VAST-FCL: Verifiable Sparse-Task Federated Continual Unlearning for Cross-Institutional Educational AI

Abstract

Cross-institutional educational models must continually absorb new curricula, semesters, and device streams while honoring requests to withdraw previously authorized data. Continual learning seeks to prevent forgetting, whereas machine unlearning requires the deliberate removal of prior influence. This creates a structural conflict that is not resolved by deleting database records alone. The supplied course architecture uses task-specific sparse masks and parameter resets to isolate knowledge, but its claim of complete forgetting conflates three different objectives: parameter removal, distributional similarity to retraining, and externally verifiable execution. This paper proposes VAST-FCL, a Verifiable Audited Sparse-Task Federated Continual Learning framework. VAST-FCL confines revocable task increments to addressable parameter subspaces, repairs downstream tasks through controlled mask reallocation and authorized rehearsal, and records each state transition in a hash-linked transparency log. We evaluate the method on six sequential synthetic educational tasks with 72 features and 12 random seeds. After deleting the third task, VAST-FCL retains 79.07% ± 1.79% mean accuracy, compared with 77.19% ± 2.20% for reset-only and 79.73% ± 1.57% for full retraining. The gain over reset-only is significant ($t(11) = 5.31$, $p < 0.001$), while estimated computation is 22% of full retraining. All protocol assertions pass across the 12 runs. The findings support efficient task-level structural deletion, but not universal sample-level distributional equivalence. Production deployment therefore requires a layered evidence package that combines structural checks, behavioral privacy tests, and independent audit.

Keywords: federated continual learning; machine unlearning; sparse task masks; educational data governance; auditability; transparency logs

Research highlights

- Separates structural deletion, statistical proximity to retraining, and verifiable execution as three distinct assurance layers.
- Reallocates the released mask under explicit dependency controls to restore downstream utility with a small authorized rehearsal buffer.
- Produces an auditable deletion receipt without writing raw educational records or complete gradients to the governance log.
- Reports seeded synthetic results, uncertainty, code, assumptions, and limitations instead of presenting simulated values as field evidence.

1. Introduction

Federated learning reduces the need to centralize sensitive records by aggregating locally computed model updates [1]. This principle is attractive in education, where institutions may wish to collaborate without transferring student interaction logs, assessments, or behavioral traces to a common repository. Data localization, however, does not make a trained contribution revocable. Once a model has passed through several semesters of sequential updates, the influence of a course, institution, or authorization batch may propagate into later parameters. Deleting the original record does not identify which model state was affected, how later tasks depend on that state, or how an external auditor can verify that the withdrawal was executed.

Machine unlearning addresses this gap by asking a trained system to remove the influence of designated training data. A common reference is the model that would have been obtained by training from scratch without the deleted data [2,3]. SISA reduces the required retraining path through sharding and isolation [2], while federated unlearning methods reconstruct a global model after removing a client's contribution [4,5]. Continual learning approaches the opposite problem. It uses regularization, rehearsal, or parameter isolation to prevent new tasks from overwriting old knowledge. Hard Attention to the Task and PackNet show that task-specific sparse subnetworks can isolate sequential tasks in a single model [6,7]. Privacy-Aware Lifelong Learning connects sparse task subnetworks to task-level unlearning [8].

The course PDF combines these ideas in a cloud-edge-end governance architecture. Its central insight is valuable: a revocable contribution is easier to remove when the system creates an addressable boundary during training. However, physically resetting a parameter subset is not automatically equivalent to retraining on the retained data, and neither operation proves to a third party that the correct request was processed. This distinction is particularly important in education, where inaccurate statements of privacy assurance could affect high-risk systems used for access, placement, learning assessment, or student monitoring.

This paper asks the following research question: **Can task masks make a withdrawal locally addressable, restore downstream utility at substantially lower cost than full retraining, and generate evidence that an independent auditor can verify?**

The contribution is threefold. First, we define a layered assurance model that distinguishes structural deletion, statistical proximity, and auditable execution. Second, we introduce a state machine that compiles a task withdrawal into mask lookup, parameter reset, controlled reallocation, federated repair, and receipt publication. Third, we evaluate the utility-cost trade-off on a reproducible sequential benchmark and report the remaining distance to full retraining rather than claiming lossless unlearning.

2. Related work

2.1 Federated learning and heterogeneous clients

FedAvg combines local stochastic optimization with weighted server aggregation [1]. It established that useful global models can be trained while keeping raw data on participating devices or institutional nodes. Yet non-independent and non-identically distributed data can create client drift. SCAFFOLD uses control variates to correct this drift and reduce the number of communication rounds [9]. These methods explain how to coordinate training, but they do not preserve an explicit contribution boundary for later withdrawal. If a server retains only the final weights, locating a specific task or authorization batch after the fact normally requires approximation or retraining.

