Journal of Cloud Computing

https://doi.org/10.1186/s13677-026-00939-w

Article in Press

# Selective semantic state editing for private edge-cloud inference in SLMs

Received: 4 May 2026

Accepted: 9 June 2026

Published online: 22 June 2026

Cite this article as: Wang F., Li D., Ma B. *et al.* Selective semantic state editing for private edge-cloud inference in SLMs. *J Cloud Comp* (2026). https://doi.org/10.1186/s13677-026-00939-w

**Feifan Wang, Delong Li, Baihe Ma, Xu Wang, Chen Li, Xuelei Qi, Yang Zhang, Kewei Sha, Haihua Chen & Guangsheng Yu**

# Selective Semantic State Editing for Private Edge-Cloud Inference in SLMs

Feifan Wang[1], Delong Li[1], Baihe Ma[1], Xu Wang[1], Chen Li[1], Xuelei Qi[2], Yang Zhang[3], Kewei Sha[3], Haihua Chen[3], Guangsheng Yu[1*]

[1]School of Electrical & Data Engineering, University of Technology Sydney, Ultimo, NSW, 2007, Australia.
[2]School of Computing, Macquarie University, NSW, 2122, Australia.
[3]The Anuradha and Vikas Sinha Department of Data Science, University of North Texas, Denton, 76207, USA.

*Corresponding author(s). E-mail(s): Guangsheng.Yu@uts.edu.au;



**Abstract**

Edge-cloud collaborative inference in small language models offloads intermediate latent representations to bypass on-device computational bottlenecks. However, this transparent transmission exposes sensitive user attributes to latent-space inference attacks. In this paper, to effectively protect latent states during transmission, we propose Selective Semantic State Editing (SSSE), an inference-time privacy mechanism that intervenes exclusively at the offload boundary. Our framework first employs a learnable risk scorer to identify privacy-sensitive token-level latent states, then constructs on-device semantic alternatives by combining an input-conditioned local reservoir with a static prototype cache, and finally performs risk-aware selective latent state editing to suppress sensitive-attribute leakage while preserving task-relevant information. Experimental results show that SSSE provides a favorable privacy–utility balance. On the topic classification task, SSSE achieves an accuracy of 0.4691 and a macro-F1 of 0.1890, incurring only a 0.0030 macro-F1 drop from baseline. Meanwhile, SSSE reduces attack macro-F1 from 0.6539 to 0.6288 on gender, and from 0.5403 to 0.5053 on age-group. The ablation results further verify the effectiveness of the risk scorer and candidate construction. Deployment overhead study finds that SSSE introduces a practical privacy-intervention overhead of approximately 2.8 ms/sample. Our work demonstrates the practical potential of selective semantic state editing for privacy-preserving edge-cloud inference.

**Keywords:** Edge-cloud inference, language models, privacy leakage, semantic state

# 1 Introduction

Small Language Models (SLMs) [1–3] are increasingly deployed in edge-cloud environments to support low-latency and resource-efficient intelligent services [4]. Compared with large centralized models, SLMs enable highly flexible deployment on user devices, including mobile assistants, wearable systems, and personalized devices [5–7]. However, edge devices remain constrained by limited computation, memory, and energy resources, especially for long-context and retrieval-augmented tasks. This limitation has increased the importance of collaborative processing across devices, edge nodes, and cloud servers in edge-intelligence applications [8–11]. As a result, edge-cloud collaborative inference has emerged as an important deployment paradigm [12–14], where the device executes the initial layers of the model and offloads intermediate latent representations to an edge or cloud server for the remaining computation.

Split inference and collaborative inference improve efficiency and responsiveness in edge-cloud computing, but the transmission of intermediate representations introduces a new privacy surface [15–17]. Although raw inputs remain on-device, transmitted latent states can still encode private attributes, user intent, and contextual information, enabling attacks such as attribute inference, membership inference, and input reconstruction [18, 19]. Privacy in edge-cloud SLM systems should therefore be considered not only at the input level but also at the level of transmitted latent representations [20–22]. Language models make this problem more severe because privacy leakage may concentrate in specific parts of the transmitted representation rather than spread uniformly across all latent states [23]. These observations motivate a selective protection mechanism for edge-cloud SLM inference that targets privacy-relevant latent states without indiscriminately perturbing the full transmitted representation.

Existing privacy defenses overlook the need to protect localized high-risk latent states. Some defenses for edge-cloud computing operate on raw inputs before transmission or model training, using mechanisms such as text sanitization or differential privacy [24–26]. Such methods intervene before latent representations are formed or constrain privacy during optimization, but do not directly protect transmitted latent states in split inference. Other defense methods perturb the latent representation with additive noise, through learned noise distributions or differential-privacy-based privatization [27–29]. These methods usually treat transmitted latent states as a homogeneous object and apply representation-wide perturbation. In structured language representations, global perturbation can disrupt task-relevant semantics, introduce unnecessary distortion into low-risk states, and reduce downstream utility [28, 29]. Recent studies indicate that privacy leakage in language-model representations is not uniformly distributed across the transmitted representation, and latent states may carry disproportionate sensitive information [23, 30, 31].

To address the lack of selective protection for privacy-sensitive latent states in edge-cloud SLM inference, we propose Selective Semantic State Editing (SSSE), an

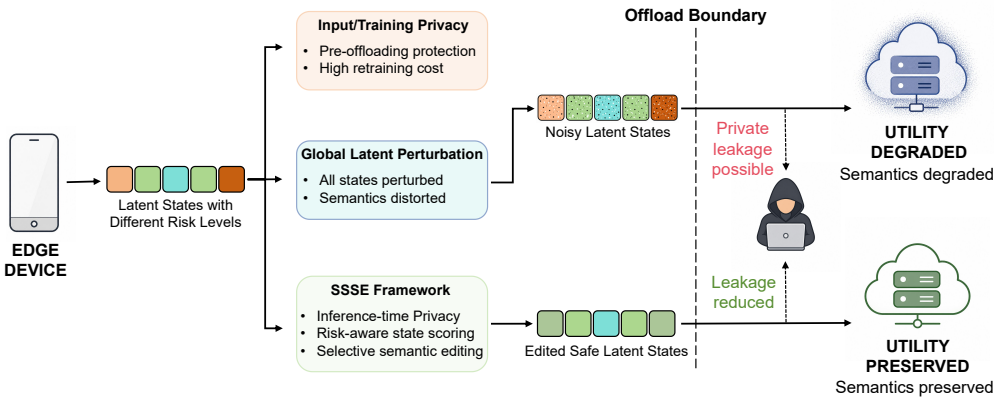


**Fig. 1** Overview and motivation of SSSE for edge-cloud split inference. Existing input/training-stage privacy mechanisms and global latent perturbation either operate before latent offloading or perturb the transmitted representation broadly, which may leave private attributes recoverable or degrade task semantics. SSSE instead performs inference-time, risk-aware semantic editing at the offload boundary, selectively transforming high-risk latent states to reduce sensitive-attribute leakage while preserving task-relevant semantics.

inference-time privacy-preserving framework for edge-cloud SLM deployments. SSSE is a post-hoc defense that operates without modifying or retraining the pre-trained SLM backbone, making it suitable for practical deployment scenarios where backbone retraining is economically costly or operationally infeasible. SSSE performs privacy intervention at the offload boundary, where latent states are exposed to the untrusted server, and edits a subset of latent states with elevated leakage risk to avoid unnecessary distortion of the full transmitted representation. Given the hidden states produced by the device-side layers, SSSE uses a learnable risk scorer to identify privacy-sensitive token positions, enabling data-driven localization of vulnerable states. For each high-risk state, SSSE constructs a privacy-aware semantic candidate bank by combining a dynamic input-conditioned local reservoir with a static on-device prototype cache, thereby providing context-relevant and device-local candidates without introducing new cross-boundary exposure. SSSE then transforms vulnerable states into task-consistent semantic alternatives before transmission. This design replaces indiscriminate global perturbation with selective semantic editing, enabling SSSE to protect privacy-sensitive latent states while more effectively preserving task-relevant semantics during edge-cloud SLM inference. Figure 1 illustrates the overview and motivation of our framework.

In this paper, our contributions can be summarized as follows:

- We formulate representation-level privacy protection in edge-cloud collaborative inference as a selective semantic editing problem, highlighting that privacy protection in SLM split inference should operate on transmitted latent states rather than only on raw inputs or global perturbations.

- We propose SSSE, a post-hoc inference-time defense that provides risk-aware protection at the offload boundary, reducing leakage of sensitive attributes while preserving task-relevant semantic structure without retraining the backbone.
- We develop an on-device semantic editing mechanism that combines contextual candidate retrieval with prototype-based support, enabling privacy-preserving latent transformation without introducing additional cross-boundary communication overhead and yielding a favorable privacy–utility balance in deployment.

We evaluate SSSE against representative baselines under the split-inference protocol described in this paper. Experimental results show that our framework consistently reduces the recoverability of sensitive attributes while preserving downstream task performance. On the topic classification task, SSSE achieves an accuracy of 0.4691 and a macro-F1 of 0.1890, compared with a baseline of 0.4771 and 0.1920. On the gender privacy task, it reduces attacker accuracy and macro-F1 from 0.6689 and 0.6539 to 0.6234 and 0.6288, respectively. On the more challenging age-group privacy task, SSSE further reduces attacker accuracy and macro-F1 from 0.6654 and 0.5403 to 0.6273 and 0.5053. In addition, deployment profiling under a heterogeneous edge-cloud setting shows that SSSE incurs a practical privacy-intervention overhead of approximately 2.8 ms/sample, supporting its feasibility for privacy-preserving edge-cloud inference. These results indicate that SSSE achieves near-baseline task utility through selective semantic editing while reducing privacy leaks.

The remainder of this paper is organized as follows. Section 2 reviews related work in split inference defenses and latent representation privacy leakages. Section 3 introduces the system model, threat model, and problem formulation. Section 4 presents the core components of the SSSE framework. Section 5 presents our empirical results. Finally, Section 6 concludes the paper.

## 2 Related Work

Split inference and collaborative inference have become practical paradigms for deploying deep models across resource-constrained devices and edge-cloud servers [4, 7, 9, 10]. Although split inference improves efficiency and reduces on-device overhead, intermediate representations transmitted during offloading still pose privacy risks. Prior studies have shown that latent representations remain vulnerable to inference attacks, including recent attacks on distributed language-model inference [32–34]. Therefore, related work centers on two themes: defenses for communicated intermediate representations and the analysis of privacy leakage in language model representations.

### 2.1 Defenses for Intermediate-Representation Privacy

Prior defenses related to intermediate-representation privacy commonly exhibited two limitations: direct latent-state defenses usually relied on representation-wide perturbation rather than selective protection, while input-side and training-stage methods did not directly protect transmitted latent states during split inference.

Some studies protected privacy by directly modifying the communicated representation itself. NoPeek [27] reduced leakage by minimizing dependence between raw

inputs and shared activations during distributed inference. This design established intermediate-representation protection as a valid defense target, but its protection objective remained coarse-grained and did not distinguish higher-risk from lower-risk latent states. Shredder et al. [28] advanced this direction by learning noise distributions for exposed intermediate states under an explicit privacy–utility trade-off. The learned perturbation was more adaptive than fixed noise injection, but the method still protected privacy mainly through global perturbation of the communicated representation. Mai et al. [29] proposed Split-and-Denoise (SnD), which extended this direction to the language-model setting by applying local differential privacy (LDP) privatization to transmitted embeddings and recovering part of the utility through denoising. The denoising stage partially mitigated performance loss, but the defense still treated transmitted embeddings as a homogeneous entity and did not model token-level privacy risk.

