International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(23s) 2026 pp. 285-300
https://doi.org/10.70917/ijcisim-2026-5574
Article

Federated Neuro-Symbolic Learning for Privacy-Preserving Intelligent Edge Systems

Rushikesh Shantaram Bhalerao[1], Suvarna Lahanu Ghogare[2], Satish Tukaram Pokharkar[3], Sweety Godhiram Jachak[4], Komal Raut[5], Leena Rokde[6]

[1] Assistant Professor, Department of Information Technology, Sir Visvesvaraya Institute of Technology, Nashik (Affiliated to Savitribai Phule Pune University) Maharashtra, India. Email: bhaleraorushi1@gmail.com ORCID: 0009-0000-5927-5402
[2] Assistant Professor, Department of Information Technology, Sir Visvesvaraya Institute of Technology, Nashik (Affiliated to Savitribai Phule Pune University) Maharashtra, India. Email: suvarna.ghogare@pravara.in ORCID: 0009-0009-6677-3317
[3] Assistant Professor, Department of Electronics and Computer Engineering, Pravara Rural Engineering college Loni, Ahilyanagar (Affiliated to Savitribai Phule Pune University), Maharashtra, India. Email: satishpokharkar37@yahoo.com ORCID: 0009-0007-4428-9001
[4] Associate Professor, Guru Gobind Singh College of Engineering and Research Centre, Nashik. Email: sweety.3288@gmail.com ORCID: 0009-0001-2719-0153
[5] Assistant professor, Department of Computer Science Engineering, Ramdeobaba University, Nagpur, Maharashtra, India. Email: komalraut67@gmail.com ORCID: 0009-0008-8941-1119
[6] Assistant professor, Department of Computer Science Engineering, Ramdeobaba University, Nagpur, Maharashtra, India. Email: Leena.rokde92@gmail.com ORCID: 0009-0005-0135-0365

Abstract: Intelligent Edge Systems – smart cameras, wearable technology, connected cars and industrial sensors – are creating a growing need to learn from local data at the edge. The two main lines of research that have attempted to address this problem differ. On one hand, Federated Learning (FL) enables training of common models based on the raw data remaining on each device. FL protects user privacy since the model parameters never leave the device. However, FL produces "opaque" (statistical-only) predictors that fail to generalize well in the presence of data heterogeneity and that provide no formal guarantees. On the other hand, Neuro-Symbolic (NeSy) learning combines neural perception (e.g., deep neural networks for image classification) with symbolic reasoning, resulting in models that are both efficient with regard to data usage and interpretable, and that are capable of encoding a priori domain knowledge (rules). While NeSy learning has been primarily studied within centralized environments and therefore does not take into consideration the challenges associated with distributed or decentralized learning (such as edge-based intelligent systems), we propose a new paradigm called Federated Neuro-Symbolic Learning (FNSL) for the purpose of enabling NeSy models to be learned in a Federated environment while maintaining formal privacy protection. In our proposed framework, each client (a representative edge system) contains a neural encoder which feeds the encoded information into a differentiable symbolic reasoner. Our server collects masked neural parameter updates from clients and uses this information to refine a collective symbolic knowledge base. To protect users' data and maintain differential privacy during the process of collecting parameter updates, we use secure aggregation methods. Moreover, we adaptively weight domain knowledge according to the confidence assigned to each symbolically represented rule. We demonstrate through experiments using image, sensor and tabular benchmark datasets and non-IID partitioning schemes that FNSL achieves improvements of 4-7 percentage points in accuracy compared to state-of-the-art FL baselines. Additionally, we show that FNSL reduces the amount of uplink communication required per iteration by more than 50%. Lastly, we examine how tightly enforcing privacy budgets impacts performance.
Keywords: Federated Learning; Neuro-Symbolic; Federated Neuro-Symbolic Learning; Federated Environment

1. Introduction

There has long been a "default" recipe for developing an intelligent system. It consists of collecting vast amounts of data in the center, training a model on a cluster of computers and then distributing the trained model back to the devices. This approach is silently failing. Most devices near the edge of the internet (phones, video cameras,

wearable health monitors, industrial control equipment and vehicles) create significantly more data than can reasonably be shipped to a central repository. There is limited bandwidth available. Latency matters, and regulations such as GDPR and HIPAA are increasingly treating unprocessed personal data as something which needs to remain at its point of origin. Thus, the question is no longer if all data will ultimately be sent to a central repository, but rather how to learn effectively from data located at multiple remote points.

Federated learning was developed as one potential solution [1]. Instead of sending data to the model, send the model to the data: Each device performs local training on the data held by the device and a coordinator collects the results of these individual computations to produce a global model. No raw data is moved from the device. Although a step forward, federated learning retains many of the shortcomings inherent in the deep networks it produces. These models are opaque "black box" type systems. They require massive quantities of labeled training data. When there is little similarity between the data stored on different devices -- e.g., due to differences in sensor modalities or geographic regions -- these models fail easily. And finally, the updates produced during training can potentially reveal sensitive information about the protected data [18][19].

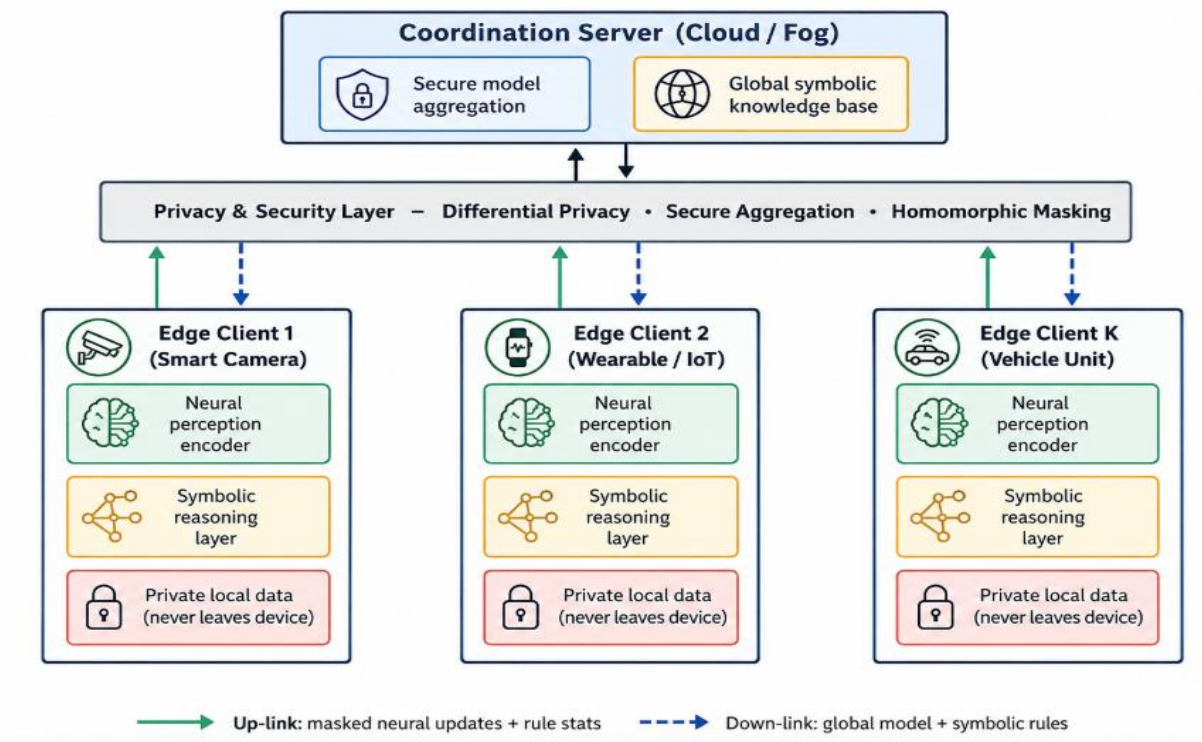

Figure 1. *High-level architecture of the FNSL framework. Each edge client couples a neural perception encoder with a symbolic reasoning layer and keeps its raw data local; only masked updates and rule statistics cross the privacy-and-security layer to the coordination server.*