2.2 Parameter isolation in continual learning

Hard Attention to the Task learns masks that protect parameters associated with previous tasks [6]. PackNet iteratively prunes a network, frees unused capacity, and freezes surviving parameters for each task [7]. Their important property is not sparsity alone. It is addressability: the system knows which coordinates were first allocated to a task and which later tasks depend on them. Privacy-Aware Lifelong Learning extends this principle to task-level removal and rehearsal-based recovery [8]. Once later tasks reuse earlier parameters, however, a reset can cause secondary utility loss. A deployable system must therefore record dependencies and repair only the affected retained tasks.

2.3 Machine unlearning and deletion guarantees

SISA treats data shards as isolation units and retrains only the affected slices after deletion [2]. Sekhari et al. provide a formal study of deletion, generalization, storage, and computation [3]. FedEraser reconstructs a federated model by calibrating retained historical client updates [4], while SIFU studies sequential client unlearning with provable guarantees in federated optimization [5]. These methods use different deletion units and assumptions. A task mask is suitable when an entire course, client phase, or authorization batch is withdrawn, but it does not solve arbitrary sample-level deletion unless sample contributions are separately isolated.

2.4 Verification and governance evidence

Final model behavior alone is insufficient evidence that an unlearning procedure was honestly executed. Verifiable unlearning reframes the problem as a security protocol in which the trainer commits to data or model state and later proves correct deletion [10]. Append-only Merkle logs can provide efficient inclusion and consistency proofs [11]. A log nevertheless proves only that a particular statement was committed. It does not prove that an implementation is free from hidden state or that a deleted sample cannot be inferred. VAST-FCL therefore treats the log as an execution-evidence layer that complements, rather than replaces, statistical privacy tests.

3. Problem formulation and assurance boundaries

Let the sequential task stream be $T=T\_1,ldots,T\_K$ and the participating institutions be C. The global parameter vector is $\theta R \wedge d$. Task k has a binary mask $m\_k0,1 \wedge d$ and an increment Δ_k. The effective model for task t is

$$\theta^{(t)} = \theta_0 + \sum_{k=1}^{t} m_k \odot \Delta_k.$$

When task r is withdrawn, the structural deletion operation is

$$U_r(\theta) = \theta - m_r \odot \Delta_r, \qquad \Delta_r \leftarrow 0.$$

We distinguish three assurance levels.

1. **Structural deletion,** G_s**.** The task-exclusive increment is set to zero, task-specific buffers are destroyed, and the state assertions can be checked deterministically.
2. **Statistical proximity,** G_d**.** The unlearned model is sufficiently close to the reference model $\theta \wedge -r$ trained without task r, according to declared behavioral or distributional metrics. The present experiment reports a normalized parameter gap but does not claim a general (ε,δ) unlearning guarantee.
3. **Auditable execution,** G_a**.** An external party can verify the request signature, old state, mask commitment, new state, and append-only log consistency. This proves the recorded transition under stated assumptions, not the absence of covert copies or implementation backdoors.

The three levels prevent a common category error. A successful zero reset establishes G_s; it does not automatically establish G_d or G_a.

4. The VAST-FCL framework

4.1 Sparse task allocation

For each arriving task, an edge node computes an importance score vector s_k. Coordinates already owned by earlier non-revoked tasks are read-only. The task selects the top q fraction of the remaining candidate set F_k:

$$m\_k,j=I[s\_k,j \geq Q\_1\text{-}q(s\_k| jF\_k)].$$

After training, the institution commits to the task identifier, data-version identifier, update digest, and $H(m\_k)$. Raw samples and complete gradients remain local. Later tasks may read frozen increments for forward transfer but can write only to their authorized masks.

4.2 Withdrawal and dependency resolution

A signed withdrawal enters a five-stage state machine:

1. Resolve the task-data-mask dependency graph and identify m_r and the downstream set $D^+(r)$.
2. Set the task increment to zero and destroy its local rehearsal buffer.
3. Release the deleted coordinates to the first direct successor under an exclusive reallocation rule.
4. Repair affected retained tasks using only currently authorized rehearsal samples and mask-restricted updates.
5. Evaluate utility, zero-mask, and log-consistency gates before publishing the new model version.

The exclusive reallocation rule prevents several successor increments from claiming the same released coordinate. If any gate fails, the candidate version remains quarantined for human review.

