastic controlled averaging for federated learning. In International Conference on Machine Learning (PMLR), pp. 5132–5143.

28. Reisizadeh, A., Farnia, F., Pedarsani, R., and Jadbabaie, A. (2020). Robust federated learning: the case of affine distribution shifts. In Advances in Neural Information Processing Systems, *33*, pp. 21554–21565.

29. Sahu, A.K., Li, T., Sanjabi, M., Zaheer, M., Talwalkar, A., and Smith, V. (2018). On the convergence of federated optimization in heterogeneous networks. Preprint at arXiv. https://doi.org/10.48550/arXiv.1812.06127.

30. Sarkar, D., Narang, A., and Rai, S. (2020). Fed-focal loss for imbalanced data classification in federated learning. Preprint at arXiv. https://doi.org/10.48550/arXiv.2011.06283.

31. Fallah, A., Mokhtari, A., and Ozdaglar, A.E. (2020). Personalized federated learning with theoretical guarantees: a model-agnostic meta-learning approach. In Advances in Neural Information Processing Systems, *33*, pp. 3557–3568.

32. Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017). Focal loss for dense object detection. In Proceedings of the IEEE international conference on computer vision, pp. 2980–2988.

33. Miyato, T., Maeda, S.-i., Koyama, M., and Ishii, S. (2018). Virtual adversarial training: a regularization method for supervised and semi-supervised learning. IEEE Trans. Pattern Anal. Mach. Intell. *41*, 1979–1993.

34. Mukhoti, J., Kulharia, V., Sanyal, A., Golodetz, S., Torr, P.H.S., and Dokania, P.K. (2020). Calibrating deep neural networks using focal loss. In Annual Conference on Neural Information Processing Systems, *33*, pp. 15288–15299.

35. Sagawa, S., Koh, P.W., Hashimoto, T.B., and Liang, P. (2019). Distributionally robust neural networks for group shifts: on the importance of regularization for worst-case generalization. Preprint at arXiv. https://doi.org/10.48550/arXiv.1911.08731.

36. Peng, X., Huang, Z., Zhu, Y., and Saenko, K. (2020). Federated adversarial domain adaptation. In 8th International Conference on Learning Representations (OpenReview.net).

37. Song, L., Ma, C., Zhang, G., and Zhang, Y. (2020). Privacy-preserving unsupervised domain adaptation in federated setting. IEEE Access *8*, 143233–143240.

38. Yao, C.-H., Gong, B., Qi, H., Cui, Y., Zhu, Y., and Yang, M.-H. (2022). Federated multi-target domain adaptation. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 1424–1433.

39. Chen, M., Zhang, W., Yuan, Z., Jia, Y., and Chen, H. (2021). Fede: embedding knowledge graphs in federated setting. In IJCKG'21: The 10th International Joint Conference on Knowledge Graphs (ACM), pp. 80–88.

40. He, C., Balasubramanian, K., Ceyani, E., Rong, Y., Zhao, P., Huang, J., Annavaram, M., and Avestimehr, S. (2021). Fedgraphnn: a federated learning system and benchmark for graph neural networks. Preprint at arXiv. https://doi.org/10.48550/arXiv.2104.07145.

41. Lalitha, A., Kilinc, O.C., Javidi, T., and Koushanfar, F. (2019). Peer-to-peer federated learning on graphs. Preprint at arXiv. https://doi.org/10.48550/arXiv.1901.11173.

42. Pei, Y., Mao, R., Liu, Y., Chen, C., Xu, S., Qiang, F., and Tech, B.E. (2021). Decentralized federated graph neural networks. In International Workshop on Federated and Transfer Learning for Data Sparsity and Confidentiality in Conjunction with IJCAI.

43. Wang, B., Li, A., Li, H., and Chen, Y. (2020a). Graphfl: a federated learning framework for semi-supervised node classification on graphs. Preprint at arXiv. https://doi.org/10.48550/arXiv.2012.04187.

44. Wang, C., Chen, B., Li, G., and Wang, H. (2021). FL-AGCNS: federated learning framework for automatic graph convolutional network search. Preprint at arXiv. https://doi.org/10.48550/arXiv.2104.04141.

45. Ma, R., Li, Y., Li, C., Wan, F., Hu, H., Xu, W., and Zeng, J. (2020). Secure multiparty computation for privacy-preserving drug discovery. Bioinform *36*, 2872–2880.

46. Xiong, Z., Cheng, Z., Xu, C., Lin, X., Liu, X., Wang, D., Luo, X., Zhang, Y., Qiao, N., Zheng, M., et al. (2020). Facing small and biased data dilemma in drug discovery with federated learning. Preprint at BioRxiv. https://doi.org/10.1101/2020.03.19.998898.

