# Outsourcing Training without Uploading Data via Efficient Collaborative Open-Source Sampling

**Junyuan Hong**[*]
Michigan State University
hongju12@msu.edu

**Lingjuan Lyu**
Sony AI
lingjuan.lv@sony.com

**Jiayu Zhou**
Michigan State University
jiayuz@msu.edu

**Michael Spranger**
Sony AI
michael.spranger@sony.com

## Abstract

As deep learning blooms with growing demand for computation and data resources, outsourcing model training to a powerful cloud server becomes an attractive alternative to training at a low-power and cost-effective end device. Traditional outsourcing requires uploading device data to the cloud server, which can be infeasible in many real-world applications due to the often sensitive nature of the collected data and the limited communication bandwidth. To tackle these challenges, we propose to leverage widely available *open-source data*, which is a massive dataset collected from public and heterogeneous sources (e.g., Internet images). We develop a novel strategy called Efficient Collaborative Open-source Sampling (ECOS) to construct a proximal proxy dataset from open-source data for cloud training, in lieu of client data. ECOS probes open-source data on the cloud server to sense the distribution of client data via a communication- and computation-efficient sampling process, which only communicates a few compressed public features and client scalar responses. Extensive empirical studies show that the proposed ECOS improves the quality of automated client labeling, model compression, and label outsourcing when applied in various learning scenarios.

## 1 Introduction

Nowadays, powerful machine learning services are essential in many devices that supports our daily routines. Delivering such services is typically done through client devices that are power-efficient and thus very restricted in computing capacity. The client devices can collect data through built-in sensors and make predictions by machine learning models. However, their stringent computing power often makes the local training prohibitive, especially for high-capacity deep models. One widely adopted solution is to outsource the cumbersome training to cloud servers equipped with massive computational power, using machine-learning-as-a-service (MLaaS). Amazon Sagemaker [29], Google ML Engine [6], and Microsoft Azure ML Studio [4] are among the most successful industrial adoptions, where users upload training data to designated cloud storage, and the optimized machine learning engines then handle the training. One major challenge of the outsourcing solution in many applications is that the local data collected are sensitive and protected by regulations, therefore prohibiting data sharing. Notable examples include General Data Protection Regulation (GDPR) [1] and Health Insurance Portability and Accountability Act (HIPPA) [2].

[*] Work done during internship at Sony AI. Corresponding to: Lingjuan Lyu.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

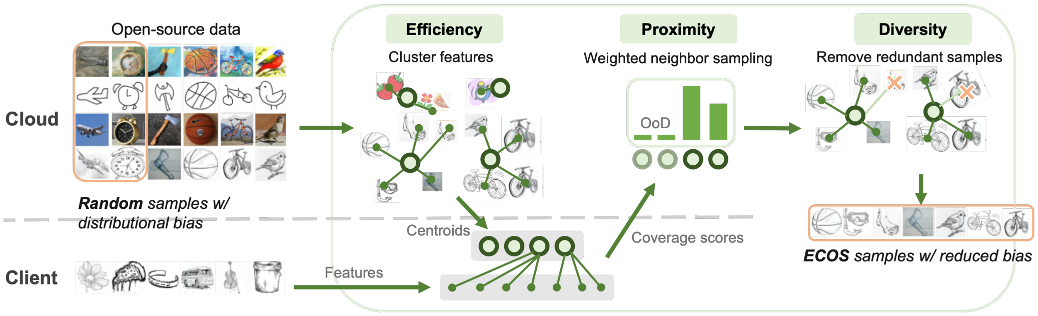


Figure 1: Illustration of the proposed ECOS framework. Instead of uploading local data for cloud training, ECOS downloads the centroids of clustered open-source features to *efficiently* sense the client distribution, where the client counts the local neighbor samples of the centroids as the coverage score. Based on the the scores of centroids, the server adaptively samples *proximal* and *diverse* data for training a transferable model on the cloud.

On the other hand, recent years witnessed a surging amount of general-purpose and massive datasets authorized for public use, such as ImageNet [15], CelebA [31], and MIMIC [26]. Moreover, many task-specific datasets used by local clients can be well considered as biased subsets of these large public datasets [41, 28]. Therefore, the availability of these datasets allows us to use them to model confidential local data, facilitating training outsourcing without directly sharing the local data. One approach is to use the private client dataset to craft pseudo labels for a public dataset in a confidential manner [56, 40], assuming that the public and local data are identically-and-independently-distributed (*iid*). In addition, Alon *et al.* showed that an *iid* public data can strongly supplement client learning, which greatly reduces the private sample complexity [3]. However, the *iid* assumption can often be too strong for general-purpose *open-source* datasets, since they are usually collected from heterogeneous sources with distributional biases from varying environments. For example, a search of 'digits' online yields digits images from handwriting scans, photos, to artwork of digits.

In this paper, we relax the *iid* assumption in training outsourcing and instead consider the availability of an open-source dataset. We first study the gap between the *iid* data and the heterogeneous open-source data in training outsourcing, and show the low sample efficiency of open-source data. We show that in order to effectively train a model from open-source data that is transferable to the client data, the open-source data needs to communicate more samples than those of *iid* data. The main reason behind such low sample efficiency is that we accidentally included out-of-distribution (OoD) samples, which poison the training and significantly degrade accuracy at the target (client) data distribution [5]. We propose a novel framework called Efficient Collaborative Open-source Sampling (ECOS) to tackle this challenge, which filters the open-source dataset through an efficient collaboration between the client and server and does not require client data to be shared. During the collaboration, the server sends compressed representative features (centroids) of the open-source dataset to the client. The client then identifies and excludes OoD centroids and returns their privately computed categorical scores to the server. The server then adaptively and diversely decompresses the neighbors of the selected centroids. The main idea is illustrated in Fig. 1.

Our major contributions are summarized as follows:

• *New problem and insight*: Motivated by the strong demands for efficient and confidential outsourcing, using public data in place of the client data is an attractive solution. However, the impact of heterogeneous sources of the public data, namely open-source data, is rarely studied in existing works. Our empirical study shows the potential challenges due to such heterogeneity.

• *New sampling paradigm*: We propose a new unified sampling paradigm, where the server only sends very few query data to the client and requests very few responses that efficiently and privately guide the cloud for various learning settings on open-source data. To our best knowledge, our method enables efficient cloud outsourcing under the most practical assumption of open-source public data, and does not require accessing raw client data or executing cumbersome local training.

• *Compelling results*: In all three practical learning scenarios, our method improves the model accuracy with pseudo, manual or pre-trained supervisions. Besides, our method shows competitive efficiency in terms of both communication and computation.

## 2 Related Work

There are a series of efforts studying how to leverage the data and computation resources on the cloud to assist client model training, especially when client data cannot be shared [53, 48]. We categorize them as follows: *1) Feature sharing*: Methods like group knowledge transfer [21], split learning [47] and domain adaptation [18, 17] transfer edge knowledge by communicating features extracted by networks. To provide a theoretical guarantee of privacy protection, [37] proposed an advanced information removal to disentangle sensitive attributes from shared features. In the notion of rigorous privacy definition, Liu *et al.* leveraged public data to assist private information release [30]. Earlier, data encryption was used for outsourcing, which however is too computation-intensive for a client and less applicable for large-scale data and deep networks [12, 27]. Federated Learning (FL) [34] considers the same constraint on data sharing but allocates the burdens of training [23] and communication [57] to clients and opens up a series of challenges on privacy [11], security [33, 10] and knowledge transfer [24]. *2) Private labeling:* PATE and its variants were proposed to generate client-approximated labels for unlabeled public data, on which a model can be trained [39, 40]. Without training multiple models by clients, Private kNN was a more efficient alternative which explored the private neighborhood of public images for labeling [56]. These approaches are based on a strong assumption of the availability of public data that is *iid* as the local data. This paper considers a more practical yet challenging setting where public data are from multiple agnostic sources with heterogeneous features.

Sampling from public data has been explored in central settings. For example, Xu *et al.* [51] used a few target-domain samples as a seed dataset to filter the open-domain datasets by positive-unlabeled learning [32]. Yan *et al.* [52] used a model to find the proxy datasets from multiple candidate datasets. In self-supervised contrastive learning, model-aware $K$-center (MAK) used a model pre-trained on the seed dataset to find desired-class samples from open-world dataset [25]. Though these methods provided effective sampling, they are less applicable when the seed dataset is placed at the low-energy edge, because the private seed data at the edge cannot be shared with the cloud for filtering and the edge device is incapable of computation-intensive training. To address these challenges, we develop a new sampling strategy requiring only light-weight computation at the edge.

## 3 Outsourcing Model Training With Open-Source Data

### 3.1 Problem Setting and Challenges

Motivated in Section 1, we aim to outsource the training process from computation-constrained devices to the powerful cloud server, where a proxy public dataset without privacy concerns is used in place of the client dataset for cloud training. One solution is (private) client labeling by k-nearest-neighbors (kNN) [56], where the client and cloud server communicate the pseudo-label of a public dataset privately and the server trains a classifier by the labeled and unlabeled samples in a semi-supervised manner. The success of this strategy depends on the key assumptions that public data in the cloud and private data in the client are *iid*, which are rather strong in practice and thus prevent it from many real-world applications. In this work, we make a more *realistic* assumption that the public datasets are as accessible as *open-source* data. An open-source dataset consists of

Table 1: Test accuracy (%) with different client domains (columns). Cloud data are identically distributed as the client data (*ID*) or including more data from 5 distinct domains (*ID+OoD*) without overlapped samples. We first label a number of randomly selected cloud examples (i.e., sampling budget) privately by client data [56], and then train a classifier to recognize digit images. The privacy cost $\epsilon$ is accounted for in the notion of differential privacy. Larger budgets imply more privacy and communication costs. More results on different settings are enclosed in Appendix B.3.

| Cloud | Sampling | MNIST | | SVHN | | USPS | | SynthDigits | | MNIST-M | | Average | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Data | Budget | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ |
| ID | 1000 | **84.3**±2.4 | 4.48 | **51.6**±1.4 | 4.08 | **87.1**±0.5 | 4.51 | **73.2**±1.5 | 4.57 | **55.5**±1.0 | 4.46 | **70.4** | 4.42 |
| ID+OoD | 1000 | 78.0±3.5 | 4.30 | 40.6±1.6 | 3.75 | 82.2±2.7 | 4.32 | 62.1±1.6 | 4.41 | 49.1±1.0 | 4.27 | 62.4 | 4.21 |
| | 8000 | 82.2±4.1 | 5.89 | 47.9±1.8 | 5.89 | 85.4±0.5 | 5.89 | 64.4±3.6 | 5.89 | 53.3±2.2 | 5.89 | 66.6 | 5.89 |
| | 16000 | 82.6±1.4 | 7.17 | 48.5±1.7 | 7.17 | 86.7±1.9 | 7.17 | 67.5±2.3 | 7.17 | 52.0±3.0 | 7.17 | 67.4 | 7.17 |
| | 32000 | 84.1±1.6 | 9.32 | 49.4±0.2 | 9.32 | 86.8±2.0 | 9.32 | 68.5±0.1 | 9.32 | 53.0±2.7 | 9.32 | 68.4 | 9.32 |

biased features from multiple heterogeneous sources (feature domains), and therefore includes not only in-distribution (ID) samples similar to the client data but also multi-domain OoD samples.

