Entropy-Regularized Probabilistic Gates for Sparse Model Discovery in Scarce-Data Federated Learning

Krishna Harsha Kovelakuntla Huthasana[1,*], Alireza Olama[1] and Andreas Lundell[1]

[1]Åbo Akademi University, Rantakatu 2, 65100, Vaasa, Finland

Abstract

Federated Learning (FL) is a distributed machine learning (ML) paradigm with collaboration among multiple clients without sharing data. FL is challenging under data heterogeneity and partial client participation. Learning sparse models is useful for communication and computational efficiency in FL, but it is especially difficult in the small-sample high-dimensional regime ($d \gg N$) where optimization can yield parameter configurations that fail to generalize to unseen test data. While magnitude-based pruning doesn't account for uncertainty exploration in the parameter space, a formulation with probabilistic gates and an L_0 constraint allows sampling from competing sparse configurations during training. In this work, we study entropy regularization of gate distributions as a mechanism to maintain uncertainty in sparse federated optimization by preventing early commitment to sparse support. We examine its impact under data heterogeneity, client participation heterogeneity, and sparsity. Experiments on synthetic and real-world benchmarks show consistent improvements over federated iterative hard thresholding (Fed-IHT) and pruning after dense federated averaging (FedAvg) training, both in statistical performance on test data and in sparsity recovery accuracy.

Keywords

Entropy Regularization, Sparsity, Federated Learning, Uncertainty, Parameter Exploration, Probabilistic Gates, Entropy Maximization, Sparse Federated Learning, L_0 Constraint

1. Introduction

Federated Learning (FL) algorithms operate in a distributed machine learning (ML) setting in which multiple clients collaborate to train [1]. This framework is characterized by privacy requirements of each client and avoids data sharing. While not all distributed settings are privacy-sensitive, FL is still beneficial because it eliminates the need for centralized data [2]. FL can be coordinated either by a single server or by clients communicating with one another. In this work, we study FL with central orchestration by a server to obtain a single global model as shown in Figure 1. A global model is typically learned by iteratively averaging parameters or gradients from clients and redistributing the global model to clients for further learning. However, statistical heterogeneity across clients and partial client participation during training pose challenges to the learning process in FL. Furthermore, sparse training and inference are desirable to improve generalizability [3] and enhance computational and communication efficiency in FL, thereby posing an additional challenge of discovering sparse models [4].

A common approach to inducing sparsity relies on L_1 and L_2 norms[1] for regularization, which depend directly on parameter magnitudes and offer varying levels of shrinkage. In contrast, using a magnitude-independent L_0 pseudo-norm is advantageous because it imposes a constant penalty on nonzero parameters and is useful for learning a model with a desired parameter density ρ. The Lagrangian for the L_0 density-constrained optimization problem in FL can be defined as:

$$\mathfrak{L}(\theta, \lambda) = \sum_{c=1}^{C} \frac{n_c}{N} \mathcal{L}^{(c)}(\theta) + \lambda \left(\|\theta\|_0 - \rho|\theta| \right), \quad \|\theta\|_0 = \sum_{j=1}^{|\theta|} \mathbb{I}[\theta_j \neq 0], \tag{1}$$

The Annual Symposium of Computer Science 2026

*Corresponding author.

kkovelak@abo.fi (K. H. K. Huthasana); alireza.olama@abo.fi (A. Olama); andreas.lundell@abo.fi (A. Lundell)

0009-0006-7106-3748 (K. H. K. Huthasana); 0000-0002-1787-3487 (A. Olama); 0000-0002-1548-1893 (A. Lundell)

© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

[1]For $\theta \in R^d$, L_1 norm is $\|\theta\|_1 = \sum_{i=1}^{d} |\theta_i|$ and L_2 norm is $\|\theta\|_2 = \left(\sum_{i=1}^{d} \theta_i^2\right)^{1/2}$.

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

published 2026-11-02

CEUR-WS.org/Vol-4309/paper14.pdf

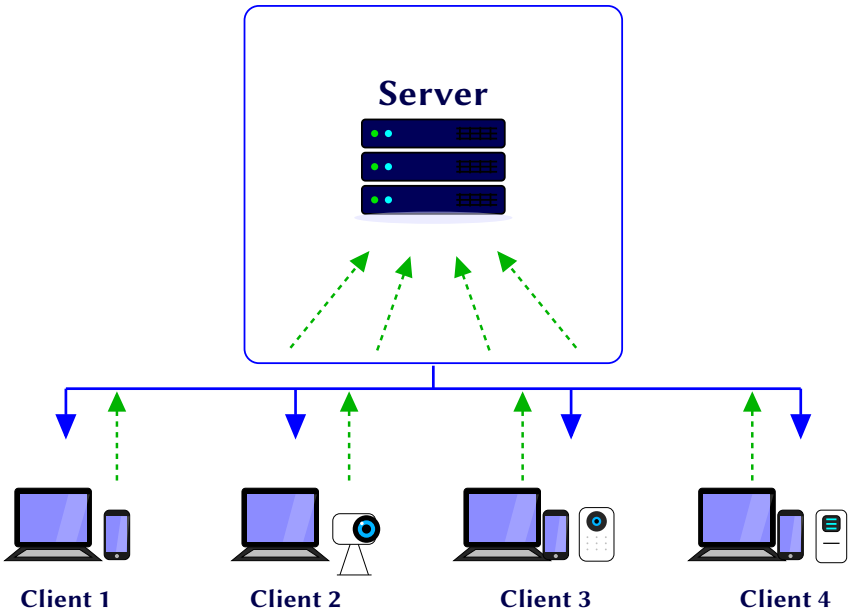


Figure 1: Client–server federated learning architecture with central orchestration. Solid arrows indicate the aggregated global model distributed to clients, while dotted arrows indicate local model updates sent from clients to the server.

where, $\mathcal{L}^{(c)}(\theta)$ denotes the normalized loss at client c, defined as:

$$\mathcal{L}^{(c)}(\theta) = \frac{1}{n_c} \sum_{i=1}^{n_c} \ell\big(h(x_i^{(c)}; \theta), y_i^{(c)}\big). \tag{2}$$

Here, C is number of clients in FL, each holding a local dataset $(D^{(c)})_{c=1}^{C} = (X^{(c)}, Y^{(c)})$, where $X^{(c)} \in \mathbb{R}^{n_c \times in}$, $Y^{(c)} \in \mathbb{R}^{n_c \times out}$, and $\sum_{c=1}^{C} n_c = N$. We assume a model $h(x;\theta) : \mathbb{R}^{in} \to \mathbb{R}^{out}$ and a loss function $\ell(h(x;\theta), y)$, where $x \in \mathbb{R}^{in}$, $y \in \mathbb{R}^{out}$, and $\theta \in \mathbb{R}^{d}$. The above formulation with Lagrange multiplier λ leads to a min-max optimization problem that can, in principle, be solved using gradient descent-ascent. However, the non-differentiability of the L_0 pseudo-norm complicates optimization within standard gradient-based learning frameworks.