Related privacy mechanisms were also studied outside the split-inference setting. For input-level privacy, Harel et al. [24] protected privacy through token manipulation before downstream encoding. Kan et al. [35] reduced exposure in remote conversational systems by sanitizing text. Xu et al. [36] further explored privacy-aware text rewriting as an input-side transformation. These input-side methods reduced raw-text exposure before encoding, but did not directly control privacy leakage that persisted in intermediate representations after encoding. For training-stage privacy, Shi et al. [25] introduced selective differential privacy for language modeling by protecting designated sensitive spans rather than uniformly privatizing all text. Wu et al. [37] further advanced this direction by applying adaptive differential privacy to language model training. These training-stage methods constrained memorization and leakage during optimization, but did not provide direct inference-time protection for offloaded latent states.

## 2.2 Privacy Leakage in Language Model Representations

Studies on language-model representations provided strong evidence of privacy leakage, but most of them focused on diagnosing vulnerability rather than enabling direct protection during split inference.

Wan et al. [30] showed that input text could be reconstructed from LLM embeddings and further showed that recoverability varied across model layers. This result demonstrated that encoded representations still retained substantial input information, but the study remained centered on leakage characterization rather than protection. Dong et al. [31] further showed that internal states remained vulnerable to sensitive-attribute attacks even at deeper layers, challenging the assumption that deeper representations were inherently privacy-safe. This analysis extended the leakage picture beyond shallow embeddings, but it still did not provide a mechanism for inference-time intervention. Li et al. [21] and Chen et al. [38] studied embedding inversion attacks and showed that text embeddings preserved rich semantic information that could support reconstruction of original inputs, including in multilingual settings. These results strengthened the evidence that latent representations remained privacy-sensitive even after encoding.

Recent work further indicated that privacy leakage in language models was heterogeneous rather than uniformly distributed. Harel et al. [23] showed that privacy risk could arise at token granularity and was not evenly distributed across a sequence. Token-level heterogeneity, therefore, challenged representation-wide defenses and motivated protection mechanisms that could localize and intervene on high-risk latent states.

Taken together, existing leakage studies revealed where privacy risk lay in language-model representations, but they did not provide an inference-time mechanism to protect transmitted latent states during edge-cloud inference.

## 2.3 Significance of the Proposed SSSE

The studies reviewed above have shown that existing privacy-preserving methods for edge-cloud inference still leave a gap in the protection of selective latent states. Input-side sanitization and training-stage privacy mechanisms reduce exposure before latent representations are transmitted, but they do not directly protect the intermediate states exposed at the split boundary. Noise-based latent defenses operate closer to the communicated representation, but they usually perturb the transmitted states as a whole and may therefore introduce unnecessary distortion of task-relevant information. Meanwhile, recent analyses of language-model representations have shown that privacy leakage can be heterogeneous across token-level states, motivating protection mechanisms that can identify and intervene in the vulnerable parts of the transmitted representation.

Table 1 summarizes this positioning by comparing representative privacy-preserving paradigms for edge-cloud inference. This comparison considers whether each paradigm operates directly on transmitted latent states, supports selective intervention, incorporates semantic awareness, is applicable during deployment, and avoids backbone retraining. These dimensions reflect the main requirements of privacy protection in split-inference SLM deployment, where the exposed object is the offloaded latent representation and the defense should preserve downstream semantic utility.

Against this background, our work can be distinguished from prior paradigms in three aspects.

i We operate directly on intermediate latent states at inference time, targeting the representation actually exposed during offloading without requiring backbone retraining.



**Table 1** Comparison of representative privacy-preserving paradigms for edge-cloud inference. We compare whether a method directly operates on transmitted latent states, supports selective intervention, incorporates explicit semantic awareness, can be applied during deployment, and avoids retraining the backbone model.

| Method Paradigm | Latent-level | Selective | Semantic-aware | Post-hoc | Backbone-free |
|---|---|---|---|---|---|
| Noise-based latent defenses [22, 28, 29] | ✓ | ✗ | ✗ | ✓ | ✓ |
| Training-time latent learning [18, 19, 27] | ✓ | ✓ | ✗ | ✗ | ✗ |
| Text sanitization [24, 35, 36] | ✗ | ✓ | ✓ | ✓ | ✓ |
| Training-stage DP [25, 37] | ✗ | ✓ | ✗ | ✗ | ✗ |
| **Ours** | ✓ | ✓ | ✓ | ✓ | ✓ |

ii We perform risk-aware selective intervention, so that latent states with elevated privacy risk are edited while lower-risk states are largely preserved.
iii We edit vulnerable states toward privacy-aware semantic alternatives constructed locally on-device, reducing sensitive-attribute recoverability without introducing additional cross-boundary exposure.

This design combines deployment-time latent-state protection, selective intervention, and semantic-aware editing within a single edge-cloud split-inference framework.

The experimental evaluation further examines this positioning through representative baselines, including global perturbation, selective perturbation, masking, and replacement strategies.

# 3 System Model

This section formalizes the edge-cloud split-inference setting considered in this paper. We first define the collaborative inference pipeline and the intermediate representation exposed at the split boundary. We then present the threat model and define privacy leakage as sensitive-attribute recoverability from the transmitted representation.

**Table 2** Summary of Key Notations

| Notation | Description |
|---|---|
| $\mathcal{D}_i$ | The $i$-th user device in the edge-cloud split-inference system. |
| $x_i, y_i, \hat{y}_i$ | Input sequence, ground-truth task label, and predicted task output for device $\mathcal{D}_i$. |
| $l, n, d$ | Split layer index, sequence length, and hidden-state dimension. |
| $f_{\theta_c}^{(\leq l)}, f_{\theta_s}^{(>l)}$ | Device-side encoder up to split layer $l$ and server-side continuation after split layer $l$. |
| $H_i^{(l)}, h_{i,j}^{(l)}$ | Split-layer representation and the hidden state of the $j$-th token. |
| $\mathcal{T}$ | Device-side privacy-preserving transformation applied to the split-layer representation. |
| $\tilde{H}_i^{(l)}, \tilde{h}_{i,j}^{(l)}$ | Protected/edited split-layer representation and the edited state of the $j$-th token. |
| $s_i, \hat{s}_i$ | Ground-truth sensitive attribute and attacker-inferred sensitive attribute. |
| $\mathcal{A}_\omega$ | Attribute-inference adversary operating on the transmitted representation. |
| $\mathcal{L}_{\text{leak}}$ | Empirical sensitive-attribute recoverability from the transmitted representation. |
| $\Delta\mathcal{U}, \mathcal{E}$ | Utility degradation and intervention cost induced by the transformation $\mathcal{T}$. |
| $\epsilon_u, \epsilon_e$ | Maximum tolerable utility degradation and intervention cost. |
| $g_\psi$ | Learnable token-level risk scorer. |
| $r_{i,j}^{(l)}, \hat{r}_{i,j}^{(l)}$ | Raw and normalized privacy risk scores of token state $h_{i,j}^{(l)}$. |
| $m_{i,j}^{(l)}$ | Continuous editing intensity mask for the $j$-th token state. |
| $\mathcal{M}_i^{(l)}(x_i)$ | Device-local candidate bank containing semantic alternatives. |
| $\mathcal{R}_i^{(l)}(x_i)$ | Input-conditioned local reservoir constructed from low-risk states. |
| $\mathcal{P}^{(l)}$ | Static semantic prototype cache maintained on-device. |
| $z$ | Candidate latent state from the device-local candidate bank. |
| $u(h_{i,j}^{(l)}, z)$ | Privacy-aware compatibility score between target state $h_{i,j}^{(l)}$ and candidate $z$. |
| $\mathcal{N}_K(h_{i,j}^{(l)})$ | Top-$K$ privacy-aware semantic neighborhood for target state $h_{i,j}^{(l)}$. |
| $q_\phi(z \mid h_{i,j}^{(l)}, x_i)$ | Normalized aggregation weight assigned to candidate $z$. |
| $\overline{h}_{i,j}^{(l)}$ | Edited prototype aggregated from the selected semantic neighborhood. |
| $p_{\theta_s}(\cdot)$ | Predictive distribution induced by the server-side continuation. |
| $\mathcal{L}_{\text{task}}, \mathcal{L}_{\text{attr}}$ | Downstream task loss and sensitive-attribute inference loss. |
| $\mathcal{L}_{\text{priv}}, \mathcal{L}_{\text{cons}}, \mathcal{L}_{\text{sparse}}$ | Privacy loss, consistency regularization loss, and sparsity penalty. |
| $\lambda_1, \lambda_2, \lambda_3$ | Trade-off coefficients for privacy, consistency, and sparsity terms. |
| $\Delta_u, \Delta_p$ | Utility drop and privacy gain used in experimental evaluation. |

Finally, we formulate the system-level objective that motivates selective latent-state editing. For clarity, the key notations used in this paper are summarized in Table 2.

## 3.1 Collaborative Split Inference

We consider an edge-cloud split-inference setting with a set of user devices $\{\mathcal{D}_i\}_{i=1}^{N_{\text{dev}}}$ and an edge-cloud server. Each device supports low-latency local interaction but may not always execute the full SLM inference pipeline due to compute, memory, or energy constraints. Partial offloading is used when the device-side execution budget is insufficient.

Let $x_i$ denote an input sequence on device $\mathcal{D}_i$. The SLM is partitioned at split layer $l$. We denote the device-side encoder by $f_{\theta_e}^{(\leq l)}$ and the server-side continuation, including the remaining layers and task head, by $f_{\theta_s}^{(>l)}$. Here, $\theta_e$ and $\theta_s$ are the device-side and server-side model parameters, respectively.

The device-side computation up to the split layer is represented as

$$H_i^{(l)} = f_{\theta_e}^{(\leq l)}(x_i) = \left[h_{i,1}^{(l)}, \ldots, h_{i,n}^{(l)}\right] \in \mathbb{R}^{n \times d}. \tag{1}$$

Here, $H_i^{(l)}$ stacks $n$ token-level latent states with hidden dimension $d$, and $h_{i,j}^{(l)}$ denotes the latent state of the $j$-th token at the split layer. To protect the representation exposed at the split boundary, the device-side module maps $H_i^{(l)}$ to a protected representation:

$$\tilde{H}_i^{(l)} = \mathcal{T}\left(H_i^{(l)}; x_i\right). \tag{2}$$

The protected representation $\tilde{H}_i^{(l)}$ preserves the tensor format expected by the server-side model, so it can be used as a drop-in replacement for $H_i^{(l)}$ in the remaining inference pipeline. The raw input $x_i$, the original split-layer representation $H_i^{(l)}$, and any device-local information used by $\mathcal{T}$ are kept on the user device.

The server-side continuation then produces the public-task prediction from the protected representation:

$$\hat{y}_i = f_{\theta_s}^{(>l)}\left(\tilde{H}_i^{(l)}\right), \tag{3}$$

where $\hat{y}_i$ denotes the prediction for the public downstream task. This formulation identifies the transmitted intermediate representation as the main exposed object in split inference. Accordingly, the privacy problem studied in this paper is not direct raw-input exposure, but the recoverability of sensitive information from the offloaded latent representation.

