Federated Learning with Client-Exclusive Classes

Jiayun Zhang[1], Xiyuan Zhang[1], Xinyang Zhang[2], Dezhi Hong[3†], Rajesh K. Gupta[1], Jingbo Shang[1]
[1]University of California, San Diego, La Jolla, CA, USA
[2]University of Illinois at Urbana-Champaign, Champaign, IL, USA [3]Amazon
{jiz069,xiyuanzh,gupta,jshang}@ucsd.edu,xz43@illinois.edu,hondezhi@amazon.com

ABSTRACT

Existing federated classification algorithms typically assume the local annotations at every client cover the same set of classes. In this paper, we aim to lift such an assumption and focus on a more general yet practical non-IID setting where every client can work on non-identical and even disjoint sets of classes (i.e., *client-exclusive classes*), and the clients have a common goal which is to build a global classification model to identify the union of these classes. Such heterogeneity in client class sets poses a new challenge: how to ensure different clients are operating in the same latent space so as to avoid the drift after aggregation? We observe that the classes can be described in natural languages (i.e., class names) and these names are typically safe to share with all parties. Thus, we formulate the classification problem as a matching process between data representations and class representations and break the classification model into a data encoder and a label encoder. We leverage the natural-language class names as the common ground to anchor the class representations in the label encoder. In each iteration, the label encoder updates the class representations and regulates the data representations through matching. We further use the updated class representations at each round to annotate data samples for locally-unaware classes according to similarity and distill knowledge to local models. Extensive experiments on four real-world datasets show that the proposed method can outperform various classical and state-of-the-art federated learning methods designed for learning with non-IID data.

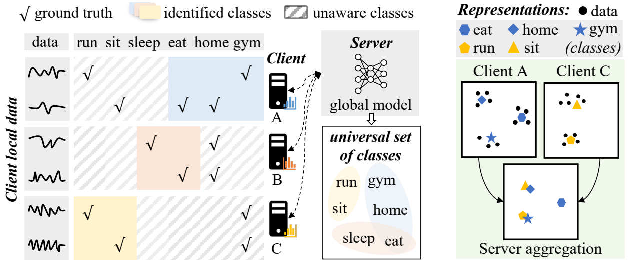
KEYWORDS

Federated Learning, non-IID, Label Name Sharing, Knowledge Distillation

1 INTRODUCTION

Federated learning [30] emerges as a distributed learning paradigm that allows multiple parties to collaboratively learn a global model effective for all parties while ensuring the privacy of local data. This benefits applications in a wide range of domains, such as recommendation [23, 27, 47], ubiquitous sensing [10, 16, 17, 39] and mobile computing [15, 19, 46].

Existing federated classification methods, even those designed for non-IID clients [9, 20, 21, 25, 28, 43, 44, 50], typically assume that the local annotations at each client follow the same set of classes; however, this assumption does not hold true in many real-world applications. For example, a smartwatch company wants to build a human activity classifier for all activity types, as shown in Figure 1(a). Although their smartwatch users as clients could experience almost all types of daily activities, each user may only opt to report (i.e., annotate) a subset of activities. Another example is a federated medical diagnosis system, which attempts to infer all types of diseases of a patient for comprehensive health screening. Physicians and specialist groups with different expertise can participate in this federated learning system as clients. As one can see here, different specialists will only offer disease annotations within their domains, even if a patient may have several types of diseases at the same time. This makes the class sets at many clients non-identical and even non-overlapping.



(a) The problem setting of non-identical client class sets. (b) Unique challenge.

Figure 1: (a) illustrates our problem setting using a behavioral context recognition system where users have different preferences in reporting (i.e., annotating) labels. (b) demonstrates the unique challenge here — local models at different clients may operate in different and even independent latent spaces, making it hard to aggregate at the server.

We aim to lift this assumption and work on a new and rather practical federated learning setting, **client-exclusive classes**, where the local annotations at each client cover different and even non-overlapping sets of classes. We denote the classes that are not covered in the local annotations as *locally-unaware classes* at each client. Each client can hold local data without any annotations, no matter whether their true labels are among the locally-unaware classes or not. Also, the classification task here can be either single-label or multi-label. When it is multi-label, the local data might be only partially labeled due to the locally-unaware classes. Therefore, this new setting is more general and challenging than the missing class scenario [22], which is only applicable to single-label classification.

The non-identical client class sets pose a significant challenge of huge variance in local training across different clients. As shown in Figure 1(b), one can view classification as a matching process between data representations and label representations in a latent space. Because of the non-identical client class sets, locally trained classifiers are more likely to operate in drastically different latent spaces. Moreover, when the class sets are non-overlapping, it is

[†]Work unrelated to Amazon.

possible that the latent spaces at different clients are completely independent. This would result in inaccurate classification boundaries after aggregation at the server, making our setting more challenging than non-IID clients with identical client class sets.

We propose a novel federated learning framework FedAlign to better align the latent spaces across clients from both label and data perspectives as follows:

(1) *Anchor the label representations using label names.* We observe that the natural-language class names (i.e., label names) often carry valuable information for understanding label semantics, and more importantly, they are typically safe to share with all parties. Therefore, we break the classification model into a data encoder and a label encoder as shown in Figure 2, and then leverage the label names as the common ground for label encoders. The server initializes the label encoder with pre-trained text representations, such as word embedding. The label encoder will be then distributed to different clients and updated alternatively with data encoders during local training and global aggregation, mutually regulating the latent space.
(2) *Connect the data representations via locally-unaware classes.* The representations given by the label encoder are representative of the classes. Their distances to the data samples in the latent space can tell how likely a sample belongs to a class. Therefore, we introduce regularization in a knowledge distillation manner. Specifically, as shown in Figure 2, at each client, we first annotate local data based on their similarities with the representations of the locally-unaware classes, and then add another cross-entropy loss between the pseudo-annotations and the predictions by local models. Such regularization connects data encoders across different clients, so they have a better chance to be in the same latent space.

We conduct experiments on four real-world applications, including the most challenging scenario of multi-label classification with non-overlapping client class sets. We show that FedAlign can outperform various state-of-the-art federated learning algorithms. Our contributions are as follows:

- We lift the common assumption of identical client class sets in federated learning and study a more general yet practical setting, non-identical client class sets. Such heterogeneity in client class sets poses a new challenge — local models at different clients may operate in different and even independent latent spaces.
- We propose a novel framework FedAlign to better align the latent spaces across clients by anchoring the label representations using label names and connecting data representations via locally-unaware classes.
- We conduct extensive experiments on four real-world datasets of different tasks and show that FedAlign can further improve various state-of-the-art federated learning algorithms.

2 PRELIMINARIES

Problem Formulation. We aim to generate a global classification model using federated learning with non-identical class sets, where each client only identifies part of the classes from its dataset. Given the universal set of classes as C, the set of classes that are identified on client m is C_m, and the set of locally-unaware classes is $\overline{C_m}$, where $C_m \cup \overline{C_m} = C$. The objective is to learn a global model $g : X \rightarrow C$ that given input feature X, all positive labels from the class set C can be inferred.