Louizos et al. [5] introduce a reparameterization using stochastic gates $z \in \mathbb{R}^{p}$ with $\theta = \tilde{\theta} \odot z$[2] as the effective parameters of the model. By assuming a hard Concrete distribution over z (an approximation of the Bernoulli distribution), the L_0 norm is approximated by the expected number of active gates, i.e., $\sum_j \mathbb{E}_{q(z)}[z_j]$, enabling gradient-based optimization of the L_0-regularized objective. Gallego-Posada et al. [7] extended this framework by using an L_0 density constraint and solving the resulting min–max problem, thereby enabling the user to define the target sparsity in a centralized setting. Huthasana et al. [8] further adapted this approach to sparsity learning in the FL context, accounting for heterogeneity in data and client participation.

However, learning sparse models is challenging, particularly in the small-sample high-dimensional regime ($d \gg N$) [9, 10, 11, 12], where the optimization is unstable, resulting in multiple solutions of competing parameter configurations with poor sparsity recovery and generalization behaviour to unseen test data. Bao et al. [13] work on statistical recovery in a low sample–high dimensional regime, but a low sample size for the individual client relative to the number of parameters is considered, while the total sample size N far exceeds the number of parameters. We aim to study a low total sample size N relative to the number of parameters $d = |\theta|$ similar to a centralized setting, which is extremely challenging under data and client participation heterogeneity in FL.

Louizos et al. [5] points to penalizing relative entropy or the Kullback-Leibler divergence $\mathrm{KL}(q(z)\|p(z))$ in the optimization where $p(z)$ is the prior and $q(z)$ is the approximate posterior for uncertainty

[2] $\odot$ denotes the elementwise product [6].

exploration. The concept of entropy regularization is not new, and it is commonly used to encourage diversity and exploration at the decision level, distributions of class or action, in reinforcement learning (RL), and adapted to uncertainty exploration in input space, distributions of latent variables dependent on data, in variational auto–encoders [14, 15]. It is also used in Bayesian inference in centralized and FL settings [16, 17]. In their approach, the hard concrete gate distributions $q(z)$ are not directly dependent on the data and can be sampled independently of it post-training. In this work, the formulations at Louizos et al. [5], Gallego-Posada et al. [7] are adapted with relative-entropy penalization in a communication-efficient FL setting to explore uncertainty in non-zero parameter configurations and show that it consistently outperforms iterative hard-thresholding-based pruning during training [18] and post-training pruning of a dense model using the classic federated averaging algorithm [1].

The remainder of this paper is organized as follows. We first present the entropy-regularized L_0-constrained formulation for federated optimization, then the proposed distributed algorithm, followed by experiments in heterogeneous FL settings, and conclude.

2. Formulation

Assuming a model $h(x;\theta) : \mathbb{R}^{in} \to \mathbb{R}^{out}$ and a loss function $\ell(h(x;\theta), y)$, where $x \in \mathbb{R}^{in}$, $y \in \mathbb{R}^{out}$, and $\theta \in \mathbb{R}^{d}$, consider a centralized dataset $D = (X, Y)$ with $X \in \mathbb{R}^{N \times in}$ and $Y \in \mathbb{R}^{N \times out}$. Using [5, 7], the min–max objective can be defined using the expectation of the loss with respect to the distribution of gates, and the L_0 pseudo-norm approximated by the expected number of active gates, i.e., $\sum_j \mathbb{E}_{q(z|\phi)}[z_j]$ as shown in eq. 3. Since each z_j is a deterministic transformation of parameter-free noise, the expectation can be optimized using Monte Carlo sampling and reparameterized gradients [19, ch. 3.3.3].

$$\hat{\mathfrak{L}}(\tilde{\theta}, \phi, \lambda) = \frac{1}{R}\sum_{r=1}^{R}\left[\frac{1}{N}\sum_{i=1}^{N}\ell\big(h(x_i; \tilde{\theta} \odot z^{(r)}), y_i\big)\right] + \lambda\left(\sum_{j=1}^{|\theta|}\mathbb{E}_{q(z|\phi)}[z_j] - \rho|\theta|\right). \tag{3}$$

In a federated learning (FL) setting with C clients holding datasets $(D^{(c)})_{c=1}^{C} = (X^{(c)}, Y^{(c)})$, we consider a reparameterized linear model $h(x; \tilde{\theta} \odot z) : \mathbb{R}^{in} \to \mathbb{R}^{out}$ and a loss function $\ell(h(x; \tilde{\theta} \odot z), y)$. Here, $X^{(c)} \in \mathbb{R}^{n_c \times in}$, $Y^{(c)} \in \mathbb{R}^{n_c \times out}$, $\sum_{c=1}^{C} n_c = N$, $x \in \mathbb{R}^{in}$, $y \in \mathbb{R}^{out}$, and $\theta = \tilde{\theta} \odot z \in \mathbb{R}^{in}$. The gate parameters are defined as $\phi = \log \alpha \in \mathbb{R}^{in}$.