47. Hao, Z., Lu, C., Huang, Z., Wang, H., Hu, Z., Liu, Q., Chen, E., and Lee, C. (2020). Asgn: an active semi-supervised graph neural network for molecular property prediction. In Proceedings of the 26th ACM SIGKDD

International Conference on Knowledge Discovery & Data Mining, pp. 731–752.

48. Klicpera, J., Giri, S., Margraf, J.T., and Günnemann, S. (2020a). Fast and uncertainty-aware directional message passing for non-equilibrium molecules. Preprint at arXiv. https://doi.org/10.48550/arXiv.2011.14115.

49. Anderson, B.M., Hy, T., and Kondor, R. (2019). Cormorant: covariant molecular neural networks. In Advances in Neural Information Processing Systems, pp. 14510–14519.

50. Miller, B.K., Geiger, M., Smidt, T.E., and Noé, F. (2020). Relevance of rotationally equivariant convolutions for predicting molecular properties. Preprint at arXiv. https://doi.org/10.48550/arXiv.2008.08461.

51. Honda, S., Shi, S., and Ueda, H.R. (2019). SMILES transformer: pre-trained molecular fingerprint for low data drug discovery. Preprint at arXiv. https://doi.org/10.48550/arXiv.1911.04738.

52. Mayr, A., Klambauer, G., Unterthiner, T., Steijaert, M., Wegner, J.K., Ceulemans, H., Clevert, D.-A., and Hochreiter, S. (2018). Large-scale comparison of machine learning methods for drug target prediction on chembl. Chem. Sci. *9*, 5441–5451.

53. Seo, H., Park, J., Oh, S., Bennis, M., and Kim, S. (2020). Federated knowledge distillation. Preprint at arXiv. https://doi.org/10.48550/arXiv.2011.02367.

54. Jorgensen, W.L. (2009). Efficient drug lead discovery and optimization. Accounts of chemical research *42*, 724–733.

55. Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2018). How powerful are graph neural networks? In International Conference on Learning Representations. https://doi.org/10.48550/arXiv.1810.00826.

56. Kipf, T.N., and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In 5th International Conference on Learning Representations (OpenReview.net). https://doi.org/10.48550/arXiv.1609.02907.

57. Zhou, T., Wang, S., and Bilmes, J.A. (2020). Curriculum learning by dynamic instance hardness. Adv. Neural Inf. Process. Syst. *33*, 8602–8613.

58. Arjovsky, M., Bottou, L., Gulrajani, I., and Lopez-Paz, D. (2019). Invariant risk minimization. Preprint at arXiv. https://doi.org/10.48550/arXiv.1907.02893.

59. Krueger, D., Caballero, E., Jacobsen, J., Zhang, A., Binas, J., Zhang, D., Priol, R.L., and Courville, A.C. (2021). Out-of-distribution generalization via risk extrapolation (rex). In Proceedings of the 38th International Conference on Machine Learning, *139* (PMLR), pp. 5815–5826.

60. Nam, J.H., Cha, H., Ahn, S., Lee, J., and Shin, J. (2020). Learning from failure: training debiased classifier from biased classifier. Preprint at arXiv. https://doi.org/10.48550/arXiv.2007.02561.

61. Zhang, J., Zhu, J., Niu, G., Han, B., Sugiyama, M., and Kankanhalli, M. (2020). Geometry-aware instance-reweighted adversarial training. In International Conference on Learning Representations. https://doi.org/10.48550/arXiv.2010.01736.

62. Wei, C., Shen, K., Chen, Y., and Ma, T. (2021). Theoretical analysis of self-training with deep networks on unlabeled data. In 9th International Conference on Learning Representations (OpenReview.net). https://doi.org/10.48550/arXiv.2010.03622.

63. Li, Q., He, B., and Song, D. (2021). Model-contrastive federated learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10713–10722.

64. Kingma, D.P., and Ba, J. (2015). Adam: a method for stochastic optimization. In 3rd International Conference on Learning Representations, Y. Bengio and Y. LeCun, eds. (San Diego, CA, USA: ICLR). May 7-9, 2015, Conference Track Proceedings.

65. Wang, M., Yu, L., Zheng, D., Gan, Q., Gai, Y., Ye, Z., Li, M., Zhou, J., Huang, Q., Ma, C., et al. (2019). Deep graph library: towards efficient and scalable deep learning on graphs. Preprint at arXiv. https://doi.org/10.48550/arXiv.1909.01315.

66. Fey, M., and Lenssen, J.E. (2019). Fast graph representation learning with PyTorch Geometric. In ICLR Workshop on Representation Learning on Graphs and Manifolds. https://doi.org/10.48550/arXiv.1903.02428.

67. DiMasi, J.A., Grabowski, H.G., and Hansen, R.W. (2016). Innovation in the pharmaceutical industry: new estimates of r&d costs. J. Health Econ. *47*, 20–33.