The training set on client m is denoted as $\mathcal{D}_m = \{(x_i, y_i)\}_{i=1}^{N}$, where x_i is the data feature and $y_i = (y_{i,c_1}, \ldots, y_{i,c_{|C|}})$ is a vector showing the labels of each class. If $c_j \in C_m, y_{i,c_j} \in \{0, 1\}$. If $c_j \in \overline{C_m}$, y_{i,c_j} is unknown. It is possible that some data samples $x_i \in D_m$ do not belong to any of the classes in C_m, i.e., $\forall c \in C_m : y_{i,c} = 0$.

Backbone Classification Model. Generally, the classification model can be decomposed into a data encoder and a classification layer. The data encoder takes the input data x_i and generates a data representation. Then, a linear layer (i.e., classifier) transforms the representation into prediction logits. We discuss two types of classification tasks as follows.

Single-label multi-class classification. In this setting, each sample is associated with only one positive class. In other words, the classes are mutually exclusive. We use the softmax activation function to get the prediction probability. Denote $g_m(\cdot)$ as the classification model at client m and the prediction probability of x_i on class c as $g(x_i)_c$. The class with the maximum probabilities is predicted as positive, i.e., $\tilde{y}_i = \underset{c}{argmax}(g_m(x_i)_c)$. The loss function at client m is:

$$\mathcal{L}_{C_m} = -\frac{1}{N_m} \sum_{i=1}^{N_m} \sum_{c \in C_m} y_{i,c} \log g_m(x_i)_c$$

Multi-label classification. In this setting, each sample may be associated with a set of positive classes. For example, a person may have both diabetes and hypertension. The sigmoid function is applied to get the prediction probability, each element of which is the probability that the input data x_i is associated with each class. The final predictions are achieved by thresholding the probabilities at 0.5. If $g(x_i)_c > 0.5$, sample x_i is predicted to be in class c, i.e., $\tilde{y}_{i,c} = 1$. The learning objective is to minimize the binary cross-entropy loss over the identified classes:

$$\mathcal{L}_{C_m} = -\frac{1}{N_m} \sum_{i=1}^{N_m} \sum_{c \in C_m} [y_{i,c} \log g_m(x_i)_c + (1 - y_{i,c}) \log(1 - g_m(x_i)_c)].$$

3 THE FEDALIGN FRAMEWORK

3.1 Overview

The pseudo code of FedAlign can be found in Algorithm 1. Learning with FedAlign framework consists of the following steps:

(1) **Label name sharing and label encoder initialization:** Before training, the server collects the natural language label names from the clients. The server initializes the label encoder θ^0 via pre-trained text representations, such as word embedding. We expect more advanced techniques like pre-trained neural language models could make the learning converge even faster, but we leave it as future work. The data encoder ω^0 is randomly initialized.
(2) **Client selection and model communication:** At t-th round, the server randomly selects a subset of clients S_t and sends the global model parameters to them.
(3) **Local training:** Client $m \in S_t$ independently trains its local model and returns the parameters.
(4) **Model aggregation:** The server aggregates the parameters of client models into global parameters.

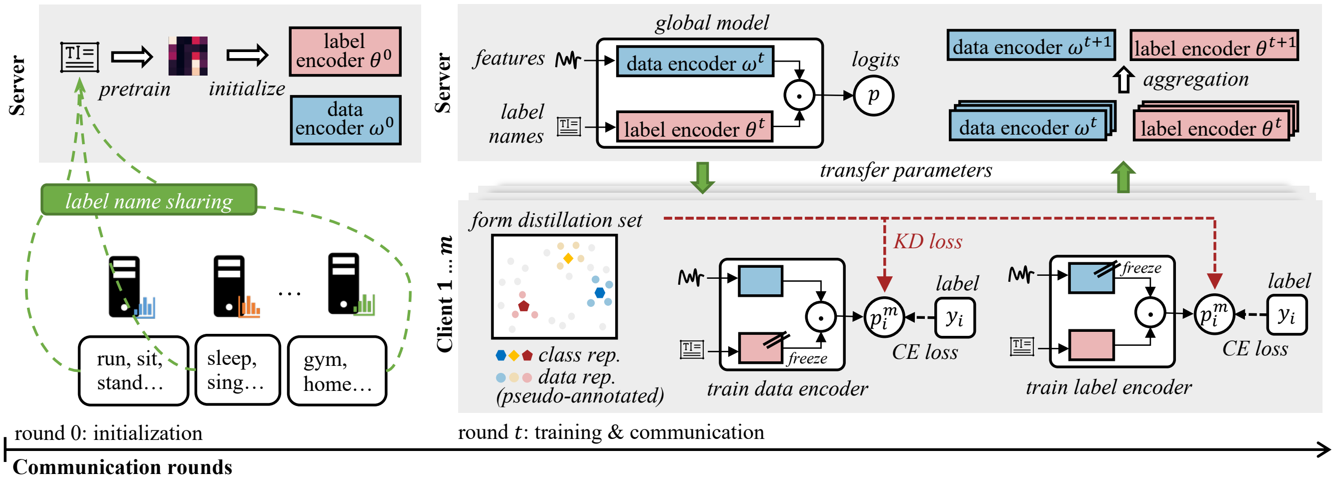


Figure 2: Overview of FedAlign framework. The label names are leveraged as a common ground for label encoders to anchor class representations. During local training, the two encoders perform alternating training to mutually regulate the latent spaces. The labels about locally-unaware classes are assigned to the local samples based on the similarities between the data representations and class representations and the knowledge is transferred to local models via knowledge distillation objectives.

```
Algorithm 1: FedAlign Framework
  Input  : Communication rounds $T$, number of selected
           clients each round $|S_t|$, local training epochs $E$.
  Output : The final global model $g(x; \theta^T, \omega^T)$.
Server executes:
Collect label names from clients and pre-train text
  representations to initialize label encoder $\theta^0$ ;   // Step (1)
Randomly initialize data encoder $\omega^0$;
for $t = 0, 1, ..., T-1$ do
    Select a subset $S_t$ of clients at random ;             // Step (2)
    for $m \in S_t$ do
        $\theta_m^{(t+1)}, \omega_m^{(t+1)} \leftarrow$ ClientUpdate$(m, \theta^{(t)}, \omega^{(t)})$;
        // Step (3)
    $\omega^{(t+1)} \leftarrow \frac{1}{|S_t|} \sum_{m \in S_t} \omega_m^{(t+1)}$;   // Step (4)
    $\theta_c^{(t+1)} \leftarrow \frac{\sum_{m \in S_t, c \in C_m} \theta_{m,c}^{(t+1)}}{|\{m | m \in S_t, c \in C_m\}|}$;   // Step (4)
return $\theta^T, \omega^T$
ClientUpdate$(m, \theta^{(t)}, \omega^{(t)})$:
$\theta_m^{(t)}, \omega_m^{(t)} \leftarrow \theta^{(t)}, \omega^{(t)}$;
for $epoch \in \{1, 2, ..., E\}$ do
    Calculate similarities of data and class representations
      and form distillation set $\tilde{\mathcal{D}}_m^{(t)}$ ;   // Step (a)
    Freeze $\theta_m^{(t)}$ and update $\omega_m^{(t+1)}$ ;   // Step (b)
    Freeze $\omega_m^{(t+1)}$ and update $\theta_m^{(t+1)}$ ;   // Step (c)
return $\theta_m^{(t+1)}$, and $\omega_m^{(t+1)}$ to server ;   // Step (d)
```