The Lagrangian corresponding to the entropy regularized L_0 density-constrained optimization problem is:

$$\hat{\mathfrak{L}}(\tilde{\theta}, \phi, \lambda) = \sum_{c=1}^{C}\frac{n_c}{N}\mathcal{L}^{(c)}(\tilde{\theta}, \phi) + \lambda\left(\sum_{j=1}^{|\theta|}\mathbb{E}_{q(z|\phi)}[z_j] - \rho|\theta|\right) + T\sum_{j=1}^{|\theta|}KL(q(z_j|\phi)||p(z_j|\phi_{init})). \tag{4}$$

Here, $p(z|\phi_{init})$ is a prior and is also a hard concrete distribution, and $\mathcal{L}^{(c)}(\tilde{\theta}, \phi)$ denotes the Monte Carlo estimate of the normalized loss at client c, defined as:

$$\mathcal{L}^{(c)}(\tilde{\theta}, \phi) = \frac{1}{R}\sum_{r=1}^{R}\frac{1}{n_c}\sum_{i=1}^{n_c}\ell\big(h(x_i^{(c)}; \tilde{\theta} \odot z^{(r)}), y_i^{(c)}\big). \tag{5}$$

The stochastic gates z are sampled using the Hard Concrete distribution, applying a hard-sigmoid transformation to a stretched Binary Concrete random variable [5, 20], defined as:

$$\begin{aligned} s &= \sigma\left(\frac{\log\frac{u}{1-u} + \log\alpha}{\beta'}\right), \quad u \sim \mathcal{U}(0,1), \\ \bar{s} &= s(\zeta - \gamma) + \gamma, \quad z = \min(1, \max(0, \bar{s})). \end{aligned} \tag{6}$$

The expectation of a gate being active [3] is derived at Louizos et al. [5] using the cumulative distribution function $Q(\bar{s})$. We introduced $T >= 0$, which we treat as a constant with or without decay to penalize

[3] $\mathbb{E}_{q(z|\phi)}[z_j] = 1 - Q(\bar{s}_j \le 0 \mid \phi_j) = \sigma\left(\log\alpha_j - \beta' \log\left(-\frac{\gamma}{\zeta}\right)\right)$

entropy. The resulting min–max optimization problem is:

$$\tilde{\theta}^*, \phi^*, \lambda^* = \arg\min_{\tilde{\theta},\phi} \arg\max_{\lambda \geq 0} \hat{\mathfrak{L}}(\tilde{\theta}, \phi, \lambda). \tag{7}$$

The parameters $\tilde{\theta}$ and $\phi = \log \alpha$ are jointly optimized using gradient descent with reparameterized gradients. λ is updated via gradient ascent and a restart strategy of resetting its value to 0 as and when the sparsity constraint is satisfied [7]. Since the hard concrete distribution is a continuos approximation of Bernoulli distribution the computation of $KL(q(z_j)||p(z_j))$ involves an additional term involving $\bar{s}$, and thus, $z \in (0, 1)$ which can be computed using truncated distribution $q(\bar{s}|\bar{s} \in (0, 1))$ or a Monte-carlo estimate of the same. We used the closed-form expressions presented in the appendix A provided at Louizos et al. [5].

3. Algorithm

We use the notations $\mathcal{L}_{\text{Con}}(\phi)$ and $\mathcal{L}_{\text{KL}}(\phi)$ for L_0 density constraint and the $KL(q(z)||p(z))$ in (4)[8]. The Lagrangian can then be written as:

$$\hat{\mathfrak{L}}(\tilde{\theta}, \phi, \lambda) = \sum_{c=1}^{C} \frac{n_c}{N} \mathcal{L}^{(c)}(\tilde{\theta}, \phi) + \lambda \, \mathcal{L}_{\text{Con}}(\phi) + T \, \mathcal{L}_{\text{KL}}(\phi). \tag{8}$$

McMahan et al. [1] propose federated averaging (FedAvg), a distributed algorithm for learning a global model via synchronous updates from clients using gradient averaging. In this setting, a central server coordinates training across C clients each holding a local dataset $D^{(c)}$. Each client performs stochastic gradient descent (SGD) updates locally for few iterations before communicating the parameters to the server, to reduce the communication between server and clients by increasing computation at clients. The server aggregates the parameters by averaging to obtain the global model.

At iteration t, each client performs the following updates:

$$\tilde{\theta}_c^{t+1} = \tilde{\theta}_c^t - \eta_{\tilde{\theta}} \nabla_{\tilde{\theta}} \mathcal{L}^{(c)}(\tilde{\theta}^t, \phi^t), \tag{9}$$

$$\phi_c^{t+1} = \phi_c^t - \eta_\phi \left(\nabla_\phi \mathcal{L}^{(c)}(\tilde{\theta}^t, \phi^t) + \lambda \, \nabla_\phi \mathcal{L}_{\text{Con}}^{(c)}(\phi^t) + T \, \nabla_\phi \mathcal{L}_{\text{KL}}^{(c)}(\phi^t) \right), \tag{10}$$

$$\lambda_c^{t+1} = \lambda_c^t + \eta_\lambda \, \mathcal{L}_{\text{Con}}^{(c)}(\phi^t). \tag{11}$$

At each round, a fraction γ_c of clients is selected uniformly at random, resulting in $K = \lfloor \gamma_c C \rfloor$ participating clients. In practice, a few mini-batches of uniform size B sampled iteratively at each client to locally update the model $n(B)$ times per communication round or a full pass over client data amount to a local epoch. The server gathers all updates from clients and performs a synchronous update of global model , using the averages of the updates from clients, after each communication round or global epoch keeping T constant or decaying it according to a predetermined schedule. The Lagrange parameter λ is reset to zero when the constraint is satisfied [7]. The aggregation weights w_k for clients sampled in an epoch or round can be uniform or proportional to the number of samples each client holds.

This approach enables learning a global sparse model in FL with entropy regularization and an L_0 constraint using probabilistic gates. We refer to this variant of FedAvg as E-FLoPS, where E stands for entropy regularization. The learning rates $\eta_{\tilde{\theta}}$ and η_ϕ needs to be appropriately tuned. The learning rate η_λ for the Lagrange parameter updates is set in the order of $1/|\theta|$. The temperature $T \in (0, \infty)$ is also initialized at $1/|\theta|$ and increased if needed to encourage uncertainty exploration as the total sample size N decreases. The gate parameters $\phi = \log \alpha$ are initialized from a normal distribution with mean $\log \rho_{\text{init}} - \log(1 - \rho_{\text{init}})$ and variance 0.01, where ρ_{init} controls the initial density. The target density is denoted by $\rho_{\text{targ}} = \rho$. The Hard Concrete distribution parameters are set to $\gamma = -0.1$, $\zeta = 1.1$, and β' is recommended to be set at 0.66, following Louizos et al. [5]. The Lagrange multiplier is initialized as $\lambda = 0$. Client participation variability is simulated by randomly selecting a fraction of clients at each training round.