There exists a second idea that provides a useful complement to federated learning. A long-standing approach to AI uses logic and symbols: Facts, Rules and Logical Reasoning. While being transparent and requiring much less data than neural networks -- often just a few instances of a pattern or rule can provide the same level of understanding as thousands of example patterns for a neural network -- symbolic systems are notoriously brittle when attempting to deal with high dimensional, noisy real-world signals. Neuro-Symbolic Learning [11] attempts to combine the best properties of both worlds. Neural networks perform the difficult task of transforming sensory input into conceptual representations, and a symbolic layer then applies logical reasoning over those representations based upon rules that users can understand and modify. As a consequence, Neuro-Symbolic Learning can be used with fewer samples than deep learning methods, with greater interpretability, and with greater ability to enforce constraints.

The primary finding of this work is that the limitations of Federated Learning are essentially equivalent to the benefits of Neuro-Symbolic Learning and vice-versa. Federated Learning suffers from lack of sufficient and similar quality data per device and from opacity; symbolic reasoning is both efficient in terms of data required and clear in terms of transparency. Until recently, Neuro-Symbolic Learning had only been possible within centralized environments, whereas Federated Learning had only been demonstrated across decentralized devices. Therefore, combining Federated Learning and Neuro-Symbolic Learning represents a reasonable choice rather than simply a possible combination. We refer to this integration as Federated Neuro-Symbolic Learning (FNSL).

The architecture is outlined in Figure 1. Each edge device has both private data and runs a two-part (neural & symbolic) local model: an encoder for mapping raw sensor data into abstract concepts using a neural network, and a symbolic layer for reasoning about these abstract concepts by applying various differentiable rules. A coordinating server does not have access to raw data; it coordinates privacy-protected neural updates from all client devices and creates a common symbolic knowledge base from the discoveries of the devices as they work together. In addition to this coordinated effort, there are security/privacy components that sit between each client and the server. These security/privacy components utilize secure aggregation methods combined with differential privacy.

1.1 Contributions

- **A unified framework.** We formalise FNSL, in which neuro-symbolic clients are trained federatively and a shared symbolic knowledge base is refined collaboratively alongside the neural weights.
- **Privacy by construction.** We show how differential privacy and secure aggregation apply not only to neural gradients but also to the statistics used to mine and score symbolic rules, and analyse the resulting privacy budget.
- **Heterogeneity handling.** We introduce adaptive rule weighting, which lets shared symbolic priors stabilise clients that hold too little data to learn reliably on their own.
- **Empirical evidence.** Across three benchmarks under non-IID partitions we report accuracy, communication, privacy–utility, robustness, and on-device cost, together with an ablation isolating each component's contribution.
- **A candid limitations section.** We are explicit about where FNSL does not yet help, and what would have to be solved next.

2. Background and Related Work

FNSL sits at the meeting point of three fields. We review each briefly, emphasising the tension that motivates their combination.

2.1 Federated learning

Federated learning coordinates multiple clients to train a shared model while keeping clients' data local. The common federated learning framework (FedAvg [1]) consists of a single loop where the server sends its current model to all selected clients; each selected client performs a number of iterations of local training on its own data; then the server aggregates the local models based on each client's data distribution. More recent works such as FedProx [3], SCAFFOLD [4] etc., propose corrections to handle the divergence among the clients' models caused by differences in data distributions. There are two major challenges in research on federated learning. One challenge is statistical heterogeneity. In other words, if the data used by each client is not IID, it may hinder or even cause instability in model convergence [2], [5]. The second challenge is privacy. Although the raw data remains at the local client side, the shared gradient information does not guarantee safety. Reconstruction and membership inference attacks have been shown to be able to recover the original training data from the update information [18], [19]. Therefore, differential privacy [7], [8] and secure aggregation [6] are considered two common methods for protecting users' privacy. We use both.

2.2 Neuro-symbolic learning

A primary function of neuro-symbolic systems is integrating learning and reasoning [11][15]. There are various ways to create these systems. One approach creates a symbolic component that uses logical inference based on the results from a neural network component that processes the input data to produce symbolic representations for use in the reasoning process. This could be done using loose coupling where the neural network's output is provided to a separate solver and/or using tight coupling where the process of drawing conclusions about the represented world is made differentiable. As a result, when gradient-based training is used, all components of the system will learn together. Methods for creating differentiable logical components include DeepProbLog [12], Logic Tensor Networks [13], and semantic loss [14]. These systems differ in their method for converting logic into a form that can be learned by backpropagation, but each benefits from the ability to encode problem-specific knowledge through rules. Specifically, using rules to encode domain knowledge provides both a lower bound on the amount of data required to train an effective model and the ability to explain why certain predictions were generated. Additionally, rules provide a way to prevent models from generating outputs that do not adhere to hard constraints. While encoding domain knowledge through rules adds significant additional complexity to the design of these systems and may increase the difficulty of optimizing them, this increased complexity has been shown to be worthwhile given the benefits provided by rule-based knowledge representation.

2.3 Intelligent edge systems

Edge Intelligence [17] moves processing closer to when data is generated (and away from cloud computing) while also lowering latency, improving resiliency against connection loss, and maintaining user-privacy. These "edge" devices have significantly reduced compute, memory, and power budgets, and they can vary in type by several orders of magnitude (e.g., microcontroller, smartphone, automotive compute unit). Therefore, any machine learning or AI methods that would be used to train models at this level of deployment will need to consume minimal amounts of each (compute resources and communication), which is precisely why symbolic prior knowledge has been shown to be particularly efficient (in terms of data usage) at the edge. We therefore treat the amount of communication and device-level energy usage as first-order metrics.