4.3 Authorized rehearsal repair

Let B_k be a small buffer for retained task k. The post-deletion repair objective is

$$\min_{\Delta_{D^+(r)}} \sum_{k \in D^+(r)} \sum_{(x,y) \in B_k} \ell(y, f_{\theta^{-r}}(x)) + \lambda \|\Delta_k\|_2^2.$$

Buffers are authorization-scoped, purpose-limited, and deleted after the repair gate. The model may update only the successor's existing mask and the coordinates explicitly released by the deleted task. This restriction reduces both computation and the risk of reintroducing the withdrawn contribution through unrestricted fine-tuning.

4.4 Transparency receipt

Each event e_i stores the previous digest h_{i-1}, event type, model version, task and mask digests, actor role, and timestamp:

$$h_i = H(h_{i-1} \,\|\, \mathrm{canon}(e_i)).$$

The resulting receipt contains the request digest, pre- and post-deletion state roots, $H(m_r)$, gate outcomes, code version, and institutional signatures. A Merkle transparency log supports inclusion and consistency checks [11]. The receipt is deliberately minimal. It must not contain student identifiers or reversible feature encodings.

5. Experimental design

5.1 Synthetic sequential tasks

The benchmark contains six sequential binary classification tasks. Each task has 900 training samples, 700 test samples, and 72 standardized Gaussian features. The first eight dimensions represent cross-curricular latent ability. Each task adds a nine-dimensional course-specific component. Later tasks reuse earlier components with a decay factor of 0.82^{t-k}, creating a measurable forward dependency. Labels are sampled from a logistic model with Gaussian noise. Each task selects 11 previously unallocated coordinates with the highest residual correlation.

5.2 Withdrawal scenario and baselines

After all six tasks are trained, the third task is withdrawn. We compare:

- **Reset-only:** erase the deleted increment without downstream repair.
- **VAST-FCL:** reset, reallocate the released coordinates to the direct successor, and repair each affected task with 260 authorized samples and 135 optimization steps.
- **Full retraining:** rebuild the sequence from all retained training data after excluding the deleted task.

Each method is evaluated on 12 paired random seeds. The primary outcome is mean retained-task accuracy. Secondary outcomes are worst-task accuracy, normalized parameter distance to full retraining, relative computational cost, and audit-assertion pass rate. Relative computation is based on gradient-sample evaluations, not measured GPU energy.

5.3 Reproducibility

Seeds range from 4100 to 4111. The experiment script is analysis/simulate_studies.py; raw and aggregated outputs are stored in analysis/results/study1_raw.csv and analysis/results/study1_summary.csv.

6. Results

6.1 Retained utility and cost

Method	Mean retained accuracy	Standard deviation	Worst-task accuracy	Parameter gap to retraining	Relative compute	Audit assertion pass rate
Reset-only	77.19%	2.20%	73.27%	Not applicable	0.6%	0%
VAST-FCL	**79.07%**	1.79%	**75.48%**	0.260	**22.0%**	**100%**
Full retraining	79.73%	1.57%	76.60%	0.000	100%	Not applicable

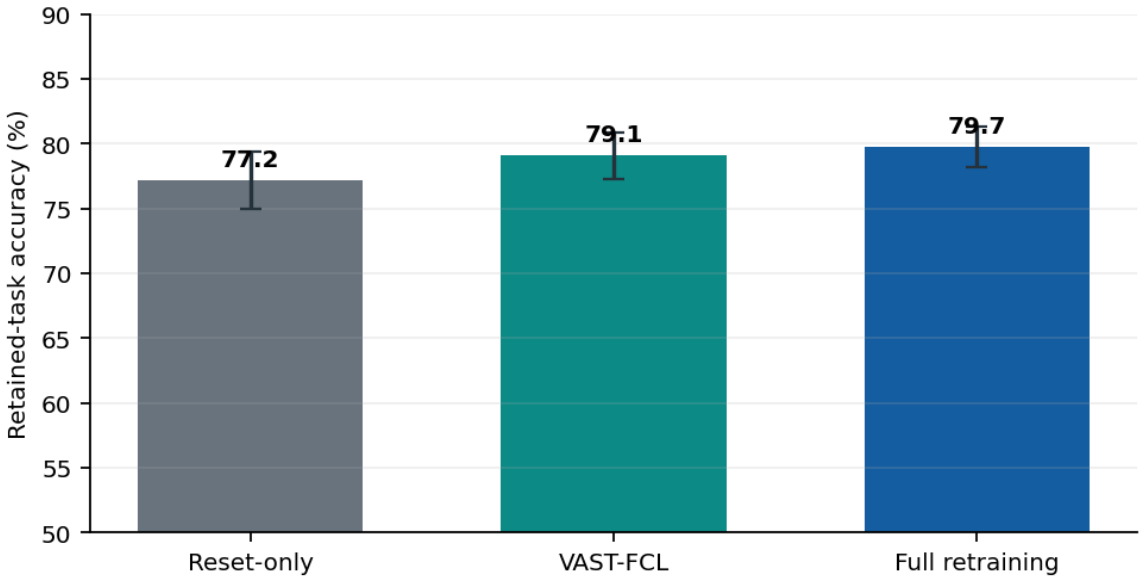

Figure 1. Mean retained-task accuracy after deleting the third task. Error bars show the standard deviation across 12 seeds.