E-FLoPS achieves test-time sparsity by using deterministic gates $\hat{z}$ sampled without noise or smoothing [7]. For exact sparsity, the raw parameters $\tilde{\theta}$ for the top-m indices of effective parameters θ are pruned, where $m = \lfloor \rho_{\text{targ}} \cdot |\theta| \rfloor$. This pruning mechanism can be applied from a pre-defined threshold of epochs, referred to as the *prune start* epoch, which is set to 0 for E-FLoPS to improve communication efficiency via serialization for message compression [8]. At this stage, the serialized mean gate values, from repeated sampling, corresponding to top-m indices of gates z are retained, and the rest of the gates are replaced with their average, denoted by z_{-m}^{avg}. Only the pruned $\tilde{\theta}$ and z, along with their non-zero indices and the scalar z_{-m}^{avg}, are communicated. Upon reception, parameters are reconstructed via $\theta = \tilde{\theta} \odot z$ and aggregated, and ϕ is recovered from z using $\phi{=}\beta' \log\left(\frac{z}{1-z}\right)$, ignoring the noise component of the gates. The communication cost reduction by this approach in FL is significant for small ρ_{targ} with minimal meta data overhead.

Algorithm 1 FedAvg variant E-FLoPS. E and B denote the number of epochs and the mini-batch size.

Initialization: $(\tilde{\theta}^{(0)}, \phi^{(0)})$, compute $\theta^{(0)} = \tilde{\theta}^{(0)} \odot z^{(0)}$
for epoch $b = 1$ to E **do**
  Sample client subset S_t
  for each client $k \in S_t$ **do**
    $(k, \tilde{\theta}_k, z_k, \lambda_k) \leftarrow$ **ClientCompute**$(k, \tilde{\theta}^{(t)}, z^{(t)}, \lambda^{(t)})$
  end for
  Server aggregation:

$$\theta = \sum_{k \in S_t} w_k \tilde{\theta}_k$$
$$z = \sum_{k \in S_t} w_k z_k$$

  Reconstruct $\tilde{\theta} = \theta \oslash z$[4]and recover ϕ from z using $\phi = \beta' \log\left(\frac{z}{1-z}\right)$
  Perform server-side updates on $(\tilde{\theta}, \phi, \lambda)$
  Either decay or keep T constant
  if $b >$ prune start **then**
    prune $\tilde{\theta}$ using top-m θ top-m indices of θ
    prune z with tail statistic z_{-m}^{avg}
    Serialize and communicate
  end if
end for
ClientCompute$(k, \tilde{\theta}, z, \lambda)$:
  Recover ϕ_k from z using $\phi{=}\beta' \log\left(\frac{z}{1-z}\right)$
  Sample mini-batches $b \sim D^{(k)}$ of size B
  Perform $n(B)$ local SGD steps on $(\tilde{\theta}_k, \phi_k)$ and ascent for λ_k
  return Communicate pruned and serialized $(\tilde{\theta}_k, \phi_k)$ and λ_k

4. Experiments

We include experiments on synthetic and real-world datasets. We evaluate true sparsity recovery in linear regression (LR) on synthetically generated data, convolutional neural network (CNN) on the MNIST digit classification data, and softmax multi-class classifier on the Golub leukemia cancer classification data. We compare our method with the federated iterative hard thresholding algorithm (Fed-IHT) proposed by Tong et al. [18], where a hard-thresholding operation is used to retain only top-m parameters in absolute magnitude at each iteration or epoch along with federated averaging to enforce

[4] $\oslash$ represents an element wise division.

sparsity, and with classic federated averaging (`Fed-Avg`) with dense training and pruning after the last training epoch. For `FedAvg` the training time performance on test data is evaluated using the top-m parameters after each epoch though the training is dense without imposing sparsity. An approximate upper bound is established by centralized training by pooling data from clients with distributional shifts. A tuning phase, marked by a vertical dotted line in all figures in the experiments section, is conducted to further improve the statistical performance of the models with the fixed sparse support discovered by the end of the training phase.

In FL, comunication is a bottle neck and needs to be minimized. The federated averaging variant `E-FLoPS` enables sparse communication throughout training without compromising statistical performance. The theoritical uplink and downlink communication costs can be estimated as multiples of the message size and the number of communication rounds, assuming 4 bytes per parameter and index each. The total two way communication cost per client in each server round in FL is,

$$\texttt{FedAvg} : \text{epochs} \times 4|\theta|, \ \texttt{E-FLoPS} : \text{epochs} \times 4 \cdot (2\rho_{\text{targ}}|\theta|), \text{ and } \texttt{FedIter-HT} : \text{epochs} \times 4 \cdot (\rho_{\text{targ}}|\theta|).$$

Thus, the communication cost of `E-FLoPS` is more than `FedIter-HT`, but significantly lower than dense training via `FedAvg`. For large model sizes $|\theta|$ and small target densities ρ_{targ}, the gap between `FedAvg` and `E-FLoPS` becomes substantial, while small with that of `FedIter-HT`.

The experiments are conducted on an Apple MacBook with an M4 Pro chip (12-core CPU) and 24 GB unified memory, running macOS 15.5. The implementation uses Flower (v1.29.0) framework for simulation of FL and PyTorch (v2.7.0) framework for training, with Python (v3.12.7).