## 3.2 Threat Model and Privacy Leakage

In the considered split-inference setting, raw inputs remain on the user device and only the protected intermediate representation $\tilde{H}_i^{(l)}$ is transmitted to the edge-cloud server. The server has no direct access to $x_i$ or the original split-layer representation $H_i^{(l)}$. However, the transmitted representation may still encode semantic and

attribute information about the user. Although communication encryption can protect the transmission channel, it does not prevent a legitimate receiver from analyzing the decrypted latent representation. The privacy risk studied in this paper is the recoverability of private information from $\tilde{H}_i^{(l)}$.

The primary adversary is an honest-but-curious edge-cloud server. It follows the prescribed inference protocol and returns the downstream prediction, while also attempting to infer private user attributes from the received representation. A passive on-path eavesdropper may pose an additional risk if channel protection is absent or compromised [32, 33].

We consider the adversary to be algorithm-aware but device-private. The adversary is aware of the split-inference setting and the device-side privacy transformation's high-level design. The transformation parameters, input-dependent auxiliary states, and intermediate transformation decisions are kept private on the user device. At runtime, the adversary observes the protected representation transmitted to the server and performs sensitive-attribute inference from this representation.

Let $s_i \in \mathcal{S}$ be the sensitive attribute associated with input $x_i$, with $\mathcal{S}$ denoting the sensitive-attribute label space. Given the representation $\tilde{H}_i^{(l)}$, the adversary produces an attribute prediction through a probing model:

$$\hat{s}_i = \mathcal{A}_\omega \left( \tilde{H}_i^{(l)} \right), \tag{4}$$

where $\mathcal{A}_\omega$ is the attribute-inference model parameterized by $\omega$, and $\hat{s}_i$ is the inferred sensitive attribute. A closer match between $\hat{s}_i$ and $s_i$ indicates stronger sensitive-attribute recoverability from the offloaded representation. We therefore define privacy leakage as the empirical recoverability of $s_i$ from the transmitted representation:

$$\mathcal{L}_{\text{leak}}(\mathcal{T}) = \mathcal{M}_{\text{att}} \left( \hat{s}_i, s_i \right), \tag{5}$$

where $\mathcal{M}_{\text{att}}(\cdot, \cdot)$ denotes an attacker-performance measure for sensitive-attribute inference. A lower value of $\mathcal{L}_{\text{leak}}$ indicates that the transmitted representation exposes less information about the sensitive attribute.

Other attacks on intermediate representations, such as prompt inference, are also relevant to distributed language-model inference [34]. This paper focuses on attribute inference because it provides a concrete and measurable representation-level privacy risk for evaluating split-boundary protection.

## 3.3 Problem Formulation

The system objective is to protect the representation transmitted during split inference. A privacy transformation cannot simply destroy or heavily randomize the latent states, because the transmitted representation is also the input required by the server-side model to complete the downstream task. This creates a central tension: the transformation should reduce private leakage at the offload boundary while preserving the task-relevant semantic structure needed for inference.

The desired transformation must balance utility preservation, privacy reduction, and intervention control. Utility preservation requires that the protected representation remain compatible with the server-side model, as the server relies on it to complete the downstream task. Privacy reduction requires the transmitted representation to expose less information about the sensitive attribute $s_i$, since the attribute-inference adversary observes the same representation used for inference. Intervention control is also necessary because broad modification of the full representation can distort task-relevant semantics and increase edge-side computation.

Therefore, we express this system-level objective as

$$\min_{\mathcal{T}} \ \mathcal{L}_{\text{leak}}(\mathcal{T}) \qquad \text{s.t.} \qquad \Delta\mathcal{U}(\mathcal{T}) \le \epsilon_u, \quad \mathcal{E}(\mathcal{T}) \le \epsilon_e, \tag{6}$$

where $\mathcal{L}_{\text{leak}}(\mathcal{T})$ measures sensitive-attribute recoverability from the transmitted representation, $\Delta\mathcal{U}(\mathcal{T}) \ge 0$ measures the downstream utility degradation relative to raw split inference, and $\mathcal{E}(\mathcal{T}) \ge 0$ measures the intervention cost introduced by the transformation. The intervention cost captures the scale of representation modification, such as how broadly or strongly latent states are edited. The constants $\epsilon_u \ge 0$ and $\epsilon_e \ge 0$ specify the maximum tolerable utility degradation and intervention cost, respectively. The formulation emphasizes a balanced transformation that reduces leakage within a minimal utility loss and intervention cost budget.

This objective motivates a selective split-boundary transformation that localizes privacy-sensitive latent states and edits them without broadly disrupting task-relevant semantics.

## 4 Selective Semantic State Editing Framework

The previous sections identify a central challenge in edge-cloud split inference with SLMs: transmitted latent states can leak sensitive information. At the same time, broad perturbation of the full representation can unnecessarily damage task-relevant semantics. Therefore, this section introduces Selective Semantic State Editing, an inference-time privacy-preserving framework that performs localized semantic intervention on transmitted latent representations. SSSE aims to reduce the recoverability of sensitive attributes from transmitted latent states while preserving downstream utility and limiting unnecessary intervention.

Figure 2 illustrates the overall workflow. At the split boundary, we first apply a learnable risk scorer to identify privacy-sensitive token-level latent states. We then construct an on-device candidate bank by combining an input-conditioned local reservoir with a static prototype cache, and retrieve privacy-aware semantic alternatives for high-risk states through top-$K$ ranking. Finally, we perform risk-aware semantic state editing before transmitting the edited representation to the server. The full inference procedure is summarized in Algorithm 1. The remaining subsections detail the three modules and the corresponding training objective.

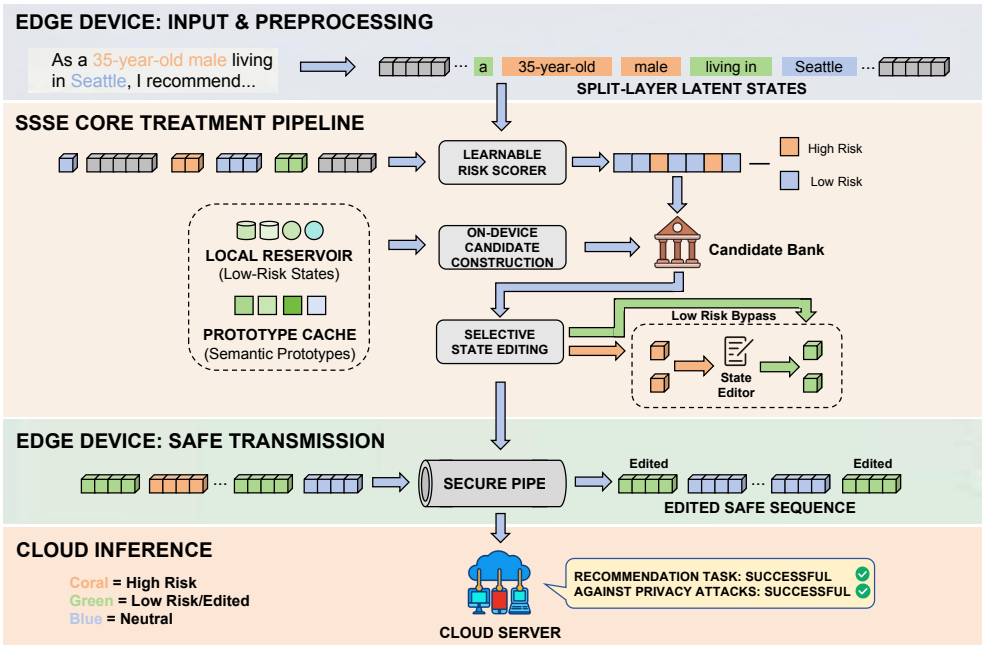


**Fig. 2** Framework of SSSE in edge-cloud split inference. At the offloading boundary, the framework first assigns token-level privacy risk scores, then constructs a device-local semantic neighborhood using an input-conditioned local reservoir and a static prototype cache, and finally performs selective state editing, transmitting only the edited representation to the cloud server for downstream inference.

## 4.1 Learnable Risk Scorer

We begin by estimating the privacy sensitivity of each token state in the split-layer representation. For device $\mathcal{D}_i$, the risk scorer maps each token state $h_{i,j}^{(l)} \in H_i^{(l)}$ to a scalar privacy-risk value:

$$r_{i,j}^{(l)} = g_\psi\left(h_{i,j}^{(l)}\right), \tag{7}$$

where $g_\psi$ denotes the learnable token-level risk scoring module. The purpose of this module is to localize latent states that are more likely to carry sensitive-attribute information before the representation is transmitted to the server. Such localization is important because privacy leakage in language-model representations can be concentrated in a subset of latent states, while applying the same intervention to all states can unnecessarily distort task-relevant information.

The risk score serves as the basis for token-level intervention at the split boundary. After sequence-wise normalization and mask generation, a higher score leads to a larger editing intensity for the corresponding token state, while a lower score keeps the state close to its original representation. In the current implementation, $g_\psi$ is instantiated as a lightweight MLP:

$$g_\psi\left(h_{i,j}^{(l)}\right) = \mathbf{v}_r^\top \delta\left(\mathbf{W}_r h_{i,j}^{(l)} + \mathbf{b}_r\right) + c_r, \tag{8}$$

**Algorithm 1** Selective Semantic State Editing for Split Inference

**Require:** Input sequence $x_i$; split layer $l$
**Ensure:** Output prediction $\hat{y}_i$

```
Compute split-layer hidden states H_i^(l) = {h_{i,1}^(l), ..., h_{i,n}^(l)}
if offloading is not required then
    Perform local inference and return ŷ_i
end if
Compute token-level risk scores r_{i,j}^(l) and editing masks m_{i,j}^(l)
Construct on-device candidate bank M_i^(l)(x_i)
for j = 1 to n do
    Retrieve neighborhood N_K(h_{i,j}^(l))
    Compute edited prototype h̄_{i,j}^(l)
    Set h̃_{i,j}^(l) = (1 − m_{i,j}^(l)) h_{i,j}^(l) + m_{i,j}^(l) h̄_{i,j}^(l)
end for
Form H̃_i^(l) = {h̃_{i,1}^(l), ..., h̃_{i,n}^(l)}
Transmit H̃_i^(l) to the edge or cloud server
Complete the remaining inference and return ŷ_i
```

where $\mathbf{W}_r$ and $\mathbf{b}_r$ are the hidden-layer weight matrix and bias vector, $\mathbf{v}_r$ and $c_r$ project the hidden feature to a scalar risk score, and $\psi = \{\mathbf{W}_r, \mathbf{b}_r, \mathbf{v}_r, c_r\}$ denotes the scorer parameters. This formulation provides a compact nonlinear mapping from the hidden-state space to a scalar intervention score. The scorer is optimized using the following training objective, which encourages edits that reduce sensitive-attribute recoverability while preserving task-relevant semantics. Thus, the MLP learns to assign higher scores to token states whose editing provides greater privacy intervention value under the utility constraint.

The scale of raw risk scores can vary across input sequences, which leads to unstable intervention intensity when scores are used directly for editing. To reduce this sequence-level scale effect, we normalize the scores within each input before mask generation:

$$\mu_i^{(l)} = \frac{1}{n}\sum_{t=1}^{n} r_{i,t}^{(l)}, \qquad \varsigma_i^{(l)} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}\left(r_{i,t}^{(l)} - \mu_i^{(l)}\right)^2 + \varepsilon}, \tag{9}$$

$$\hat{r}_{i,j}^{(l)} = \frac{r_{i,j}^{(l)} - \mu_i^{(l)}}{\varsigma_i^{(l)}}, \tag{10}$$

where $\mu_i^{(l)}$ and $\varsigma_i^{(l)}$ denote the mean and standard deviation of the raw risk scores for the current sequence, and $\varepsilon > 0$ is a small constant for numerical stability. This normalization places the risk scores of each input into a standardized score space, making the subsequent intervention less sensitive to sequence-level score-scale variations.