The immediate question is how the OoD samples affect the outsourced training. In Table 1, we empirically study the problem by using a 5-domain dataset, Digits, where 50% of one domain is used on the client and the remained 50% together with the other 4 domains serve as the public dataset on the cloud. To conduct the cloud training, we leverage the client data to generate pseudo labels for the unlabeled public samples, following [56]. It turns out that the presence of OoD samples in the cloud greatly degrades the model accuracy. The inherent reason for the degradation is that the distributional shift of data [43] compromised the transferability of the model to the client data [50].

**Problem formulation by sampling principles**. Given a client dataset $D^p$ and an open-source dataset $D^q$, the goal of open-source sampling is to find a proper subset $S$ from $D^q$, whose distribution matches $D^p$. In [25], Model-Aware K-center (MAK) formulated the sampling as a principled optimization:

$$\min_{S \subseteq D^q} \Delta(S, D^p) - H(S \cup D^p; D^q), \tag{1}$$

where $\Delta(S, D^p) := \mathbb{E}_{x' \in D^p}[\min_{x \in S} \|\phi(x) - \phi(x')\|^2]$ measures *proximity* as the set difference between $S$ and $D^p$ using a feature extractor $\phi$, and the latter $H(S \cup D^p; D^q) := \max_{x' \in D^q} \min_{x \in S \cup D^p} \|\phi(x) - \phi(x')\|^2$ measures *diversity* by contradicting $S \cup D^p$ and $D^q$ (suppose $D^q$ is the most diverse set)[2]. Solving Eq. (1) results in an NP-hard problem that is intractable [13], and MAK provides an approximated solution by a coordinate-wise greedy strategy. It first pre-trains the model representations on $D^p$ and finds a large candidate set with the best proximity to extracted features. Then, it incrementally selects the most diverse samples from the candidate set until the sampling budget is used up.

Though MAK is successful in the central setting, it is not applicable when $D^p$ is isolated from cloud open-source data and is located at a resource-constrained client for two reasons: 1) *Communication inefficiency*. Uploading client data may result in privacy leakage, sending public data to the client is a direct alternative but the cost can be prohibitive. 2) *Computation inefficiency*. Pre-training a model on $D^p$ or proximal sampling (which computes the distances between paired samples from $D^q$ and $D^p$) induces unaffordable computation overheads for the low-energy client.

### 3.2 Proposed Solution: Efficient Collaborative Open-Source Sampling (ECOS)

To address the above challenges, we design a new strategy that 1) uses compressed queries to reduce the communication and computation overhead and 2) uses a novel principled objective to effectively *sample* from open-source data with the client responses of the compressed queries.

**Construct communication-efficient and an informative query set $\hat{\Phi}^q$ at cloud**. Let $d$ be the number of pixels of an image, the communication overhead of transmitting $D^q$ to the client is given by $\mathcal{O}(d|D^q|)$. For communication efficiency, we optimize the following two factors:
i) *Data dimension $d$*. First, we transmit extracted features $\Phi^q = \{\phi(x) | x \in D^q\}$ instead of images to reduce the communication overhead to $\mathcal{O}(d_e|D^q|)$, where $d_e$ is a much smaller embedding dimension. For accurate estimation of the distance $\Delta$, a pre-defined discriminative feature space is essential without extra training on the client. Depending on resources, one may consider hand-crafted features such as HOG [14], or an off-the-shelf pre-trained model such as ResNet pre-trained on ImageNet.
ii) *Data size $|D^q|$*. Even with compression, sending all data for querying is inefficient due to the huge size of open-source data $|D^q|$. Meanwhile, too many queries would cast unacceptable privacy costs to the client. As querying on similar samples leads to redundant information in querying, we propose to reduce such redundancy by selecting informative samples. We use the classic clustering method KMeans [20] for compressing similar samples by clustering them, and collect the $R$ mean vectors or *centroids* into $\hat{\Phi}^q = \{c_r\}_{r=1}^R$. We denote $R$ as the *compression size* and $\hat{D}^q$ as the set of original samples corresponding to $\hat{\Phi}^q$.

**New sampling objective**. We note that sending the compact set $\hat{\Phi}^q$ in place of $D^q$ prohibits the client from optimizing $\Delta(S, D^p)$ in Eq. (1) for $S \in D^q$. Instead, we sample a set of centroids $\hat{S} \in \hat{\Phi}^q$ and decompress them by the cluster assignment into corresponding original samples with rich features

[2] Note that we use $L_2$-norm distance instead of normalized cosine similarity in $\Delta(S, D^p)$ in contrast to MAK, since normalized cosine similarity is not essential if the feature space is not trained under the cosine metric. We also omit the tailedness objective which is irrelevant in our context.

afterwards. In principle, we leverage the inequality $\Delta(S, D^p) \leq \Delta(\hat{S}, D^p) + \Delta(\hat{S}, S)$ to attain a communication-efficient surrogate objective as follows:

$$\min_{\hat{S}\subseteq\hat{D}^q, S\subseteq D^q} \underbrace{\Delta(\hat{S}, D^p) + \Delta(\hat{S}, S)}_{\text{proximity}} - \underbrace{H(S; D^q)}_{\text{diversity}}, \tag{2}$$

where $\hat{S}$ (or $\hat{D}^q$) is the compressed centroid substitute of $S$ (or $D^q$). Different from Eq. (1), we decompose the proximity term into two in order to facilitate communication efficiency leveraging an informative subset $\hat{D}^q$. We solve the optimization problem in a greedy manner by two steps at the client and the cloud, respectively:

i) At the *client step*, we optimize $\Delta(\hat{S}, D^p)$ to find a subset of centroids ($\hat{S} \subset \hat{D}^q$) that are proximal to the client set $D^p$. Noticing that $\hat{D}^q$ contains the cluster centroids, we take advantage of the property to define a novel proximity measure of the cluster $r$: Centroid Coverage (CC), denoted as $v_r$. Upon receiving centroids from the cloud, the client uses them to partition the local data into $\{\mathcal{C}_r^p\}_{r=1}^R$ where $\mathcal{C}_r^p$ denotes the $r$-th cluster partition of local data. We compute the CC score by the cardinality of the neighbor samples of the centroid $r$, i.e., $v_r = |\mathcal{C}_r^p|$. To augment the sensitivity to the proximal clusters, we scale the CC score by a non-linear function $v'_r = \psi_s(|\mathcal{C}_r^p|)$, where the scale function $\psi_s(x) = x^s$ is parameterized by $s$.

ii) At the *cloud step*, we optimize the proximity of $S$ w.r.t. the proxy set $\hat{S}$, i.e., $\Delta(\hat{S}, S)$, and remove redundant and irrelevant samples from the candidate set to encourage diversity, i.e., $-H(S; D^q)$. As samples among clusters are already diversified by KMeans, we only need to promote the in-cluster *diversity*. To this end, we reduce the sample redundancy per cluster at cloud by K-Center [44], which heuristically finds the most diverse samples. Such design transfers the diversity operation to cloud and thus reduces the local computation overhead. To maintain the *proximity*, the K-Center is applied within each cloud cluster and the sampling budget per cluster is proportional to their vote numbers and the original cluster sizes. With the normalized scores, we compute the sampling budget per cluster which is upper bounded by the ratio of the cluster in the cloud set.

**Algorithm 1** Efficient collaborative open-source sampling (ECOS)

```
Input: Client dataset $D^p$, cloud query dataset $D^q$, sampling budget $B$, compression size $R$, feature
    extractor $\phi(\cdot)$, distance function $\Delta(x, S) = \min_{y\in S} \|\phi(x) - \phi(y)\|$, initial sample set $S = \emptyset$,
    score scale function $\psi_s(x) = x^s$.
Extract features $\Phi^q = \{\phi(x) | x \in D^q\}$;
Cloud creates a compressed dataset $\hat{\Phi}^q = \mathrm{KMeans}_R(\Phi^q)$;          ▷ Compress $R$ Centroids
▷ ▷ ▷ Client End ▷ ▷ ▷
Download the feature extractor $\phi$ and $\hat{\Phi}^q$;
Use centroids $\hat{\Phi}^q$ to partition $\Phi^p = \{\phi(x) | x \in D^p\}$ into clusters $\{\mathcal{C}_r^p\}_{r=1}^R$;
Compute the Centroid Coverage (CC) scores: $v_r = |\mathcal{C}_r^p|, \forall r \in \{1, \dots, R\}$;
Upload scaled cluster scores $\{v'_r = \psi_s(v_r)\}_{r=1}^R$;          ▷ Proximity
◁ ◁ ◁ Cloud End ◁ ◁ ◁
Partition $D^q$ into clusters $\{\mathcal{C}_r^q\}_{r=1}^R$ by centroids $\hat{\Phi}^q$;
Compute per-cluster sampling budget $b_r = \min\left\{\frac{|\mathcal{C}_r^q|}{\sum_j |\mathcal{C}_j^q|}, \frac{v'_r}{\sum_j v'_j}\right\} \cdot B$;
for $r$ in $\{1, \dots, R\}$ do          ▷ Decompress Centroids
    Initialize $S' = \{x\}$ by a sample randomly picked from $\mathcal{C}_r$;
    while $|S'| < b_r$ do          ▷ Diverse Sampling
        $u = \arg\max_{x\in\mathcal{C}_r^q} \Delta(x, S')$;
        $S' = \{u\} \cup S'$;
    $S = S' \cup S$;
return $S$
```

We summarize our algorithm in Algorithm 1, which enables the clients to enjoy better computation efficiency than local training and better communication efficiency than the centralized sampling (e.g., MAK). **1) Computation efficiency**. Since our method only requires inference operations on the client device, which should be efficiently designed for the standard predictive functions of the device, and is training-free for the client, the major complexity of ECOS is on computing centroid coverage and is much lower than gradient-based algorithms whose complexity scales with the model size and training iterations. As computing the CC scores only requires the nearest centroid estimation and ranking, the filtering can be efficiently done. The total *time complexity* is $\mathcal{O}(C_\phi|D^p|+(d_e+1)R|D^p|)$, dominated by the first term, where $C_\phi$ is the complexity of extracting features depending on the specific method. The second term $(d_e+1)R|D^p|$ is for computing the pair-wise distances between $\Phi^p$ and $\hat{\Phi}^q$ and estimating the nearest centroids per sample (or partitioning client data). In a brief comparison, the complexity of local $T$-iteration gradient-descent training could be approximately $\mathcal{O}(TC_\phi|D^p|)$ which is much more expensive since typically $TC_\phi \gg d_e$. To complete the analysis, the *space complexity* is $\mathcal{O}(C'_\phi + (d+d_e)|D^p| + d_eR + |D^p|R)$ for the memory footprint of $\phi$, the images $(d)$ or features $(d_e)$ of client and cloud centroid data, and the distance matrix. **2) Communication efficiency**. The downloading complexity will be $\mathcal{O}(d_eR)$ for $R$ $d_e$-dimensional centroid features and the uploading complexity is $\mathcal{O}(R)$ including indexes of samples. Thus, the data that will be communicated between the client and cloud is approximately of $\mathcal{O}(d_eR+R)$ complexity in total. In comparison, downloading the whole open-source dataset by central sampling (e.g., MAK) requires $\mathcal{O}(d|D^q|+B)$ complexity. As $R < d_eR \ll d_e|D^q| \ll d|D^q|$, our method can significantly scale down the computation cost.

