Choose Before You Label: Efficient Node Selection in Constrained Federated Learning

Francesco Malandrino
CNR-IEIIT
Torino, Italy
francesco.malandrino@cnr.it

Carla Fabiana Chiasserini
Politecnico di Torino and CNR-IEIIT
Torino, Italy
carla.chiasserini@polito.it

Jayadev Naram
Chalmers University
Goteborg, Sweden
jayadev@chalmers.se

Giuseppe Durisi
Chalmers University
Goteborg, Sweden
durisi@chalmers.se

***Abstract*—In many cases, federated learning (FL) has to take place in *communication constrained* scenarios, where we must select a small number of learning nodes to reduce bandwidth consumption. Furthermore, such nodes may also have *computational constraints*, i.e., they can store small datasets and process and perform as little data processing as possible. In this context, it is of paramount importance to make node selection decisions *before* the learning process begins, and *without* labeling information. We tackle this daunting task through a two-pronged approach, where we (i) introduce a new metric called *loneliness*, defined on unlabeled datasets, and (ii) propose a novel algorithm called Goldilocks to make node selection decisions and identify the data to be labeled. Through both a theoretical and an experimental analysis, we show that loneliness is strongly linked with learning performance (i.e., test accuracy). Furthermore, our performance evaluation, including three state-of-the-art datasets and a comparison against centralized learning, demonstrates that Goldilocks outperforms approaches based upon a balanced label distribution by providing over 70% accuracy improvement, in spite of being efficient to compute and not using labeling information.**

I. Introduction

In many relevant and critical scenarios, including healthcare [1] and mobile autonomous systems, it is necessary to perform distributed (often, federated) learning in conditions that are *doubly* constrained. First, due to high communication costs, limited bandwidth, and/or harsh propagation conditions, we have to choose a small number of learning nodes. Second, due to computational (memory and processing) capabilities of mobile devices, there may be limited data availability, i.e., local datasets at such devices will be small, and a limited amount of data can be processed. Third, *labeling* such datasets, while necessary for learning, might be a long, slow, and/or costly process—due to, e.g., the need to involve human experts. It follows that, for constrained FL to succeed, it is of paramount importance to make high-quality node selection decisions using as little information as possible and, crucially, no labeling information at all.

The latter requirement rules out traditional approaches to node and dataset selection, which are predicated upon evaluating the balance between classes in local datasets [2], [3], [4] and/or assessing how well different labels correspond to different classes [5], [6]. Furthermore, we cannot rely on approaches evaluating how different nodes influence the training process (e.g., by looking at their gradients [2], [7], [4]), because we need to choose the nodes *before* the learning process starts—intuitively, we cannot afford to label samples we end up not using.

In this work, we propose a way out of this conundrum, predicated on two pillars. The first one is a new metric, called *loneliness* and satisfying our main requirements, i.e., it is extremely lightweight to compute and does not require labeling information. Through a set of experiments, we show that (i) loneliness exhibits a stronger correlation than label balance with the testing performance of a learning task, and that (ii) the best performance is associated with intermediate loneliness values. As a theoretical explanation of this second finding, we provide a probably-approximately correct (PAC) Bayes bound, based on adaptive subspace compression [8]. Through this bound, we illustrate that while the training error increases monotonically with the loneliness, the number of compressed bits to represent the weights of the deep neural network (DNN) (which we use as complexity term in the PAC-Bayes bound) decreases with the loneliness.

The second pillar is represented by an efficient, iterative algorithm called Goldilocks and able to iteratively add new nodes to the training process so as to reach a target loneliness value. A pictorial sketch of how our approach and the Goldilocks procedure improve over the state of the art is provided in Fig. 1. The traditional approach, represented on the left, is geared towards scenarios where we must select tens or hundreds of learning nodes and makes decisions based on how balanced labels are. While it is true that better-balanced datasets yield better performance, there is still a significant variability within the accuracy yielded by similarly-balanced datasets. Our approach, represented on the right, leverages the loneliness metric, resulting in a much more accurate knowledge of which datasets yield the best test accuracy, especially when datasets are small and only a small number thereof can be selected.

We evaluate the performance of Goldilocks in both centralized and federated scenarios, using state-of-the-art DNN models and datasets, and find it to consistently outperform approaches only considering label information. Importantly, the performance metrics we consider go beyond mere classification accuracy, and include the number of learning nodes to involve in the learning process.

In summary, our main contributions can be summarized as

978-3-903176-75-1 ©2025 IFIP

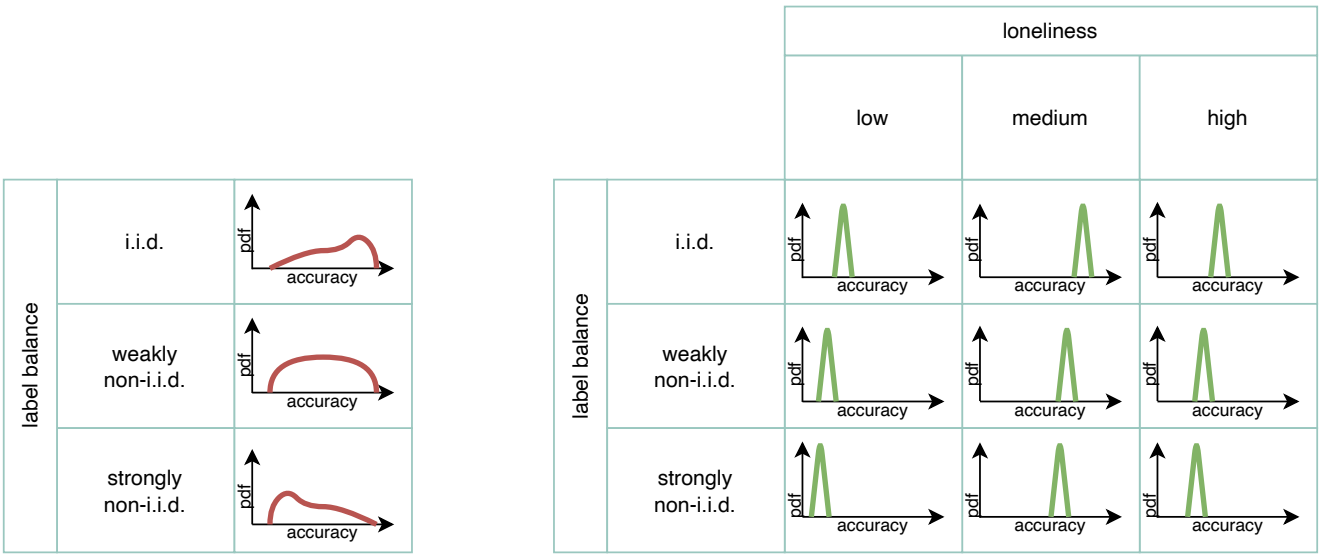


Fig. 1. A qualitative depiction of how using loneliness improves testing accuracy. The figure compares the information accounted for by state-of-the-art approaches based on label balance (left) and Goldilocks (right); each plot illustrates pictorially the empirical distribution of test accuracy achievable by training sets with a given level of loneliness and label balance. Accounting for label distribution only provides partial information on the learning outcome. On the contrary, loneliness has a much stronger correlation with accuracy; therefore, by leveraging on it, Goldilocks is able to make better decisions, hence, achieving higher accuracy.