4.1. Experiments on Synthetic Data

We generated synthetic data for sparse linear regression following the procedure from Bertsimas et al. [21] for a range of $\frac{N}{|\theta|}$. For a parameter vector dimension $d = |\theta|$, each row $x_i \in \mathbb{R}^d$ of $X \in \mathbb{R}^{N \times d}$ is sampled from a zero-mean Gaussian distribution with covariance matrix Σ. We use a Toeplitz covariance structure defined as $\Sigma_{ij} = \rho_{\text{cor}}^{|i-j|}, \quad i, j = 1, \ldots, d$. An m-sparse coefficient vector $w_{\text{true}} \in \mathbb{R}^d$ is constructed, where $m = \lfloor \rho \cdot d \rfloor$. A subset of indices $S_m \subseteq \{1, \ldots, d\}$ is selected uniformly at random, and coefficients are assigned as $(w_{\text{true}})_j \sim \text{Unif}\{-1, 1\}$ for $j \in S_m$ and $(w_{\text{true}})_j = 0$ otherwise. The response vector is generated as $y = Xw_{\text{true}} + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, \sigma^2 I_N)$. The signal-to-noise ratio (SNR) is defined as $\text{SNR}=\frac{\|Xw_{\text{true}}\|_2^2}{\|\varepsilon\|_2^2}$, and the noise level is set to $\sigma=\frac{\|Xw_{\text{true}}\|_2}{\sqrt{\text{SNR}}\sqrt{N}}$. For $|\theta|$=1000, we generate a number of total training samples N for varying ratios of $\frac{N}{|\theta|}$ with 5000 samples of test data in all cases. The generated training data is then distributed among clients.

We employed affine shifting for attribute distribution skew, Dirichlet partitioning protocol (DPP) for skew in the number of samples available at a client [22, 23]. A fraction of clients (0.6) is randomly sampled in each epoch to introduce participation heterogeneity [22]. We evaluate performance using mean squared error (MSE) and R^2. Experiments are conducted at a correlation level of $\rho_{\text{cor}} = 0.2$ and an SNR of 20, with a true sparsity level of 95% or target density of 5%.

As shown in Figure 2, `E-FLoPS` achieves a higher R^2 on unseen test data than `FedIter-HT` at all $\frac{N}{|\theta|}$. `E-FLoPS` also exceeds top-m model of densely training `FedAvg` during training and the pruned and fine-tuned model discovered post training. The Figure 2 also shows the stability of across varying learning rates of $\tilde{\theta}$, the raw parameters, and ϕ, the parameters of the gates. For stable training of `E-FLoPS`, we fix $\eta_\lambda=\frac{0.01}{|\theta|}$, and tune learning rates within the ranges $\eta_{\tilde{\theta}} \in [0.05, 0.5]$ and $\eta_\phi \in [0.1, 0.9]$. Table 1 shows the accuracy of sparsity recovery $A(\theta)=\frac{TP}{(TP+FN)}$, the fraction of true features recovered. `E-FLoPS` has higher accuracy than both `Fed-IHT` and `FedAvg` except at smallest $\frac{N}{|\theta|}$.

4.2. Experiments on Real Data

4.2.1. Image Classification

We consider publicly available *MNIST* data [24]: A multi-class classification dataset of handwritten digits with 28×28 gray-scale pixel values for each. We use a CNN with two 5×5 convolutional layers

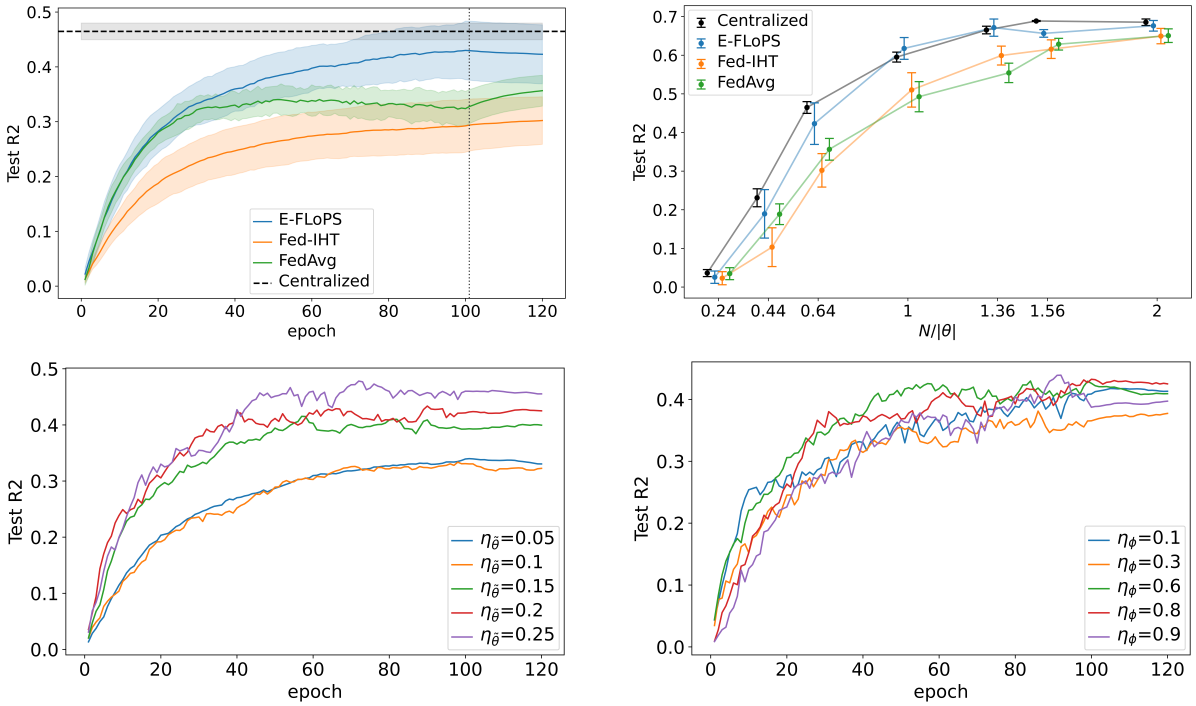


Figure 2: The figures show (a) mean test R^2 and standard deviation for $\frac{N}{|\theta|}$=0.64 over 30 runs of all the algorithms, (b) test R^2 over varying $\frac{N}{|\theta|}$ by changing total number of samples available across all clients,(c) test R^2 at $\eta_{\phi} = 0.85$ for varying $\eta_{\tilde{\theta}}$, and (d) test R^2 at $\eta_{\tilde{\theta}} = 0.25$ for varying η_{ϕ}.

Table 1
Final support recovery across total sample sizes (N) at $d = |\theta|$ showing mean ± standard deviation of $A(\theta)$ over 30 different run seeds. The federated method with the best support recovery is indicated in bold in each row, and the centralized method is shown as a reference upper bound.