VAST-FCL improves mean retained accuracy by 1.88 percentage points over reset-only. The paired comparison gives $t(11) = 5.31$ and $p = 0.00025$. VAST-FCL remains 0.66 points below full retraining, with $t(11) = -3.50$ and $p = 0.00494$. The repair therefore produces a meaningful recovery but does not reproduce the full-retraining distribution without loss.

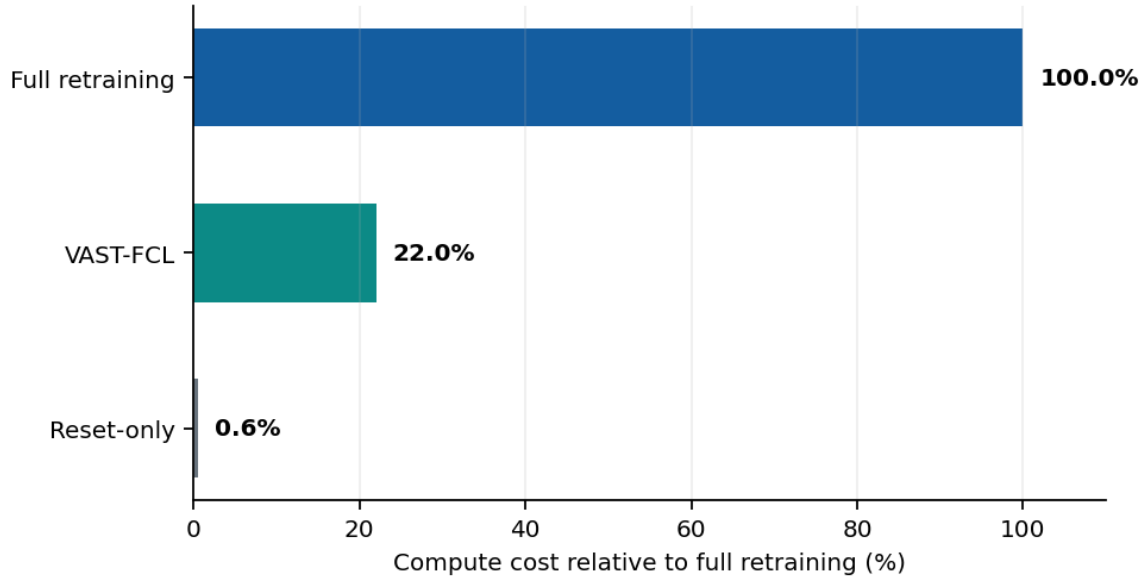

Figure 2. Estimated gradient computation relative to full retraining. VAST-FCL uses 22% of the retraining budget.

6.2 Interpretation of deletion evidence

All VAST-FCL runs satisfy three structural assertions: the withdrawn increment is zero, the deletion event refers to the correct mask digest, and the hash chain recomputes from its initial value to the final receipt. The defensible claim is therefore that task-level structural deletion and protocol execution were verified in this implementation.

The normalized parameter gap to full retraining is 0.260. This nonzero value matters. It demonstrates why successful structural checks should not be described as universal exact unlearning. A production evidence package should additionally include membership-inference performance, backdoor success on the deleted task, retained and deleted loss distributions, output divergence from retraining, and independent monitoring of the transparency log.

7. Discussion

7.1 Why masks help

Task masks transform an ambiguous historical influence into a queryable ownership map. The control plane can locate the revocable increment without scanning or retraining the complete model. Controlled reallocation then uses the released capacity to absorb downstream concepts that had depended on the deleted task. The increase in worst-task accuracy from 73.27% to 75.48% suggests that recovery is not confined to the average task.

7.2 Relationship to SISA, FedEraser, and SIFU

SISA isolates data shards and retrains affected slices [2]. FedEraser calibrates retained federated updates [4]. SIFU provides a formal path for sequential client deletion [5]. VAST-FCL uses task parameter increments as its isolation unit. It is consequently well matched to course-level, client-phase, or authorization-batch withdrawal, but not to an arbitrary individual record. A layered platform could use SISA for sample-level requests, SIFU for client-level requests, task masks for curricular withdrawal, and a common receipt format for execution evidence.