Pre-training text representations and label encoder initialization in (1) are conducted only once at the beginning. Steps (2)-(4) repeat for T rounds until the global model converges. During local training in (3), each client $m \in S_t$ conducts the following steps:

(a) **Form distillation dataset for locally-unaware classes:** Client m forms a distillation set $\tilde{\mathcal{D}}_m^{(t+1)}$ for locally-unaware classes $\overline{C_m}$ by using the latest class representations.
(b) **Train data encoder:** Client m freezes the label encoder parameters and trains the data encoder.
(c) **Train label encoder:** Client m freezes the parameters in the data encoder and trains the label encoder.
(d) **Model communication after local updates:** Client m sends the updated model parameters to the server.

3.2 Semantic Label Name Sharing

The vanilla classification model in Section 2 learns latent label spaces merely based on the local training data with numerical label IDs. With non-identical client class sets, local models at different clients are likely to form different and even independent label spaces, making the classification boundaries aggregated at the server inaccurate. To better align the label spaces, we turn to the semantics of label names as a common ground to anchor class representations. The natural language label names carry valuable information for understanding the label correlations. For example, in behavioral context recognition, the activity of "lying down" is likely to indicate the person is "sleeping", and the possible location of the activity is "at home". Such knowledge about label correlations not only exists in the datasets to investigate, but also can be mined through analyzing the semantics of the label names.

Incorporating Label Encoder to Classification Model. We replace the classification layer in a conventional classification model with a label encoder as shown in Figure 2. The label encoder

$f_\theta : w_c \rightarrow r_c \in R^d$ takes the natural language label names w_c as the inputs and maps them into representations r_c. Prior knowledge about label semantics can be inserted into the label encoder by initializing it with pre-trained label embeddings. Inspired by existing works that learn semantic word embeddings based on word-word co-occurrence and point-wise mutual information (PMI) [1, 14, 34], we use an external text corpus related to the domain of the classification task to extract knowledge of label co-occurrence and pre-train the label embeddings for initializing the label encoder. The details about label embedding pre-training can be found in Appendix.

For each input data x_i, the model calculates the matching scores of x_i and every class by taking the dot product of the data representation $h_\omega : x_i \rightarrow z_i \in \mathbb{R}^d$ output from the data encoder and the class representations output by the label encoder. This way, it calculates the similarity between the representations of input data and classes. Then, an activation function is applied to the dot product to get the prediction probabilities of x_i belonging to each class. Same as described in Section 2, for multi-label classification, the sigmoid function is used and the final prediction is achieved by thresholding the probabilities at 0.5. For single-label classification, the softmax function is used and the final prediction is the class with the maximum probabilities.

Alternating Training of Encoders. The data encoder and label encoder are two branches in the classification model. We want the representations obtained by one encoder to regulate the training of the other while preventing mutual interference. Thus, at each learning iteration, we first freeze the parameters in the label encoder θ and update the parameters of the data encoder ω. Then, we freeze the parameters in the data encoder and update the label encoder parameters.

3.3 Knowledge Distillation for Locally-unaware Classes

Due to the lack of label information of certain classes to support supervised training, the training at each client is biased toward the identified classes. To mitigate such drift, we design a knowledge distillation strategy for unaware classes at each client by further utilizing the class representations. The class representations are updated with the training of the label encoder at each round of communication. As the classification problem is formed as a matching between the embeddings of data and label names in Section 3.2, the distance between the label names and the data samples in the latent space can tell how likely a sample belongs to a class. Thus, we use the class representations to annotate samples for the locally-unaware classes and distill knowledge to local models.

Forming Local Distillation Dataset. When the client receives the parameters of the data encoder $\omega^{(t)}$ and label encoder $\theta^{(t)}$ at t-th round, it uses the label encoder to get the updated class representations $\{f_\theta^{(t)}(w_c) | c \in C\}$ and uses the data encoder to generate the data representations of its local data $\{h_\omega^{(t)}(x_i) | (x_i, y_i) \in \mathcal{D}_m\}$. Then, for each locally-unaware class $\dot{c}$, the client calculates its cosine similarities to the data:

$$s_{i,\dot{c}}^{(t)} = \frac{f_\theta^{(t)}(w_{\dot{c}}) \cdot h_\omega^{(t)}(x_i)}{\|f_\theta^{(t)}(w_{\dot{c}})\| \cdot \|h_\omega^{(t)}(x_i)\|}.$$

Samples are annotated according to their similarities to the class. The samples with the highest similarities with the class representation $f_\theta^{(t)}(w_{\dot{c}})$ are annotated as positive samples of class $\dot{c}$. The client also annotates the negative samples for the locally-unaware classes. The sample with the least similarities with $f_\theta^{(t)}(w_{\dot{c}})$ is annotated as a negative sample of $\dot{c}$.

The number of samples to be annotated depends on the percentile of similarity. We set the thresholds $\hat{\tau}_{\dot{c}}^{(t)}$ and $\check{\tau}_{\dot{c}}^{(t)}$ as the q_1-th and q_2-th percentile of the similarities over all samples for annotating positive samples and negative samples respectively. The client annotates a subset of the samples whose similarities are higher than $\hat{\tau}_{\dot{c}}^{(t)}$ as the positive samples in class $\dot{c}$ (i.e., $\tilde{y}_{i,\dot{c}}^{(t)} = 1$) and those with selection scores lower than $\check{\tau}_{\dot{c}}^{(t)}$ as the negative samples in class $\dot{c}$ (i.e., $\tilde{y}_{i,\dot{c}}^{(t)} = 0$). The distillation dataset after the t-th round is:

$$\tilde{\mathcal{D}}_m^{(t)} \leftarrow \{(x_i, \tilde{y}_i^{(t)}) | s_{i,\dot{c}}^{(t)} > \hat{\tau}_{\dot{c}}^{(t)} \text{ or } s_{i,\dot{c}}^{(t)} < \check{\tau}_{\dot{c}}^{(t)}, \dot{c} \in \overline{C_m}\}$$

In single-label classification, there is another constraint that the unlabeled sample can be annotated as a positive sample of class $\dot{c}$ only if the class is the closest to the data representation among all classes, i.e., $\dot{c} = \underset{c \in C}{argmax}(s_{i,c}^{(t)})$.