$\frac{N}{d}$	E-FLoPS	Fed-IHT	FedAvg	Centralized
0.24	0.140 ± 0.039	0.129 ± 0.039	**0.179 ± 0.038**	0.195 ± 0.027
0.44	0.379 ± 0.082	0.292 ± 0.044	**0.427 ± 0.036**	0.473 ± 0.027
0.64	**0.626 ± 0.060**	0.490 ± 0.055	0.566 ± 0.038	0.736 ± 0.026
1.00	**0.828 ± 0.034**	0.687 ± 0.072	0.741 ± 0.023	0.840 ± 0.016
1.36	**0.925 ± 0.026**	0.831 ± 0.032	0.803 ± 0.030	0.945 ± 0.010
1.56	**0.920 ± 0.016**	0.874 ± 0.032	0.890 ± 0.110	0.980 ± 0.000
2.00	**0.941 ± 0.021**	0.913 ± 0.028	0.917 ± 0.027	0.987 ± 0.010

of 6 and 16 channels, each followed by 2×2 max pooling, and three fully connected layers with widths 120, 84, and 10 for this classification task resulting in a non-linear model with 44,426 parameters. The targeted density is 2.5% (sparsity of 97.5%).

The data is downsampled per class at a ratio of ~0.02, resulting in 1184 training samples and 10000 test samples. The data is distributed to clients using DPP for label skew and only a fraction of clients(0.6) are randomly sampled to train in each round. The affine shifts for further distributional heterogeneity are not performed in this case. For evaluation, we use cross-entropy (CE) loss with classification accuracy. Figure 3 shows that `E-FLoPS` and `FedIter-HT` have similar test-time performance and communication efficiency. However, `E-FLoPS` shows a greater reduction of Multiply Accumulate Operations (MACs) leading to computational efficiency at inference time.

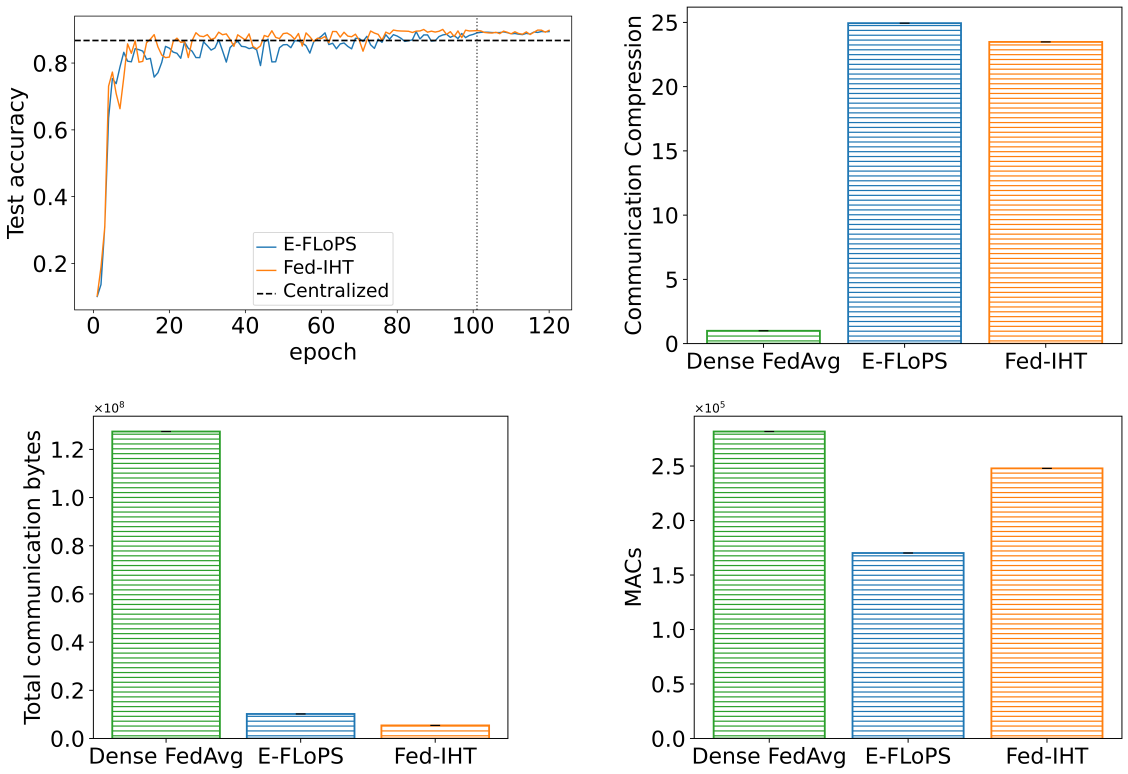


Figure 3: The figures show: a) Test accuracy over epochs at 97.5% sparsity at $\frac{N}{\theta}$~0.02, b) Communication compression compared to dense training, c) Total communication bytes between server and clients in training and fine-tuning, and d) MACs reduction.

4.2.2. Leukemia Classification

We use the publicly available Golub leukemia gene-expression dataset from the study on molecular classification of acute leukemia types, acute lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML), using gene expression [25]. The data consist of 72 patient samples with 3571 features after initial preprocessing. Each feature represents the measured expression level of a gene/probe in a patient sample; larger values indicate higher abundance of the corresponding gene transcript. These gene-expression profiles are used to classify samples as ALL or AML. This dataset combines a very small sample-to-dimension ratio, $\frac{N}{|\theta|}$=0.02, with dense, high-dimensional gene-expression features. It also represents a realistic setting in which data privacy may be a central concern, since the samples correspond to patient-level biomedical measurements.

We randomly select 15 samples for the test set and distribute the remaining samples across 5 clients using DPP at a fixed split seed. We train a softmax classifier at a target density of 0.1% (99.9% sparsity), with all clients participating. We do not synthetically alter client data using affine shifts. Figure 4 shows that E-FLoPS outperforms the other variants over 30 runs with different seeds for run-time stochastic processes. We confirmed that the difference between E-FLoPS and Fed-IHT is significant using a paired t-test, with a t-statistic of 3.79 and a p-value of 7×10^{-4}. We also performed experiments across 30 train/test splits generated with different split seeds to ensure that the results were not driven by a particularly favorable random partition of the data. Across these splits, E-FLoPS again outperforms Fed-IHT and FedAvg. Table 2 presents the most frequently selected genes across different run seeds. The most frequently selected genes by E-FLoPS, cystatin C and myeloperoxidase, are supported by prior experimental research on their roles in promoting metastasis and cancer spread [26].