**Privacy protection and accountant**. When the cloud server is compromised by an attacker, uploading CC scores leak private information of the local data samples, for example, the presence of an identity [45]. To mitigate the privacy risk, we protect the uploaded scores by a Gaussian noise mechanism, i.e., $\tilde{v}_r = v_r + \mathcal{N}(0,\sigma^2)$, and account for the privacy cost in the notion of differential privacy (DP) [19]. DP quantifies the numerical influence of the absence of a private sample on $[v_1,\cdots,v_R]$, which is connected to the chance of exposing the sample to the attacker. To obtain a tight bound on the privacy cost, we utilize the tool of Rényi Differential Privacy (RDP) [36] and leverage the Poisson sampling to further amplify the privacy [55]. With the noise mechanism governed by $\sigma$, the resultant privacy cost in the sense of $(\epsilon,\delta)$-DP can be accounted as $\epsilon = \mathcal{O}(\gamma\sqrt{\log(1/\delta)}/\sigma)$ where $\gamma$ is the Poisson subsampling rate and $\delta$ is a user-specified parameter. A larger $\epsilon$ implies higher risks of privacy leakage in the probability of $\delta$. Formal proofs can be found in Appendix A.3.

**Generalization error of models trained on cloud.** Our work can be viewed as knowledge transfer from the open-source domain to the private client domain. Therefore, we present Theorem 3.1 based on prior domain-adaptation theoretical results [7, 18, 17].

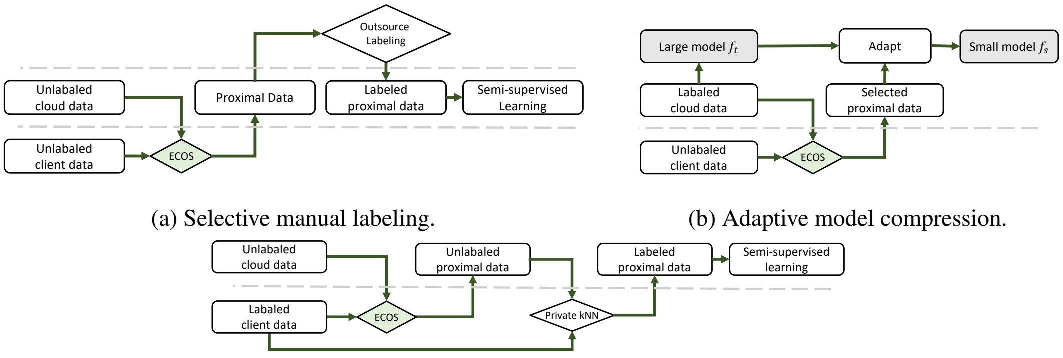
**Theorem 3.1.** *Assume that a open-source dataset $S$ is induced from a mixture of cluster distributions, i.e., $\sum_{r=1}^R \alpha_r \mathcal{D}_r^q$ with cluster distribution $\mathcal{D}_r^q$, $\alpha_r \in [0,1]$ and $\sum_{r=1}^R \alpha_r = 1$. Suppose client data are sampled from $\mathcal{D}^p$. Let $L(\cdot,\cdot)$ be a loss function on a hypothesis and a dataset (for empirical error) or a distribution (for generalization error). If $f$ is governed by the parameter $\theta$ trained on $S$ and belongs to a hypothesis space $\mathcal{H}$ of $VC$-dimension d, then with probability at least $1-p$ over the choice of samples, the inequality holds,*

$$L(f_\theta,\mathcal{D}^p) \le L(f_\theta,S) + \frac{1}{2}\sum\nolimits_{r=1}^{R} \alpha_r d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_r^q,\mathcal{D}^p) + 4\sqrt{\frac{2d\log(2|S|)+\log(4/p)}{|S|}} + \xi, \quad (3)$$

*where $\xi = \min_\theta\{L(f_\theta,S) + L(f_\theta,D^p)\}$, and $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D},\mathcal{D}')$ denotes the distribution divergence.*

The proof of Theorem 3.1 is deferred to Appendix A.4. Theorem 3.1 shows that given a model trained on cloud, its generalization error on client data hinges on the quality and size of the ECOS-sampled subset. Informally, if the sampled set has enough samples (the 3rd term) and follows a similar distribution as the client dataset (the 2nd term), then the model generalizes better via only training on the proxy cloud dataset (the 1st term). In Theorem 3.1, the model $f_\theta$ can be trained by any task-specific optimization method. For example, minimizing a $\mu$-strongly-convex and $G$-smooth loss function $L(f_\theta,S)$ w.r.t. $\theta$ via $T$-iteration gradient descent leads to $L(f_\theta,S) = (1-\mu/G)^T|L(f_{\theta_0},S) - L^*|$ where $L^*$ is the optimal loss.

Given Eq. (3), it can be shown that minimizing the upper bound requires a dedicated trade-off between sample size and quality. Consider a simple case, only minimizing $\sum_{r=1}^R \alpha_r d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_r^q,\mathcal{D}^p)$. The solution is that $\alpha_r$ equals $1$ if $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_r^q,\mathcal{D}^p)$ is smallest among all choices of clusters, otherwise zero. However, the number of samples in the cluster $r$ is limited as $|\mathcal{C}_r^q|$ which is much smaller than the whole open-source dataset $|D^q|$, and therefore $|S|$ will be too small to enlarge the third term in Eq. (3). Thus, $\alpha_r$ should be smoothed to increase sample size by trading in some quality (distribution divergence). In our implementation, we approximate $\alpha_r$ via the CC score $v_r^s$, where an $s < \infty$ smooths the $\alpha_r$ to trade off sample size and quality. When $s$ vanishes, all clusters will be selected at the same chance and the divergence could be large.



(a) Selective manual labeling.

(b) Adaptive model compression.

(c) Automated client labeling.

Figure 2: Our method is applicable to various cloud training cases, where ECOS filters the open-source samples to improve the model performance trained on (a) manual, (b) pre-trained model (teacher), and (c) pseudo supervisions.

# 4 Empirical Results

**Datasets.** We use datasets from two tasks: digit recognition and object recognition. Distinct from prior work [56], in our work, the open-source data contains samples out of the client's distribution. With the same classes as the client dataset, we assume open-source data are from different environments and therefore include different feature distributions, for example, DomainNet [41] and Digits [28]. DomainNet includes large-sized $244 \times 244$ everyday images on 6 domains and Digits consists of $28 \times 28$ digit images on 5 domains. Instead of using an overly large volume of data from a single domain like [56], we tailor each-domain subset to contain fewer images than standard digit datasets, for instance, MNIST with 7438 images, which was previously adopted in the distributed learning setting [28] and mitigates the hardness of collecting enormous data. In practice, collecting tens of thousands of images by a single client, e.g., 50000 images from MNIST domain, will be unrealistic. Similarly, DomainNet will be tailored to only include 10 classes with 2000-5000 images per domain.
**Splits of client and cloud datasets**. For Digits, we use one domain for the client and the rest domains for the cloud as open-source set. For DomainNet, we randomly select 50% samples from one domain for the client and leave the rest samples together with all other domains to the cloud. The difference of configurations for the two datasets is caused by their different domain gaps. Even without ID data, it is possible for Digits to transfer the knowledge across domains.
**Baselines.** For a fair comparison, we compare our method to baselines with the same sampling budget. Each experiment case is repeated for three times with seeds $\{1, 2, 3\}$. We account for the privacy cost by Poisson-subsampling RDP [55] and translate the cost to the general privacy notion, $(\epsilon, \delta)$-DP when $\delta = 10^{-5}$. Here, we use the *random* sampling as a naive baseline. We also adopt a coreset selection method, K-Center [44], to select informative samples within the limited budget. Both baselines are perfectly private, because they do not access private information from clients. Details of hyper-parameters are deferred to Appendix B.1.

## 4.1 Evaluations on Training Outsourcing

To demonstrate the general applicability of ECOS, we present three practical use cases of outsourcing, categorized by the form of supervisions. The conceptual illustrations are in Fig. 2. Per use case, we train a model on partially labeled cloud data (outsourced training) and accuracy on the client test set is reported together with standard deviations. We present the results in Tables 2 to 4 case by case, where we vary the domain of the client by columns. In each column, we highlight the best result unless the difference is not statistically significant.

**Case 1: Selective manual labeling**. We assume that the cloud will label the filtered in-domain samples by using a third-party label service, e.g., Amazon Mechanical Turk [38], or by asking the end clients for manual labeling. As the selected samples are non-private, they can be freely shared with a third parity. But the high cost of manual labeling or service is the pain point, which should be carefully constrained within a finite *budget* of demanded labels.