follows:

- We propose a new metric, called *loneliness*, estimating the suitability of an *unlabeled* dataset (and, hence, of the learning node owning the data) for a given learning task;
- We perform a set of experiments, demonstrating that a strong link exists between loneliness and learning performance;
- We provide a theoretical explanation for the effectiveness of loneliness, based on model compression and a PAC-Bayes bound;
- We leverage loneliness and the insights provided by the theoretical analysis to design a procedure, called Goldilocks, that makes high-quality node selection decisions;
- We evaluate the performance of Goldilocks under both centralized and FL tasks using multiple datasets and neural networks, demonstrating that it consistently finds the best trade-offs between the resources needed for training and the resulting test accuracy. Notably, Goldilocks yields over 70% better accuracy improvement, while requiring to disclose no data about labels or label distribution.

II. The Loneliness Metric

We consider a typical distributed ML task where a set of learning nodes $\{n^k\} \in \mathcal{N}$, equipped with local datasets X^k and labels y^k, have to optimize the average value of loss function L by choosing the weights W of a parameterized learning model, i.e.,

$$\min_W \frac{1}{|\mathcal{N}|} \sum_{n^k \in \mathcal{N}} L(X^k, y^k, W).$$

Weights themselves can be set through any distributed learning algorithm, e.g., the classic FedAvg.

To characterize the quality of each local dataset, we start by introducing a sample-specific quantity. Specifically, we define the *loneliness* $\ell(i,k)$ of sample x_i^k in dataset X^k as the distance between x_i^k and the closest other sample in X^k:

$$\ell(i,k) = \min_{x_j^k \in X^k \setminus \{x_i^k\}} \left\| x_i^k - x_j^k \right\|. \tag{1}$$

It follows from (1) that the further away a sample is from the others, the higher its loneliness is. On the contrary, samples with low loneliness are very similar to other samples.

We further extend the notion of loneliness to the dataset X^k owned by node $n^k \in \mathcal{N}$, by considering the mean sample-wise loneliness in X^k:

$$\hat{\ell}(X^k) = \min_{x_i^k \in X^k} \ell(i,k). \tag{2}$$

Note that the loneliness metric does not depend on the label of the points in the datasets. The relationship between sample- and dataset-wise loneliness is exemplified in Fig. 2.

Complexity: Importantly, loneliness can be computed as a byproduct of the κ-nearest neighbors algorithm, with κ denoting the number of neighbors to consider, e.g., $\kappa = 5$. Recall that computing κ-NN values has a complexity (which is independent from κ) of $O(nd)$, with n being the number of points and d denoting the dimension of data. It follows that obtaining the loneliness value of a dataset is (i) free, if κ-NN information is already available and/or is computed for other purposes, and (ii) very efficient even if κ-NN must be ran *ad hoc*.

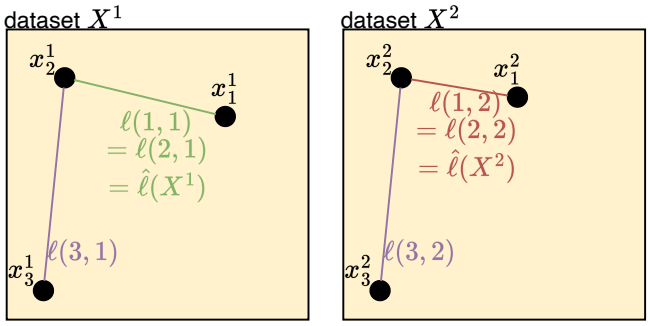


Fig. 2. The relationship between distance between samples (e.g., x_1^1 and x_2^1) in a dataset, represented as a yellow box, and loneliness values. The distance from each sample to the closest sample to it corresponds to the sample-wise loneliness defined in (1); the smallest sample-wise loneliness corresponds to the dataset-wise loneliness as defined in (2).

Alternatives and discussion: Metrics with a similar purpose to loneliness include those used for outlier detection [9] and quality assurance [5], [6]; however, loneliness has a more specific purpose and is much simpler to compute. Indeed, outlier detection entails first clustering the data, and *then* marking as outliers the nodes that are too far away from any cluster. Similarly, quality assurance methodologies like [5], [6] first identify neighborhoods in both the input and output spaces, and then quantify the complexity of the relationship between them. In all cases, the first step of the algorithm is clustering, which (taking the simplest viable algorithm, i.e., κ-means) has a complexity of $O(\kappa n i)$, with i being the number of iterations to run. Comparing that with the $O(\kappa n d)$ complexity of loneliness, and recalling that $i \gg d$ in most cases, we can conclude that computing loneliness is more efficient that even performing the *preliminary* step, with the *simplest* algorithm, of alternative approaches.

To illustrate the usefulness of the loneliness metric, we conduct in the next section an experiment illustrating its impact on the performance of learning algorithms operating on small training sets.

III. Experimental Analysis

The micro-datasets: To ascertain the effect of loneliness on the learning performance and to compare it with the traditionally used label balance, we start from the popular MNIST dataset and create a total of 90 micro-datasets, according to the following rules:

- all micro-datasets have 500 samples, extracted from the 60 000 samples of the MNIST training set; every micro-dataset has a different combination of label balance and loneliness level as specified below;
- in each micro-dataset, one of the 10 classes is over-represented, and $\alpha \in [1, 5]$ is the unbalance factor, i.e., the ratio between the number of samples of the most and least represented classes;
- each micro-dataset k has a loneliness level $\lambda(X^k)$ ranging between 1 and 10 and defined as

$$\lambda(X^k) = 1 + \left\lfloor 10 \frac{\hat{\ell}(X^k) - \min_{n^h \in \mathcal{N}} \hat{\ell}(X^h)}{\max_{n^h \in \mathcal{N}} \hat{\ell}(X^h) - \min_{n^h \in \mathcal{N}} \hat{\ell}(X^h)} \right\rfloor . \quad (3)$$