Knowledge Distillation. The pseudo annotations about the locally-unaware classes $\overline{C_m}$ are then used as knowledge to transfer to client m's local model. Let $N_{\dot{c}}^{(t)}$ be the number of samples in $\tilde{\mathcal{D}}_m^{(t)}$, $g_m^{(t+1)}(\cdot)$ be the local model with parameters being updated. We use binary cross-entropy loss for multi-label classification as the distillation loss:

$$\mathcal{L}_{\overline{C_m}}^{(t)} = -\frac{1}{N_m^{'}} \sum_{i=1}^{N_m^{'}} \sum_{\dot{c} \in \overline{C_m}} [\tilde{y}_{i,c}^{(t)} \log g_m^{(t+1)}(x_i)_{\dot{c}} + \\ (1 - \tilde{y}_{i,c}^{(t)}) \log(1 - g_m^{(t+1)}(x_i)_{\dot{c}})],$$

where $N_m^{'}$ is the number of samples in the distillation set $\tilde{\mathcal{D}}_m^{(t)}$.

For single-label classification, we use cross-entropy loss as distillation loss:

$$\mathcal{L}_{\overline{C_m}}^{(t)} = -\frac{1}{N_m^{'}} \sum_{i=1}^{N_m^{'}} \sum_{c \in C} \tilde{y}_{i,c}^{(t)} \log g_m^{(t+1)}(x_i)_c.$$

We use a parameter α to control the weight of the distillation loss term. The learning objective of the (t)-th round on client m is to minimize the following loss term:

$$\mathcal{L}_m^{(t)} = \mathcal{L}_{C_m}^{(t)} + \alpha \mathcal{L}_{\overline{C_m}}^{(t)}$$

3.4 Model Aggregation

After the clients finish the local training, they send the updated parameters to the server. After the server collects the parameters from all participating clients $m \in S_t$, it conducts model aggregation. For the data encoder, the parameters are averaged across all client models:

$$\omega^{(t+1)} \leftarrow \frac{1}{|S_t|} \sum_{m \in S_t} \omega_m^{(t+1)}.$$

For the label encoder, the representations of class c are averaged only among the client models that have class c as its locally-identified class:

$$\theta_c^{(t+1)} \leftarrow \frac{\sum_{m \in S_t, c \in C_m} \theta_{m,c}^{(t+1)}}{|\{m | m \in S_t, c \in C_m\}|}.$$

4 EXPERIMENTS

We first introduce the applications and the datasets we use. Then, we describe the compared methods and experimental setup. We show the experimental results with different settings and conduct an ablation study and to show the effectiveness of the designs in FedAlign.

4.1 Datasets

We conduct experiments on 6 datasets covering 4 different application scenarios and both single-label and multi-label classification problems. Table 1 offers an overview and the details are as follows.

(1) **Behavioral Context Recognition.** The task is to infer the context of human activity, e.g., what the person is doing, where and with whom the person is. **ExtraSensory** [40] is a benchmark dataset for this application scenario. It contains 51 classes in total which can be partitioned into 5 main categories: *posture*, *location*, *activity*, and *companion*. Based on ExtraSensory, we construct 3 datasets with non-overlapping client class sets as follows.
   - **ES-5**. In this dataset, we have 5 clients and every client only has annotations from a different main category (i.e., one main category to one client). Training samples are then assigned to clients according to their associated classes. Since ExtraSensory is a multi-label dataset, the same training sample may have multiple classes from different main categories. In this case, we assign the training sample based on the most infrequent class among these multiple labels to ensure that each locally-identified class will have at least one positive sample. To make this dataset more realistic, we always assign all the data of each human subject to the same client.
   - **ES-15 and ES-25**. We also increase the number of clients to 15 and 25 to further challenge the compared methods. We start with the 5 class groups as ES-5 and iteratively split the class groups until the number of class groups is the same as the number of clients. For every split, we select the group with the most number of classes and randomly separate it into two sub-groups. Every class group is visible and only visible to one client. One can then apply a similar process as ES-5 to assign training samples to different clients.

(2) **Medical Code Prediction.** Medical codes describe whether a patient has a specific medical condition or is at risk of development. The task aims at automatically annotating medical codes from clinical notes that record information about what happened during a patient's hospitalization. We start with the MIMIC-III database [7] and follow the preprocessing procedure in [32] to form the benchmark MIMIC-III 50-label dataset that contains the 50 most frequent labels. These classes can be grouped into 10 categories according to the ICD-9 taxonomy[1]. We construct the **MIMIC-III-10** dataset by partitioning the dataset into 10 clients following the same strategy as that in ES-5.

Table 1: Our datasets cover application scenarios of behavioral context recognition, clinical phenotype classification, human activity recognition, and text classification. The imbalance factor of a dataset refers to the ratio of its smallest class size to the largest class size.

Dataset	$\|C\|$	# of clients	Avg. N_m	Avg. $\|C_m\|$	Imbalance	Remarks
ES-5	51	5	5,446	10.2	0.0013	Multi-label, non-overlapping client class sets
ES-15		15	1,769	3.4		
ES-25		25	1,073	2.04		
MIMIC-III-10	50	10	807	5	0.1157	
PAMAP2-9	18	9	1,287	5	0.2049	Single-label
R8-8	8	8	617	3	0.0130	

(3) **Human Activity Recognition.** The task aims at identifying the movement or action of a person based on sensor data. We start with the PAMAP2 [38] dataset, which contains data of 18 different physical activities collected from 9 subjects. We construct the **PAMAP2-9** dataset by using the data from each subject as a client, which results in 9 clients. For each client, we randomly select 5 classes to be its locally-identified classes.

(4) **Text Classification.** The task is to categorize text into organized groups. We use the Reuters-21578 R8 dataset [2], which is a collection of news articles. We construct the **R8-8** dataset by setting the number of clients as 8, randomly dividing the data into 8 partitions, and assigning one partition to each client. For each client, we randomly select 3 classes to be the identified classes.

More details about data preprocessing are described in Appendix.

4.2 Compared Methods

We compare FedAlign with various classical [30] and state-of-the-art federated learning methods [9, 20, 21] designed for non-IID data problems as follows.

- **FedAvg** [30] is a classical federated learning method. The weights of local models uploaded by clients in each round are averaged to get the global model. Then, the server sends the updated model to each client.
- **FedProx** [21] deals with the heterogeneity in the client models by enforcing a L_2 regularization term in local optimization to limit the distance between local model and global model.
- **MOON** [20] adds a contrastive loss term to maximize the consistency of representations learned by the local model and that by the global model and minimize the consistency between representations learned by the local models of consecutive rounds.
- **Scaffold** [9] addresses the client-variance problem by maintaining control variates to estimate the update directions of the global model and the client model. The drift of local training is approximated by the difference between the local and global update directions and added to the local updates to mitigate drift.