Table 2: Test accuracy by selective labeling on Digits (top) and DomainNet (bottom).

| Sampling | | MNIST | | SVHN | | USPS | | SynthDigits | | MNIST-M | | Average |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Budget | Method | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ |
| | Ours | **97.3**±0.1 | 0.22 | **68.7**±0.3 | 0.22 | 90.8±0.1 | 0.22 | **84.4**±0.6 | 0.22 | 70.4±0.6 | 0.22 | **82.3** |
| 2000 | K-Center | 96.7±0.3 | 0.00 | 65.1±1.3 | 0.00 | 90.1±0.7 | 0.00 | 80.2±1.1 | 0.00 | 70.1±0.3 | 0.00 | 80.4 |
| | Random | 96.5±0.3 | 0.00 | 64.0±0.3 | 0.00 | 91.6±1.0 | 0.00 | 83.8±0.3 | 0.00 | 70.9±0.6 | 0.00 | 81.4 |
| | Ours | **98.1**±0.2 | 0.22 | **74.6**±1.0 | 0.22 | **93.5**±0.3 | 0.22 | **91.2**±0.4 | 0.22 | 74.5±0.5 | 0.22 | **86.4** |
| 5000 | K-Center | 97.9±0.2 | 0.00 | 72.3±0.6 | 0.00 | 92.7±0.9 | 0.00 | 89.6±0.3 | 0.00 | 74.0±0.5 | 0.00 | 85.3 |
| | Random | 97.6±0.3 | 0.00 | 70.0±0.3 | 0.00 | 93.0±0.6 | 0.00 | 89.7±0.4 | 0.00 | 73.9±0.7 | 0.00 | 84.8 |
| - | Local | 99.1±0.1 | 0.00 | 88.8±0.2 | 0.00 | 98.9±0.1 | 0.00 | 96.4±0.2 | 0.00 | 88.8±0.2 | 0.00 | 94.4 |

| Sampling | | Clipart | | Infograph | | Painting | | Quickdraw | | Real | | Sketch | | Average |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Budget | Method | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ |
| | Ours | **88.4**±1.5 | 0.58 | **52.6**±0.9 | 0.58 | **90.4**±1.7 | 0.58 | **84.3**±1.6 | 0.58 | 92.1±1.2 | 0.58 | **87.2**±0.5 | 0.58 | **82.5** |
| 1000 | K-Center | 86.8±0.3 | 0.00 | 50.5±0.9 | 0.00 | 89.1±1.4 | 0.00 | 27.2±1.8 | 0.00 | **92.5**±0.1 | 0.00 | 85.6±1.4 | 0.00 | 72.0 |
| | Random | 86.9±0.8 | 0.00 | 47.4±2.7 | 0.00 | 88.6±0.1 | 0.00 | 77.9±2.4 | 0.00 | 91.4±0.3 | 0.00 | 86.9±0.5 | 0.00 | 79.9 |
| | Ours | 93.2±0.4 | 0.58 | **58.1**±0.6 | 0.58 | 92.5±1.1 | 0.58 | **89.2**±0.9 | 0.58 | **94.4**±0.2 | 0.58 | 92.8±0.2 | 0.58 | **86.7** |
| 3000 | K-Center | 93.5±1.1 | 0.00 | 56.3±0.3 | 0.00 | **92.9**±0.3 | 0.00 | 60.5±8.7 | 0.00 | 94.1±0.2 | 0.00 | 92.1±0.7 | 0.00 | 81.6 |
| | Random | 92.5±0.6 | 0.00 | 53.6±1.4 | 0.00 | 91.7±0.8 | 0.00 | 86.1±0.4 | 0.00 | 93.5±0.3 | 0.00 | 93.0±0.2 | 0.00 | 85.1 |
| - | Local | 87.0±0.3 | 0.00 | 51.7±0.7 | 0.00 | 85.9±0.4 | 0.00 | 83.5±0.3 | 0.00 | 93.5±0.1 | 0.00 | 81.7±0.6 | 0.00 | 80.6 |

Given a specified sampling budget, we compare the test accuracy of semi-supervised learning (FixMatch) on sampled data in Table 2. Since the ECOS tends to select in-distribution samples, it eases the transfer of cloud-trained model to the client data. On the Digits dataset, we find that our method attains more accuracy gains as budget increases, demonstrating that more effective labels are selected. On the DomainNet dataset, our method outperforms baselines on 5 out of 6 domains and is stable in most domains given a small budget (1000) and is superior on average. Given higher budgets, the accuracy of all methods are improved, and our method is outstanding on Infograph and Quickdraw domains and is comparable to the baselines on other domains.

Table 3: Test accuracy by adaptive model compression on DomainNet.

| Sampling | | Clipart | | Infograph | | Painting | | Quickdraw | | Real | | Sketch | | Average |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Budget | Method | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ | $\epsilon$ ↓ | Acc (%) ↑ |
| | Ours | 82.9±0.7 | 0.58 | **48.5**±2.7 | 0.58 | **85.4**±1.0 | 0.58 | **81.3**±2.5 | 0.58 | 91.4±0.6 | 0.58 | **82.4**±1.1 | 0.58 | **78.6** |
| 1000 | K-Center | 81.2±0.9 | 0.00 | 44.6±0.9 | 0.00 | 84.5±2.2 | 0.00 | 41.7±1.9 | 0.00 | **92.4**±0.5 | 0.00 | 80.1±2.1 | 0.00 | 70.8 |
| | Random | **83.8**±0.7 | 0.00 | 44.4±2.1 | 0.00 | 83.8±1.6 | 0.00 | 76.3±2.2 | 0.00 | 90.1±0.6 | 0.00 | 80.1±0.7 | 0.00 | 76.4 |
| | Ours | **90.6**±0.6 | 0.58 | **51.4**±1.9 | 0.58 | **89.6**±1.2 | 0.58 | **87.6**±0.3 | 0.58 | 93.6±0.7 | 0.58 | **88.4**±1.5 | 0.58 | **83.5** |
| 3000 | K-Center | 88.9±2.3 | 0.00 | 51.2±0.4 | 0.00 | 89.4±0.9 | 0.00 | 57.6±4.4 | 0.00 | **94.5**±0.5 | 0.00 | 86.9±0.6 | 0.00 | 78.1 |
| | Random | 88.4±0.8 | 0.00 | 47.6±1.9 | 0.00 | 89.4±1.0 | 0.00 | 84.7±0.2 | 0.00 | 93.0±1.0 | 0.00 | 86.3±1.2 | 0.00 | 81.6 |

**Case 2: Adaptive model compression**. Due to the large volume of the open-source dataset, a larger model is desired for better capturing the various features, which however is so inefficient to fit into the resource-constrained client devices or specialize for the data distribution of the client. Confronting this challenge, model compression [9] is a conventional idea to forge a memory-efficient model by transferring knowledge from large models to small ones. Specifically, we first pre-train a large *teacher* model $f_t$ on all cloud data by the supervised learning, assuming labels are available in advance. Still, we use an ImageNet-pre-trained model to initialize the feature extractor $\phi(\cdot)$ of a *student* model $f_s$. Then the client will use the downloaded feature extractor $\phi$ to filter samples. Here, we utilize knowledge distillation [22] to finetune the $f_s$ with an additional classifier head upon the $\phi$. On the selected samples, we train a linear classifier head for 30 epochs under the supervision of true labels and the teacher model $f_t$, and then fine-tune the full network $f_s$ for 500 epochs. The major challenge comes from distributional biases between the multi-source open-source data and the client data. Leveraging ECOS, we may sample data near the client distribution and reduce the bias in the follow-up compression process.

We simulate the case on the large-sized image dataset, DomainNet, which is demanding for large-scale networks, e.g., ResNet50, to effectively learn the complicated features. Here, we compress ResNet50 into a smaller network, ResNet18, by using an adaptively selected subset of the cloud dataset. We omit the experiment for digit images where a large model may not be necessary for such a small image size. In Table 3, we present the test accuracy of compressed ResNet18 using 3000 samples from DomainNet in finetuning. With a small portion of privacy cost ($\epsilon < 0.6$), our method improves the accuracy on Clipart and Real domains against the baselines. Note that the model accuracy here is lower than label outsourcing in Table 2, and reason is that the supervisions from the larger models are just an approximation of the full dataset. Without using the full dataset for compression, the training can be completed fast and responsively on the demand of a client.

Table 4: Test accuracy of client labeling on two datasets: Digits (top) and DomainNet (bottom).

| Budget | Sampling<br>Method | MNIST<br>Acc (%) ↑ | $\epsilon \downarrow$ | SVHN<br>Acc (%) ↑ | $\epsilon \downarrow$ | USPS<br>Acc (%) ↑ | $\epsilon \downarrow$ | SynthDigits<br>Acc (%) ↑ | $\epsilon \downarrow$ | MNIST-M<br>Acc (%) ↑ | $\epsilon \downarrow$ | Average<br>Acc (%) ↑ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5000 | Ours | **84.2**±2.3 | 5.35 | 47.9±3.1 | 5.32 | **86.1**±1.0 | 5.35 | 68.6±1.6 | 5.35 | **58.4**±1.9 | 5.35 | **69.0** |
| | K-Center | 81.9±3.4 | 5.34 | 48.4±1.2 | 5.33 | 82.1±1.2 | 5.34 | 69.4±1.9 | 5.34 | 55.4±2.0 | 5.34 | 67.4 |
| | Random | 81.8±4.1 | 5.34 | 45.3±3.0 | 5.29 | 81.2±2.3 | 5.34 | 65.9±2.7 | 5.34 | 55.5±2.6 | 5.34 | 65.9 |

| Budget | Sampling<br>Method | Clipart<br>Acc (%) ↑ | $\epsilon \downarrow$ | Infograph<br>Acc (%) ↑ | $\epsilon \downarrow$ | Painting<br>Acc (%) ↑ | $\epsilon \downarrow$ | Quickdraw<br>Acc (%) ↑ | $\epsilon \downarrow$ | Real<br>Acc (%) ↑ | $\epsilon \downarrow$ | Sketch<br>Acc (%) ↑ | $\epsilon \downarrow$ | Average<br>Acc (%) ↑ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3000 | Ours | 33.2±5.9 | 4.46 | **23.8**±2.2 | 3.50 | **47.4**±3.3 | 4.51 | **39.8**±7.7 | 4.87 | 62.9±1.9 | 4.92 | **51.7**±1.2 | 4.94 | **43.2** |
| | K-Center | **39.3**±3.6 | 4.57 | 18.2±2.7 | 3.61 | 46.6±2.5 | 4.52 | 36.1±2.9 | 4.94 | **63.8**±1.9 | 4.96 | 47.7±2.8 | 4.96 | 42.0 |
| | Random | 30.7±2.2 | 4.41 | 21.6±4.9 | 3.43 | 44.0±3.1 | 4.53 | 39.0±5.6 | 4.82 | 59.9±3.6 | 4.75 | 47.2±3.4 | 4.94 | 40.4 |

**Case 3: Automated client labeling**. When the client obtained labeled samples, for example, photos labeled by phone users, the cloud only needs to collect an unlabeled public dataset. Therefore, we may automate the labeling process leveraging the client supervision knowledge to reduce cost or users' efforts on manual labeling. To be specific, we let the client generate pseudo labels for the cloud data based on their neighbor relation, as previously studied by [56] (private kNN). However, the private kNN assumes that the client and the cloud follow the same distribution, which weakens its applicability confronting the heterogeneity and the large scale of open-source data. Thus, we utilize ECOS as a pre-processing of open-source data before being labeled by private kNN. Therefore, we have two rounds of communication for transferring client knowledge: proximal-data sampling by the ECOS and client labeling by the private kNN [56]. To compose the privacy costs from these two steps, we utilize the analytical moment accountant technique to get a tight privacy bound [49]. Interested readers can refer to Appendix A.1 for a brief introduction to private kNN and our implementations.