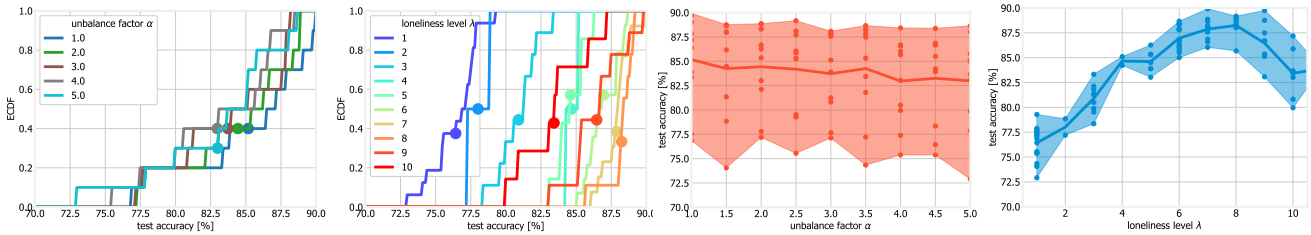
The micro-datasets so obtained reproduce those cases where there is a large number of potential learning nodes, all equipped with datasets that are (i) small and (ii) defective in different ways. In such scenarios, it is often impractical or impossible to query a large number of learning nodes. Hence, choosing them wisely is crucial, especially considering the need for dependable learning involving heterogeneous nodes.

In view of the relative simplicity of the MNIST dataset, we use the LeNet5 DNN for our tests; such DNN includes three convolutional layers and two fully-connected ones. Additional details on the data and DNN models used for all tests conducted in the paper can be found in the Appendix.

Loneliness and test accuracy: Fig. 3 summarizes the main results of our MNIST tests. We start from the relationship between label distribution and learning performance: in Fig. 3(a) and Fig. 3(b), each line corresponds to one value of the unbalance factor α (in Fig. 3(a)) and loneliness level λ (in Fig. 3(b)), and depicts the empirical cumulative density function (ECDF) of the test accuracy yielded by the corresponding micro-datasets. The value of α or λ tells us on which of the ECDFs the actual accuracy is; as an example, for a loneliness level equal to 2, the accuracy is between 76% and 78% with 90% probability, and its expected value is 77.4%.

The difference between the two plots is striking: ECDFs corresponding to different values of the unbalance factor α in Fig. 3(a) almost always overlap and cover virtually all accuracy values on the x-axis. Furthermore, the mean values (i.e., the dots) are very concentrated. All values of α result in accuracy levels between 77% and 88% with a mean around 83%. We note that knowing the exact value of α does not help make that information more specific. On the other hand, the ECDFs corresponding to different values of loneliness level λ are much more far apart, and overlap to a very limited extent. Similarly, the mean values are also far from each other (also note that we depict ten levels of loneliness, but only five values of α). It is thus evident that loneliness offers more detailed information on where the resulting accuracy will be. Hence, it is not only a less expensive metric to compute, since it does not require labeling. It is also more useful.

Fig. 3(c) and Fig. 3(d) provide a more detailed view of this effect. Each marker in the plots corresponds to a dataset, and its position along the x- and y-axis corresponds to the value of the metric (unbalance factor α in the former plot, loneliness level λ in the latter) and resulting accuracy, respectively. We note that the shaded area is almost rectangular for the unbalance factor, which confirms how different values of α correspond to similar values of accuracy; for loneliness, instead, we observe a much more narrow, arch-like shape.



(a) (b) (c) (d)

Fig. 3. MNIST experiments with the LeNet DNN: loneliness is more useful than label balance as a metric to predict accuracy. Distribution of the test accuracy for different values of the unbalance factor α (a) and loneliness level λ (b), with dots representing mean values; test accuracy levels achieved by micro-datasets with different unbalance factors (c) and loneliness (d).

The narrowness of the shaded area in Fig. 3(d) further indicates that λ serves as an excellent proxy for the resulting accuracy. Interestingly, the arch-like shape of the area shows that the best accuracy is achieved for *intermediate* levels of loneliness (between 5 and 7). The behavior in Fig. 3(d) makes intuitive sense if we consider that high levels of loneliness might be associated with the presence of outliers. A whole dataset composed of outliers is in fact harder to generalize from than one with a more balanced composition.

In summary, we can conclude that **loneliness has a much stronger correlation with test accuracy. Hence, it is much more useful than class balance when predicting it. Furthermore, the best accuracy is reached for intermediate levels of loneliness.**

IV. Loneliness Through the Lenses of a Compression-Based PAC-Bayes Bound

In this section, we use a PAC-Bayes generalization bound to explain why the test accuracy of micro-datasets peaks at intermediate loneliness $\hat{\ell}$. Let $\mathcal{Z}$ be the instance space and P_Z be the unknown distribution on $\mathcal{Z}$ that generates the training data $\mathbf{Z}=(Z_1,\dots,Z_m)\in\mathcal{Z}^m$ independently. Let $\mathcal{W}$ be the hypothesis space and $P_{W|\mathbf{Z}}$ be a probabilistic learning algorithm that takes $\mathbf{Z}$ and outputs a hypothesis $W\in\mathcal{W}$. Finally, let $c:\mathcal{W}\times\mathcal{Z}\to\mathbb{R}_+$ be the loss function. Then the population loss of w is defined as $L_{P_Z}(w)=\mathbb{E}_{P_Z}[c(w,Z)]$ and the training loss is defined as $L_{\mathbf{Z}}=\frac{1}{m}\sum_{i=1}^{m}c(w,Z_i)$. Given a prior distribution Q_W and a posterior distribution $P_{W|\mathbf{Z}}$ over $\mathcal{W}$, [10] states that with probability at least $1-\delta$ under $P_{\mathbf{Z}}$,

$$\mathbb{E}_{P_{W|\mathbf{Z}}}[L_{P_Z}(W)] \le \mathbb{E}_{P_{W|\mathbf{Z}}}[L_{\mathbf{Z}}(W)]+ \sqrt{\frac{\mathrm{KL}(P_{W|\mathbf{Z}}\|Q_W)+\log(m/\delta)+2}{2m-1}}. \quad (4)$$

In words, the population loss can be upper-bounded by the sum of a training loss and a complexity term that quantifies, via the Kullback-Leibler (KL) distance, the penalty incurred in assuming $W\sim P_{W|\mathbf{Z}}$ when $W\sim Q_W$. Bounds on the population error similar to (4) are usually referred to as PAC-Bayes bounds.