To use the normalized risk value as a continuous editing controller, we convert it into a differentiable editing mask:

$$m_{i,j}^{(l)} = \sigma\left(\frac{\hat{r}_{i,j}^{(l)} - \tau}{\gamma}\right), \tag{11}$$

where $\sigma(\cdot)$ denotes the sigmoid function, $\tau \in \mathbb{R}$ is a risk threshold in the normalized score space, and $\gamma > 0$ controls the softness of the transition. The mask value $m_{i,j}^{(l)} \in (0,1)$ determines the editing intensity assigned to token state $h_{i,j}^{(l)}$. A larger mask value moves the state more strongly toward a privacy-aware semantic alternative, while a smaller mask value preserves more of the original representation.

The soft mask serves as a differentiable relaxation of discrete state selection. It allows Selective Semantic State Editing to concentrate intervention on high-risk latent states while preserving low-risk states that may still contain task-relevant semantics. The resulting masks $\{m_{i,j}^{(l)}\}_{j=1}^{n}$ therefore define both the scope and the strength of the subsequent semantic editing operation.

## 4.2 On-Device Candidate Construction

For token states assigned non-negligible editing intensity, SSSE requires replacement candidates that are both privacy-safer and semantically compatible. All candidates are constructed locally on the device to avoid additional cross-boundary exposure and communication overhead, which remains a key concern in edge-cloud transmission systems [39]. The candidate bank combines an input-conditioned local reservoir with a static prototype cache: the former provides context-specific alternatives, while the latter expands semantic support when the current input contains few suitable low-risk states.

For device $\mathcal{D}_i$ and input sequence $x_i$, the on-device candidate bank at split layer $l$ is defined as

$$\mathcal{M}_i^{(l)}(x_i) = \mathcal{R}_i^{(l)}(x_i) \cup \mathcal{P}^{(l)} = \{z_{i,a}^{(l)}\}_{a=1}^{M_i}, \tag{12}$$

where $\mathcal{R}_i^{(l)}(x_i)$ denotes the input-conditioned local reservoir, $\mathcal{P}^{(l)}$ denotes the static prototype cache, and $M_i$ is the number of available candidates for the current input.

The local reservoir is constructed from token states in the current sequence whose editing masks indicate low privacy risk:

$$\mathcal{R}_i^{(l)}(x_i) = \left\{ h_{i,t}^{(l)} \;\middle|\; m_{i,t}^{(l)} \le \rho,\ t = 1, \ldots, n \right\}, \tag{13}$$

where $\rho \in (0,1)$ is a reservoir threshold applied to the soft editing mask. This construction uses low-risk states from the same input as candidate alternatives, so the reservoir provides context-compatible semantic support without requiring any additional cross-boundary communication. The threshold $\rho$ controls the strictness of candidate admission: a smaller value selects only states with lower estimated privacy risk, while a larger value provides a broader candidate set.

The prototype cache complements the local reservoir by storing fixed semantic representatives on-device. Let $\mathcal{D}_{\text{pub}}$ denote a sanitized public corpus used only for offline prototype construction, and let $\mathcal{Z}_{\text{pub}}^{(l)}$ be the collection of split-layer token states extracted from this corpus:

$$\mathcal{Z}_{\text{pub}}^{(l)} = \left\{ h^{(l)}(x,t) \;\middle|\; x \in \mathcal{D}_{\text{pub}},\ t = 1, \ldots, n_x \right\}. \tag{14}$$

The public states are partitioned into $N$ clusters $\{\mathcal{C}_k^{(l)}\}_{k=1}^{N}$, where $N \in \mathbb{N}_+$ denotes the prototype cache size, and each prototype is computed as the centroid of one cluster:

$$p_k^{(l)} = \frac{1}{|\mathcal{C}_k^{(l)}|} \sum_{h \in \mathcal{C}_k^{(l)}} h, \qquad \mathcal{P}^{(l)} = \{p_k^{(l)}\}_{k=1}^{N}. \tag{15}$$

Because $\mathcal{P}^{(l)}$ is built offline and stored locally, it enlarges the semantic candidate space without adding runtime communication overhead. The cache is particularly useful when the current input provides only limited low-risk states suitable for replacement.

The candidate bank provides a pool of device-local alternatives, but candidate construction alone does not determine which alternatives are suitable for a given high-risk state. A suitable candidate should satisfy three requirements: it should remain semantically close to the target state, avoid excessive representation shift, and have a lower estimated privacy risk. We therefore define a privacy-aware compatibility score for a target state $h_{i,j}^{(l)}$ and a candidate $z \in \mathcal{M}_i^{(l)}(x_i)$:

$$u\left(h_{i,j}^{(l)}, z\right) = -\alpha\, d_{\text{sem}}\left(h_{i,j}^{(l)}, z\right) - \eta\, c\left(h_{i,j}^{(l)}, z\right) + \beta\, s_{\text{priv}}(z), \tag{16}$$

where $\alpha, \eta, \beta \geq 0$ control the relative importance of semantic fidelity, replacement cost, and privacy preference, respectively.

The semantic distance is computed by cosine distance:

$$d_{\text{sem}}\left(h_{i,j}^{(l)}, z\right) = 1 - \frac{\left(h_{i,j}^{(l)}\right)^{\mathsf{T}} z}{\left\|h_{i,j}^{(l)}\right\|_2 \|z\|_2 + \varepsilon}. \tag{17}$$

This term favors candidates that preserve the semantic direction of the original token state. To prevent an edited state from moving too far from the original representation, we further define a normalized replacement cost:

$$c\left(h_{i,j}^{(l)}, z\right) = \frac{\left\|h_{i,j}^{(l)} - z\right\|_2^2}{\left\|h_{i,j}^{(l)}\right\|_2^2 + \varepsilon}. \tag{18}$$

This cost penalizes candidates that would introduce a large displacement in the latent space.

The privacy preference term promotes candidates with lower estimated privacy risk:

$$s_{\mathrm{priv}}(z) = 1 - \sigma\left(\frac{g_\psi(z) - \tau_p}{\gamma_p}\right), \tag{19}$$

where $\tau_p \in \mathbb{R}$ and $\gamma_p > 0$ are calibration parameters for candidate privacy scoring. A larger value of $s_{\mathrm{priv}}(z)$ indicates that the candidate is estimated to be less privacy-sensitive. Therefore, the compatibility score in Eq. (16) selects candidates that are semantically close to the target state, avoid excessive distortion, and are less likely to preserve sensitive-attribute information.

Based on this score, we define the privacy-aware semantic neighborhood of the target state as the top-$K$ candidates in the local candidate bank:

$$\mathcal{N}_K\left(h_{i,j}^{(l)}\right) = \mathrm{TopK}_{z \in \mathcal{M}_i^{(l)}(x_i)}\left[u\left(h_{i,j}^{(l)}, z\right)\right]. \tag{20}$$

The neighborhood $\mathcal{N}_K(h_{i,j}^{(l)})$ contains the most compatible semantic alternatives for editing the target state, where $K \in \{1, \ldots, M_i\}$ denotes the neighborhood size.

To aggregate information from this neighborhood, we assign normalized weights to the selected candidates:

$$q_\phi\left(z \mid h_{i,j}^{(l)}, x_i\right) = \frac{\exp\left(u\left(h_{i,j}^{(l)}, z\right)/\kappa\right)}{\sum_{z' \in \mathcal{N}_K(h_{i,j}^{(l)})} \exp\left(u\left(h_{i,j}^{(l)}, z'\right)/\kappa\right)}, \qquad z \in \mathcal{N}_K(h_{i,j}^{(l)}), \tag{21}$$

where $\kappa > 0$ is a temperature parameter controlling the sharpness of candidate aggregation, and $\phi$ denotes the set of parameters and hyperparameters involved in candidate scoring and aggregation. Candidates with higher compatibility scores receive larger aggregation weights. The resulting neighborhood and aggregation weights are then used to construct the edited prototype during the selective state-editing stage.

## 4.3 Selective State Editing

The preceding modules provide three quantities for each token state: the editing mask $m_{i,j}^{(l)}$, the privacy-aware semantic neighborhood $\mathcal{N}_K(h_{i,j}^{(l)})$, and the corresponding aggregation weights $q_\phi(z \mid h_{i,j}^{(l)}, x_i)$. Selective state editing uses these quantities to construct a privacy-aware alternative for each token state and then applies a risk-controlled interpolation before offloading. The purpose of this step is to suppress sensitive-attribute cues in high-risk states while preserving task-relevant information in low-risk states.

For each target state $h_{i,j}^{(l)}$, we first aggregate the selected semantic neighborhood into an edited prototype:

$$\overline{h}_{i,j}^{(l)} = \sum_{z \in \mathcal{N}_K(h_{i,j}^{(l)})} q_\phi\left(z \mid h_{i,j}^{(l)}, x_i\right) z. \tag{22}$$

The prototype $\overline{h}_{i,j}^{(l)}$ represents a privacy-aware semantic alternative to the original state. Because the compatibility score in Eq. (21) determines the aggregation weights, candidates that are semantically close, less disruptive, and less privacy-sensitive contribute more strongly to the edited prototype. We then define the semantic editing direction as the displacement from the original state to the edited prototype:

$$\Delta h_{i,j}^{(l)} = \overline{h}_{i,j}^{(l)} - h_{i,j}^{(l)}. \tag{23}$$

This displacement specifies how the original latent state should move toward a privacy-aware alternative in the latent space. Instead of applying this displacement uniformly to all token states, SSSE scales the editing direction by the learned risk-aware mask:

$$\tilde{h}_{i,j}^{(l)} = h_{i,j}^{(l)} + m_{i,j}^{(l)} \Delta h_{i,j}^{(l)}. \tag{24}$$

Equivalently,

$$\tilde{h}_{i,j}^{(l)} = \left(1 - m_{i,j}^{(l)}\right) h_{i,j}^{(l)} + m_{i,j}^{(l)} \overline{h}_{i,j}^{(l)}. \tag{25}$$

This formulation provides a continuous transition from retaining the original representation to adopting the privacy-aware semantic alternative. When $m_{i,j}^{(l)}$ is close to zero, the edited state remains close to the original state, which preserves task-relevant semantics for low-risk tokens. When $m_{i,j}^{(l)}$ is large, the state is moved more strongly toward $\overline{h}_{i,j}^{(l)}$, which suppresses sensitive cues in high-risk token states. Thus, the mask controls both the scope and the strength of the intervention.

Applying the same editing rule to all token states gives a representation-level transformation. For the matrix form, we stack token states row-wise as $H_i^{(l)} \in \mathbb{R}^{n \times d}$ and $\overline{H}_i^{(l)} \in \mathbb{R}^{n \times d}$, where the $j$-th rows correspond to $h_{i,j}^{(l)}$ and $\overline{h}_{i,j}^{(l)}$, respectively. Let $\mathbf{m}_i^{(l)} = [m_{i,1}^{(l)}, \ldots, m_{i,n}^{(l)}]^\mathsf{T}$ be the token-level editing mask. We expand it to $\mathbf{G}_i^{(l)} = \mathbf{m}_i^{(l)} \mathbf{1}_d^\mathsf{T}$, where $\mathbf{1}_d \in \mathbb{R}^d$ is an all-one vector. The edited split-layer representation is then given by

$$\tilde{H}_i^{(l)} = H_i^{(l)} + \mathbf{G}_i^{(l)} \odot \left(\overline{H}_i^{(l)} - H_i^{(l)}\right), \tag{26}$$

where $\odot$ denotes element-wise multiplication.