In experiments, we use the state-of-the-art private pseudo-labeling method, private kNN [56], to label the subsampled open-source data with the assistance from the labeled client dataset. To reduce the sensitivity of private kNN w.r.t. the threshold, we instead enforce the number of the selected labels to be 600 and balance the sizes by selecting the top-60 samples with the highest confidence per class. Then, we adopt the popular semi-supervised learning method, FixMatch [46], to train the classifier. In Table 4, we report the results when cloud features are distributionally biased from the client ones but they share the same class set. For each domain choice of client data, we will use the other domains as the cloud dataset. Distinct from prior studies, e.g., in [56] or [40], we assume 80-90% of the cloud data are out of the client distribution and are heterogeneously aggregated from different domains, casting greater challenges in learning. On the Digits, we eliminate all ID data from the cloud set to harden the task. Both on Digits and DomainNet datasets, our method consistently outperforms the two sampling baselines under the similar privacy costs. The variance of privacy costs is mainly resulted from the actual sampling sizes. Though simply adopting K-Center outperforms the random sampling, it still presents larger gaps compared to our method in multiple domains. For instance, in Quickdraw domain, given 3000 sampling budget, the K-Center method performs poorly and is even worse than a random sampling.

## 4.2 Qualitative Study

To better understand the proposed method, we conduct a series of qualitative studies on client labeling. We use two DomainNet datasets as a representative benchmark in the studies.

Table 5: Ablation study of the proposed method on DomainNet. Test accuracy of the client labeling case is reported.

| Proximity | Diversity | Clipart | Infograph | Painting | Quickdraw | Real | Sketch | Average |
|---|---|---|---|---|---|---|---|---|
| ✗ | ✗ | 30.7±2.2 | 21.6±4.9 | 44.0±3.1 | 39.0±5.6 | 59.9±3.6 | 47.2±3.4 | 40.4 |
| ✓ | ✗ | 25.5 ±5.7 | 21.1±0.5 | 41.4±5.3 | 31.3±1.3 | 60.3±2.3 | 31.9±2.9 | 35.2 |
| ✗ | ✓ | **39.3**±3.6 | 18.2±2.7 | 46.6±2.5 | 36.1±2.9 | **63.8**±1.9 | 47.7±2.8 | 42.0 |
| ✓ | ✓ | 33.2±5.9 | **23.8**±2.2 | **47.4**±3.3 | **39.8**±7.7 | **62.9**±1.9 | **51.7**±1.2 | **43.2** |

1) **Ablation study**. In Table 5, we conduct ablation studies to evaluate the effect of different objectives introduced by ECOS, following the client labeling benchmark with a 3000 sample budget. Without proximity and diversity objectives, we let the baseline be the naive random sampling. We first include the proximity objective, where we greedily select samples from top-scored clusters until the budget is fulfilled. However, we find that the naive proximity objective results in a quite negative effect compared to the random baseline. The failure can be attributed to the nature of clustering, which includes more similar samples and thereby lacks diversity. When diversity is encouraged and combined with the proximal votes, we find the performance is improved significantly in multiple domains and on average. Now with diversity, the proximity objective can further improve the sampling in Painting, Quickdraw, and Sketch domains significantly. 2) **Visualize cluster selection**. In Fig. 3, we demonstrate that the CC can effectively reject OoD centroids. Also, it is interesting to observe that when multiple centroids are distributed closely, then they will compete with each other and reject the redundant ones consequently. 3) **Efficiency**. In Section 3, we studied the communication efficiency theoretically. Empirically, ECOS only needs to upload 100 bytes of the CC scores in all experiments, while traditional outsourcing needs to upload the dataset, which is about 198MB for the lowest load in DomainNet (50% of Sketch domain data for client). Even counting the download load, ECOS only needs to download 45MB of the pre-trained ResNet18 feature extractor together with 51KB data of centroid features, which is much less than data uploading. More detailed evaluation of the efficiency is placed in Appendix B.4 and we study how sample budgets and privacy budgets trade off the accuracy of ECOS models in Appendix B.3.

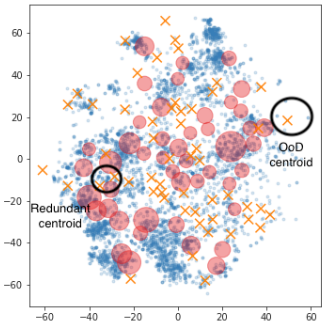


Figure 3: Demonstration of the centroids qualified by private CC. We use blue dots to represent the client data from Real domain of DomainNet. For the data of the cloud domains, larger circles represent centroids with higher CC and orange crosses are rejected OoD centroids.

## 5 Conclusion

In this paper, we explore the possibility of outsourcing model training without access to the client data. To reconcile the data shortage from the target domain, we propose to find a surrogate dataset from the source agnostic public dataset. We find that the heterogeneity of the open-source data greatly compromises the performance of trained models. To tackle this practical challenge, we propose a collaborative sampling solution, ECOS, that can efficiently and effectively filter open-source samples and thus benefits follow-up learning tasks. We envision this work as a milestone for the private and efficient outsourcing from low-power and cost-effective end devices.

We also recognize open questions of the proposed solution for future studies. For example, the public dataset may require additional data processing, e.g., aligning and cropping for improved prediction accuracy. In our empirical studies, we only consider the computer vision tasks, though no assumption was made on the data structures. We expect the principles to be adapted to other data types with minimal efforts. More data types, including tabular and natural-language data, will be considered in the follow-up works. Take the language data for example: BERT is among the most popular pre-trained models for extracting features from sentences [16], upon which ECOS can assistant to sample proxy data from massive public dataset, for instance, WikiText-103 [35] based on 28,475 Wikipedia articles. Alternative to the $L_2$-norm based proximity objectives, advanced semantic features could also enhance the sampling effectiveness of ECOS in language data [54]. More discussions on the social impacts of this work are enclosed in Appendix A.5.

## Acknowledgments and Disclosure of Funding

This research was funded by Sony AI. This material is based in part upon work supported by the National Science Foundation under Grant IIS-2212174, IIS-1749940, Office of Naval Research N00014-20-1-2382, and National Institute on Aging (NIA) RF1AG072449.

## References

[1] General data protection regulation. https://gdpr-info.eu/.

[2] A. Act. Health insurance portability and accountability act of 1996. *Public law*, 104:191, 1996.

[3] N. Alon, R. Bassily, and S. Moran. Limits of private learning with access to public data. *Advances in neural information processing systems*, 32, 2019.

[4] J. Barnes. Azure machine learning. *Microsoft Azure Essentials. 1st ed, Microsoft*, 2015.

[5] B. Biggio, B. Nelson, and P. Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

[6] E. Bisong. Google cloud machine learning engine (cloud mle). In *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, pages 545–579. Springer, 2019.

[7] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman. Learning bounds for domain adaptation. *Advances in neural information processing systems*, 20, 2007.

[8] S. Boyd, S. P. Boyd, and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[9] C. Buciluǎ, R. Caruana, and A. Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 535–541, 2006.

[10] C. Chen, Y. Liu, X. Ma, and L. Lyu. Calfat: Calibrated federated adversarial training with label skewness. In *Advances in Neural Information Processing Systems*, 2022.

[11] C. Chen, L. Lyu, H. Yu, and G. Chen. Practical attribute reconstruction attack against federated learning. *IEEE Transactions on Big Data*, 2022.

[12] F. Chen, T. Xiang, X. Lei, and J. Chen. Highly efficient linear regression outsourcing to a cloud. *IEEE transactions on cloud computing*, 2(4):499–508, 2014.

[13] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver. Combinatorial optimization. *Oberwolfach Reports*, 5(4):2875–2942, 2009.

[14] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[17] J. Dong, Y. Cong, G. Sun, Z. Fang, and Z. Ding. Where and how to transfer: Knowledge aggregation-induced transferability perception for unsupervised domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021.

[18] J. Dong, Y. Cong, G. Sun, B. Zhong, and X. Xu. What can be transferred: Unsupervised domain adaptation for endoscopic lesions segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4022–4031, June 2020.

[19] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*, Lecture Notes in Computer Science, pages 265–284. Springer Berlin Heidelberg, 2006.

[20] E. W. Forgy. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *biometrics*, 21:768–769, 1965.

[21] C. He, M. Annavaram, and S. Avestimehr. Group knowledge transfer: Federated learning of large cnns at the edge. *Advances in Neural Information Processing Systems*, Nov. 2020.

[22] G. Hinton, O. Vinyals, J. Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

[23] J. Hong, H. Wang, Z. Wang, and J. Zhou. Efficient split-mix federated learning for on-demand and in-situ customization. *arXiv preprint arXiv:2203.09747*, 2022.

[24] J. Hong, Z. Zhu, S. Yu, Z. Wang, H. H. Dodge, and J. Zhou. Federated adversarial debiasing for fair and transferable representations. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 617–627, 2021.

[25] Z. Jiang, T. Chen, T. Chen, and Z. Wang. Improving contrastive learning on imbalanced data via open-world sampling. In *Advances in Neural Information Processing Systems*, May 2021.

[26] A. E. Johnson, T. J. Pollard, L. Shen, L.-w. H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. Anthony Celi, and R. G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.

[27] X. Lei, X. Liao, T. Huang, H. Li, and C. Hu. Outsourcing large matrix inversion computation to a public cloud. *IEEE Transactions on cloud computing*, 1(1):1–1, 2013.

[28] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou. Fedbn: Federated learning on non-iid features via local batch normalization. In *International Conference on Learning Representations*, Sept. 2020.

[29] E. Liberty, Z. Karnin, B. Xiang, L. Rouesnel, B. Coskun, R. Nallapati, J. Delgado, A. Sadoughi, Y. Astashonok, P. Das, et al. Elastic machine learning algorithms in amazon sagemaker. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 731–737, 2020.