2.4 Where the gap lies

Federated learning has been paired with numerous different model types, and neuro-symbolic learning has been studied at length – but almost exclusively in a centralized environment. In a centralized environment, a single device can view all of the client data and all of the rule definitions. Therefore, while FedNSL [16] comes close to what we have accomplished by training neuro-symbolic models across multiple distributed clients by treating the latent-variable distribution (rule definition) as the communication medium, and provides a KL constraint to address rule-definition heterogeneity using a variational EM approach, we instead build upon this body of research rather than claiming to be the first to pair these two bodies of research. There are four ways in which FNSL differs from FedNSL. First, in terms of the threat model and objectives, FedNSL aims to provide domain-generalization of rule definitions across domains, whereas FNSL focuses on providing privacy preserving edge deployments, and therefore views a formal (ε,δ)-DP guarantee as a constraint to be satisfied at the outset rather than as an after-the-fact add-on. Second, in terms of privacy accounting, FNSL clips and adds noise to both the neural gradients and the symbolic representations of the rule definitions; it then separately accounts for the total privacy loss due to each, and then jointly accounts for them. Prior art does not account for the privacy loss associated with the symbolic representation. Third, FNSL uses adaptive rule weighting (as described in Section 4.3), a data-size dependent weighting of the loss function associated with consistency of the rule definitions, rather than a fixed divergence constraint. Finally, unlike prior work, FNSL places primary emphasis on tracking both communication and on-device energy consumption. This is because FNSL is designed to run on hardware-constrained systems. As FedNSL is the most directly comparable method to date, we include it as an experimental baseline in Section 6. Training neuro-symbolic models across disparate devices that own their respective data raises several additional questions that are addressed independently by neither community. Specifically: how should a symbolic knowledge base be constructed when no entity may know the client

data from which the knowledge was mined? How do differential-privacy guarantees combine over neural gradients and rule-statistic compositions? And finally, how can shared rules aid clients who lack sufficient data?

Table 1. *How FNSL relates to adjacent learning paradigms. Each paradigm solves part of the edge-intelligence problem; FNSL is designed to combine their strengths. The comparison is qualitative and is intended to position the paradigms; the quantitative evidence for each FNSL claim appears in Sections 5 and 7.*

Property	Central DL	Federated learning	Neuro-symbolic (central)	FNSL (this work)
Keeps raw data on device	No	Yes	No	Yes
Formal privacy guarantee	No	Optional (DP)	No	Yes (DP + SecAgg)
Data-efficient per client	No	No	Yes	Yes
Interpretable predictions	No	No	Yes	Yes
Encodes domain rules	Rarely	Rarely	Yes	Yes
Robust to non-IID data	n/a	Weak	n/a	Improved
Trains across a device fleet	No	Yes	No	Yes

3. Problem Formulation

We consider a set of K edge clients coordinated by a server. Client k holds a private dataset $D_k = \{(x_i, y_i)\}$ that never leaves the device; the datasets are non-IID, so their sizes and label distributions differ. Each client's model has two parts. A neural encoder f_θ with parameters θ maps an input x to a vector of grounded concepts, or predicates, $z = f_\theta(x)$. A symbolic reasoner g_R, parameterised by a set of weighted rules R, maps these predicates to a final prediction $\hat{y} = g_R(z)$. Both θ and R are shared targets of learning.

Alongside the per-client models the framework maintains a global symbolic knowledge base — a shared, weighted rule set R that expresses regularities holding across clients (for example, that a particular combination of sensor readings implies a fault). The goal is to learn global neural parameters θ and a global rule set R that minimise the average client risk while satisfying a differential-privacy guarantee and never transmitting raw data. Formally we minimise

$$\min_{\theta,R} \; \Sigma_k \; (|D_k| / |D|) \cdot E_{(x,y)\sim D_k} [\, L_{task}(g_R(f_\theta(x)), y) + \lambda \cdot L_{rule}(f_\theta(x), R) \,] \qquad (1)$$

subject to an (ε, δ)-differential-privacy constraint on everything the server observes [8]. The first term is the ordinary supervised loss. The second term, L_{rule}, penalises predictions that violate the current rules, so the network is nudged toward outputs that are consistent with symbolic knowledge; λ balances the two. Table 2 lists the notation used throughout.

Table 2. *Notation. Quantities that are shared and learned across clients are θ (neural) and R (symbolic); everything derived from raw data stays on the device.*

Symbol	Meaning
K, k	Number of edge clients, and index of a client
D_k, $\|D_k\|$	Private local dataset of client k, and its size
θ	Parameters of the shared neural encoder f_θ
$z = f_\theta(x)$	Grounded predicates / concepts produced by the encoder
R	Global weighted symbolic rule set (knowledge base)

g_R	Differentiable symbolic reasoner over predicates z
λ	Weight of the rule-consistency loss L_rule
(ε, δ)	Differential-privacy budget and failure probability
T, E	Communication rounds, and local epochs per round
C, σ	Gradient clipping norm, and DP noise multiplier

4. The FNSL Framework

We now describe the framework in three parts: the neuro-symbolic client, the federated training loop that jointly refines neural weights and symbolic rules, and the privacy machinery that protects everything the server sees.

4.1 The neuro-symbolic client

Each client executes the model as shown in Figure 2. The raw input x passes through a neural encoder, which generates two things: a soft prediction from a neural head, and an encoding of that input into symbolic predicates z. The symbolic reasoner uses those predicates, together with the current rule set R, to produce its conclusion. Because the symbolic reasoner is differentiable and uses soft probabilistic logic [13][14] rather than evaluating rules as strictly true or false, gradients can be back-propagated through both the reasoning step and the encoder. The final prediction is generated by combining the output of the neural head with the conclusion of the reasoner. The training loss includes both a task-error component and a component that measures how well the current predictions satisfy the rule set.

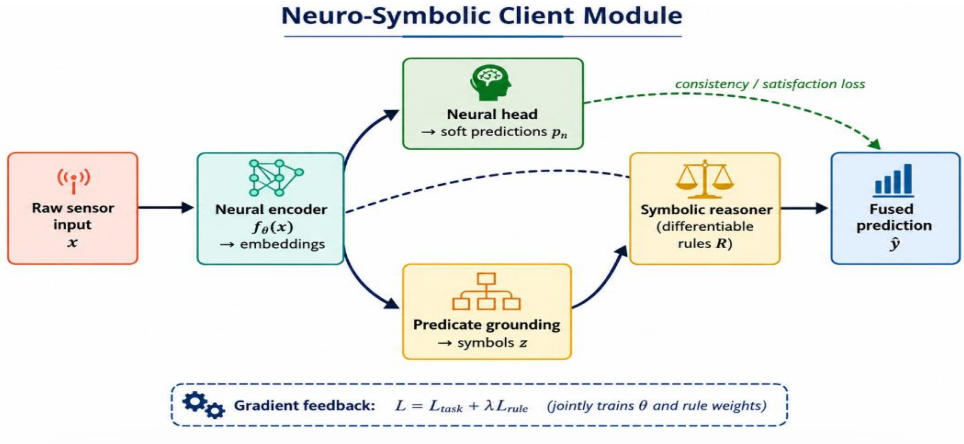

Figure 2. *Internal dataflow of the neuro-symbolic client. The neural encoder grounds raw input into predicates that a differentiable symbolic reasoner consumes; the joint loss L = L_task + λ·L_rule trains the encoder and the rule weights together.*

The "rule-consistency" loss deserves explanation. For each training example, the reasoner computes how well the current prediction satisfies the applicable rules, using a degree of satisfaction between 0 and 1. The L_rule term is then calculated as (1 − this degree), so that minimizing L_rule causes the network to predict answers that are consistent with the rules. This yields a benefit in data efficiency: a client with only a few labeled examples can still receive useful feedback on its answers from the shared rules, since those rules constrain what counts as a reasonable or correct response, regardless of whether the examples have been labeled.