[1] the International Statistical Classification of Diseases and Related Health Problems (ICD): ftp://ftp.cdc.gov/pub/Health_Statistics/NCHS/Publications/ICD-9/ucod.txt

Table 2: Main experimental results (%) on the six datasets. ExtraSensory (ES-5, ES-15, ES-25) and MIMIC-III are multi-label datasets where class sets of different clients are non-overlapping. PAMAP2 and R8 are single-label datasets where the class sets of different clients overlap. The results are averaged over 5 runs.

Method	ES-5		ES-15		ES-25		MIMIC-III-10		PAMAP2-9		R8-8	
	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc
FedAvg [30]	28.77	80.79	22.71	62.44	19.52	50.42	35.04	67.07	68.89	71.45	78.51	92.76
FedProx ($\mu = 0.001$) [21]	29.26	80.67	22.42	61.91	19.48	51.16	34.75	65.98	69.70	73.63	79.05	92.61
FedProx ($\mu = 0.0001$) [21]	28.53	79.14	22.52	61.95	19.05	52.33	33.73	64.31	69.38	71.78	75.98	91.78
MOON [20]	29.12	81.00	22.84	62.53	19.52	50.24	34.62	66.34	71.70	74.25	79.26	93.07
Scaffold [9]	28.14	77.13	23.15	61.69	19.73	48.81	33.58	62.84	73.57	75.60	82.83	94.43
FedRS ($\alpha = 0.5$) [22]	28.01	78.72	22.50	62.09	19.44	51.41	34.82	66.80	68.70	71.42	80.10	92.74
FedRS ($\alpha = 0.9$) [22]	28.25	79.25	22.55	62.17	19.40	50.87	35.44	67.45	71.81	74.44	76.68	91.81
FedPU [26]	-	-	-	-	-	-	-	-	85.39	87.59	83.17	94.23
FedAlign	**29.69**	**83.92**	**23.36**	**73.61**	**20.78**	**62.11**	**37.87**	**71.13**	**87.21**	**88.14**	**83.76**	**94.92**

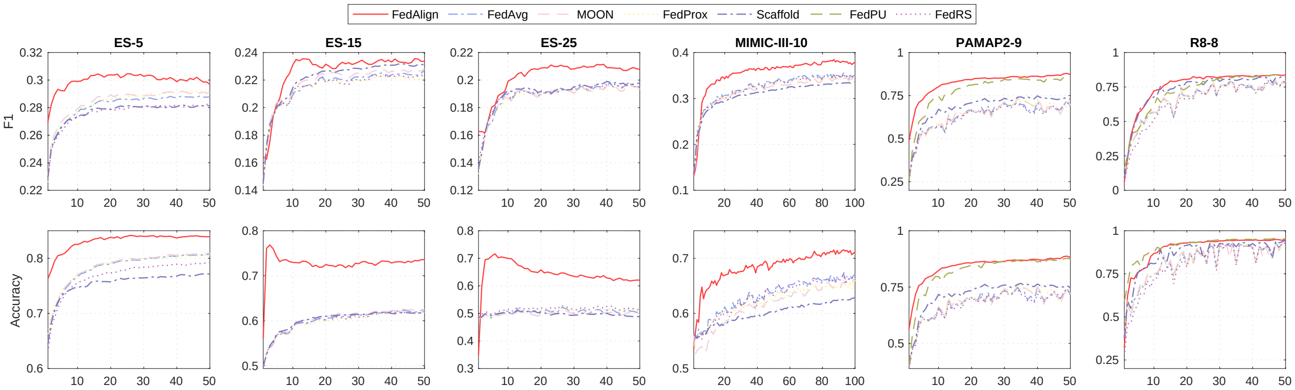


Figure 3: Performance w.r.t. Communication Rounds on Six Datasets. The results are averaged over 5 runs.

We also compare two state-of-the-art federated learning methods [22, 26] designed for classification problems with missing classes. Specifically,

- **FedRS** [22] is designed for federated learning with missing classes where each client only owns data of part of the classes (i.e., *locally-identified classes* in our terminology). It restricts the weight update of the missing classes in the classifier by adding scaling factors to the softmax operation.
- **FedPU** [26] is designed for the scenario where each client only labels a small part of their dataset and there exists unlabeled data from both locally-identified classes and locally-unaware classes. It utilizes the labeled data at each client to estimate the misclassification loss between the unaware classes of the other clients and adds the estimated loss to the local optimization objective. As the calculation of the expected risk is based on single-label classification, where each sample is associated with only one of the classes, it does not hold true in multi-label problems. While it is difficult to generalize FedPU for multi-label classification problems, we compare it with FedAlign in the single-label applications.

4.3 Experiment Setup

Base Neural Network Model. For a fair comparison, we use the same network setting for all compared methods. The data encoder is a Transformer Encoder [41]. We use one encoder layer in the data encoder, where the number of heads in multi-head attention is 4 and the dimension of the feed-forward network is 64. The label encoder is a single hidden layer neural network. The dimension of data and class representations d is 256.

Evaluation Metrics. Due to label imbalance, we adopt accuracy and F1-score to evaluate the performance. They are often used as benchmark metrics for the datasets and tasks in our experiments [32, 35, 38, 40]. We calculate the metrics for each class and report the macro-average.

Train/Test Split. For MIMIC-III and R8, we use the data split provided by the dataset. For the other datasets, we use the data from 80% of the data for training and 20% of the data for testing.

Learning Setting. For ExtraSensory (ES-5, ES-15, ES-25), PAMAP2-9, R8-8, we run $T = 50$ communication rounds. For MIMIC-III-10, we run $T = 100$ rounds as it takes longer to converge. The number of selected clients per round is $|S_t| = 5$ and the local epochs $E = 5$.

Optimizer and Hyperparameters. For compared methods, we try different values for hyperparameters μ (in FedProx and MOON)

Table 3: Results (% Averaged Over 5 Runs) of Ablation Study

Method	PAMAP2-9		R8-8	
	F1	Acc	F1	Acc
FedAvg	68.89	71.45	77.27	90.59
FedAlign w/o semantic	83.39	85.40	82.37	94.13
FedAlign w/o distillation	70.87	74.59	79.67	91.06
FedAlign w/o alternation	86.01	87.50	83.22	94.63
FedAlign	**87.21**	**88.14**	**83.76**	**94.92**

and α (in FedRS) that are often adopted in the previous papers [20–22]. For MOON, we set $\mu = 0.001$ for all datasets. For FedProx, we report the performance with $\mu = 0.001$ and $\mu = 10^{-5}$. For FedRS, we report the performance with $\alpha = 0.5$ and $\alpha = 0.9$.

4.4 Main Results and Analysis

Single-Label, Non-identical but Overlapping Client Class Sets. Table 2 shows the results. For both PAMAP2 and R8 datasets, FedAlign performs better than the baseline methods. The non-IID problems that FedRS and FedPU aim to tackle are slightly different from ours. Although they show improvements over the classical algorithm FedAvg and the methods designed for the typical non-IID setting (FedProx, MOON, and Scaffold), FedAlign shows better performance compared with FedRS and FedPU in the problem of non-identical client class sets.