[30] T. Liu, G. Vietri, T. Steinke, J. Ullman, and S. Wu. Leveraging public data for practical private query release. In *International Conference on Machine Learning*, pages 6968–6977. PMLR, 2021.

[31] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[32] M. R. Loghmani, M. Vincze, and T. Tommasi. Positive-unlabeled learning for open set domain adaptation. *Pattern Recognition Letters*, 136, June 2020.

[33] L. Lyu, H. Yu, X. Ma, C. Chen, L. Sun, J. Zhao, Q. Yang, and P. S. Yu. Privacy and robustness in federated learning: Attacks and defenses. *arXiv preprint arXiv:2012.06337*, 2020.

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

[35] S. Merity, C. Xiong, J. Bradbury, and R. Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

[36] I. Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275, Santa Barbara, CA, USA, Aug. 2017. IEEE.

[37] S. A. Osia, A. Taheri, A. S. Shamsabadi, K. Katevas, H. Haddadi, and H. R. Rabiee. Deep private-feature extraction. *IEEE Transactions on Knowledge and Data Engineering*, 32(1):54–66, Jan. 2020.

[38] G. Paolacci, J. Chandler, and P. G. Ipeirotis. Running experiments on amazon mechanical turk. *Judgment and Decision making*, 5(5):411–419, 2010.

[39] N. Papernot, M. Abadi, Ú. Erlingsson, I. Goodfellow, and K. Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *International Conference on Learning Representations*, 2016.

[40] N. Papernot, S. Song, I. Mironov, A. Raghunathan, K. Talwar, and Ú. Erlingsson. Scalable private learning with pate. *International Conference on Learning Representations*, Feb. 2018.

[41] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1406–1415, 2019.

[42] X. Peng, Z. Huang, Y. Zhu, and K. Saenko. Federated adversarial domain adaptation. *arXiv preprint arXiv:1911.02054*, 2019.

[43] J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence. *Dataset shift in machine learning*. Mit Press, 2008.

[44] O. Sener and S. Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

[45] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[46] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. A. Raffel, E. D. Cubuk, A. Kurakin, and C.-L. Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in Neural Information Processing Systems*, 33:596–608, 2020.

[47] P. Vepakomma, O. Gupta, T. Swedish, and R. Raskar. Split learning for health: Distributed deep learning without sharing raw patient data. *ICLR 2019 Workshop on AI for social good*, Dec. 2018.

[48] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen. In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning. *IEEE Network*, 33(5):156–165, 2019.

[49] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan. Subsampled rényi differential privacy and analytical moments accountant. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1226–1235. PMLR, 2019.

[50] K. Weiss, T. M. Khoshgoftaar, and D. Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.

[51] Y. Xu, Y. Wang, H. Chen, K. Han, C. XU, D. Tao, and C. Xu. Positive-unlabeled compression on the cloud. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[52] X. Yan, D. Acuna, and S. Fidler. Neural data server: A large-scale search engine for transfer learning data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3893–3902, 2020.

[53] J. Yao, S. Zhang, Y. Yao, F. Wang, J. Ma, J. Zhang, Y. Chu, L. Ji, K. Jia, T. Shen, A. Wu, F. Zhang, Z. Tan, K. Kuang, C. Wu, F. Wu, J. Zhou, and H. Yang. Edge-cloud polarization and collaboration: A comprehensive survey. *arXiv:2111.06061 [cs]*, Nov. 2021.

[54] S. Zad, M. Heidari, P. Hajibabaee, and M. Malekzadeh. A survey of deep learning methods on semantic similarity and sentence modeling. In *2021 IEEE 12th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 0466–0472. IEEE, 2021.

[55] Y. Zhu and Y.-X. Wang. Poission subsampled rényi differential privacy. In *International Conference on Machine Learning*, pages 7634–7642. PMLR, 2019.

[56] Y. Zhu, X. Yu, M. Chandraker, and Y.-X. Wang. Private-knn: Practical differential privacy for computer vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11854–11862, 2020.

[57] Z. Zhu, J. Hong, S. Drew, and J. Zhou. Resilient and communication efficient learning for heterogeneous federated systems. In *Proceedings of Thirty-ninth International Conference on Machine Learning (ICML 2022)*, 2022.

## Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We include the instructions but not codes for reproducing results.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

# A More Method Details

In this section, we elaborate on additional technical and theoretical details of our paper.

## A.1 Automated client labeling via private kNN

Here, we briefly introduce the main idea of client labeling by private kNN. Given $C$ classes, labeling is done by nearest neighbor voting:

$$f(x) = \arg\max_{c\in[C]} A_\epsilon(v_c),\ v_c = |\{(x', y') \in \mathbb{N}_K(x)|y' = c\}|,$$

where $\mathbb{N}_K(x)$ is a set including the $K$-nearest neighbors of $x$ in the client dataset. $A_\epsilon$ is a privatization mechanism complying with the notion of $\epsilon$-Differential Privacy (DP) [19]. In brief, privatization is done by adding Gaussian noise to a value with finite sensitivity. To filter out potential wrong labels, the client only returns high-confident samples by screening. Let the confidence of a pseudo label be $s(x) = \max_{c\in[C]} A_\epsilon(v_c)$ which is also privatized by the Gaussian noise mechanism. We find that the original version of screening may suffer from a large imbalance of pseudo labels. Per class, we screen the pseudo labels by selecting the top-60 confident samples given 600 labeling budget.

## A.2 Improving client labeling

Because of the noise mechanism for privacy protection, the client labeling may be quite random if the selected samples are hard to discriminate. Thus, we propose to improve the discrimination of selected samples in advance, when the ECOS selects samples for labeling. First, we estimate the discrimination by the confidence in the ECOS. The ECOS confidence is defined by the vote count of the highest-voted class subtracting the one of the second highest one, denoted as $v_r^{\text{conf}}$. To merit the balancedness of samples, we filter the clusters to keep the top-70% samples with the highest confidence *per class*. When decompressing the clusters on the cloud, we incorporate the confidence into the sampling score by $v'_r = \psi\left[(v_r^{\text{conf}} + v_r)/2\right]$ where $v_r$ is the original score.

## A.3 Privacy Accountant for ECOS

To understand the privacy cost of ECOS, we review the techniques that are essential to establish the privacy bound.

**Definition A.1** (Differential Privacy [19])**.** Suppose $\epsilon$ and $\delta$ are two positive constants. A randomized algorithm $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ is $(\epsilon, \delta)$-DP if for every pair of neighboring datasets $X, X' \in \mathcal{X}$, and every possible measurable output set $Y \subset \mathcal{Y}$ the following inequality holds:

$$\text{Pr}[\mathcal{M}(X) \in Y] \le e^\epsilon \text{Pr}[\mathcal{M}(X') \in Y] + \delta,$$

where $\text{Pr}[\cdot]$ denotes the probability of a given event.

DP provides a way to quantify the privacy risk (termed as the difference between two probability given a pair of similar but different inputs) in the probability of $\delta$. Though DP is a simple notion for risks, the estimation of a tight privacy bound is still challenging. For this reason, RDP is proposed an alternative tool.

**Definition A.2.** (Rényi Differential Privacy (RDP) [36]) A randomized algorithm $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ is $(\alpha, \epsilon)$-RDP with order $\alpha > 1$ if for all neighboring datasets $X, X'$ the following holds

$$D_\alpha(\mathcal{M}(X)||\mathcal{M}(X')) \le \epsilon,$$

where $D_\alpha(\cdot||\cdot)$ is the Rényi divergence between two distributions.

The RDP and DP can be connected by Lemma A.1.

**Lemma A.1.** *If a mechanism $\mathcal{M}$ satisfies $(\alpha, \epsilon)$-RDP, then it also satisfies $(\epsilon + \frac{\log 1/\delta}{\alpha-1}, \delta)$-DP.*

To reveal the potential privacy risks, we give a theoretical bound on the privacy cost based on DP in Lemma A.2. The proof of Lemma A.2 is similar to Theorem 8 in [56] without confidence screening.

**Lemma A.2.** *Suppose the subsampling rate $\gamma$ and noise magnitude $\sigma$ of the ECOS are positive values such that $\gamma \leq \min\left\{0.1, \sigma\sqrt{\log(1/\delta)/6}\right\}$ and $\sigma \geq 2\sqrt{5}$. The total privacy bound of the ECOS scoring $m = |\hat{D}^q|$ query samples with $n = |\hat{D}^p|$ private client samples is $(\epsilon, \delta)$-DP with $\delta > 0$ and*

$$\epsilon = \mathcal{O}(\frac{\gamma}{\sigma}\sqrt{\log(1/\delta)}), \tag{4}$$

*if $v_r$ in Algorithm 1 is estimated by using $\lceil \gamma n \rceil$ samples randomly subsampled from $\hat{D}^p$ with replacement and is processed by $\tilde{v}_r = v_r + \mathcal{N}(0, \sigma^2 I)$.*

*Proof.* When one sample is absent from the private client dataset, the scores $[v_1, \dots, v_R]$ will differ by 2 if without subsampling. By Lemma 11 of [56], the subsampled Gaussian mechanism accounted by the RDP is

$$\epsilon(\alpha) \leq \frac{24\gamma^2\alpha}{\sigma^2}$$

for all $0 < \alpha \leq \frac{\sigma^2 \log(1/\gamma)}{2}$, $\gamma \leq 0.1$ and $\sigma \geq 2\sqrt{5}$. By Lemma A.1, we can convert the RDP inequality to the standard $(\epsilon, \delta)$-DP as

$$\epsilon = \frac{24\gamma^2\alpha}{\sigma^2} + \frac{\log(1/\delta)}{\alpha - 1}.$$

Let $\alpha$ be $1 + \frac{\sqrt{\log(1/\delta)}}{\sqrt{\frac{24\gamma^2}{\sigma^2}}}$. Thus,

$$\epsilon = \frac{24\gamma^2}{\sigma^2} + 4\frac{\gamma}{\sigma}\sqrt{6\log(1/\delta)} \leq 8\frac{\gamma}{\sigma}\sqrt{6\log(1/\delta)},$$

where the last inequality is derived by the given range of $\gamma$. This thus completes the proof. □

The above bound implies that the privacy cost for our method is invariant w.r.t. the scale of the query dataset $\hat{D}^q$, and only depends on the DP parameters. Note that the Lemma A.2 is an asymptotic bound which requires some strict conditions on $\gamma$ or other variables. In practice, we leverage the tool of analytic privacy accountant through a numerical method [49], with which we can relax the strict conditions.