7.3 Compliance and educational governance

Article 17 of the General Data Protection Regulation establishes a right to erasure subject to legal grounds and exceptions [12]. No single machine-learning metric fully determines compliance. The European Union AI Act classifies several systems used for educational access, placement, outcome assessment, or exam monitoring as high risk [13]. Institutions therefore need purpose limitation, data minimization, role-based authorization, human oversight, retention schedules, and accessible appeals in addition to technical deletion.

8. Limitations

The experiment uses low-dimensional synthetic logistic tasks and cannot be extrapolated directly to vision transformers or large language models. The deletion unit is an entire task rather than an individual sample or learned fact. Compute is estimated from gradient-sample evaluations, not measured on a real federated cluster. The hash-chain implementation checks order and integrity but does not implement a SNARK, remote attestation, or Byzantine consensus. Finally, privacy leakage, poisoning, and fairness are separate dimensions that require dedicated real-distribution evaluation.

9. Conclusion

VAST-FCL provides a more defensible extension of sparse task masking for privacy-aware lifelong learning. It turns task withdrawal into a measurable state transition: locate the contribution, erase the task increment, repair authorized successors, and publish a verifiable receipt. In the seeded synthetic benchmark, VAST-FCL reaches within 0.66 percentage points of full-retraining utility at an estimated 22% of its computation. Its main contribution is not a claim of 100% universal forgetting. It is an architecture in which task-level influence is addressable, recovery is quantifiable, and execution evidence is auditable. Future work should combine this structural layer with formal deletion algorithms, membership-inference testing, adversarial evaluation, and implemented cryptographic proofs.

Data and code availability

No real student data are used. The synthetic generator, random seeds, raw results, and figures are included in the coursework directory. Any replication with real educational records would require ethics review, a data-protection impact assessment, and a purpose-limited collection design.

References

[1] McMahan, B., Moore, E., Ramage, D., Hampson, S., and Agüera y Arcas, B. (2017). Communication-Efficient Learning of Deep Networks from Decentralized Data. *AISTATS*, PMLR 54, 1273-1282. [source]

[2] Bourtoule, L., Chandrasekaran, V., Choquette-Choo, C. A., et al. (2021). Machine Unlearning. *2021 IEEE Symposium on Security and Privacy*, 141-159. [source]

[3] Sekhari, A., Acharya, J., Kamath, G., and Suresh, A. T. (2021). Remember What You Want to Forget: Algorithms for Machine Unlearning. *NeurIPS 34*, 18075-18086. [source]

[4] Liu, G., Ma, X., Yang, Y., Wang, C., and Liu, J. (2020). Federated Unlearning. arXiv:2012.13891. [source]

[5] Fraboni, Y., Van Waerebeke, M., Scaman, K., et al. (2024). SIFU: Sequential Informed Federated Unlearning for Efficient and Provable Client Unlearning in Federated Optimization. *AISTATS*, PMLR 238, 3457-3465. [source]

[6] Serra, J., Suris, D., Miron, M., and Karatzoglou, A. (2018). Overcoming Catastrophic Forgetting with Hard Attention to the Task. *ICML*, PMLR 80, 4548-4557. [source]

[7] Mallya, A., and Lazebnik, S. (2018). PackNet: Adding Multiple Tasks to a Single Network by Iterative Pruning. *CVPR*, 7765-7773. [source]

[8] Özdenizci, O., Rueckert, E., and Legenstein, R. (2025). Privacy-Aware Lifelong Learning. arXiv:2505.10941. [source]

[9] Karimireddy, S. P., Kale, S., Mohri, M., et al. (2020). SCAFFOLD: Stochastic Controlled Averaging for Federated Learning. *ICML*, PMLR 119, 5132-5143. [source]

[10] Eisenhofer, T., Riepel, D., Chandrasekaran, V., et al. (2025). Verifiable and Provably Secure Machine Unlearning. *IEEE SaTML*. [source]

[11] Laurie, B., Langley, A., and Kasper, E. (2013). Certificate Transparency. RFC 6962. [source]

[12] European Parliament and Council. (2016). Regulation (EU) 2016/679, Article 17: Right to Erasure. [source]

[13] European Parliament and Council. (2024). Regulation (EU) 2024/1689: Artificial Intelligence Act. [source]

[S1] Tao, J., Liu, Z., Lyu, R., and Cao, X. (2026). A Scalable Data Governance Architecture for Privacy-Aware Intelligent Learning Systems in Lifelong. Course-supplied PDF manuscript.