Multi-Label, Non-overlapping Client Class Sets. As one can clearly see, FedAlign always yields better performance than the baseline methods. Remarkably, with non-identical client class sets, the three state-of-the-art algorithms designed to deal with non-IID data (i.e., FedProx, MOON, and Scaffold) do not guarantee improvement over FedAvg (e.g., Scaffold loses to FedAvg on ES-5). FedRS does not show improvement over its base algorithm FedAvg. Although it is designed for federated learning with missing classes, its mechanism is specifically based on single-label (softmax) classification. In multi-label (sigmoid) classification, the weight update of class c is only affected by features from class c. Adding scaling factors to missing classes does not affect the weight update of the identified classes, thus, does not solve the non-IID problem.

For the ExtraSensory dataset, as the number of partitions increases (5, 15, 25), the performance of the methods declines. This is due to two reasons: first, the label sets are more widely distributed so the training samples with identified classes on each client become fewer; second, the label space of different clients becomes more diverse, making it difficult to learn and aggregate information. However, we observe a growing trend in the improvement of testing accuracy when increasing the number of partitions. With more widely distributed classes and fewer training samples, FedAlign show greater advantages.

Performance vs. Communication Rounds. In Figure 3, we show the test performance with respect to the number of communication rounds. We see that FedAlign outperforms the baselines since a few rounds of communication. There is a clear performance gap between the baseline methods and FedAlign.

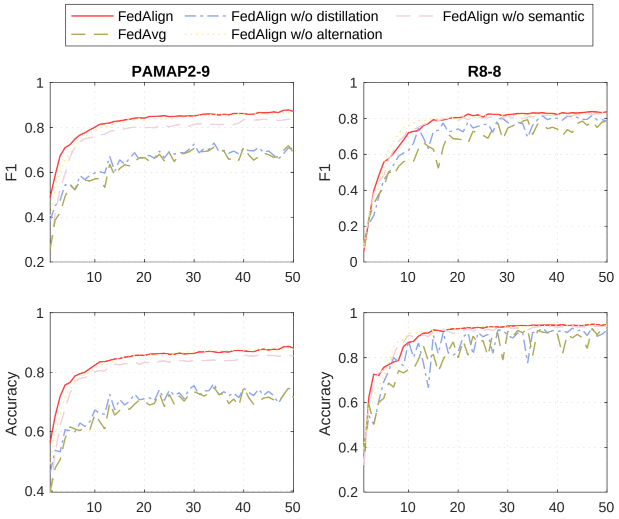


Figure 4: Performance (averaged over 5 runs) w.r.t. communication rounds in ablation study.

4.5 Ablation Study

We conduct an ablation study to evaluate the contribution of each design in FedAlign on the PAMAP and R8 datasets. First, we evaluate the performance of the method without knowledge distillation (denoted as **FedAlign w/o distillation**). The classification model consists of a label encoder and a label encoder and the framework conducts alternating training of the two modules. Second, we evaluate the performance of the method without the semantic label name sharing (denoted as **FedAlign w/o semantic**). The distillation dataset is formed by annotating the samples according to their prediction confidence given by the latest global model. Highly confident predictions are used as pseudo-labels and the confidence threshold is decided by the same percentile value as in FedAlign. Third, We evaluate the performance of the method without alternating training (denoted as **FedAlign w/o alternation**) which updates the label encoder and data encoder simultaneously. Note that the model aggregation method in FedAlign is based on FedAvg (i.e., averaging the model parameters), thus we also compare FedAvg as the baseline method.

Table 3 shows the performance at the end of 50 communication rounds and Figure 4 shows the performance with respect to the number of communication rounds. We notice the performance drops when removing any of the designs. The methods with complete features can work better than using only part of the designs. This shows the key designs (semantic label name sharing, knowledge distillation, and alternating training) in FedAlign all contribute to performance improvement, and combining them can produce the best performance.

5 RELATED WORK

We first review the important topic related to the problem we discuss, that is, federated learning with non-IID data. Then, we review two lines of related work that motivate our designs in the framework: label semantics modeling and knowledge distillation in federated learning.

5.1 Federated Learning with non-IID data

One of the fundamental challenges in federated learning is the presence of non-IID data [8]. The reasons and solutions to this challenge are being actively explored. One common solution is to mitigate client variance during local training by adding regularization [20, 21], or estimating global update directions and removing drift [9]. Another type of solution improves model aggregation at the server [25, 43, 44], such as assigning aggregation weights according to local optimization steps [44] or doing layer-wise aggregation by matching hidden elements with similar feature extraction signatures [43]. Other works propose solutions from the data perspective. They leverage public datasets to conduct distillation for model fusion [25] or calibrate classifiers using synthesized features [28, 50]. Personalization is another type of solution [16, 39], which enables different clients to have different model parameters with regard to their own data distribution. These methods tackle more relaxed non-IID problems that assume clients have the same set of classes.

We study the problem of non-identical class sets in federated learning, which is a more general case of non-IID label distribution skew. Some recent works [22, 26] consider the problem that clients can only access part of the whole class set. FedRS [22] deals with the case where each client only owns the data of certain classes. It restricted the weight update of the missing classes during the local procedure by adding a scaling factor in the softmax function. FedPU [26] focuses on a scenario where each client only labels a small part of their dataset and there exists unlabeled data from both positive classes (i.e., *identified classes* in ours) and negative (i.e., *locally-unaware* in ours) classes. It utilizes the labeled data at each client to estimate the misclassification loss between the negative classes of the other clients and adds the estimated loss to local optimization objective. The problem settings in the two related works are slightly different from ours and applying them to our problem shows less improvement compared to our proposed method.

5.2 Label Semantics Modeling

For tasks where some of the label co-occurrence patterns can not be directly observed from the training dataset, such as zero-shot learning [11], it is hard for the model to understand label correlations. To deal with the problem, several methods are proposed to leverage prior knowledge such as knowledge graphs [42] or model semantic label embedding from textual information about classes [13, 29, 36, 45, 48]. For example, Ba et al. [13] derived embedding features for classes from natural language descriptions and learned a mapping to transform the text features of classes to the visual image feature space. Radford et al [36] used contrastive pre-training to jointly train an image encoder and a text encoder and predict the correct pairings of image and text caption, which helps to produce high-quality image representations. Matsuki et al [29] and Wu et al [45] incorporate word embeddings for zero-shot learning in human activity recognition. These methods show the potential of allowing models to use semantic relationships between labels to make predictions for classes not observed in the training set, which motivates our design of semantic label name sharing.

5.3 Knowledge Distillation in Federated Learning

Knowledge distillation [5] is originally used for transferring knowledge from a large model or ensemble of models to a single smaller model. The large pre-trained models are regarded as the teachers and the smaller model (i.e., student) mimics the outputs or intermediate features of the teachers. In federated learning, knowledge distillation has been used to deal with statistical heterogeneity [50] and system heterogeneity (i.e., different client models) [18, 25]. A public dataset is usually required, or otherwise, a generator is trained to get synthetic data [50]. Recently, semi-supervised methods are also applied in federated learning for cases where labeled training data is limited [6, 24]. Our work is different from the previous works in that the non-identical class sets setting we focus on has no labeled data for the unaware classes at each client. We use the class representations to annotate data samples and add regularization in a knowledge distillation manner to transfer knowledge about unaware classes to client models.