To shed lights on the impact of the loneliness on the PAC-Bayes bound (4), we experimentally evaluate $\mathbb{E}_{P_{W|\mathbf{Z}}}[L_{\mathbf{Z}}(W)]$ and $\mathrm{KL}(P_{W|\mathbf{Z}}\|Q_W)$ for datasets with different loneliness. Specifically, following [8], we select the universal prior $Q_W(W)=2^{-\mathcal{K}(W)/\gamma}$ where $\mathcal{K}$ is the prefix Kolmogorov complexity of W [11] and $\gamma\le 1$. The learning algorithm $P_{W|\mathbf{Z}}$ is chosen to be the point mass distribution on $W^\star$, which is the hypothesis obtained by training on $\mathbf{Z}$. Training is performed in two steps, according to the adaptive subspace compression algorithm introduced in [8]: (i) first, we learn a low-dimensional linear embedding of the model weights; then (ii) we quantize the embedded weights to a fixed number of levels. The training loss is computed using the 0–1 loss function. The KL term for the selected posterior and prior is upper-bounded by an expression containing the length of the shortest program needed to reproduce $W^\star$. This is computed as the number of bits required to represent the quantized model weights extracted from step (ii) of the training procedure, using an arithmetic code.

We construct 50 micro-datasets from MNIST with varying loneliness values $\hat{\ell}$ and fixed unbalance factor $\alpha=1$ (i.e., balanced datasets). The experimental details are provided in the Appendix. In Fig. 4, we plot the training loss and the KL term for three different values of dataset size. Each dot represents the training loss and the KL term for one dataset, respectively. We also report the linear regression line for each plot, the corresponding Pearson correlation coefficient r-value, and the two-sided p-value.

We see that for all dataset sizes m considered in the figure, the training loss appears to increase as a function of the loneliness value. Intuitively, as the samples in the dataset become more dissimilar, the training process becomes slower. On the contrary, the KL term appears to decrease monotonically with the loneliness. This suggests that the model complexity decreases. Intuitively, a dataset with a higher loneliness can be classified using a more compressible neural network.

As a result of these two opposite trends, intermediate loneliness values are preferable. Unfortunately, this cannot be demonstrated by evaluating the PAC-Bayes bound in (4) directly, since the training procedure suggested in [8] yields vacuous results whenever the dataset size m is below 2000. For $m\ge 2000$, the variation in loneliness across the generated datasets is not significant and no trends can be inferred. Indeed, this is the reason why the slope of the linear regression curve

in Fig. 4 decreases when m increases.

V. Node and Dataset Selection: Goldilocks

In this section, we describe our Goldilocks node- and data-selection procedure.

Node selection: The node selection problem can be stated as follows: given a set $\mathcal{N}^{\text{curr}} \subset \mathcal{N}$ of currently-selected learning nodes, we want to identify a new node $n^* \in \mathcal{N} \setminus \mathcal{N}^{\text{curr}}$ to add to the training process, so as to optimize the learning outcome, e.g., maximize the test accuracy. Such outcome is estimated through a *proxy metric* $\hat{\mu}(\mathcal{Q})$, taking as input a set $\mathcal{Q}$ of selected datasets, i.e., $\mathcal{Q} = \bigcup_q X^q$; we are also given a target value $\hat{\tau}$ for the metric $\hat{\mu}$. The metric $\hat{\mu}$ itself can correspond to any metric taking as an input a set of datasets, including the loneliness $\hat{\ell}$ defined in (2) and the unbalance factor α. In all cases, the value of the metric of a set of datasets $\mathcal{Q}$ corresponds to the value of that metric computed over the union of all datasets therein.

Given all the above, the selected node according to the Goldilocks procedure is simply the one that results in the metric $\hat{\mu}$ being closest to the target $\hat{\tau}$, i.e.,

$$n^* \leftarrow \arg \min_{n \in \mathcal{N} \setminus \mathcal{N}^{\text{curr}}} \left| \hat{\mu}(\mathcal{N}^{\text{curr}} \cup \{n\}) - \hat{\tau} \right|. \tag{5}$$

It is worth emphasizing that the Goldilocks procedure just outlined can be performed for arbitrary metrics, including the unbalance factor α introduced in Sec. III, the loneliness level $\lambda(X^k)$ defined in (3), as well as other metrics defined in the literature, e.g., the normalized entropy used in [12]. Furthermore, Goldilocks supports arbitrary target values, i.e., it can be applied to metrics that do not need to be maximized or minimized: this is fundamental in situations like those depicted in Fig. 3(right), where the best performance is associated with intermediate loneliness values. In these cases, Goldilocks is able to select the nodes that are *just right* (hence the name of the procedure) for the task at hand.

As mentioned earlier, the steps above can be applied only if the metric to use is defined for individual samples, i.e., if $\mu(i,k)$ does exist. This is the case of loneliness (as per (1)), but not, as an example, for the label balance. We can thus remark again that, by combining the loneliness metric and the Goldilocks procedure, we are able to make fine-grained node selection decisions.

Scope and goals of Goldilocks: As per (5), Goldilocks selects nodes solely based on the metric $\hat{\mu}$, ignoring such factors as node connectivity, resources, and costs. All these factors need to be weighted against sheer learning performance (e.g., training accuracy) in real-world situations, as also discussed in Sec. VII. It is worth stressing that Goldilocks is not meant to give a complete, self-contained solution to the node selection problem in FL. Rather, it helps comparing how different metrics affect node selection decisions and the resulting performance. Such an approach can then be integrated within any of the alternative approaches discussed in Sec. VII, leaving the remaining parts thereof (dealing, for example, with connectivity) in place.

VI. Performance Evaluation

We now verify the usefulness of the Goldilocks procedure in the context of node selection, especially when loneliness is chosen as metric. To this end, we perform a set of experiments using the CIFAR10 dataset [13] and the MobileNetV2 DNN [14]. Specifically, (i) We generate from the CIFAR10 training set 100 micro-datasets with 500 samples each, following the same procedure as in Sec. III; every micro-dataset has different loneliness and label unbalance factor. (ii) For each micro-dataset, we use the Goldilocks procedure described in Sec. V, in combination with either the label unbalance factor α or the loneliness level λ, to select one or two additional datasets. (iii) We train the model using each combination of datasets, in a centralized manner, for 50 epochs, tracking the resulting test accuracy.