This matrix form shows that SSSE performs continuous, risk-controlled editing on the transmitted representation. Low-risk states are assigned small mask values and therefore remain close to their original representations, while high-risk states are moved toward privacy-aware semantic alternatives. The server receives only $\tilde{H}_i^{(l)}$. The original representation $H_i^{(l)}$, the trained risk-scorer parameters, the local reservoir, the input-specific candidate bank, the compatibility scores, and the editing masks remain on-device. Therefore, the intervention is localized to the offloading boundary and does not introduce additional cross-boundary exposure.

### 4.4 Training Objective

The preceding modules define how SSSE identifies high-risk states, constructs semantic alternatives, and edits the transmitted representation. We now optimize these components with a differentiable objective that balances task preservation, adversarial privacy suppression, semantic consistency, and sparse intervention.

SSSE is trained with

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda_1 \mathcal{L}_{\text{priv}} + \lambda_2 \mathcal{L}_{\text{cons}} + \lambda_3 \mathcal{L}_{\text{sparse}}, \tag{27}$$

where $\lambda_1, \lambda_2, \lambda_3 \geq 0$ control the operating point among privacy reduction, semantic preservation, and intervention sparsity. The coefficients $\lambda_1$, $\lambda_2$, and $\lambda_3$ are selected jointly on the validation set rather than tuned in isolation. For each candidate configuration, we evaluate four validation criteria: public-task performance after editing, predictive consistency between the original and edited representations, residual sensitive-attribute recoverability, and average editing intensity. The final configuration is chosen as the operating point that substantially reduces sensitive-attribute recoverability while keeping the degradation of public-task performance and predictive consistency within an acceptable range and avoiding excessive editing.

The role of $\lambda_1$ is to determine the strength of the privacy-suppression signal. However, optimizing this term alone may remove not only sensitive cues but also task-relevant semantic factors. Therefore, $\lambda_2$ acts as a semantic-stability anchor by penalizing the divergence between the server-side predictive distribution produced from the edited representation and that produced from the original representation. This discourages edits that alter the task-level semantic behavior of the representation. The sparsity weight $\lambda_3$ further constrains the intervention by penalizing average mask activation, thereby encouraging localized edits.

The task-preservation term is defined as

$$\mathcal{L}_{\text{task}} = \mathbb{E}_{(x_i, y_i)} \left[ \ell_{\text{task}} \left( f_{\theta_s}^{(>l)}(\tilde{H}_i^{(l)}), y_i \right) \right], \tag{28}$$

where $\ell_{\text{task}}(\cdot, \cdot)$ is the supervised loss for the public downstream task. This term keeps the edited representation useful for server-side inference.

To suppress sensitive-attribute recoverability, we use the attribute-inference loss of the probing adversary:

$$\mathcal{L}_{\text{attr}} = \mathbb{E}_{(x_i, s_i)} \left[ \ell_{\text{attr}} \left( \mathcal{A}_\omega(\tilde{H}_i^{(l)}), s_i \right) \right], \tag{29}$$

where $\ell_{\text{attr}}(\cdot, \cdot)$ is the supervised loss for sensitive-attribute prediction. The privacy term for the editing module is defined in the adversarial direction:

$$\mathcal{L}_{\text{priv}} = -\mathcal{L}_{\text{attr}}. \tag{30}$$

Thus, minimizing $\mathcal{L}_{\text{priv}}$ with respect to the SSSE parameters reduces the information available to the attribute-inference adversary.

To limit semantic drift, we add a consistency regularizer between the server-side predictive distributions induced by the original and edited split-layer representations:

$$\mathcal{L}_{\text{cons}} = \mathbb{E}_{x_i}\left[D_{\text{KL}}\left(p_{\theta_s}(H_i^{(l)}) \parallel p_{\theta_s}(\tilde{H}_i^{(l)})\right)\right], \tag{31}$$

where $p_{\theta_s}(\cdot)$ denotes the predictive distribution induced by the server-side continuation. This term preserves the task-level behavior of the original representation.

Finally, sparse intervention is encouraged by penalizing the average editing intensity:

$$\mathcal{L}_{\text{sparse}} = \mathbb{E}_{x_i}\left[\frac{1}{n}\sum_{j=1}^{n} m_{i,j}^{(l)}\right]. \tag{32}$$

Since $m_{i,j}^{(l)}$ controls the editing intensity of the $j$-th token state, this term assigns an explicit cost to mask activation. It regularizes the risk scorer by discouraging broad or weakly justified interventions across the transmitted representation. Under the full objective, increasing a mask value is encouraged only when the corresponding edit provides sufficient privacy gain to offset the sparsity cost and the utility-related penalties imposed by the task and consistency terms. Thus, $\mathcal{L}_{\text{sparse}}$ promotes compact, high-value interventions and helps preserve task utility by limiting edits to states with stronger privacy-reduction benefit.

Training follows an alternating optimization procedure. The attacker is first updated to improve sensitive-attribute prediction:

$$\min_{\omega} \ \mathcal{L}_{\text{attr}}. \tag{33}$$

The SSSE modules are then updated against this attacker:

$$\min_{\psi,\phi} \ \mathcal{L}_{\text{task}} - \lambda_1 \mathcal{L}_{\text{attr}} + \lambda_2 \mathcal{L}_{\text{cons}} + \lambda_3 \mathcal{L}_{\text{sparse}}. \tag{34}$$

The backbone SLM remains frozen during privacy-module training. SSSE therefore improves split-boundary privacy by learning the risk scorer and candidate-based editing module without retraining the backbone.



# 5 Experiments

This section evaluates SSSE in the edge-cloud SLM split-inference setting. We first describe the dataset, model split configuration, baselines, and training and evaluation protocol. We then report the main privacy–utility results on topic classification and two private-attribute inference tasks, followed by robustness analysis, ablation and sensitivity studies, and deployment overhead profiling.

## 5.1 Experimental Setup

### 5.1.1 Dataset and Preprocessing

We conduct the main experiments on the **Blog Authorship Corpus** [40], a large-scale blog dataset with author-level demographic metadata and topical labels. The raw corpus contains approximately 19,320 bloggers and 681,284 posts, with metadata fields including gender, age, topic, zodiac sign, date, and text.

We use topic classification as the public utility task, and use gender and age group as the private attributes. To construct a clean benchmark for privacy-preserving split inference, we remove empty posts, discard posts with fewer than 30 words, exclude samples with unknown topical labels (`indUnk`), and remove authors with fewer than 5 cleaned posts. For the utility task, we retain the 10 most frequent topics: *Student*, *Technology*, *Arts*, *Education*, *Communications-Media*, *Internet*, *Non-Profit*, *Engineering*, *Law*, and *Publishing*. For age-group inference, author ages are discretized into three groups: 13–22, 23–32, and 33–48.

After preprocessing, the benchmark contains 7,157 authors and 259,478 posts. The dataset is split at the author level into training, validation, and test sets using a 70/15/15 split with a random seed of 42. The split is author-disjoint, so no author appears in more than one partition. To preserve the distributions of the utility task and the primary privacy task, the partition is stratified by topic and gender. Table 3 summarizes the processed benchmark distribution.

### 5.1.2 Model and Split Configuration

We build SSSE on top of MiniLM-L12-H384-uncased [1] as the backbone SLM, given its favorable efficiency–performance trade-off for resource-constrained edge deployment. Following the split-inference setting considered in this paper, the encoder is partitioned at layer $L_s = 6$ of 12 Transformer layers, with the edge device executing the initial 6 layers and the server executing the remaining 6. The maximum input length is fixed to 256 tokens in all experiments.

SSSE is applied directly to the 384-dimensional hidden states at the split boundary. The learnable risk scorer $g_\psi$ is implemented as a lightweight MLP that produces token-level risk scores and the corresponding soft editing masks. For the main utility

**Table 3** Statistics of the processed Blog Authorship benchmark.

| Statistic | Train | Val | Test | Total |
|---|---|---|---|---|
| Authors | 5,009 | 1,074 | 1,074 | 7,157 |
| Posts | 183,469 | 39,650 | 36,359 | 259,478 |
| Male authors | 2,738 | 587 | 586 | 3,911 |
| Female authors | 2,271 | 487 | 488 | 3,246 |
| Age 13–22 | 2,559 | 553 | 547 | 3,659 |
| Age 23–32 | 1,877 | 406 | 409 | 2,692 |
| Age 33–48 | 573 | 115 | 118 | 806 |

task, we use a split MiniLM topic classification model with a linear classification head over the mean-pooled final hidden representation. For privacy evaluation, we use a separate MLP-based attribute inference model operating on the mean-pooled split-layer representation to predict private attributes from the offloaded latent states.

### 5.1.3 Baselines

We compare SSSE against seven representation baselines under the same backbone, split layer, data split, and attacker protocol. Raw Transmission offloads split-layer hidden states without modification and serves as the utility upper bound and privacy lower bound. We first include global representation defenses that modify the transmitted states as a whole. Global Gaussian Perturbation (G-All) adds isotropic Gaussian noise to all transmitted token states and represents the conventional global perturbation paradigm in split inference. Shredder-inspired Learned Noise (Shredder-LN) adapts the method of Shredder [28] as a classical global learned-noise perturbation baseline. SnD-inspired LDP-Denoising (SnD-LDP) implements an LDP perturbation and denoising baseline inspired by Split-and-Denoise [29]. Shredder-LN and SnD-LDP are trained before deployment and evaluated as global representation defenses.

We further compare SSSE with budgeted selective defenses under the same intervention budget. For these baselines, T$r$% denotes a selective setting in which the top $r$% token states are modified according to the corresponding selection rule. Selective Gaussian Perturbation (G-T$r$%) ranks token states by latent norm and injects Gaussian noise into the selected states. Random Masking (Mask-T$r$%) zeros out the selected states. Random Replacement (Replace-T$r$%) substitutes the selected states with randomly sampled states from the same sequence.

## 5.2 Training and Evaluation Protocol

Training follows the joint optimization objective defined in the previous section, which balances task utility, privacy reduction, semantic consistency, and sparse intervention. We use AdamW with $5 \times 10^{-5}$ learning rate, 32 batch size, and 0.01 weight decay. Hyperparameters are selected on the validation set using discrete candidate sets. For the joint loss, we search $\lambda_1 \in \{0.25, 0.5, 0.75, 1.0\}$, $\lambda_2 \in \{0.25, 0.5, 0.75, 1.0\}$, and $\lambda_3 \in \{0.01, 0.02, 0.05, 0.1, 0.2\}$, and use $\lambda_1 = 0.5$, $\lambda_2 = 1.0$, and $\lambda_3 = 0.05$ in the main experiments. For the semantic candidate module, we search neighborhood sizes $K \in \{8, 12, 16, 24, 32\}$ and prototype cache sizes $N \in \{64, 128, 256, 512\}$, and use $K = 16$ and $N = 256$ as the default configuration. All experiments are conducted on a local machine equipped with an Intel Core Ultra 5 125H CPU, an NVIDIA GeForce RTX 4060 Laptop GPU with 8 GB memory, and 32 GB of system RAM.