[5]The gene "major histocompatibility complex, class I, B" is represented as "histocompatibility complex" for legibility in the Table 2.

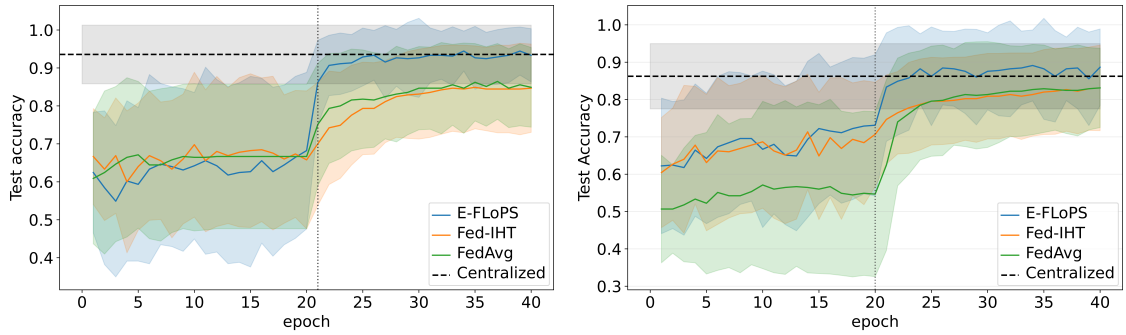


Figure 4: The figure shows a) Test accuracy over epochs at 99.9% sparsity at $\frac{N}{|\theta|}$~0.02 at a fixed data/split seed and 30 different run seeds and b) Test accuracy over epochs at 99.9% sparsity at $\frac{N}{|\theta|}$~0.02 at a fixed run seed and 30 different data/split seeds.

Table 2
The table shows the top genes selected by each method. Values in parentheses indicate the proportion of runs in which a particular gene is selected[5].

E-FLoPS	Fed-IHT	FedAvg	Centralized
cystatin C (0.90)	ferritin, light polypeptide (0.70)	azurocidin 1 (0.33)	interleukin 8 (0.73)
myeloperoxidase (0.80)	**cystatin C** (0.67)	interleukin 8 (0.33)	ferritin, light polypeptide (0.70)
ferritin, light polypeptide (0.80)	azurocidin 1 (0.57)	ferritin, light polypeptide (0.30)	**myeloperoxidase** (0.67)
interleukin 8 (0.53)	histocompatibility complex (0.57)	**myeloperoxidase** (0.27)	**cystatin C** (0.63)
defensin, alpha 1 (0.43)	hemoglobin, beta (0.57)	**cystatin C** (0.23)	glycoprotein B (0.53)
azurocidin 1 (0.33)	interleukin 8 (0.47)	interleukin 8 (0.17)	azurocidin 1 (0.47)
histocompatibility complex (0.23)	**myeloperoxidase** (0.43)	elastase 2, neutrophil (0.17)	histocompatibility complex (0.37)

5. Conclusion

Prior work has shown that sparse models can reduce communication overhead, memory footprint, and possibly inference cost, while also improving generalization. However, the challenge of sparse recovery in ($d \gg N$) regime under heterogeneous FL conditions remains underexplored. We study entropy regularization of probabilistic gates for sparse FL under an L_0 density constraint. The proposed federated averaging algorithm, `E-FLoPS`, extends sparse federated optimization with a mechanism for uncertainty-driven exploration of sparse parameter configurations. This is particularly relevant in ($d \gg N$) regime, where optimization may otherwise commit early to suboptimal sparse configurations with poor generalization.

We formulate a differentiable, entropy-regularized, L_0-constrained objective using Hard Concrete gates, and show how to optimize it in a federated setting using reparameterized gradients at clients and aggregation of sparse updates. The resulting method helps achieve a user-defined target sparsity while improving exploration of the parameter space during learning.

Experiments on synthetic linear regression, CNN for image classification and softmax classifier for leukemia classification show that `E-FLoPS` consistently improves sparsity recovery and statistical performance at the target sparsity compared with iterative hard-thresholding-based sparse federated training, `Fed-IHT`, and post-training pruning of dense `FedAvg`. These gains are achieved with communication efficiency comparable to `Fed-IHT`.

We show that `E-FLoPS` is effective for both linear and non-linear models at high sparsity levels. Understanding how entropy regularization affects sparse recovery differently in data-scarce versus overparameterized regimes is an interesting direction for future work, along with extending the method to structured sparsity.

Acknowledgments

This work has been supported by FAST, the Finnish Software Engineering Doctoral Research Network,

funded by the Ministry of Education and Culture in Finland.

Declaration on Generative AI

During the preparation of this work, the author(s) used Grammarly in order to: Grammar and spelling check in Overleaf. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

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

[2] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al., Advances and open problems in federated learning, Foundations and trends® in machine learning 14 (2021) 1–210.

[3] R. Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society Series B: Statistical Methodology 58 (1996) 267–288.

[4] J. Wang, Z. Charles, Z. Xu, G. Joshi, H. B. McMahan, B. A. y Arcas, M. Al-Shedivat, G. Andrew, S. Avestimehr, K. Daly, D. Data, S. Diggavi, H. Eichner, A. Gadhikar, Z. Garrett, A. M. Girgis, F. Hanzely, A. Hard, C. He, S. Horvath, Z. Huo, A. Ingerman, M. Jaggi, T. Javidi, P. Kairouz, S. Kale, S. P. Karimireddy, J. Konecny, S. Koyejo, T. Li, L. Liu, M. Mohri, H. Qi, S. J. Reddi, P. Richtarik, K. Singhal, V. Smith, M. Soltanolkotabi, W. Song, A. T. Suresh, S. U. Stich, A. Talwalkar, H. Wang, B. Woodworth, S. Wu, F. X. Yu, H. Yuan, M. Zaheer, M. Zhang, T. Zhang, C. Zheng, C. Zhu, W. Zhu, A field guide to federated optimization, 2021. URL: https://arxiv.org/abs/2107.06917. arXiv:2107.06917.

[5] C. Louizos, M. Welling, D. P. Kingma, Learning sparse neural networks through l_0 regularization, arXiv preprint arXiv:1712.01312 (2017).