## A.4 Theoretical Analysis

Though empirical results show that more accurate models can be trained on ECOS-sampled datasets, it remains unclear how the cloud dataset and the training process affect the model performance on the client. In this section, we provide the proof of Theorem 3.1 using the domain generalization bound as stated in Theorem A.1.

**Theorem A.1** (Domain-adaptation learning bound from [7])**.** *Suppose two domains with distributions $\mathcal{D}_s$ and $\mathcal{D}_t$. Let $\mathcal{H}$ be a hypothesis space of $VC$-dimension $d$ and $D_s$ be the dataset induced by samples of size $N$ drawn from $\mathcal{D}_s$, respectively. Then with probability at least $1 - p$ over the choice of samples, for each $f \in \mathcal{H}$,*

$$L(f, \mathcal{D}_t) \leq L(f, D_s) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t) + 4\sqrt{\frac{2d\log(2N) + \log(4/p)}{N}} + \xi, \tag{5}$$

*where $\xi$ is the optimal loss, i.e., $\min_f L(f, \mathcal{D}_s) + L(f, \mathcal{D}_t)$, and $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_t)$ denotes the divergence between domain $s$ and $t$.*

*Proof of Theorem 3.1.* Our proof is mainly based on Theorem A.1. By definition, $S$ contains data sampled from the mixture of distributions, $\sum_{r=1}^R \mathcal{D}_r^q$. Apply Theorem A.1 to attain

$$L(f_\theta, \mathcal{D}^p) \leq L(f_\theta, S) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\sum_{r=1}^R \mathcal{D}_r^q, \mathcal{D}^p) + 4\sqrt{\frac{2d\log(2|S|) + \log(4/p)}{|S|}} + \xi. \tag{6}$$

And we have

$$d_{\mathcal{H}\Delta\mathcal{H}}\left(\sum\nolimits_{r=1}^{R} \mathcal{D}_r^q, \mathcal{D}_t\right) \le \sum\nolimits_{r=1}^{R} \alpha_r d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}_r, \mathcal{D}_t), \tag{7}$$

which was proved in [42] (proof of Theorem 2). Plugging Eq. (7) into Eq. (6), we can get the conclusion. □

Now, we consider $f_\theta$ to be trained by the widely-adopted gradient descent method. We first present Theorem A.2 which characterizes the empirical error bound after $T$ iterations. Substitute Eq. (8) into Eq. (3). Then we can obtain Lemma A.3.

**Theorem A.2** (Convergence bound, e.g., from [8])**.** *Suppose the model $f$ is parameterized by $\theta$ initialized as $\theta_0$. Let the loss function $L(f_\theta, D)$ be $\mu$-strongly convex and $G$-smooth w.r.t. $\theta$, and assume that the global minima $\theta^*$ exists. Then after $T$ iterations, gradient descent with a fixed learning rate $\eta \le 1/G$ satisfies*

$$L(f_{\theta_T}, D) \le L^* + (1 - \mu/G)^T \ell_0(\theta_0, D) \tag{8}$$

*where $L^* = \min_\theta L(f_\theta, D)$ and $\ell_0(\theta_0, D)$ is the initial error gap, i.e., $|L(f_{\theta_0}, D) - L^*|$.*

**Lemma A.3.** *Suppose assumptions in Theorems 3.1 and A.2 holds. Let $L(\cdot,\cdot)$ be a loss function on a hypothesis and a dataset (for empirical error) or a distribution (for generalization error). If $f$ is governed by the parameter $\theta$ trained on $S$ via $T$-iteration gradient descent and belongs to a hypothesis space $\mathcal{H}$ of $VC$-dimension $d$, then with probability at least $1 - p$ over the choice of samples, the inequality holds,*

$$\begin{aligned} L(f_{\theta_T}, \mathcal{D}^p) \le (1 - \frac{\mu}{G})^T \ell_0(\theta_0, S) + \frac{1}{2} \sum\nolimits_{r=1}^{R} \alpha_r d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_r^q, \mathcal{D}^p) \\ + 4\sqrt{\frac{2d \log(2|S|) + \log(4/p)}{|S|}} + \xi', \end{aligned} \tag{9}$$

*where $\xi' = \min_\theta \{L(f_\theta, S) + L(f_\theta, D^p)\} + \min_\theta L(f_\theta, S)$, and $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}, \mathcal{D}')$ denotes the distribution divergence.*

### A.5 Social Impacts

The conflict between the concerns on data privacy and demands for intensive computation resources for machine learning has composed the main challenge in training outsourcing. In this work, we devote our efforts to outsourcing with uploading data by leveraging authorized or public datasets. As the public datasets commonly available in many applications are collected from multiple data sources and thus tend to be non-identically distributed as the client data, it casts new challenges to use the public in place of the client dataset. Our method addresses this problem with accountable privacy cost and low communication and computation complexity. Therefore, the proposed ECOS provides a promising solution to mitigate the aforementioned conflict between the privacy and computation desiderata. Therefore, users from a broader spectrum can benefit from such a method to confidentially conduct cloud training.

### A.6 Connection to Federated Learning

Both our method and federated learning (FL) [34] consider protecting data privacy via not sharing data with the cloud. The key difference between FL and our concerned problem (training outsourcing) is that FL requires clients to conduct training while ECOS outsources the training to the cloud server. Since ECOS does not require local training, it can be ad-hocly plugged into FL to obtain an auxiliary open-source dataset for enhancing the federated training. ECOS can be used either before federated training (when a pre-trained model is required) or during federated training (when the pre-trained model can be replaced by the on-training model).

## B Experimental Details and More Experiments

Complementary to the main content, we provide the details of the experiment configurations to merit the reproducibility. We also conduct more qualitative experiments to understand the efficiency and effectiveness of the proposed method.

### B.1 Experimental Details

We organize the case-specified configurations into three cases and discuss the general setups first.

For Digits, we train the model for 150 epochs. We adopt a convolutional neural network for Digits in Table 6 and ResNet18 for DomainNet. To solve the learning problems including FixMatch and distillation-based compression, we use stochastic gradient descent with the momentum of $0.9$ and the weight decay of $5 \times 10^{-4}$. We use $s = 5$ for the scale function $\psi_s$ on DomainNet and $s = 1$ on Digits. When not specified, we noise the ECOS query with the magnitude as $25$. In Case $1$ and $2$, we reduce the noise magnitude to $10$ for DomainNet, since the two queries can bear more privacy costs to trade for higher accuracy.

**Case 1: Selective manual labeling**. We make use of the off-the-shelf ResNet18 is pre-trained on the ImageNet, which is widely accessible online. We adopt FixMatch for semi-supervised learning with the coefficient of $0.1$ on the pseudo-labeled loss, the moving average factor of $0.9$, and the batch size of $64$ for DomainNet and $128$ for Digits. To avoid feature distorting, we warm up the fine-tuning by freezing all layers except the last linear layer with a learning rate of $0.01$. After $30$ epochs, we fine-tune the model end to end until $80$ epochs to avoid overfitting biased data distributions.

**Case 3: Adaptive model compression**. We first pre-train a ResNet50 using all labeled open-source data for $100$ epochs with a cosine-annealed learning rate from $0.1$. The same warm-up strategy as Case $1$ is used here. To extract the knowledge from ResNet50, we combine the knowledge-distillation (KD) loss $L_{KD}$ and cross-entropy loss $L_{CE}$ by $0.1 \times L_{KD} + 0.9 \times L_{CE}$ and calculate the losses on the selected samples only. The temperature in the KD loss is set to be $10$.

**Case 2: Automated client labeling**. For the cloud training, we adopt the same configuration as the selective manual labeling. For private kNN, we let the client release $600$ labels with class-wise confidence thresholds described in the last section. We noise the labeling in the magnitude of $25$ and the confidence in the magnitude of $75$. For both datasets, we subsample $80\%$ client data per labeling query to reduce the privacy cost.

Table 6: The structure of the conventional neural network for the Digits dataset.

| Layer name | PyTorch pseudo code |
|---|---|
| conv1 | Conv2d(1, 64, kernel_size=(5, 5), stride=(1, 1)) |
| bn1 | BatchNorm2d(64, eps=1e-05, momentum=0.1) |
| conv1_drop | Dropout2d(p=0.5, inplace=False) |
| conv2 | Conv2d(64, 128, kernel_size=(5, 5), stride=(1, 1)) |
| bn2 | BatchNorm2d(128, eps=1e-05, momentum=0.1) |
| conv2_drop | Dropout2d(p=0.5, inplace=False) |
| fc1 | Linear(in_features=2048, out_features=384, bias=True) |
| fc2 | Linear(in_features=384, out_features=192, bias=True) |
| fc3 | Linear(in_features=192, out_features=11, bias=True) |

We conduct our experiments on the Amazon Web Service platform with 4 Tesla T4 GPUs with 16GB memory and a 48-thread Intel CPU. All the code is implemented with `PyTorch` $1.11$. To account for the privacy cost, we utilize the open-sourced `autodp` package following the private kNN.

### B.2 Effect of Parameters

To better understand the proposed method, we study the effect of the important hyper-parameters. To this end, we consider the selective labeling task with Digits, keeping $50\%$ of the SVHN dataset at the client end. Both the ID+OoD and OoD cases are evaluated to reveal the method's effectiveness under circumstances with various hardness. Also, we study how the score scale $s$ affects the ID ratios (denoted as the ID TPR) in the selected set and the number of effective samples. We only examine the ID TPR corresponding to the proximity objective in Eq. (2) if known ID samples are present on the cloud, namely in the *ID+OoD* case. In the middle panes of Figs. 4 and 5, we show that our method can effectively improve the ID TPR against the inherent ratio of ID samples on the cloud.

**Effect of the compression size** $R$. In Fig. 4, we evaluate $R$ in terms of the test accuracy. When the budget is small (1k and 2k budgets in the ID+OoD case), it is essential for the cloud server to sense the client distribution with higher accuracy via more queries. Therefore, a larger $R$ is desired, which can increase the portion of ID samples in the selected set, as shown in the middle pane of Fig. 4. Considering that a higher burden on communication comes with a larger $R$, the value of $100$ leads to a fair trade-off to the accuracy in which case the ID TPR reaches a peak.