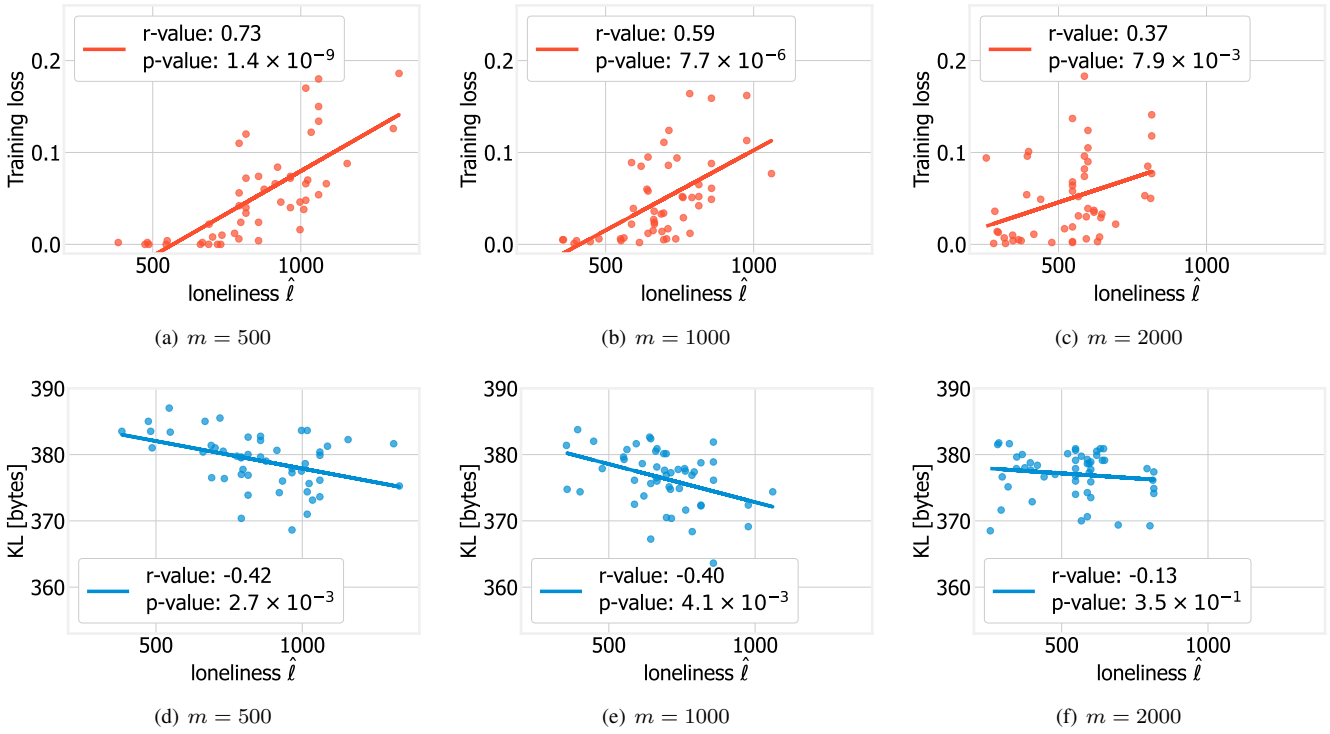
Fig. 5(a) shows the distribution of accuracy for one (dashed line), two (solid lines), and three (dotted lines) micro-datasets. The color of the lines reflects the metric employed to select the additional data: red for the label unbalance α (for which a target of $\hat{\tau}=1$ is set), and blue for the loneliness level λ (for which we set, based upon Fig. 3, $\hat{\tau}=8$). As expected, adding more data results in better accuracy. More interestingly, using loneliness *in lieu* of label unbalance results in a significantly better accuracy, for the same quantity of data.

Fig. 5(b), which summarizes the accuracy improvement, offers a more detailed view. We can observe that adding one micro-dataset to the training yields an average accuracy improvement of 4% when the micro-dataset is chosen considering the label unbalance factor α, and over 7% when loneliness level λ is accounted for. In a situation where the datasets are not labeled and labeling is expensive, a Goldilocks procedure based on loneliness has the additional advantage that only the selected datasets will need to be labeled. On the contrary, a procedure based on the class imbalance factor would require one to label all datasets. This is especially advantageous in training scenarios where only few datasets can be used due to complexity constraints.

Next, in Fig. 5(c) and Fig. 5(d), we look in more detail at *when* extra data result in the largest accuracy improvement. Each marker in the plots corresponds to a micro-dataset, and its position along the x- and y-axes corresponds to the accuracy obtained using that micro-dataset alone and combining that dataset with additional one(s) using the Goldilocks procedure, respectively; the color of the marker corresponds to the metric employed. We can observe that datasets with lower accuracy tend to benefit the most from extra data. It is also interesting to observe how the improvement yielded by loneliness over label unbalance factor is larger when we have to select two micro-datasets than three. This confirms that loneliness is especially useful when the number of nodes that can be selected is small.

A. FL and additional datasets

To better assess how general our results are, we extend our performance evaluation to: (i) a FL scenario still using CIFAR, where each micro-dataset belongs to a different learning



(a) $m = 500$ (b) $m = 1000$ (c) $m = 2000$

(d) $m = 500$ (e) $m = 1000$ (f) $m = 2000$

Fig. 4. Training loss and KL terms in (4) for different micro-datasets generated from MNIST. The parameter m indicates the size of the micro-dataset.

node and learning nodes cooperate via the FedAvg algorithm; (ii) a centralized scenario using the CINIC10 dataset [15], developed as a more diverse, drop-in alternative to CIFAR; (iii) a centralized scenario using the GTSRB dataset [16], including real-world pictures of road signs. In FL, nodes cooperate through the FedAvg algorithm as implemented by the `flwr` library. Additional details on the datasets and implementation can be found in the Appendix.

The test accuracy achieved in the aforementioned cases is summarized in Fig. 6. Consistently with Fig. 5(a), more data always result in better accuracy; also, using loneliness as a metric yields consistently a better accuracy. In agreement with Fig. 5(c) and Fig. 5(d), the effect is more significant when choosing two micro-datasets than when choosing three.

Results with the FL, CINIC10 and GTSRB experiments are presented in Fig. 7, Fig. 8 and Fig. 9 (respectively), and are consistent with those of Fig. 5.