6 CONCLUSIONS AND FUTURE WORK

We studied the problem of federated classification with non-identical class sets. We propose the FedAlign framework and demonstrated its use in federated learning for various applications. FedAlign breaks the classification model into a data encoder and a label encoder. Semantic label learning is conducted by leveraging a domain-related corpus and shared label names. The pre-trained semantic label embeddings contain the knowledge of label correlations and are then used to guide the training of data encoder. Moreover, the knowledge distillation strategy complements the training of unaware classes at each client. These two modules are a key to mitigating client variance in FedAlign, which addresses the challenge of non-identical class sets. We show that FedAlign improves the baseline algorithms for federated learning with non-IID data and achieves new state-of-the-art.

For future directions, we consider more general system heterogeneity where the participants have different network architectures, different training processes, and different tasks at the same time. We plan to extend our study to make federated learning compatible with such heterogeneity.

REFERENCES

[1] John A Bullinaria and Joseph P Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior research methods* 39, 3 (2007), 510–526.

[2] Ana Cardoso-Cachopo. 2007. Improving Methods for Single-label Text Categorization. PdD Thesis, Instituto Superior Tecnico, Universidade Tecnica de Lisboa.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[4] Felix Hamborg, Norman Meuschke, Corinna Breitinger, and Bela Gipp. 2017. news-please: A Generic News Crawler and Extractor. In *Proceedings of the 15th International Symposium of Information Science* (Berlin). 218–223. https://doi.org/10.5281/zenodo.4120316

[5] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* 2, 7 (2015).

[6] Wonyong Jeong, Jaehong Yoon, Eunho Yang, and Sung Ju Hwang. 2020. Federated semi-supervised learning with inter-client consistency & disjoint learning. *arXiv preprint arXiv:2006.12097* (2020).

[7] Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and

Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data* 3, 1 (2016), 1–9.

[8] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2021. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning* 14, 1–2 (2021), 1–210.

[9] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. 2020. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*. PMLR, 5132–5143.

[10] Stefanos Laskaridis, Dimitris Spathis, and Mario Almeida. 2021. Federated mobile sensing for activity recognition. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. 858–859.

[11] Chung-Wei Lee, Wei Fang, Chih-Kuan Yeh, and Yu-Chiang Frank Wang. 2018. Multi-label zero-shot learning with structured knowledge graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1576–1585.

[12] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* 36, 4 (2020), 1234–1240.

[13] Jimmy Lei Ba, Kevin Swersky, Sanja Fidler, et al. 2015. Predicting deep zero-shot convolutional neural networks using textual descriptions. In *Proceedings of the IEEE international conference on computer vision*. 4247–4255.

[14] Omer Levy and Yoav Goldberg. 2014. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the eighteenth conference on computational natural language learning*. 171–180.

[15] Ang Li, Jingwei Sun, Pengcheng Li, Yu Pu, Hai Li, and Yiran Chen. 2021. Hermes: an efficient federated learning framework for heterogeneous mobile clients. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. 420–437.

[16] Ang Li, Jingwei Sun, Xiao Zeng, Mi Zhang, Hai Li, and Yiran Chen. 2021. FedMask: Joint Computation and Communication-Efficient Personalized Federated Learning via Heterogeneous Masking. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. 42–55.

[17] Chenglin Li, Di Niu, Bei Jiang, Xiao Zuo, and Jianming Yang. 2021. Meta-har: Federated representation learning for human activity recognition. In *Proceedings of the Web Conference 2021*. 912–922.

[18] Daliang Li and Junpu Wang. 2019. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581* (2019).

[19] Liang Li, Dian Shi, Ronghui Hou, Hui Li, Miao Pan, and Zhu Han. 2021. To talk or to work: Flexible communication compression for energy efficient federated learning over heterogeneous mobile edge devices. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 1–10.

[20] Qinbin Li, Bingsheng He, and Dawn Song. 2021. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10713–10722.

[21] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems* 2 (2020), 429–450.

[22] Xin-Chun Li and De-Chuan Zhan. 2021. Fedrs: Federated learning with restricted softmax for label distribution non-iid data. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 995–1005.

[23] Feng Liang, Weike Pan, and Zhong Ming. 2021. Fedrec++: Lossless federated recommendation with explicit feedback. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 4224–4231.

[24] Haowen Lin, Jian Lou, Li Xiong, and Cyrus Shahabi. 2021. Semifed: Semi-supervised federated learning with consistency and pseudo-labeling. *arXiv preprint arXiv:2108.09412* (2021).

[25] Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. 2020. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems* 33 (2020), 2351–2363.

[26] Xinyang Lin, Hanting Chen, Yixing Xu, Chao Xu, Xiaolin Gui, Yiping Deng, and Yunhe Wang. 2022. Federated Learning with Positive and Unlabeled Data. In *International Conference on Machine Learning*. PMLR, 13344–13355.

[27] Shuchang Liu, Shuyuan Xu, Wenhui Yu, Zuohui Fu, Yongfeng Zhang, and Amelie Marian. 2021. FedCT: Federated collaborative transfer for recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*. 716–725.

[28] Mi Luo, Fei Chen, Dapeng Hu, Yifan Zhang, Jian Liang, and Jiashi Feng. 2021. No fear of heterogeneity: Classifier calibration for federated learning with non-iid data. *Advances in Neural Information Processing Systems* 34 (2021).

[29] Moe Matsuki, Paula Lago, and Sozo Inoue. 2019. Characterizing word embeddings for zero-shot sensor-based human activity recognition. *Sensors* 19, 22 (2019), 5043.

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

[31] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems* 26 (2013).

[32] James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable Prediction of Medical Codes from Clinical Text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 1101–1111.

[33] Bethesda (MD): National Library of Medicine. 2003. *PMC Open Access Subset [Internet]*. https://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/ [accessed 7-June-2022].

[34] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[35] Yinhua Piao, Sangseon Lee, Dohoon Lee, and Sun Kim. 2022. Sparse Structure Learning via Graph Neural Networks for Inductive Document Classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 11165–11173.

[36] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*. PMLR, 8748–8763.

[37] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. (2018).

[38] Attila Reiss and Didier Stricker. 2012. Introducing a new benchmarked dataset for activity monitoring. In *2012 16th international symposium on wearable computers*. IEEE, 108–109.

[39] Linlin Tu, Xiaomin Ouyang, Jiayu Zhou, Yuze He, and Guoliang Xing. 2021. FedDL: Federated Learning via Dynamic Layer Sharing for Human Activity Recognition. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. 15–28.

[40] Yonatan Vaizman, Katherine Ellis, and Gert Lanckriet. 2017. Recognizing detailed human context in the wild from smartphones and smartwatches. *IEEE pervasive computing* 16, 4 (2017), 62–74.