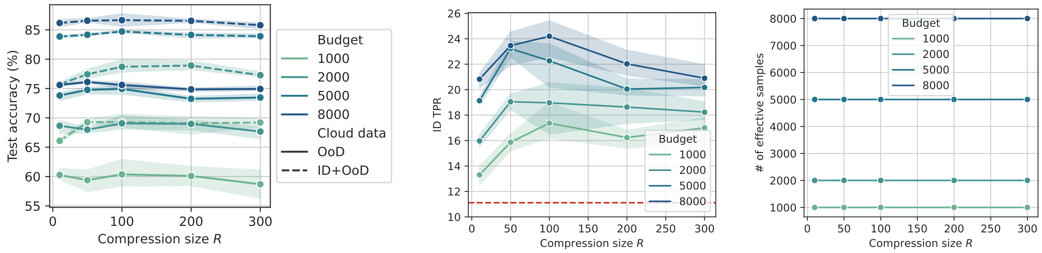


Figure 4: Vary the compression size $R$ and evaluate the test accuracy, ID ratios (%) in selected samples and the number of effectively selected samples. The red horizontal line indicates the ratio of ID samples in the whole cloud dataset.

Given a larger budget, e.g., 8000, increasing $R$ may lower the ID TPR. We attribute the decline to the limited size of the client dataset and privacy constraints. Given more clusters (i.e., $R$), the expected number of votes (proportional to the score) for each cluster will be reduced and is badly blurred by the DP noise. Thus, the ID TPR will decrease simultaneously, regardless of which the test accuracy is not significantly affected.

For the OoD case which is relatively harder for sampling due to the lack of true ID data, the parameter sensitivity is weakened, though the compression size of 100 is still a fair choice, for example, bringing in $1-2\%$ gains in the 5k, 8k cases comparing the worst cases.

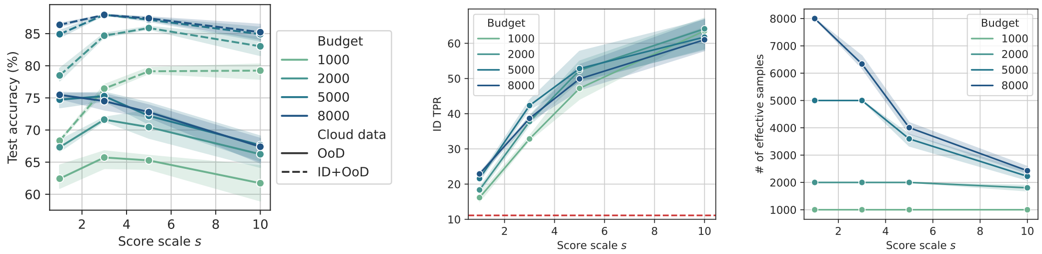


Figure 5: Vary the score scale $s$ in terms of test accuracy, ID ratios (%) in selected samples and the number of effectively selected samples (which could be smaller than the budget). The red horizontal line indicates the ratio of ID samples in the whole cloud dataset.

**Effect of the score scale** $s$. The score scale $s$ decides the sensitivity of the sampling in the sense of proximity. A larger $s$ means that the ECOS will prioritize the proximity more during sampling. In Fig. 5, we present the ablation study of $s$. A larger $s$ is preferred when the budget becomes limited because it increases the ID TPR effectively. Though not significantly, an overly large $s$ has a significantly negative influence on the accuracy, especially for the OoD case. The reason for the negative impact of $s$ on a large budget can be understood by probing the number of effective samples. For budgets larger than 2000, the effectively selected samples are reduced with heavily scaled scores (e.g., $s \geq 3$) where the ECOS will concentrate its selection into very few clusters and eliminate the rest clusters strictly.

### B.3 Evaluation of Sample and Privacy Efficiency

**Effects of sample budgets.** In Fig. 6, we compare the sample efficiency in the selective labeling task with Digits, keeping 50% of the SVHN dataset at the client end. We obtain the upper-bound accuracy in the ideal case via random sampling when the cloud dataset distributes identically (ID) as the client dataset. When OoD data are included in the open-source cloud dataset (ID+OoD), the training becomes more demanding for the labeled samples. If none of the iid samples presents in the cloud set (OoD), the accuracy decreases quickly with the same labeled samples. In comparison, informative sampling by K-Center slightly improves the accuracy by different budgets and the proposed ECOS significantly promotes the sample efficiency. With ECOS, $8 \times 10^3$ labeled samples in the ID+OoD case achieves comparable accuracy as the ideal case, while baselines remain large gaps. Both in

ID+OoD and OoD cases, our method yields competitive accuracy (at the $4 \times 10^3$ budget) versus the best baseline results using only half of the labeled data (at the $1 \times 10^4$ budget), dramatically cutting down the cost for manual labeling.

On observing the gains in sample efficiency, readers may also notice that our method induces additional costs at privacy, as compared to the baselines. We point out that the cost is constant w.r.t. the sampling budget and is as neglectable as $(0.22, 10^{-5})$-DP. It is worth noticing that the cost is independent of the hyper-parameters of the ECOS because the ECOS communication is a *single* query for each private sample (so as for the private dataset), even if we increase the size of the query set (i.e., the compression size $R$). In practice, the client can control the privacy risk (namely, the privacy cost) flexibly by adjusting the noise magnitude and the subsampling rate.

**Privacy-accuracy trade-off.** Additionally, we evaluate how the privacy costs affect the In Fig. 7, we study the relationship between privacy cost and performance by varying the noise scale $\sigma$ of the ECOS in label outsourcing. The experiment is conducted on SVHN client data for label outsourcing. Interestingly, the sampling effectiveness is not very sensitive to the noise scale. A very low privacy cost (0.08) can be achieved with noise as large as 70 and accuracy as high as $83.7\%$. The success could be attributed to the low dimension of queries (100 clusters) to the private dataset, resulting in the efficiency of privacy costs.

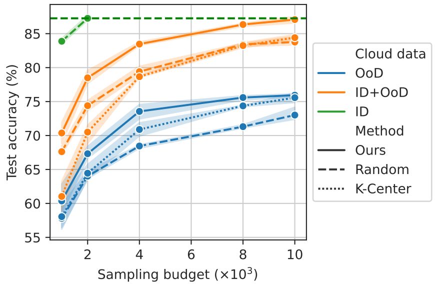


Figure 6: Evaluation of the sample efficiency on selective labeling. The green horizontal line implies the ideal case when all ID cloud data are labeled.

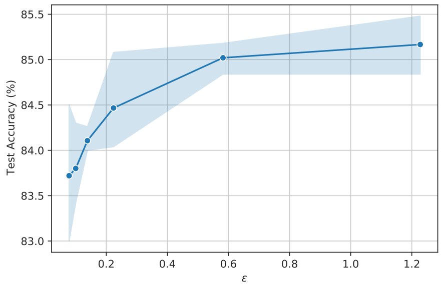
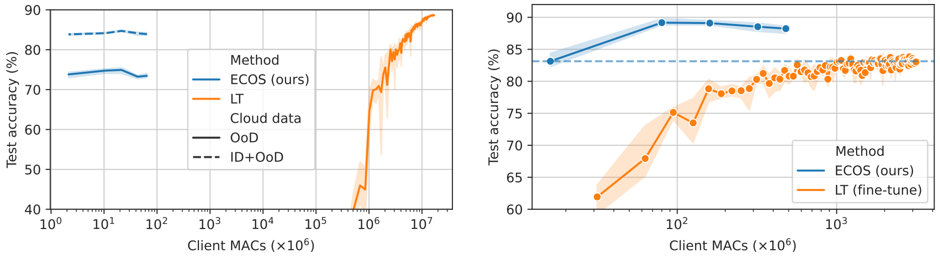


Figure 7: Evaluation on how the privacy costs affect the performance.

### B.4 Evaluation of Communication and Computation Efficiency

When improving the sample efficiency, we also need to take care of the communication and computation overheads brought by the ECOS. We examine the two kinds of efficiency by the same experiment configurations as in the last section.



(a) Digits

(b) DomainNet

Figure 8: With the 5000 budget, we evaluate the computation efficiency. The efficiency of locally-training (LT) on the DomainNet is enhanced by linear fine-tuning where only the linear classifier head is locally trained.

**Computation efficiency.** In Fig. 8, we compare the computation efficiency of our method to the local training (LT). We utilize the multiplication-and-addition counts (MACs) as the metric of computation (time) complexity, which is hardware-agnostic and therefore is preferred here. For a fair comparison, we tune the learning rate in $\{0.1, 0.01, 0.001\}$ with the cosine annealing during training and the number of epochs in $\{20, 50, 100\}$ of the LT to achieve a fair trade-off between the computation cost and test accuracy. For ECOS, since the computation cost linearly increases by the compression size (as shown in Fig. 9), we vary the compression size to check the performance when increasing computation costs. On Digits, we observe a large computation save by our method, even if the cost of our method will gradually increase by the size $R$ of the compressed query set.

Similar experiments are also run on the large-sized images using the DomainNet dataset (ID+OoD case), where the cost for extracting features is steeply increased by using a deep network (ResNet18). Recently, the most popular strategy for cloud training is two-phase learning: pre-training a model on the cloud using ImageNet and fine-tuning the linear classifier head on the client. Considering the large cost of feature extraction, we only let the client pre-extract features once only. Thus, the local training is as efficient as training a *linear* layer on extracted features. In Fig. 8, our method outperforms the local training a lot using much fewer MACs for data matching. Because all training is outsourced to the cloud, our method enables the end-to-end fine-tuning of the model resulting in better test accuracy. Even if the LT trains longer with higher computation costs, the test accuracy of the ECOS with the least MACs is comparable to the best performance of LT at around $10^9$ MACs, where the ECOS only utilizes the 10% of MACs by LT.

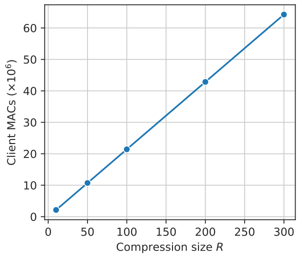


Figure 9: The linearly growing computation cost by increasing the compression size on the Digits dataset.

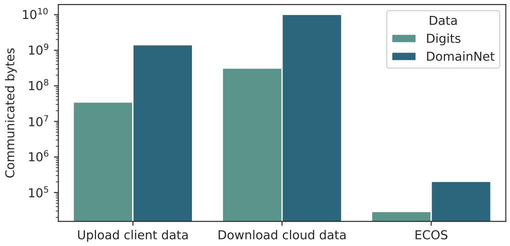


Figure 10: Evaluate the communication efficiency.

**Communication efficiency.** We also compare the communication efficiency to the full cloud training (via uploading the whole client dataset) and fully client training (via downloading cloud dataset) in Fig. 10. For the ECOS, we let the size of the query set be $100$, which is the default configuration in our experiments. Because the ECOS only communicates a few low-dimensional features (for example, 512-dimensional ResNet-extracted features for DomainNet and 72-dimensional HOG features for Digits), it costs much fewer bytes compared to traditional outsourcing by uploading the client data. To be concrete, we also present the cost of downloading the cloud data and it is way more expensive than the rest two methods.