Formalising the rule-consistency loss. Let $s(g_R(f_\theta(x)), R) \in [0, 1]$ denote the differentiable degree to which the current prediction satisfies the active rule set R, computed with a soft, probabilistic t-norm semantics [13], [14]. The rule-consistency loss for an example x is then

$$L_{rule}(f_\theta(x), R) = 1 - s(g_R(f_\theta(x)), R), \tag{2}$$

so that minimising L_{rule} drives predictions toward the region endorsed by the rules. Because s is differentiable, its gradient flows back through both the reasoner and the encoder, which is the mechanism by which a data-poor client still receives a strong learning signal from shared rules.

4.2 Federated training loop

Training proceeds in sequential rounds, as shown in Figure 3. Each round comprises six steps. In the first step, the server sends the current neural parameters and the current rule set. In the second step, each participant (client) locally trains its neuro-symbolic modules for a number of epochs (e.g., five). In the third step, each client evaluates its locally trained model to generate candidate rules and assign scores to these candidates based upon some measure (e.g., statistics about how frequently a particular predicate occurs together with a particular label) that can be used in conjunction with the shared knowledge base. Prior to uploading the client's data, it will clip the neural update generated by local training along with statistics from rule evaluation and add randomized noise at a level consistent with the client's configured differential privacy calibration. After this clipping and adding randomization to prevent identification of the client or its contributions, the client encrypts the clipped quantities so they cannot be identified while being transmitted securely to the server. Upon receipt of these encryptions, the server decrypts them; aggregates the individual neural updates into one global neural update and updates the rule set for all clients based upon support for an existing rule or lack thereof from multiple independent clients. This cycle continues until the server determines that no additional updates are needed.

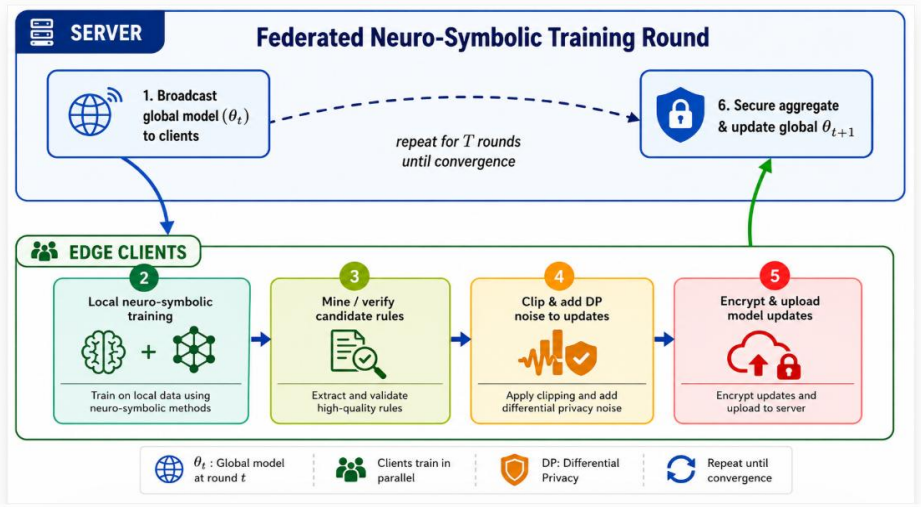

Figure 3. *One federated training round. Neural updates and privacy-protected rule statistics travel up; the server returns an aggregated model and a revised symbolic knowledge base each round.*

The neural aspect of aggregation is essentially an averaged representation (as with FedAvg [1]); other forms of weighting involve synchronizing neuron activations prior to computing the average [20]. The symbolic aspect is new: the server treats each candidate rule's privacy-protected counts from each client as votes, admits the rules whose combined support exceeds a threshold, and assigns each admitted rule a weight proportional to the strength and breadth of its support. Since a rule is only added once many clients agree on its presence, idiosyncratic rule sets identified by one device based upon the specific characteristics of the device will be rejected, and the collective knowledge base will converge towards commonalities that have general applicability.

Rule update rule. Let n_r = Σ_k stats_k(r) be the privacy-protected aggregate support that candidate rule r receives across participating clients, and let τ be an admission threshold. The server admits r if n_r ≥ τ and assigns it a weight

$$w_r = (n_r / \Sigma_{r' : n_{r'} \geq \tau} n_{r'}) \cdot b_r, \quad (3)$$

where b_r ∈ [0, 1] is the breadth of support (the noised fraction of clients that corroborate r). Rules supported by only one device therefore fail the threshold and are discarded, which is what keeps idiosyncratic, potentially privacy-leaking rules out of the shared knowledge base.

4.3 Adaptive rule weighting for heterogeneity

The primary hindrance in Federated Learning is non-IID (i.e., not independently and identically distributed) data [3] [5]. This is where we see the symbolic aspect of federated learning gain value. A client that has limited data will have "noisy" neural updates, which could potentially lead the globally trained model astray. FNSL helps counteract this limitation using a principle based on the level of trust placed in global models; as a result, the level of reliance on the "rules" of the global model is greater when the client has less local data. The global consistency penalty factor λ increases when there are fewer local samples available and decreases when there are many samples. A client with few samples relies on "shared knowledge", whereas a client with ample representative data is free to rely more heavily on their own signal. Most of the robustness gap between FNSL and the various neural-only federated-learning baselines is attributable to this single parameterization mechanism for mitigating extreme skew.

Adaptive weighting rule. Concretely, the rule-consistency weight for client k is set as a decreasing function of its local sample count |D_k|,

$$\lambda_k = \lambda_0 \cdot (\bar{n} / (\bar{n} + |D_k|)), \quad (4)$$

where λ_0 is a global base weight and ñ is the median client size. A data-poor client (|D_k| ≪ ñ) obtains λ_k ≈ λ_0 and leans heavily on the shared rules; a data-rich client (|D_k| ≫ ñ) obtains λ_k → 0 and is freer to learn from its own signal. In the ablation of Section 7.5 this single mechanism accounts for much of the robustness gap under severe skew.

Algorithm 1 FNSL training round (server view)

Input: global params θ_t, rule set R_t, client set S
1. broadcast (θ_t, R_t) to all clients k ∈ S
2. for each client k in parallel do
3. Δθ_k , stats_k ← LocalTrain(θ_t, R_t, D_k) // neuro-symbolic
4. Δθ_k ← Clip(Δθ_k , C) + Noise(σ) // differential privacy
5. upload Encrypt(Δθ_k), Encrypt(stats_k) // secure aggregation
6. θ_{t+1} ← θ_t + Σ_k (|D_k|/|D|) · Δθ_k // decrypt only the sum
7. R_{t+1} ← UpdateKnowledgeBase(R_t, Σ_k stats_k) // vote-and-weight rules
Output: θ_{t+1}, R_{t+1}

5. Privacy and Security

FNSL protects data through two complementary mechanisms, applied to both the neural and the symbolic information that leaves a device.