Privacy evaluation is conducted on two attribute inference tasks. Gender inference is used as the primary privacy task because it provides a clean binary setting for measuring empirical reductions in the recoverability of sensitive attributes. Age-group inference is used as a more challenging multi-class task to evaluate whether the defense remains effective when demographic leakage is more entangled. To reduce sensitivity to attacker initialization, privacy results are reported as the mean over five independent attacker runs with different random seeds unless otherwise stated. This protocol follows

**Table 4** Main benchmark results on the topic classification task and the two privacy inference tasks. Topic utility is measured by classification accuracy and macro-F1. Privacy is measured by attacker accuracy and macro-F1 for gender and age-group inference. Utility drop is computed as $\Delta_u = \mathrm{F1}_{\mathrm{Raw}} - \mathrm{F1}_{\mathrm{Topic}}$, and privacy gain is computed as $\Delta_p = \mathrm{F1}_{\mathrm{Raw}} - \mathrm{F1}_{\mathrm{Attack}}$ within each privacy task. Larger $\Delta_p$ and smaller $\Delta_u$ are preferred. For budgeted selective defenses, T20% denotes a fixed intervention budget of 20% token states.

| Method | Topic Utility | | | Gender Privacy | | | Age-Group Privacy | | |
|---|---|---|---|---|---|---|---|---|---|
| | Topic Acc ↑ | Topic F1 ↑ | $\Delta_u$ ↓ | Attack Acc ↓ | Attack F1 ↓ | $\Delta_p$ ↑ | Attack Acc ↓ | Attack F1 ↓ | $\Delta_p$ ↑ |
| Raw | 0.4771 | 0.1920 | - | 0.6689 | 0.6539 | - | 0.6654 | 0.5403 | - |
| Shredder-LN [28] | 0.4676 | 0.1873 | 0.0047 | 0.6382 | 0.6362 | 0.0177 | 0.6439 | 0.5344 | 0.0059 |
| SnD-LDP [29] | 0.4240 | 0.1538 | 0.0382 | 0.6522 | 0.6438 | 0.0101 | 0.6348 | **0.5017** | **0.0386** |
| G-All | 0.4445 | 0.1799 | 0.0121 | 0.6561 | 0.6487 | 0.0052 | 0.6549 | 0.5299 | 0.0104 |
| G-T20% | 0.4587 | 0.1840 | 0.0080 | 0.6508 | 0.6452 | 0.0087 | 0.6583 | 0.5342 | 0.0061 |
| Mask-T20% | 0.4594 | 0.1843 | 0.0077 | 0.6468 | 0.6367 | 0.0172 | 0.6453 | 0.5259 | 0.0144 |
| Replace-T20% | 0.4574 | 0.1834 | 0.0086 | 0.6567 | 0.6490 | 0.0049 | 0.6592 | 0.5368 | 0.0035 |
| **SSSE** | **0.4691** | **0.1890** | **0.0030** | **0.6234** | **0.6288** | **0.0251** | **0.6273** | 0.5053 | 0.0350 |

the threat model adopted in this paper, where privacy leakage is measured by the recoverability of private attributes from offloaded split-layer representations.

Evaluation covers three dimensions: utility, privacy, and system efficiency. Utility is measured by topic classification accuracy and macro-F1. Privacy leakage is measured by attacker accuracy and macro-F1 for gender and age-group inference. Lower attacker performance, together with smaller degradation in topic performance, indicates a stronger privacy–utility trade-off. Macro-F1 is reported together with accuracy because both topic labels and privacy attributes remain imbalanced after preprocessing and author-level splitting. System efficiency is measured by end-to-end latency, reported in milliseconds per sample (ms/sample), and throughput, reported in samples per second (samples/s), which assesses whether the additional privacy intervention remains compatible with practical edge-cloud collaborative inference.

## 5.3 Main Results

### 5.3.1 Benchmark Results

We first compare SSSE with representative latent-space baselines under the same split-inference protocol. The baselines include global perturbation, budgeted selective perturbation, masking, replacement, Shredder-LN, and SnD-LDP. Table 4 summarizes the results on topic classification together with the two privacy inference tasks. The main observation is that SSSE provides a more favorable privacy–utility trade-off, reducing sensitive-attribute recoverability on both privacy tasks while keeping topic classification performance close to Raw transmission.

Raw transmission provides the expected utility upper bound because no privacy intervention is applied. Among all privacy-preserving methods, SSSE remains closest to this upper bound, achieving a topic accuracy of 0.4691 and a macro-F1 of 0.1890, with only a 0.0030 drop in topic macro-F1. Shredder-LN also preserves topic utility relatively well, with a topic macro-F1 of 0.1873, but its privacy gain is limited, especially on age-group inference. SnD-LDP achieves a stronger age-group privacy reduction, but this comes with a much larger utility cost, reducing topic macro-F1 to 0.1538 and causing a 0.0382 utility drop. This result indicates that denoising can recover some

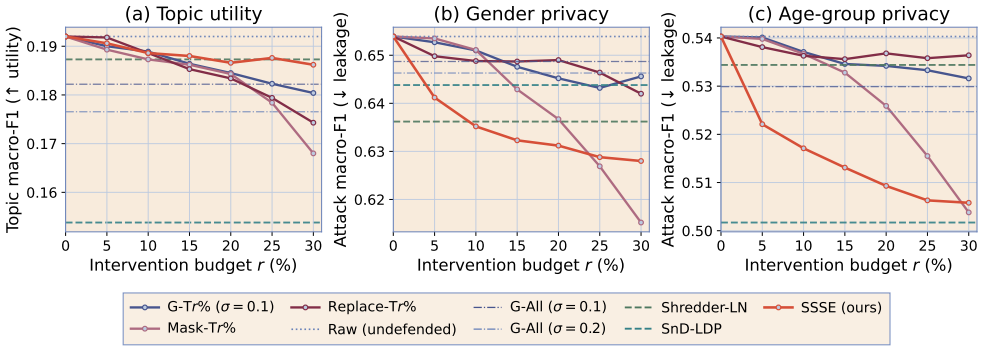


**Fig. 3** Privacy–utility trade-off across intervention budgets. Panel (a) reports topic macro-F1, while panels (b) and (c) report attacker macro-F1 on gender and age-group inference, respectively. The budget $r$ denotes the proportion of token states modified by each budgeted selective defense. Raw transmission, global Gaussian perturbation, Shredder-LN, and SnD-LDP are included as budget-independent references. Budgeted selective baselines reduce attacker macro-F1 at larger budgets but usually introduce greater topic-utility degradation. SSSE achieves a more balanced reduction in sensitive-attribute recoverability while preserving stronger topic utility across the budget range.

utility after an LDP perturbation, but strong representation-level privatization may still remove task-relevant information in the split-layer setting.

The privacy results show a clearer separation between SSSE and the baselines. On gender inference, SSSE reduces attacker accuracy and macro-F1 from 0.6689 and 0.6539 to 0.6234 and 0.6288, respectively. On age-group inference, it further reduces these metrics from 0.6654 and 0.5403 to 0.6273 and 0.5053. Although SnD-LDP achieves the lowest age-group attacker macro-F1 of 0.5017, its utility drop is much larger than SSSE's drop. Overall, global perturbation, budgeted selective perturbation, masking, replacement, learned noise, and LDP-denoising each provide partial protection, but they do not consistently balance privacy reduction and semantic utility preservation. SSSE is more effective under the split-inference setting via selectively editing high-risk latent states toward semantically compatible alternatives. This design improves privacy gain under a small utility budget, reducing private-attribute recoverability while preserving the representation structure needed for topic inference.

### 5.3.2 Budgeted Privacy–Utility Trade-off

Table 4 reports one operating point for each method, but budgeted selective defenses are inherently affected by the proportion of token states being modified. We therefore vary the intervention budget $r$ from 5% to 30% and compare SSSE with budgeted selective baselines under matched editing ratios. Raw transmission, global Gaussian perturbation, Shredder-LN, and SnD-LDP are included as budget-independent references because their protection modules are applied to the transmitted representation as a whole rather than to a budgeted subset of token states.

The results highlight a difference between semantic editing and standard noise injection under latent-state inversion attacks. Inversion relies on sensitive evidence remaining recoverable from the transmitted representation. Gaussian perturbation

reduces recoverability through general representation disturbance. Under moderate perturbation, part of the sensitive evidence can remain in the latent representation. Increasing the perturbation strength further suppresses recoverability, but also introduces larger deviations from the task-relevant semantic structure.

SSSE selectively edits token states associated with higher estimated privacy risk. The edited states are replaced with compatibility-selected semantic alternatives that remain close to the local semantic context while carrying weaker sensitive cues. As a result, the transmitted representation preserves more task-relevant structure while reducing the sensitive evidence available to the attacker. Unlike masking, which creates semantic holes by zeroing selected token states, this compatibility-aware editing helps preserve utility under larger intervention budgets. This behavior is reflected in Figure 3, where SSSE achieves lower attacker macro-F1 than masking, replacement, and Gaussian perturbation across most intervention budgets while maintaining stronger topic macro-F1.

### 5.3.3 Robustness to Attacker Initialization

We further examine whether the privacy reduction achieved by SSSE remains stable across attacker initializations. While the main benchmark reports the default evaluation setting, this robustness analysis retrains the auxiliary attribute-inference attacker with five independent random seeds and reports the mean and standard deviation of attacker performance. We compare Raw transmission, G-T20%, and SSSE under the same split-inference protocol. G-T20% is included as a representative budgeted selective Gaussian baseline.

Table 5 shows that the privacy advantage of SSSE remains stable across attacker initializations. On both gender and age-group inference, SSSE consistently achieves lower attacker accuracy and macro-F1 than Raw transmission and G-T20%. The standard deviations are small across the five attacker runs, indicating that the observed privacy reduction is not tied to a favorable attacker seed. Compared with the representative budgeted selective Gaussian baseline, SSSE provides stronger average privacy protection and more consistent reductions in attacker performance.

**Table 5** Robustness to attacker initialization over five independent attacker seeds. Results are reported as mean ± standard deviation. G-T20% denotes the budgeted selective Gaussian baseline with a fixed 20% intervention budget.

| Method | Gender Privacy | | Age-Group Privacy | |
|---|---|---|---|---|
| | Attack Acc ↓ | Attack F1 ↓ | Attack Acc ↓ | Attack F1 ↓ |
| Raw | $0.6690 \pm 0.0040$ | $0.6552 \pm 0.0049$ | $0.6651 \pm 0.0053$ | $0.5409 \pm 0.0057$ |
| G-T20% | $0.6511 \pm 0.0026$ | $0.6455 \pm 0.0029$ | $0.6507 \pm 0.0064$ | $0.5341 \pm 0.0105$ |
| **SSSE** | $\mathbf{0.6259 \pm 0.0025}$ | $\mathbf{0.6320 \pm 0.0036}$ | $\mathbf{0.6288 \pm 0.0047}$ | $\mathbf{0.5070 \pm 0.0024}$ |

**Table 6** Risk scorer ablation on the gender privacy task under a fixed-attacker protocol. High-Risk, Low-Risk, and Random denote budget-matched state selection policies before semantic editing. Utility drop and privacy gain are reported as $\Delta_u$ and $\Delta_p$, respectively.

| Method | Topic Acc ↑ | Topic F1 ↑ | $\Delta_u$ ↓ | Attack Acc ↓ | Attack F1 ↓ | $\Delta_p$ ↑ |
|---|---|---|---|---|---|---|
| Raw | 0.4771 | 0.1920 | - | 0.6689 | 0.6539 | - |
| Edit-All | 0.4424 | 0.1798 | 0.0122 | 0.6205 | 0.6204 | 0.0335 |
| High-Risk 10% | 0.4708 | 0.1897 | 0.0023 | 0.6362 | 0.6352 | 0.0187 |
| High-Risk 20% | 0.4683 | 0.1880 | 0.0040 | 0.6321 | 0.6312 | 0.0227 |
| High-Risk 30% | 0.4642 | 0.1861 | 0.0059 | 0.6283 | 0.6280 | 0.0259 |
| Low-Risk 10% | 0.4678 | 0.1840 | 0.0080 | 0.6639 | 0.6509 | 0.0030 |
| Low-Risk 20% | 0.4651 | 0.1831 | 0.0089 | 0.6588 | 0.6527 | 0.0012 |
| Low-Risk 30% | 0.4611 | 0.1804 | 0.0116 | 0.6492 | 0.6463 | 0.0076 |
| Random 10% | 0.4712 | 0.1890 | 0.0030 | 0.6655 | 0.6524 | 0.0015 |
| Random 20% | 0.4663 | 0.1832 | 0.0088 | 0.6535 | 0.6515 | 0.0024 |
| Random 30% | 0.4589 | 0.1760 | 0.0160 | 0.6483 | 0.6476 | 0.0063 |

## 5.4 Ablation and Sensitivity Analysis

### 5.4.1 Risk Scorer Ablation

We next examine whether the learned risk scorer identifies the token states that contribute most to privacy leakage. To isolate the effect of state selection, we decouple selection from semantic editing and compare High-Risk, Low-Risk, and Random policies under fixed editing budgets. Table 6 reports the results on the gender task.