VII. Related Work

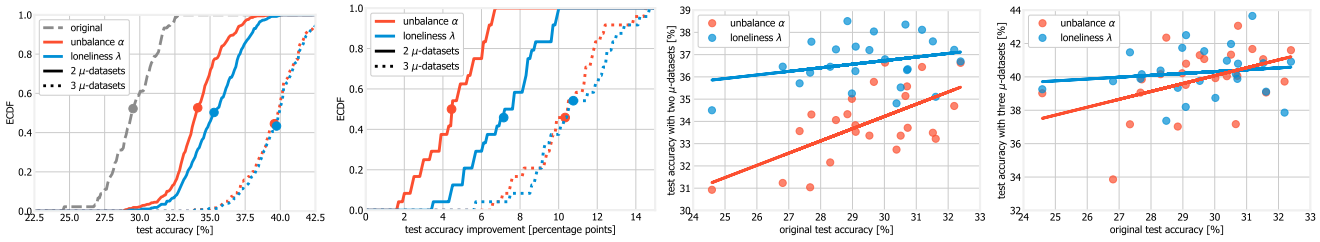
The problem of node selection in distributed learning and of dataset selection in conventional learning is well investigated. We review them here, organizing the literature into a taxonomy that highlights where our contribution fits.

Node selection: Node selection in cross-device FL is critical when data is heterogeneous and participation is intermittent. Naïve selection, as in FedAvg, can suffer from client drift under non-IID data [17], and uneven client participation creates a participation gap on top of the usual generalization gap [18]. Prior work can be grouped into four broad categories.

Label-aware approaches: These rely directly on labeled data or loss signals. For example, [18] propose sharing held-out labeled data at the server to estimate participation gaps, enabling sophisticated client-selection strategies [19]. In [20], instead, clients are selected on the basis of the local training loss, avoiding data sharing but still relying on label-based loss evaluation.

Data valuation and utility-based approaches: These methods quantify the marginal contribution of clients to the global model, typically inspired by data valuation or game-theoretic frameworks. Shapley-value estimators [21] and gradient-based utility estimates fall in this category. Reference [19] propose a greedy algorithm that directly leverages Shapley-value approximations for client selection. Reference [22] select clients with high local gradient norms at each round, while GPFL [23] compares local and global descent directions to judge client utility. FedGCS [24] further uses gradient-based optimization in a learned continuous space to score and select clients. Beyond these, Oort [25] guides client selection using a combined measure of statistical utility and system efficiency, while FedBalancer [26] adjusts client pacing and data contribution to optimize efficiency under heterogeneous data. FedGRA [27] selects clients using Grey Relational Analysis to combine device resource availability, training loss, and weight divergence.

System-aware approaches: FedCS [28] selects clients based on resource availability, while TiFL [29] tiers clients to mitigate stragglers. The primary goal of these works is to optimize runtime and resource efficiency.



(a) (b) (c) (d)

Fig. 5. Goldilocks performance on the CIFAR dataset: distribution of test accuracy (a) and of accuracy improvement (b), under different metrics and for different numbers of micro-datasets; relationship between initial accuracy and accuracy improvement when adding one (c) and two (d) micro-datasets.

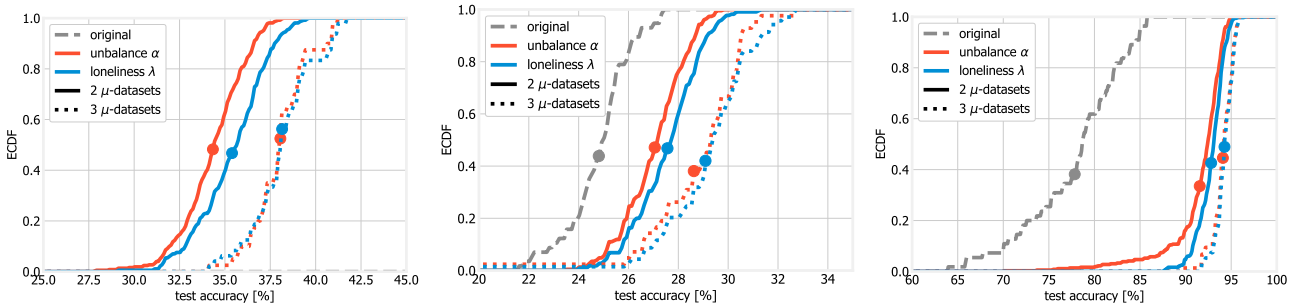


Fig. 6. Distribution of the test accuracy achieved for different numbers of micro-datasets and metrics in a FL scenario using the CIFAR dataset (left), a centralized scenario using the CINIC10 dataset (center), and a centralized scenario using the GTSRB dataset (right).

Privacy-aware approaches: Strategies that involve sharing held-out data [18] enable sophisticated selection but run counter to FL's central purpose of preserving privacy.

In contrast to these categories, our loneliness-based method provides a lightweight, privacy-preserving, and fully label-free signal that can front-load utility estimation before labels or steady participation are available.

Data selection: This is often referred to as data pruning in the literature [30], and involves pruning low-quality data, typically during the training process. An extensive review of data pruning methods can be found in [31]. Among the existing solutions, [21] uses the Shapely-value of a subset of data to estimate their utility. Reference [32] removes from the training set unforgettable examples, i.e., examples whose predicted label is correct and does not change over the training process. Reference [33], instead, retains "hard" examples, i.e., examples that have large ℓ_2-norm scores on trained models.

Coreset selection is another approach for data selection, with an extensive review provided in [34]. CRAIG [35] selects a subset of the training data that closely approximates the full gradient. K-Center Greedy [36] aims to select a subset such that every data point is close to at least one chosen point, thereby minimizing the largest distance between any data point and its nearest representative. Reference [37] proposes a squared-loss-minimization objective with loss-reduction attribution and the MRMC criterion, achieving strong performance across image recognition tasks.

In contrast, our loneliness-based method provides a simple and truly label-independent criterion that operates directly in the input space, without requiring loss signals, labels, or gradient information.

VIII. Conclusion

We have considered the problem of node selection in cooperative learning scenarios where only a small number of small datasets can be used. In this scenario, it is especially advantageous to select the datasets (and, hence, the nodes) *without* using label information, for both efficiency and privacy reasons. To this end, we proposed a metric called *loneliness*, which, using unlabeled data, computes the distance between samples of the same dataset. We integrated loneliness with a node- and data-selection strategy called Goldilocks, and found—across multiple datasets and in both centralized and federated schemes—that using loneliness consistently results in higher improvement of training accuracy, by over 70%, in spite of requiring no label information.

Acknowledgment

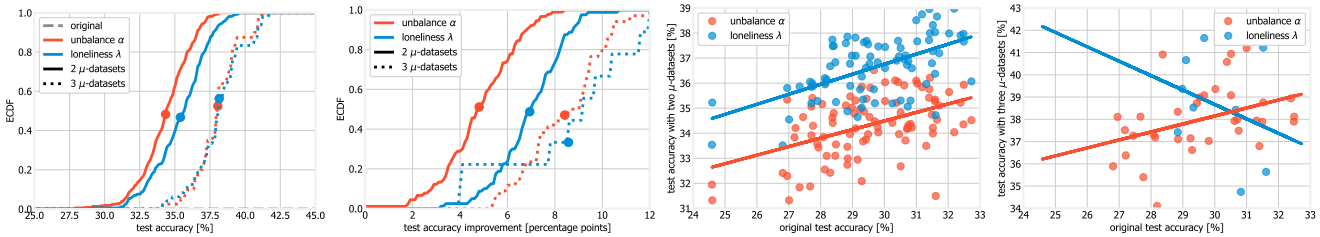
This work was supported by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU (PE00000001 – program "RESTART").