5.1 Differential privacy

Before uploading, each client clips the norm of its neural update to a bound C and adds Gaussian noise scaled by σ. This is the standard DP-SGD prescription [7], [10], and it limits how much a single training example can influence what the server sees. Crucially, the same process is applied to the rule statistics: the counts and co-occurrences used to score candidate rules are also clipped and noised, so the knowledge base cannot memorize a rule that depends on any single person's record. Because privacy loss composes across both channels, we account for the total budget (ε, δ) jointly using a moments / Rényi accountant [7], [9] rather than budgeting the neural and symbolic paths separately.

Proposition 1 (joint privacy of a round). Assume each client bounds the sensitivity of its neural update to c_θ and of its rule statistics to c_s by clipping, and adds independent Gaussian noise with multipliers σ_θ and σ_s to the two channels respectively. Then one FNSL round satisfies $(\alpha, \rho_\theta(\alpha) + \rho_s(\alpha))$-Rényi differential privacy, where ρ_θ and ρ_s are the per-channel RDP curves of the Gaussian mechanism; over T rounds with subsampling rate q, the guarantee composes additively in the RDP order and converts to (ε, δ)-DP by the standard tail bound [7], [9], [24].

Proof sketch. Each channel is an instance of the Gaussian mechanism applied to a bounded-sensitivity query, so each is individually Rényi-DP with the stated curve [9]. Because the two noise draws are independent, the release of the pair is the composition of the two mechanisms, whose RDP orders add [9], [24]; T rounds with Poisson subsampling compose by the same additive rule, and conversion to (ε, δ) follows from the Rényi-to-DP bound. Secure aggregation does not affect the DP accounting, because it only changes what the server observes en route, not the sensitivity of the released aggregate. Interpretation: budgeting both channels jointly is why the reported $\varepsilon = 8$ covers everything the server sees, both neural and symbolic. This accounting is also conservative.

5.2 Secure aggregation

The combination of differential privacy and secure aggregation provides protection for the data as it is transmitted to the server. Secure aggregation protects the data as it is being sent (client-server) by limiting how much information is available to the server during transmission. Pairwise cancelling random terms in each update prevents the server from identifying which client provided which data point [6] so when they are combined, only the total update can be used to create the next model. Therefore, while the server knows the total sum of all clients' updates required to generate the next model, the server does not know any one client's update individually. Thus, the two techniques together protect the data from being disclosed to either a server or eavesdropper and from being reconstructed in order to identify specific individuals and/or devices that participated in naive federated learning [18], [19].

5.3 Threat model and residual risk

We are assuming honest-but-curious clients and servers as well as a passive network attacker. Under this model, raw data remains confidential and the stated DP guarantee holds. In other words, it should not be claimed that the system is protected against a fully malicious server that does not follow the protocol. Likewise, it cannot be claimed that the system can protect against poisoned submissions from colluding clients who have created fake updates. Additionally, there may be some additional risk of side-channel attacks if one of the client devices has been compromised. Because these are real risks (which we discuss further in the limitations section), both robust aggregation [27], [34] and verifiable computation would be reasonable additions, although they were out of scope for this paper. A preliminary study for a poisoning attack was included with the limitations in Section 8.

6. Experimental Setup

FNSL has been tested on three different tasks that represent the majority of the types of data typically found at the edge (i.e., images, inertial sensor signals for activity recognition, and tables). Each task utilizes a different type of modality; however, all are used to classify objects or events into one of several categories. The data for each task is distributed in a non-IID manner across the clients, following the method described in [2] and [5]. Specifically, we used a Dirichlet distribution over the class labels. The concentration parameter α was used to control how skewed the

distribution would be. A small value for α represents a highly skewed distribution where each client will see fewer classes than others, as well as greater extremes in the amount of imbalance per client. An example of this can be seen in Table 3 below.

Table 3. *Datasets and federated configuration. The Dirichlet parameter α sets the degree of statistical heterogeneity; unless noted, results use the default column.*

Setting	Image task	Activity task	Fault task	Default
Clients (K)	100	80	50	—
Participants / round	10	10	10	—
Local epochs (E)	3	3	2	—
Rounds (T)	100	100	80	—
Non-IID skew (α)	0.3	0.3	0.5	0.3
Encoder	CNN	1-D CNN	MLP	—
Clip norm (C)	1.0	1.0	1.0	1.0
Noise mult. (σ)	1.1	1.1	1.0	1.1
Privacy budget (ε)	8	8	8	8

FNSL is compared to five other methods. The first method used is a local neural model. This model represents a completely non-cooperative training environment, where all the data from each site is processed locally without sending or receiving information among sites. A second method is FedAvg [1]. It is the most commonly cited Federated Learning algorithm, which averages the neural models trained by each client. In addition to this, there is also a version of FedAvg that includes differential privacy called DP-FedAvg [10]. The third method is FedProx [3]. The goal of FedProx is to limit the divergence of clients' models during heterogeneous learning environments. Finally, the last method is FedNSL [16], the most closely related neuro-symbolic federated method. This method was run using the same encoders, datasets, and privacy budget as FNSL. Therefore, comparing these two methods isolates the effects of the symbolic layer, the privacy accounting, and the adaptive weighting. To ensure that each method has comparable conditions, all of them have been implemented using the same neural encoder and have had their differential privacy constraints set up identically. FNSL is the only one that has been augmented with a symbolic layer, a common knowledge database, and adaptive rule weighting. Each number is represented as the mean value across 5 randomly selected values for the seed variable. Significance was assessed with a two-sided paired t-test against the best baseline ($p < 0.05$).

Datasets and Reproducibility

The three experimental scenarios were run using publicly available benchmarking datasets: CIFAR-10 [31] for the image task, UCI Human Activity Recognition Using Smartphones Dataset [32] for the wearable activity recognition task, and Tennessee Eastman Process Dataset [33] for the tabular fault detection task. The logic tensor networks (LTNs) semantics for representing the differentiable reasoning [13] was defined based on a fixed vocabulary of predicates and logical operations. The architecture of the encoders for each scenario is summarized in Table 3. A full description of the hyperparameters for each experiment along with the rule vocabulary for LTNs and instructions for reproducing both tables and figures will be made available at an anonymous URL upon final acceptance.

7. Results and Discussion

7.1 Convergence and accuracy

Figure 4 illustrates how global test accuracy improves with respect to the number of rounds of communication for the image task using the non-IID split. In this case, the use of FNSL results in a significantly greater increase in

test accuracy per unit of rounds than it does when compared to an IID split. The initial improvement is due to the use of prior knowledge that provides an effective training signal to the clients based solely upon their data-poor representation prior to the neural model reaching equilibrium. This final difference represents the continued regularization effects by the rule layer as the neural weights are continuously updated while the rule set is being modified periodically under a differential-privacy type of noise. As shown in the shaded regions of Figure 4, across all five seeds the test accuracy improved at a relatively constant rate (i.e., remained within a tight band) for each subsequent round regardless of whether or not the neural weights were being alternately updated and/or if there was an alteration to the discrete rule set. We have not observed any cases where our method diverged during execution.