The comparison first shows that indiscriminate editing is inefficient. Editing all token states reduces attacker macro-F1 from 0.6539 to 0.6204, but also lowers topic macro-F1 from 0.1920 to 0.1798. Under budget-constrained selection, High-Risk consistently delivers the strongest privacy reduction across all budgets. At 20%, for example, High-Risk reduces attacker macro-F1 to 0.6312, compared with 0.6527 for Low-Risk and 0.6515 for Random, while retaining competitive utility with a topic macro-F1 of 0.1880. The same pattern holds at 10% and 30%.

These results indicate that privacy cues are concentrated in a subset of transmitted latent states rather than being uniformly distributed across the representation. Editing those states is substantially more effective than editing low-risk or randomly selected states under the same budget. The ablation, therefore, supports the learned risk scorer's role as a meaningful localization mechanism for privacy latent states.

### 5.4.2 Token-level analysis of semantic preservation

To provide concrete evidence for semantic preservation, we examine the token states edited by SSSE on the gender task. For each edited token state $h_{i,j}^{(l)}$, we compute privacy and topic saliency using gradient-times-representation scores with respect to the privacy and the topic-task loss. This analysis asks whether SSSE edits token states that are more important for sensitive-attribute inference than for the public topic task.

**Table 7** Token-level diagnostic results on the gender task. Edited-token saliency is computed at the split hidden states using a gradient-times-representation score, and sample-level outcomes are measured on the test samples after editing. The Priv./Topic saliency ratio denotes the mean per-token ratio between privacy saliency and topic-task saliency over edited tokens, while Topic Saliency denotes the mean topic-task saliency of edited tokens. Topic Consistent Rate measures the proportion of samples for which the topic prediction remains unchanged after editing, and Defense Success Rate measures the proportion of samples where defense is successfully achieved.

| | Edited-Token Saliency | | Sample-Level Outcome | |
|---|---|---|---|---|
| Selection Policy | Priv./Topic Saliency Ratio ↑ | Topic Saliency ↓ | Topic Consistent Rate ↑ | Defense Success Rate ↑ |
| High-Risk | **9.04** | **0.0044** | **0.9795** | **0.1673** |
| Random | 6.62 | 0.0086 | 0.9625 | 0.1167 |
| Low-Risk | 3.59 | 0.0175 | 0.9199 | 0.0720 |

Table 7 reports the aggregate token-level diagnostics under three selection policies: high-risk, random, and low-risk selection. High-risk selection achieves the largest privacy-to-topic saliency ratio, 9.04, compared with 6.62 for random selection and 3.59 for low-risk selection. It also gives the lowest topic saliency among edited tokens, 0.0044. These results indicate that the risk scorer tends to select token states that carry stronger privacy evidence but contribute less to topic prediction. At the sample level, high-risk editing also achieves the highest topic-consistent rate and defense success rate. Thus, the selected tokens are not merely frequent or visually salient tokens; they are latent states where editing has higher privacy value and lower topic-task cost.

Table 8 further presents representative cases and their edited results. The selected tokens include pronouns, gender-associated expressions, pregnancy-related terms, and affective or stylistic cues, such as *she*, *pregnant*, *feminine*, and *beautiful*. The before/after anchors are the nearest lexical probes of the latent token representation. They are used to interpret the movement of the latent state after editing, while the original input sentence itself is not rewritten. After editing, the anchors move from strongly gender-associated or stylistically sensitive cues toward contextually related alternatives, such as broader pronoun anchors, child-related terms, political or stylistic anchors, and nearby affective concepts. This behavior indicates that SSSE performs local movement in latent space instead of direct textual replacement.



**Table 8** Representative high-risk token cases on the gender task. Anchor terms denote the nearest lexical probes of the token representation before and after SSSE editing. Δ Attack Conf. and Δ Topic Conf. denote confidence changes after SSSE editing. A negative Δ Attack Conf. indicates reduced attacker confidence, while a small Δ Topic Conf. indicates stable topic confidence.

| | | Representation Anchors | | Confidence Change | |
|---|---|---|---|---|---|
| Input Sentence | Token | Before SSSE | After SSSE | Δ Attack Conf. | Δ Topic Conf. |
| "My wife and I went to see her doctor ... She is pregnant ..." | she | she / he / her | it / its / them | −0.4110 | +0.0083 |
| "... Ms. Banks who claimed she was pregnant ..." | pregnant | pregnant / pregnancy / womb | infant / baby / kid | −0.6253 | +0.0020 |
| "... Are you feminine? politically aware? ..." | feminine | feminine / woman / feminist | politically / powerful / political | −0.5863 | +0.0046 |
| "... everything will just become a beautiful dream - a dream of love." | beautiful | beautiful / majestic / magnificent | dream / dreams / illusion | −0.5779 | +0.0086 |

**Table 9** Candidate construction ablation. Topic utility is reported by topic macro-F1 and $\Delta_u$, while privacy is reported by attack macro-F1 and $\Delta_p$.

| | Topic Utility | | Gender Privacy | | Age-Group Privacy | |
|---|---|---|---|---|---|---|
| Variant | Topic F1 ↑ | $\Delta_u$ ↓ | Attack F1 ↓ | $\Delta_p$ ↑ | Attack F1 ↓ | $\Delta_p$ ↑ |
| Raw | 0.1920 | - | 0.6539 | - | 0.5403 | - |
| SSSE | 0.1890 | 0.0030 | 0.6288 | 0.0251 | 0.5053 | 0.0350 |
| w/o Local (proto. only) | 0.1770 | 0.0150 | 0.6395 | 0.0144 | 0.5320 | 0.0083 |
| w/o Prototype (local only) | 0.1809 | 0.0111 | 0.6424 | 0.0115 | 0.5356 | 0.0047 |

The confidence changes in Table 8 provide case-level evidence for the same privacy–utility pattern observed in Table 7. Across the representative cases, SSSE consistently reduces attacker confidence, with $\Delta$ Attack Conf. ranging from $-0.4110$ to $-0.6253$, while the changes in topic confidence remain small. The aggregate diagnostics and representative token cases demonstrate that high-risk editing suppresses sensitive-attribute cues while preserving task-relevant semantic structure.

### 5.4.3 Candidate Construction Ablation

To evaluate the contribution of the candidate construction module, we ablate its two sources while keeping the learned risk scorer and the downstream editing rule unchanged. Specifically, we compare the full candidate bank with two reduced variants: *prototype only*, which removes the input-conditioned local reservoir, and *local only*, which removes the static prototype cache. Table 9 summarizes the resulting utility and privacy changes on both gender and age group inference.

The ablation shows that the two candidate sources are complementary rather than interchangeable. The full candidate bank achieves the strongest overall balance, with only a small utility drop relative to Raw transmission and the largest privacy gains on both tasks. Removing the local reservoir causes a clear utility decline, suggesting that input-conditioned candidates are important for preserving context-specific semantics during editing. Removing the prototype cache weakens privacy protection more noticeably, especially on the age-group task, indicating that the static cache provides useful semantic support beyond the current input.

Taken together, these results show that strong privacy-aware replacement requires both sources. The local reservoir contributes context-sensitive semantic coherence, while the prototype cache expands the available support when the current input alone does not provide enough suitable alternatives. The best privacy–utility trade-off is achieved only when these two components are combined in the full candidate bank.

### 5.4.4 Sensitivity to Neighborhood and Prototype

We further examine the sensitivity of the candidate construction module to two design choices: the number of retrieved neighborhood candidates used for aggregation and the size of the prototype bank stored in the static cache. In the neighborhood sweep, the prototype bank is fixed to 256, while in the prototype sweep, the neighborhood is fixed to 16. Table 10 summarizes the resulting utility, privacy, and latency trade-offs on the gender task. The sensitivity sweep is conducted under the same gender-task evaluation

**Table 10** Sensitivity to neighborhood and prototype on the gender task. In the neighborhood sweep, the prototype bank is fixed to 256. In the prototype sweep, the neighborhood is fixed to 16.

| Sweep | Setting | Topic F1 ↑ | Attack F1 ↓ | Latency (ms/sample) ↓ |
|---|---|---|---|---|
| Neighborhood | 8 | 0.1826 | 0.6406 | 6.66 |
| | 12 | 0.1800 | 0.6352 | 6.79 |
| | 16 | 0.1807 | **0.6274** | 6.86 |
| | 24 | 0.1793 | 0.6280 | 6.93 |
| | 32 | 0.1792 | 0.6289 | 7.00 |
| Prototype | 64 | 0.1794 | 0.6414 | 5.10 |
| | 128 | 0.1811 | 0.6391 | 5.71 |
| | 256 | 0.1808 | **0.6275** | 6.85 |
| | 512 | **0.1884** | 0.6420 | 11.20 |

setting but with independently trained sweep variants; therefore, the absolute values may differ slightly from the main benchmark.

The results indicate a clear saturation effect. Increasing the neighborhood size from 8 to 16 substantially improves privacy, but larger neighborhoods provide little additional benefit, slightly reduce utility and continue to increase latency. A similar pattern appears in the prototype sweep: increasing the prototype bank from 64 to 256 improves privacy while keeping utility relatively stable, but further expansion to 512 sharply increases latency and does not preserve the same privacy benefit.

These observations suggest that the candidate construction module is robust within a moderate hyperparameter range, whereas excessively large neighborhoods or prototype banks yield diminishing returns. We therefore use a neighborhood size of 16 and a prototype bank size of 256 as the default configuration, since this setting provides the strongest privacy protection in the sweeps while maintaining moderate latency.

## 5.5 Deployment Overhead

We evaluate the computational overhead of SSSE under a heterogeneous edge-cloud deployment setting. The edge-side computation is executed on CPU, including the front encoder and the privacy intervention module, while the server-side computation is executed on GPU, including the tail encoder and task head. Transmission latency is excluded from this measurement, so the reported results isolate the computational cost on both sides of split inference.