References

[1] M. Kamp, J. Fischer, and J. Vreeken, "Federated learning from small datasets," in *ICLR*, 2023.

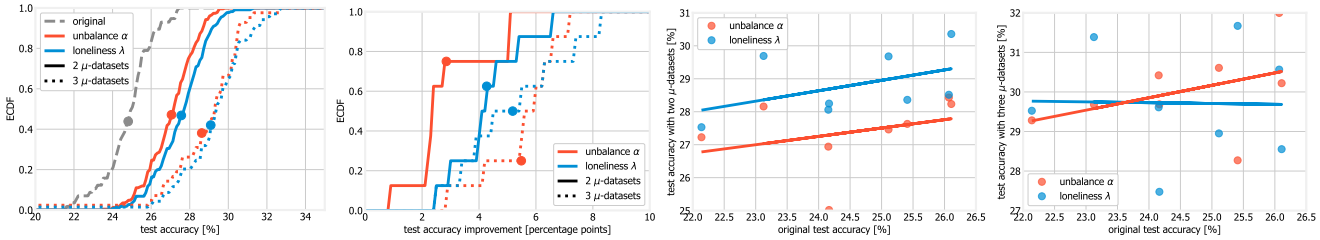
[2] H. Wu and P. Wang, "Node selection toward faster convergence for federated learning on non-iid data," *IEEE Transactions on Network Science and Engineering*, 2022.

[3] A. Li, L. Zhang, J. Tan, Y. Qin, J. Wang, and X.-Y. Li, "Sample-level data selection for federated learning," in *IEEE INFOCOM*, 2021.



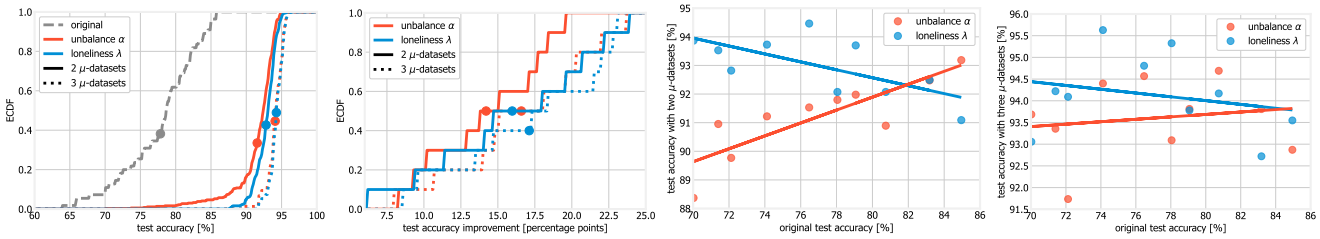
(a) (b) (c) (d)

Fig. 7. Goldilocks performance on the CIFAR dataset for a FL scenario: distribution of test accuracy (a) and of accuracy improvement (b) under different metrics and for different numbers of micro-datasets; relationship between initial accuracy and accuracy improvement when adding one (c) and two (d) micro-datasets.



(a) (b) (c) (d)

Fig. 8. Goldilocks performance on the CINIC dataset: distribution of test accuracy (a) and of accuracy improvement (b) under different metrics and for different numbers of micro-datasets; relationship between initial accuracy and accuracy improvement when adding one (c) and two (d) micro-datasets.



(a) (b) (c) (d)

Fig. 9. Goldilocks performance on the GTSRB dataset: distribution of test accuracy (a) and of accuracy improvement (b) under different metrics and for different numbers of micro-datasets; relationship between initial accuracy and accuracy improvement when adding one (c) and two (d) micro-datasets.

[4] Z. Li, Y. He, H. Yu, J. Kang, X. Li, Z. Xu, and D. Niyato, "Data heterogeneity-robust federated learning via group client selection in industrial IoT," *IEEE Internet of Things Journal*, vol. 9, 2022.

[5] S. Geerkens, C. Sieberichs, A. Braun, and T. Waschulzik, "Qi^2: an interactive tool for data quality assurance," *AI and Ethics*, 2024.

[6] C. Sieberichs, S. Geerkens, A. Braun, and T. Waschulzik, "ECS: an interactive tool for data quality assurance," *AI and Ethics*, 2024.

[7] H. Wu and P. Wang, "Fast-convergent federated learning with adaptive weighting," *IEEE TCCN*, 2021.

[8] S. Lotfi, M. Finzi, S. Kapoor, A. Potapczynski, M. Goldblum, and A. G. Wilson, "Pac-bayes compression bounds so tight that they can explain generalization," *Advances in Neural Information Processing Systems*, 2022.

[9] G. Wang and Y. Chen, "Robust feature matching using guided local outlier factor," *Pattern Recognition*, 2021.

[10] D. A. McAllester, "Pac-bayesian model averaging," in *COLT*, 1999.

[11] P. Sunehag and M. Hutter, "Algorithmic complexity," *International Encyclopedia of the Social & Behavioral Sciences*, 2015.

[12] S. Bansal, M. Bansal, R. Verma, R. Shorey, and H. Saran, "Fednse: Optimal node selection for federated learning with non-iid data," in *IEEE COMSNETS*, 2023.

[13] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[14] K. Dong, C. Zhou, Y. Ruan, and Y. Li, "MobileNetV2 model for image classification," in *IEEE ITCA*, 2020.

[15] L. N. Darlow, E. J. Crowley, A. Antoniou, and A. J. Storkey, "CINIC-10 is not imagenet or CIFAR-10," *arXiv preprint arXiv:1810.03505*, 2018.

[16] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The German traffic sign recognition benchmark: a multi-class classification competition," in *IEEE JCNN*, 2011.

[17] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International conference on machine learning*. PMLR, 2020.

[18] H. Yuan, W. Morningstar, L. Ning, and K. Singhal, "What do we mean by generalization in federated learning?" in *ICLR*, 2022.

[19] P. Singhal, S. R. Pandey, and P. Popovski, "Greedy shapley client selection for communication-efficient federated learning," *IEEE Networking Letters*, 2024.