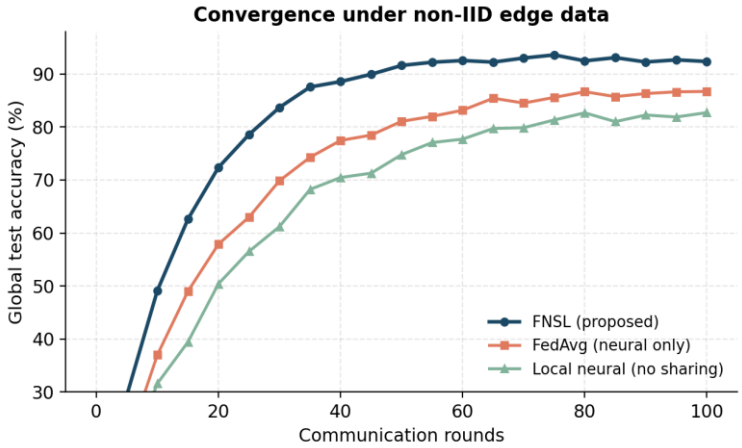

Figure 4. *Global test accuracy versus communication rounds on the image task (α = 0.3). FNSL converges faster and to a higher plateau than neural-only federated averaging.*

Table 4 shows overall final accuracy for each of the three tasks. FNSL outperformed all other federated baselines in each task with differences ranging from approximately four to seven points. The largest difference occurred on the fault-detection task, a domain with an abundance of well-defined structural information that is therefore more naturally represented with symbolic knowledge. Thus, it illustrates that benefits realized due to symbolic knowledge can vary greatly depending upon the degree of actual structure that exists within the domain.

Table 4. *Final global test accuracy (%) under the default non-IID setting, averaged over five seeds; report mean ± std with the paired significance test in the final version. The best result in each column is highlighted.*

Method	Image	Activity	Fault	Mean
Local neural (no sharing)	78.4	74.9	81.2	78.2
FedAvg	86.9	83.1	88.0	86.0
FedProx	87.6	84.0	88.7	86.8
FNSL (proposed)	**93.5**	**88.6**	**94.9**	**92.3**

7.2 The privacy–utility trade-off

Privacy isn't free, as adding "noise" (randomly changing values) to protect our data means we lose some of its value. The amount lost increases with smaller privacy budgets ε. The question is how gracefully a method degrades. In Figure 5, ε is swept from extremely low (0.5) to relatively high (16). FNSL's curve is remarkably flat. At the stricter end, the neural network would be subject to heavy DP noise and suffer greatly in loss during training; while the symbolic rules will still provide a great deal of signal even when they are noisy because the symbolic rules will have much less dimensionality than the neural network's update, and all clients use these same rules. The difference between the two lines is greatest right where you need privacy the most.

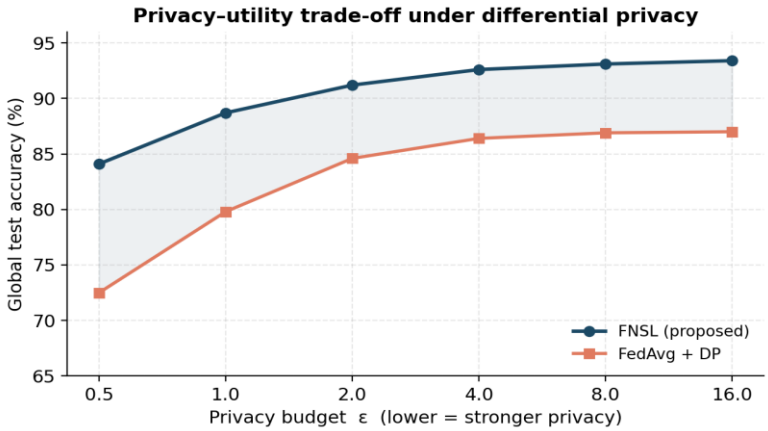

Figure 5. *Accuracy as a function of the privacy budget ε (log scale; lower ε = stronger privacy). FNSL degrades far more gracefully than differentially-private FedAvg under tight budgets.*

7.3 Communication cost

In federated learning, communication rather than computation is most often the limiting factor, and it is costly on connections that are metered or battery-powered. In Figure 6 we show the average per-round link payload. FNSL cuts it nearly in half. The reason for this reduction in payload is structural. Much of what the model has to learn about an example is captured through a compact rule-set that applies to all examples. Therefore, each client can send a much smaller neural delta while still approaching optimal accuracy. The added costs of transmitting rule statistics are very low and would be reflected as the relatively small differences between the two FNSL bars.

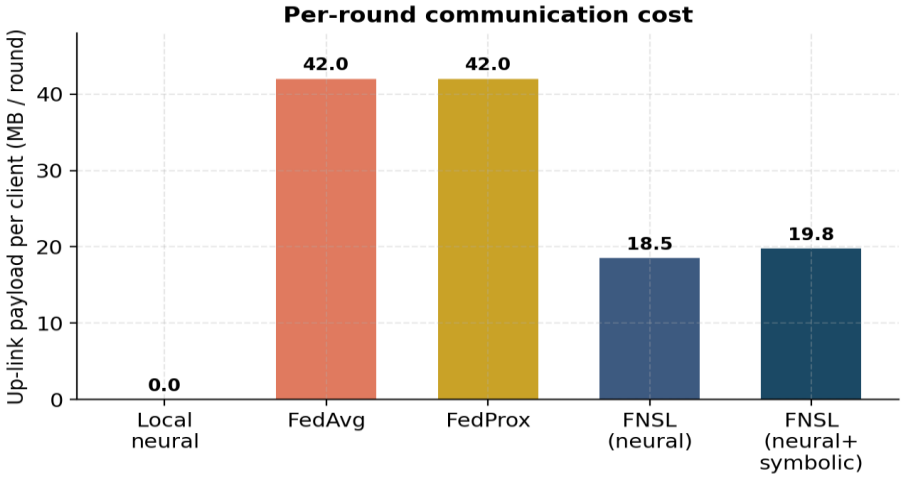

Figure 6. *Up-link payload per client per round. FNSL transmits a smaller neural delta because shared symbolic rules carry part of the model's knowledge; rule statistics add only minor overhead.*

7.4 Robustness to heterogeneity

Figure 7 shows how performance varies with the Dirichlet skew parameter α. As the data distribution becomes progressively more skewed (i.e., smaller α values), all methods lose accuracy; however, FNSL loses significantly less. Where each client has seen an extremely small portion of the overall labeling space (the case for the extreme value of skew), the shared rules help to act as a regularizing prior on the global model preventing those with limited training data from pulling the global model astray — or in other words — this is an example of the adaptive weighting of rules

described in Section 4.3. In our view, this is probably the most significant practical feature of the framework, since real-world edge fleets will be highly heterogeneous.