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[42] Laura Von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Julius Pfrommer, Annika Pick, Rajkumar Ramamurthy, et al. 2019. Informed Machine Learning–A Taxonomy and Survey of Integrating Knowledge into Learning Systems. *arXiv preprint arXiv:1903.12394* (2019).

[43] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. 2020. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440* (2020).

[44] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. 2020. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems* 33 (2020), 7611–7623.

[45] Tong Wu, Yiqiang Chen, Yang Gu, Jiwei Wang, Siyu Zhang, and Zhanghu Zhechen. 2020. Multi-layer cross loss model for zero-shot human activity recognition. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. 210–221.

[46] Chengxu Yang, Qipeng Wang, Mengwei Xu, Zhenpeng Chen, Kaigui Bian, Yunxin Liu, and Xuanzhe Liu. 2021. Characterizing impacts of heterogeneity in federated learning upon large-scale smartphone data. In *Proceedings of the Web Conference 2021*. 935–946.

[47] Jingwei Yi, Fangzhao Wu, Chuhan Wu, Ruixuan Liu, Guangzhong Sun, and Xing Xie. 2021. Efficient-FedRec: Efficient Federated Learning Framework for Privacy-Preserving News Recommendation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 2814–2824.

[48] Fengtao Zhou, Sheng Huang, and Yun Xing. 2021. Deep semantic dictionary learning for multi-label image classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 3572–3580.

[49] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*. 19–27.

[50] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. 2021. Data-free knowledge distillation for heterogeneous federated learning. In *International Conference on Machine Learning*. PMLR, 12878–12889.

A APPENDIX

A.1 Semantic Label Embedding Pre-training.

Before collaborative training, each client shares the natural language names of its locally-identified classes with the server. The server then searches for the label names in a large text corpus related to the domain of the classification task. We count the times that each label name c_i appears in each text segment (e.g., sentence or paragraph). The label names can be phrases that contain multiple words and the order of the words may change in different text segments while representing the same meaning, such as "colon cancer" and "cancer of the colon". To match phrases, we arrange the label names into sets of words and apply a sliding window of length L_w on each text segment. If the set of words in the label name is covered by the words in the sliding window, we mark that the label name appears in the text segment. The length of the sliding window L_w is varied per the label name to search. The co-occurrence of a pair of label names c_i and c_j is calculated as the point-wise mutual information (PMI):

$$\mathrm{PMI}(c_i, c_j) = \log \frac{p(c_i, c_j)}{p(c_i)p(c_j)},$$

where $p(c_i)$ and $p(c_j)$ are the individual distributions and $p(c_i, c_j)$ is the the joint distribution. The higher the $\mathrm{PMI}(c_i, c_j)$, the stronger the association between the two label names c_i and c_j. The server learns semantic label embeddings based on the co-occurrence. The principle of label embedding learning is to learn a mapping function from labels to representations $f : C \rightarrow \mathbb{R}^d$, which enforces labels with similar semantics to have similar representations. To achieve this, the server builds a label co-occurrence graph $G = \langle V, E \rangle$, where the nodes V represent the class label names and the edges E represent the co-occurrence relationship between the nodes. The PMI values are zero-centered by subtracting the mean and are used as the edge weights between label names. Edges with negative weights are removed from the graph. For every source node $c_o \in V$, we define $N_s(c_o) \subset V$ as its network neighborhood generated through simulating fixed-length random walks starting from c_o. The transition probability from node u to node v for random walk simulation is calculated by normalizing the edge weight $\pi_{u,v} = w_{u,v} / \sum_{v' \in V} w_{u,v'}$. The objective function of label embedding is:

$$\max_f \sum_{c_o \in C} (-\log Z_{c_o} + \sum_{c_i \in N_s(c_o)} f(c_i) \cdot f(c_o)),$$

where $Z_{c_o} = \sum_{v \in V} \exp(f(v) \cdot f(c_o))$ and is approximated using negative sampling [31]. The mapping function f is achieved by a single hidden layer feedforward neural network and the objective is optimized using stochastic gradient descent.

A.2 Text Corpus

Domain-related raw corpus is often easily accessible. For example, novel books that describe a variety of daily scenes can be used for understanding human activities. Academic journals are good studying resources for understanding concepts in different domains. We use the following corpora for applications in the experiments:

BookCorpus. [49] is a large collection of free novel books collected from the web. It is a popular text corpus in the domain of Natural Language Processing and has helped the training of many influential language models such as BERT [3] and GPT [37].

PubMed Open-Access (OA) subset. [33] is a text archive of journal articles and preprints in biomedical and life sciences. It has been widely used for biomedical text mining [12].

CommonCrawl (CC) News dataset. [4] is an English-language news corpus collecting news articles published between 2017 to 2019 from news sites worldwide.

We use PubMed-OA as the text corpus for clinic phenotype classification and BookCorpus for behavioral context recognition and human activity recognition, and CC News for text classification. We regard a sentence as a text segment in BookCorpus, an article as a text segment in CC News, and a paragraph as a text segment in PubMed-OA.

A.3 Details About Datasets and Preprocessing

ExtraSensory dataset. [40]: collects time-series sensory measurements of 60 participants in their natural behavior, using smartphones and smartwatches. Relevant context labels of the time-series measurements are annotated. There are in total 51 classes, which can be grouped into five main categories: posture (sitting, lying down, etc.), location (gym, beach, etc.), activity (eating, sleeping, etc.), companion (with friends, with co-workers). Each sample is associated with 3.6 classes on average. During data preprocessing, the time-series data of a subject is partitioned into several pieces according to its labels to ensure each piece of the resulting time-series data is in the same behavioral context from the beginning to the end. Each piece of data sample is a 10-minute time-series data. Data sample of less than 10 minutes is padded with zeros. The features at each timestamp represent the measurements taken within one minute. The feature dimension is 225.

MIMIC-III dataset. [7]: contains inpatient data of over $40,000$ patients. We follow the preprocessing instruction in [32] to derive the benchmark MIMIC-III 50-label dataset. The data of each patient is the clinical notes that record information about what happened during a patient's hospitalization. The vocabulary size of the clinical notes is 4,896. The notes has an average length of 1,512 words. The dataset contains 50 most frequent classes which can be grouped into 10 categories according to ICD, such as circulatory systems, digestive systems, etc. Each data sample is associated with 5.69 classes on average.

PAMAP2. [38]: collects sensory measurements of 9 subjects performing 18 different physical activities (e.g., sitting, ascending stairs, etc.). Data are collected by inertial measurement units (IMU) sensors and a heart rate monitor. The feature dimension is 52. We organize the data into time series with a window size of 100.

Reuters-21578 R8. [2]: is a collection of news articles including 8 classes (e.g. trade, grain, earn, etc.). For preprocessing, words with no less than 100 occurrences are kept. The vocabulary size is 534. Each piece of text data has an average length of 66 words.