[20] Y. J. Cho, J. Wang, and G. Joshi, "Towards understanding biased client selection in federated learning," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2022.

[21] A. Ghorbani and J. Zou, "Data shapley: Equitable valuation of data for machine learning," in *International conference on machine learning*. PMLR, 2019.

[22] O. Marnissi, H. E. Hammouti, and E. H. Bergou, "Client selection in federated learning based on gradients importance," *AIP Conference Proceedings*, 2024.

[23] S. Na, Y. Liang, and S.-m. Yiu, "GPFL: A gradient projection-based client selection framework for efficient federated learning," *IEEE Internet of Things Journal*, 2025.

[24] Z. Ning, C. Tian, M. Xiao, W. Fan, P. Wang, L. Li, P. Wang, and Y. Zhou, "FedGCS: A generative framework for efficient client selection in federated learning via gradient-based optimization," in *IJCAI*, 2024.
[25] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Oort: Efficient federated learning via guided participant selection," in *USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*, 2021.
[26] J. Shin, Y. Li, Y. Liu, and S.-J. Lee, "Fedbalancer: data and pace control for efficient federated learning on heterogeneous clients," in *Proceedings of the 20th Annual International Conference on Mobile Systems, Applications and Services (MobiSys)*, 2022.
[27] S. Chen, O. Tavallaie, M. H. Hambali, S. M. Zandavi, H. Haddadi, N. Lane, S. Guo, and A. Y. Zomaya, "Optimization of federated learning's client selection for non-iid data based on grey relational analysis," *arXiv preprint arXiv:2310.08147*, 2023.
[28] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *IEEE International Conference on Communications (ICC)*, 2019.
[29] Z. Chai, A. Ali, S. Zawad, S. Truex, A. Anwar, N. Baracaldo, Y. Zhou, H. Ludwig, F. Yan, and Y. Cheng, "TiFL: A tier-based federated learning system," in *Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing*, 2020.
[30] N. Sachdeva and J. McAuley, "Data distillation: A survey," *Transactions on Machine Learning Research*, 2023.
[31] C. Guo, B. Zhao, and Y. Bai, "Deepcore: A comprehensive library for coreset selection in deep learning," in *International Conference on Database and Expert Systems Applications*, 2022.
[32] M. Toneva, A. Sordoni, R. T. des Combes, A. Trischler, Y. Bengio, and G. J. Gordon, "An empirical study of example forgetting during deep neural network learning," in *ICLR*, 2019.
[33] M. Paul, S. Ganguli, and G. K. Dziugaite, "Deep learning on a data diet: Finding important examples early in training," *NeurIPS*, 2021.
[34] B. B. Moser, A. S. Shanbhag, S. Frolov, F. Raue, J. Folz, and A. Dengel, "A coreset selection of coreset selection literature: Introduction and recent advances," *arXiv preprint arXiv:2505.17799*, 2025.
[35] B. Mirzasoleiman, J. Bilmes, and J. Leskovec, "Coresets for data-efficient training of machine learning models," in *International Conference on Machine Learning*. PMLR, 2020.
[36] O. Sener and S. Savarese, "Active learning for convolutional neural networks: A core-set approach," in *ICLR*, 2018.
[37] J. Chen, "Efficient core-set selection for deep learning through squared loss minimization," in *International Conference on Machine Learning (ICMl)*, 2025.
[38] M. Kayed, A. Anter, and H. Mohamed, "Classification of garments from fashion MNIST dataset using CNN LeNet-5 architecture," in *IEEE ITCE*, 2020.
[39] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, 1998.
[40] T. M. Cover and A. T. Joy, *Elements of information theory*. Wiley-Interscience, 2006.

APPENDIX

Datasets, architectures, meta-parameters: For our experiments, we consider the following datasets:

MNIST [38], including $70\,000$ 28×28 black-and-white images of handwritten digits. The dataset is divided into a training set of $60\,000$ images and a testing set of $10\,000$ images, and comprises 10 classes, one per digit.

CIFAR [13], including $60\,000$ 32×32 color images of different objects and animals (airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks). The dataset is divided into a training set of $50\,000$ images and a testing set of $10\,000$ images, and comprises 10 classes, one per object.

CINIC [15], including $270\,000$ 32×32 color images: the $60\,000$ ones from CIFAR, plus $210\,000$ coming from ImageNet. It includes the same classes as CIFAR and is divided into train, test, and validation splits, each including $90\,000$ images. It is meant as a drop-in, more challenging replacement to CIFAR.

GTSRB [16] (German Traffic Sign Recognition Benchmark), including over $38\,000$ color, real-world images of road signs, belonging to 40 classes. Different images may have different size, hence, we need to resize them to 32×32; the dataset is divided into a training set of $26\,640$ images and a testing set of $12\,630$ images.

For MNIST, we use the LeNet5 convolutional network [38], including three convolutional layers and two fully-connected ones. For all other datasets, we use MobileNetV2 [14]. In all scenarios, we train for 50 epochs, and consider the one yielding the best performance. Stochastic gradient descent is used as an optimizer, with a learning rate of 10^{-3}. All experiments are performed with PyTorch, and training employs the `lightning` library.

Evaluation of the PAC-Bayes bound in (4)*::* We use the LeNet model [39] as the base model. Following [8, Sections 4.1 & 4.2], we train a compressed model (compressed size = 1000) for 1000 epochs using the Adam optimizer with a learning rate of 0.001. Then a quantized model (quantization levels = 7) is trained for 30 epochs using the Adam optimizer with a learning rate of 0.0001. The cross-entropy loss is used during training, and the 0–1 loss is used to compute the training loss in (4). The KL term in (4) is approximated as follows. Let W^* be the trained model obtained after the compression and quantization. Then the posterior is set as $P_{W|\mathbf{Z}} = \mathbb{I}_{W=W^*}$, where $\mathbb{I}$ is the indicator function. The KL term is then bounded as

$$\begin{aligned}\mathrm{KL}(P_{W|\mathbf{Z}}\|Q_W) = \mathrm{KL}(\mathbb{I}_{W=W^*}\|2^{-\mathcal{K}(W)/\gamma}) = \\ \log\left(\frac{1}{2^{-\mathcal{K}(W^*)/\gamma}}\right) \leq \mathcal{K}(W^*)\log 2 \overset{(\dagger)}{\leq} \rho(W^*)\log 2 \\ +2\log\rho(W^*),\end{aligned} \tag{6}$$

where $\rho(W^*)$ is the number of bits required to represent the weights of W^* and to obtain $(\dagger)$ we proceeded as in [40, Theorem 14.2.3]. We use arithmetic coding to compute $\rho(W^*)$.