[6] R. A. Horn, The hadamard product, in: Proc. symp. appl. math, volume 40, 1990, pp. 87–169.

[7] J. Gallego-Posada, J. Ramirez, A. Erraqabi, Y. Bengio, S. Lacoste-Julien, Controlled sparsity via constrained optimization or: How i learned to stop tuning penalties and love constraints, Advances in Neural Information Processing Systems 35 (2022) 1253–1266.

[8] K. H. K. Huthasana, A. Olama, A. Lundell, Federated learning with l0 constraint via probabilistic gates for sparsity, 2025. URL: https://arxiv.org/abs/2512.23071. arXiv:2512.23071.

[9] G. Reeves, M. C. Gastpar, Approximate sparsity pattern recovery: Information-theoretic lower bounds, IEEE Transactions on Information Theory 59 (2013) 3451–3465.

[10] A. K. Fletcher, S. Rangan, V. K. Goyal, Necessary and sufficient conditions for sparsity pattern recovery, IEEE Transactions on Information Theory 55 (2009) 5758–5772.

[11] T. J. Thomas, J. S. Rani, Recovery from compressed measurements using sparsity independent regularized pursuit, Signal Processing 172 (2020) 107508.

[12] G. Reeves, M. Gastpar, The sampling rate-distortion tradeoff for sparsity pattern recovery in compressed sensing, IEEE Transactions on Information Theory 58 (2012) 3065–3092.

[13] Y. Bao, M. Crawshaw, S. Luo, M. Liu, Fast composite optimization and statistical recovery in federated learning, in: international conference on machine learning, PMLR, 2022, pp. 1508–1536.

[14] S. An, N. Jammalamadaka, E. Chong, Maximum entropy information bottleneck for uncertainty-aware stochastic embedding, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 3809–3818.

[15] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, A. Lerchner, Understanding disentangling in β-vae, arXiv preprint arXiv:1804.03599 (2018).

[16] Z. Abboud, H. Lombaert, S. Kadoury, Sparse bayesian networks: efficient uncertainty quantification

in medical image analysis, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2024, pp. 675–684.
[17] X. Zhang, W. Li, Y. Shao, Y. Li, Federated learning via variational bayesian inference: Personalization, sparsity and clustering, arXiv preprint arXiv:2303.04345 (2023).
[18] Q. Tong, G. Liang, J. Ding, T. Zhu, M. Pan, J. Bi, Federated optimization of l0-norm regularized sparse learning, Algorithms 15 (2022) 319.
[19] R. Ranganath, Black Box variational inference: Scalable, generic Bayesian computation and its applications, Ph.D. thesis, Princeton University, 2017.
[20] C. J. Maddison, A. Mnih, Y. W. Teh, The concrete distribution: A continuous relaxation of discrete random variables, arXiv preprint arXiv:1611.00712 (2016).
[21] D. Bertsimas, J. Pauphilet, B. V. Parys, Sparse regression: Scalable algorithms and empirical performance, Statistical Science 35 (2020) pp. 555–578. URL: https://www.jstor.org/stable/26997931.
[22] A. Reisizadeh, F. Farnia, R. Pedarsani, A. Jadbabaie, Robust federated learning: The case of affine distribution shifts, Advances in neural information processing systems 33 (2020) 21554–21565.
[23] D. Solans, M. Heikkila, A. Vitaletti, N. Kourtellis, A. Anagnostopoulos, I. Chatzigiannakis, et al., Non-iid data in federated learning: A survey with taxonomy, metrics, methods, frameworks and future directions, arXiv preprint arXiv:2411.12377 (2024).
[24] Y. LeCun, The mnist database of handwritten digits, http://yann. lecun. com/exdb/mnist/ (1998).
[25] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, et al., Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, science 286 (1999) 531–537.
[26] Y. Liang, C. Liu, X.-Z. Luan, K.-S. Leung, T.-M. Chan, Z.-B. Xu, H. Zhang, Sparse logistic regression with a l1/2 penalty for gene selection in cancer classification, BMC bioinformatics 14 (2013) 198.

A. KL-Divergence for Hard Concrete Gates

The stochastic gates follow the Hard Concrete distribution [5], obtained by stretching a Binary Concrete random variable $s \in (0, 1)$:

$$\bar{s} = s(\zeta - \gamma) + \gamma, \quad z = \min(1, \max(0, \bar{s})), \tag{12}$$

with $\gamma < 0$ and $\zeta > 1$. This induces a mixed distribution over z with point masses at 0 and 1, and a continuous component on $(0, 1)$:

$$q(z \mid \phi) = Q_{\bar{s}}(0)\,\delta(z) + (1 - Q_{\bar{s}}(1))\;\delta(z-1) + (Q_{\bar{s}}(1) - Q_{\bar{s}}(0))\,q_{\bar{s}}(z \mid \bar{s} \in (0,1)), \tag{13}$$

$$\text{where,} \quad Q_{\bar{s}}(\bar{s} \mid \phi) = Q_s\left(\frac{\bar{s}-\gamma}{\zeta-\gamma}\middle|\phi\right) \quad \text{and,} \quad Q_s(s \mid \phi) = \sigma(\beta'(\log s - \log(1-s)) - \phi)\,. \tag{14}$$

we define a hard concrete prior $p(z)$ with the same support and then the KL-divergence decomposes as:

$$\mathrm{KL}(q(z)\|p(z)) = Q_{\bar{s}}(0)\log\frac{Q_{\bar{s}}(0)}{P_{\bar{s}}(0)} + (1 - Q_{\bar{s}}(1))\log\frac{1-Q_{\bar{s}}(1)}{1-P_{\bar{s}}(1)} \tag{15}$$

$$+ (Q_{\bar{s}}(1) - Q_{\bar{s}}(0))\,\mathbb{E}_{q_{\bar{s}}(z|\bar{s}\in(0,1))}\big[\log q_{\bar{s}}(z) - \log p_{\bar{s}}(z)\big]. \tag{16}$$

The first two terms correspond to masses at $z = 0$ and $z = 1$, and the last term accounts for the continuous component. In practice, this divergence is evaluated either using the closed-form expressions if available or Monte Carlo estimation. We used the closed form expressions provided at Louizos et al. [5].