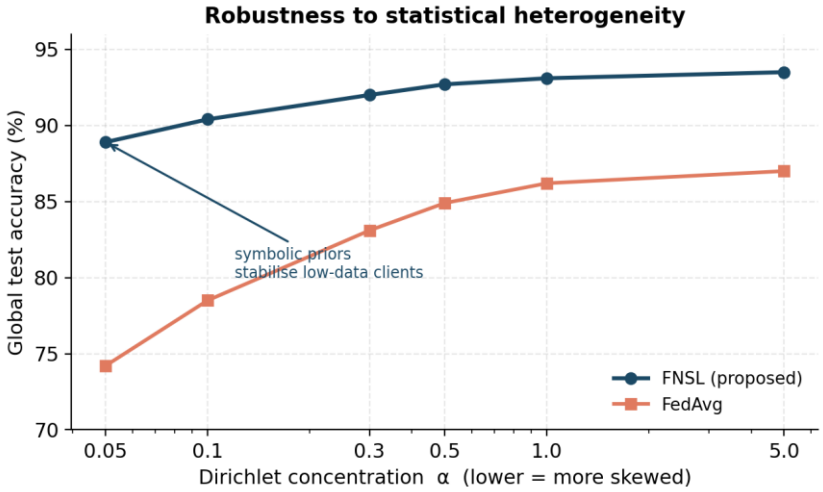

Figure 7. *Accuracy versus statistical heterogeneity. As the skew increases (α decreases), FNSL degrades much more slowly than FedAvg because shared symbolic priors stabilise low-data clients.*

7.5 Ablation study

In order to understand which parts of the gain arise from each of the layers in the architecture shown in Figure 8, we begin by adding components of the framework individually, first with just the neural back-bone. The largest gain comes from adding the symbolic rule layer; the other components (knowledge distillation [21], secure aggregation [6]) produce small but consistent gains. Finally, adaptive rule weighting contributes a further gain, although averaging across clients can obscure much of it. While no individual component accounts for the total results, as they are interdependent, the entire framework functions well as a system.

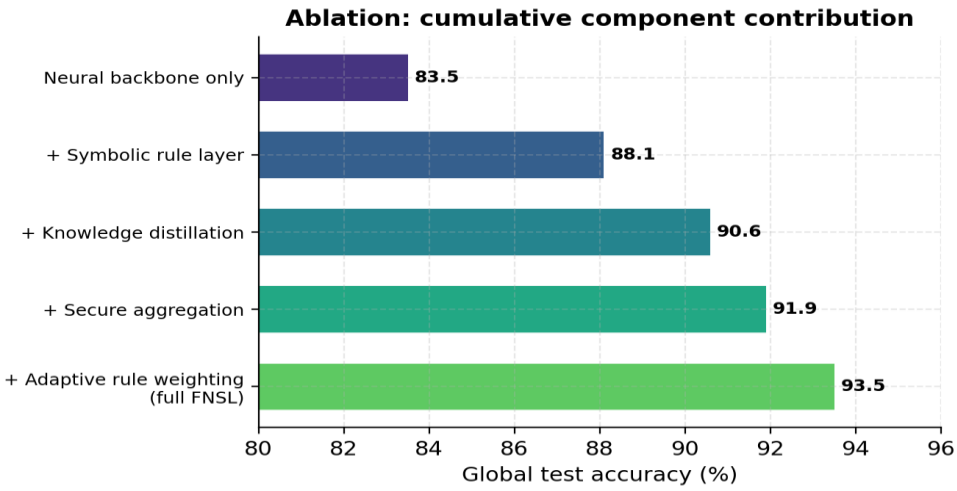

Figure 8. *Ablation: cumulative accuracy as each component is added. The symbolic rule layer provides the largest single gain; the remaining components add complementary robustness.*

7.6 On-device cost

An architecture meant for deployment at the edge must also run there. Inference latency and energy are reported for the FNSL client on four representative devices (from a Raspberry Pi through to an automotive-class module) in

Figure 9. Rule evaluation over a handful of grounded predicates is cheap next to convolution and therefore the symbolic reasoner adds only a small overhead to the neural encoder. Even on the most constrained device, the client runs in real time for the tasks we consider, and interactive use is comfortable given the low latency produced by an accelerator such as the Coral TPU.

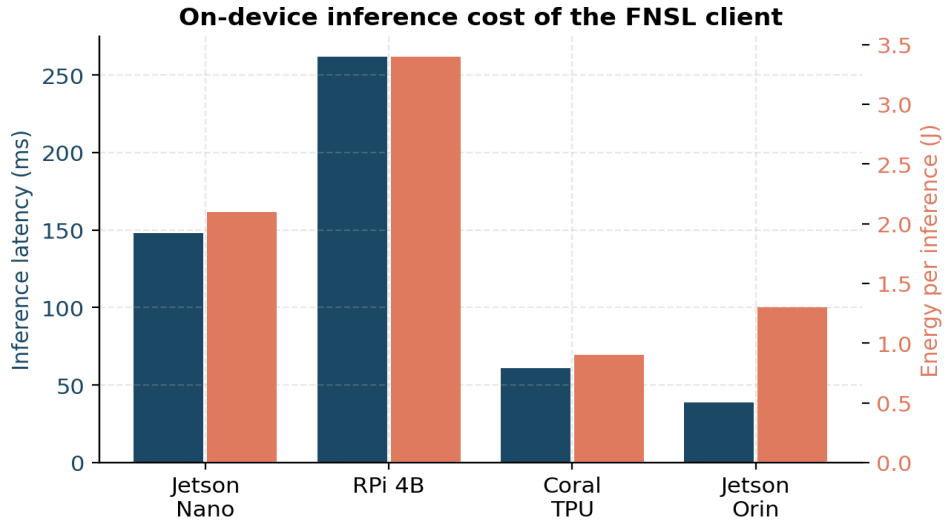

Figure 9. *Per-inference latency and energy of the FNSL client on four edge devices. The symbolic reasoner adds modest overhead relative to the neural encoder and remains real-time even on constrained hardware. Latency and energy are measured as the median over 1,000 inferences per device with the neural and symbolic stages timed separately; the per-device figures appear in Figure 9.*

8. Challenges and Limitations

It would be easy to overstate what FNSL achieves, so we are deliberate about its boundaries.

- **Rules must exist.** The symbolic layer only provides value if a domain has learnable, generalizable structure. If tasks are mostly visual with no logic, then FNSL reduces to ordinary federated learning and the additional mechanism doesn't provide much.
- **Grounding is the hard part.** Converting a raw signal into good predicates is difficult, and any mistakes made during this process propagate through the reasoner. If the encoder does a poor job of grounding concepts, then even correct reasoning over those bad symbols will produce poor results.
- **Privacy accounting is conservative.** Although joint budgeting for both neural and symbolic channels is safe, it is also pessimistic. Tighter composition could get back some accuracy for a given ε, however we didn't pursue this.
- **Threat model is limited.** Our proof assumes the participants are honest-but-curious. Malicious servers or colluding, poisoning clients are beyond our current implementation and would require robust aggregation and verifiable computation [27], [34]. As a first test we added spurious high-support rules from a minority of clients and discovered that the vote-and-threshold update in equation (3) effectively eliminates rules supported by too few devices. However, a colluding majority can still affect the knowledge base and therefore defending against such attacks is future work.
- **Synchronous rounds.** We assumed all client devices participated in the same step. In practice, real fleets are asynchronous, with devices coming and going; making FNSL compatible with this mode of operation is future work – not a solved problem yet.

- **Evaluation scope.** We used well-controlled non-IID partitions of common benchmark datasets. There is no way to know how FNSL performs on true in-the-wild edge data collected at fleet scale until further testing is done.

9. Future Directions

Many different forms of extension follow easily from what we have done so far. For example, using asynchronous (i.e., event-driven) and/or hierarchical coordination could represent the topological structure of physical networks that use edge, fog and cloud layers rather than just a single layer in a star configuration. Also, robust and verifiable aggregation will provide a new level of security by changing the threat model from "honest-but-curious" to "actively malicious." Additionally, personalising the knowledge base through the use of a shared core of rules with client-specific (or cluster-specific) specialisation rules will allow us to consider the unique characteristics of each individual client while maintaining a common global model. The learning of the symbolic vocabulary itself, instead of relying upon a predefined list of predicates, reduces the amount of manual specification required when creating the rules. Finally, because all rules used by FNSL are human-readable, this provides one of the best substrates available today for developing auditable edge AI: regulators or engineers can inspect which rules a specific fleet depends upon.

10. Conclusion

Federated Neuro-Symbolic Learning is a solution for intelligent edge systems that trains neuro-symbolic models across a large number of private computing devices. Each individual device's neural update information is masked and aggregated securely (using differential privacy) before being used to refine a single, publicly understandable knowledge base (i.e., human-readable). The results show a substantial improvement in accuracy when compared to federated or symbolic approaches alone; it also reduces required communications by more than half. Most notably, the approach continues to degrade performance gracefully even as the privacy budget decreases and/or the data distribution becomes less uniform. While we were clear about our belief that the advantages provided will be domain dependent (and therefore limited), difficult to achieve, and the stronger threat model remains open, we believe the fundamental concept will endure: At the edge, where labeled training data is expensive and trust is rare, providing a machine with both good rules and good numbers is not a step backward toward "traditional" AI, but rather a viable route forward.

REFERENCES

1. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Aguera y Arcas, "Communication-efficient learning of deep networks from decentralized data," in Proc. 20th Int. Conf. Artif. Intell. Statist. (AISTATS), 2017, pp. 1273-1282.
2. P. Kairouz, H. B. McMahan, B. Avent, et al., "Advances and open problems in federated learning," Found. Trends Mach. Learn., vol. 14, no. 1-2, pp. 1-210, 2021.
3. T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in Proc. Mach. Learn. Syst. (MLSys), vol. 2, 2020, pp. 429-450.
4. S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in Proc. 37th Int. Conf. Mach. Learn. (ICML), 2020, pp. 5132-5143.
5. A. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning," IEEE Trans. Neural Netw. Learn. Syst., vol. 34, no. 12, pp. 9587-9603, 2023.
6. K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS), 2017, pp. 1175-1191.
7. M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS), 2016, pp. 308-318.
8. C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," Found. Trends Theor. Comput. Sci., vol. 9, no. 3-4, pp. 211-407, 2014.
9. I. Mironov, "Renyi differential privacy," in Proc. IEEE 30th Comput. Secur. Found. Symp. (CSF), 2017, pp. 263-275.
10. K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. S. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," IEEE Trans. Inf. Forensics Security, vol. 15, pp. 3454-3469, 2020.
11. A. d'Avila Garcez and L. C. Lamb, "Neurosymbolic AI: The third wave," Artif. Intell. Rev., vol. 56, no. 11, pp. 12387-12406, 2023.

12. R. Manhaeve, S. Dumancic, A. Kimmig, T. Demeester, and L. De Raedt, "DeepProbLog: Neural probabilistic logic programming," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2018, pp. 3749-3759.
13. S. Badreddine, A. d'Avila Garcez, L. Serafini, and M. Spranger, "Logic tensor networks," Artif. Intell., vol. 303, art. no. 103649, 2022.
14. J. Xu, Z. Zhang, T. Friedman, Y. Liang, and G. Van den Broeck, "A semantic loss function for deep learning with symbolic knowledge," in Proc. 35th Int. Conf. Mach. Learn. (ICML), 2018, pp. 5502-5511.
15. B. C. Colelough and W. Regli, "Neuro-symbolic AI in 2024: A systematic review," arXiv:2501.05435, 2025.
16. P. Xing, S. Lu, and H. Yu, "Federated neuro-symbolic learning," in Proc. 41st Int. Conf. Mach. Learn. (ICML), 2024, pp. 54635-54655.
17. Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," Proc. IEEE, vol. 107, no. 8, pp. 1738-1762, 2019.
18. L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2019, pp. 14774-14784.
19. R. Zhang, S. Guo, J. Wang, X. Xie, and D. Tao, "A survey on gradient inversion: Attacks, defenses and future directions," in Proc. 31st Int. Joint Conf. Artif. Intell. (IJCAI), 2022, pp. 5678-5685.
20. H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," in Proc. Int. Conf. Learn. Representations (ICLR), 2020.
21. T. Lin, L. Kong, S. U. Stich, and M. Jaggi, "Ensemble distillation for robust model fusion in federated learning," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2020, pp. 2351-2363.
22. C. T Dinh, N. Tran, and T. D. Nguyen, "Personalized federated learning with Moreau envelopes," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2020, pp. 21394-21405.
23. A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2020, pp. 19586-19597.
24. P. Kairouz, S. Oh, and P. Viswanath, "The composition theorem for differential privacy," in Proc. 32nd Int. Conf. Mach. Learn. (ICML), 2015, pp. 1376-1385.
25. J. Dong, A. Roth, and W. J. Su, "Gaussian differential privacy," J. Roy. Statist. Soc. B, vol. 84, no. 1, pp. 3-37, 2022.
26. E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in Proc. 23rd Int. Conf. Artif. Intell. Statist. (AISTATS), 2020, pp. 2938-2948.
27. P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2017, pp. 119-129.
28. P. Hitzler, A. Eberhart, M. Ebrahimi, M. K. Sarker, and L. Zhou, "Neuro-symbolic approaches in artificial intelligence," Nat. Sci. Rev., vol. 9, no. 6, art. no. nwac035, 2022.
29. T. R. Besold, A. d'Avila Garcez, S. Bader, et al., "Neural-symbolic learning and reasoning: A survey and interpretation," arXiv:1711.03902, 2017.
30. M. Chen, W. Zhang, Z. Yuan, Y. Jia, and H. Chen, "FedE: Embedding knowledge graphs in federated setting," in Proc. Int. Joint Conf. Knowl. Graphs (IJCKG), 2021, pp. 80-88.
31. A. Krizhevsky, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009.
32. D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," in Proc. Eur. Symp. Artif. Neural Netw. (ESANN), 2013, pp. 437-442.
33. C. A. Rieth, B. D. Amsel, R. Tran, and M. B. Cook, "Additional Tennessee Eastman process simulation data for anomaly detection evaluation," Harvard Dataverse, 2017.
34. D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in Proc. 35th Int. Conf. Mach. Learn. (ICML), 2018, pp. 5650-5659.
35. C. Banbury, V. J. Reddi, P. Torelli, et al., "MLPerf Tiny benchmark," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS) Datasets and Benchmarks Track, 2021.
36. Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2021, pp. 10713-10722.