### 5.5.1 End-to-End Latency and Throughput

Table 11 reports the end-to-end latency and throughput on the gender task. Raw transmission provides the reference runtime of 3.57 ms/sample and a throughput of 280.5 samples/s. G-All introduces only a small increase in total latency, from 3.57 to 3.64 ms/sample, because it only performs element-wise noise injection on the transmitted states. Shredder-LN and SnD-LDP also remain relatively efficient at inference time, with total latencies of 3.86 and 3.96 ms/sample, respectively. This is expected because

**Table 11** End-to-end efficiency comparison on the gender task under a heterogeneous edge-cloud deployment setting. Edge-side computation is executed on CPU, and server-side computation is executed on GPU. Reported latency excludes transmission time. Budgeted selective baselines use the T20% setting.

| Method | Edge Cost (ms) | Cloud Cost (ms) | Total (ms) | Overhead | Samples/s |
|---|---|---|---|---|---|
| Raw | 1.79 | 1.71 | 3.57 | 1.00× | 280.5 |
| G-All | 1.87 | 1.72 | 3.64 | 1.02× | 274.6 |
| Shredder-LN | 2.04 | 1.78 | 3.86 | 1.09× | 259.1 |
| SnD-LDP | 2.16 | 1.75 | 3.96 | 1.12× | 252.4 |
| G-T20% | 2.09 | 1.70 | 3.84 | 1.08× | 260.5 |
| Mask-T20% | 2.11 | 1.69 | 3.84 | 1.08× | 260.6 |
| Replace-T20% | 2.44 | 1.70 | 4.19 | 1.18× | 238.6 |
| **SSSE** | 4.60 | 1.74 | 6.40 | 1.79× | 156.3 |

their main privacy transformations are applied as compact perturbation or denoising modules, prepared before deployment. The budgeted selective baselines increase total latency to between 3.84 and 4.19 ms/sample, mainly because they require token ranking or state replacement in addition to the standard split-inference pipeline.

SSSE increases the total latency to 6.40 ms/sample, corresponding to a 1.79× overhead relative to Raw transmission. This additional cost is mainly concentrated on the edge side. The edge-side cost increases from 1.79 ms/sample under Raw transmission to 4.60 ms/sample under SSSE, while the cloud-side cost remains close to the other methods, ranging from 1.69 to 1.78 ms/sample. This pattern is consistent with the design of SSSE: risk scoring, candidate construction, and semantic editing are all performed before offloading, whereas the server receives the edited representation in the same tensor format as the original split-layer states.

Although SSSE is more expensive than simple perturbation baselines, the overhead should be interpreted together with its privacy–utility behavior. Compared with G-All and the budgeted selective baselines, SSSE introduces additional edge-side computation to construct semantically compatible alternatives rather than directly injecting noise, masking states, or replacing states randomly. This extra computation is the price paid for avoiding unnecessary semantic distortion. As shown in the main benchmark and budget sweep, this design yields a stronger privacy–utility balance while keeping the server-side inference cost almost unchanged.

### 5.5.2 Component-wise Overhead

To identify the source of the additional runtime cost, we further profile the main stages of the SSSE intervention module under the same heterogeneous edge-cloud setting. Table 12 reports the per-sample latency of risk scoring, local reservoir construction, prototype scoring, candidate fusion, neighborhood selection, candidate aggregation, and final gate-based interpolation.

The component-wise results show that most SSSE operations are lightweight. Risk scoring requires only 0.05 ms/sample because the scorer is implemented as a compact token-level MLP. Prototype scoring, neighborhood selection, candidate aggregation, and gate-based interpolation also contribute only a small fraction of the runtime.

**Table 12** Component-wise latency breakdown of the SSSE intervention block on the gender task under the heterogeneous edge-cloud deployment setting. The last column reports the percentage contribution of each stage to the instrumented SSSE runtime.

| Stage | Latency (ms/sample) | Fraction of SSSE runtime (%) |
|---|---|---|
| Risk scoring | 0.05 | 1.8 |
| Local reservoir | 0.31 | 10.9 |
| Prototype scoring ($N$=256) | 0.04 | 1.4 |
| Candidate fusion | 2.14 | 75.1 |
| Neighborhood selection | 0.12 | 4.2 |
| Candidate aggregation | 0.13 | 4.6 |
| Gate + interpolation | 0.06 | 2.1 |
| **Instrumented SSSE sum** | **2.85** | **100.0** |

These results indicate that the learnable risk scorer and the final interpolation rule are not the main sources of deployment overhead.

Candidate fusion is the dominant cost of the intervention module. It requires 2.14 ms/sample and accounts for 75.1% of the instrumented SSSE runtime. This overhead arises because the module merges local and prototype candidates into a unified pool and evaluates compatibility-related costs between target states and candidate states. In particular, candidate-wise distance computation involves repeated tensor construction, broadcasting, and memory access over the candidate bank.

The result is useful for deployment, as this stage can be optimized without altering the primary privacy objective. Possible system-level optimizations include caching prototype norms, precomputing prototype-side quantities, reducing redundant tensor concatenation, and replacing exhaustive candidate scoring with approximate nearest-neighbor retrieval. Therefore, while the current implementation introduces a measurable edge-side overhead, the profiling results indicate a clear optimization path for improving runtime efficiency.

# 6 Conclusion

In this paper, we presented SSSE, an inference-time privacy defense for edge-cloud split inference in SLMs. SSSE protects offloaded intermediate representations by selectively editing semantic state at the split boundary. By combining a learnable risk scorer with on-device candidate construction from a local reservoir and a prototype cache, SSSE selectively edits high-risk latent states into privacy-aware semantic alternatives while preserving task-relevant information. Extensive experiments show that SSSE achieves a more favorable constrained privacy–utility trade-off than the compared baselines. Across both gender and age-group privacy tasks, it consistently reduces sensitive-attribute recoverability while maintaining topic classification performance, and the ablation results further verify the effectiveness of risk-aware state selection and candidate construction. The deployment study also shows that SSSE introduces a practical overhead under a realistic heterogeneous edge-cloud setting. In future work, the efficiency of candidate construction and fusion can be further optimized to improve deployment on more resource-constrained edge platforms.

# Declarations

## Availability of data

The empirical evaluations in this study are conducted on the publicly available Blog Authorship Corpus dataset.

## Competing interests

The authors declare that they have no competing interests.

## Funding

This work received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

## Authors' contributions

All authors contributed to the study conception, experimental design, data analysis, and manuscript preparation.

# References

[1] Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., Zhou, M.: MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. Advances in Neural Information Processing Systems **33**, 5776–5788 (2020)

[2] Gu, Y., Dong, L., Wei, F., Huang, M.: MiniLLM: Knowledge Distillation of Large Language Models (2023)

[3] Hui, B., Yang, J., Cui, Z., Yang, J., Liu, D., Zhang, L., Liu, T., Zhang, J., Yu, B., Lu, K., et al.: Qwen2.5-Coder Technical Report (2024)

[4] Cai, G., Tian, R., Yang, L., Jia, Y., Li, L., Wang, J.: Efficient inference for edge large language models: A survey. Tsinghua Science and Technology **31**(3), 1365–1380 (2026)

[5] Pan, Y., Su, Z., Wang, Y., Guo, S., Liu, H., Li, R., Wu, Y.: Cloud-edge collaborative large model services: Challenges and solutions. IEEE Network **39**(4), 182–191 (2025) https://doi.org/10.1109/MNET.2024.3442880

[6] Kakhi, K., Jagatheesaperumal, S.K., Khosravi, A., Alizadehsani, R., Acharya, U.R.: Fatigue monitoring using wearables and AI: Trends, challenges, and future opportunities. Computers in Biology and Medicine **195**, 110461 (2025) https://doi.org/10.1016/j.compbiomed.2025.110461

[7] Li, C., Gu, B., Zhao, Z., Qu, Y., Xin, G., Huo, J., Gao, L.: Federated transfer learning for on-device llms efficient fine tuning optimization. Big Data Mining and Analytics **8**(2), 430–446 (2025)

[8] Qi, L., Yan, B., Wang, W., Hu, C., Dai, F., Xu, X., Dou, W., Zhou, X.: St-bernt: A spatiotemporal $\lambda$-bernstein graph convolutional network with transformer for multi-site air quality prediction in distributed unmanned agent systems. IEEE Internet of Things Journal (2025)

[9] Jia, X., Gu, B., Chen, J., Gao, L., Pang, W., Lv, G., Qu, Y., Cui, L.: Dynamic batch processing with flexidecode scheduler for efficient llm inference in iiot. Big Data Mining and Analytics **8**(6), 1307–1323 (2025)

[10] Luo, C., Zhang, J., Guo, J., Hong, Y., Chen, Z., Gu, S.: Energy efficiency maximization in riss-assisted uavs-based edge computing network using deep reinforcement learning. Big Data Mining and Analytics **7**(4), 1065–1083 (2024)

[11] Li, S., Liu, G., Li, L., Zhang, Z., Fei, W., Xiang, H.: A review on air-ground coordination in mobile edge computing: Key technologies, applications and future directions. Tsinghua Science and Technology **30**(3), 1359–1386 (2024)

[12] Xiao, Y., Xiao, L., Wan, K., Yang, H., Zhang, Y., Wu, Y., Zhang, Y.: Reinforcement learning based energy-efficient collaborative inference for mobile edge computing. IEEE Transactions on Communications **71**(2), 864–876 (2022) https://doi.org/10.1109/TCOMM.2022.3229033

[13] Ren, W.-Q., Qu, Y.-B., Dong, C., Jing, Y.-Q., Sun, H., Wu, Q.-H., Guo, S.: A survey on collaborative DNN inference for edge intelligence. Machine Intelligence Research **20**(3), 370–395 (2023) https://doi.org/10.1007/s11633-022-1391-7

[14] Xu, X., Hu, Y., Cui, G., Qi, L., Dou, W., Cai, Z.: Cadec: a combinatorial auction for dynamic distributed dnn inference scheduling in edge-cloud networks. IEEE transactions on Mobile computing (2025)

[15] Zhan, S., Huang, L., Luo, G., Zheng, S., Gao, Z., Chao, H.-C.: A review on federated learning architectures for privacy-preserving AI: Lightweight and secure cloud–edge–end collaboration. Electronics **14**(13), 2512 (2025) https://doi.org/10.3390/electronics14132512

[16] Alzu'Bi, A., Alomar, A., Alkhaza'Leh, S., Abuarqoub, A., Hammoudeh, M.: A review of privacy and security of edge computing in smart healthcare systems: issues, challenges, and research directions. Tsinghua Science and Technology **29**(4), 1152–1180 (2024)

[17] Al-Yarimi, F.A., Salah, R., Mohamoud, K.: Blockchain-driven secure data sharing framework for edge computing networks. Tsinghua Science and Technology **30**(3), 978–997 (2024)

[18] Zhao, Y., Liu, B., Zhu, T., Ding, M., Zhou, W.: Private-encoder: Enforcing privacy in latent space for human face images. Concurrency and Computation: Practice and Experience **34**(3), 6548 (2022) https://doi.org/10.1002/cpe.6548

[19] Shan, B., Yuan, X., Ni, W., Wang, X., Liu, R.P., Dutkiewicz, E.: Preserving the privacy of latent information for graph-structured data. IEEE Transactions on Information Forensics and Security **18**, 5041–5055 (2023) https://doi.org/10.1109/TIFS.2023.3302508

[20] Bailey, L., Serrano, A., Sheshadri, A., Seleznyov, M., Taylor, J., Jenner, E., Hilton, J., Casper, S., Guestrin, C., Emmons, S.: Obfuscated Activations Bypass LLM Latent-Space Defenses (2024)

[21] Li, H., Xu, M., Song, Y.: Sentence embedding leaks more information than you expect: Generative embedding inversion attack to recover the whole sentence. In: Findings of the Association for Computational Linguistics: ACL 2023, pp. 14022–14040 (2023)

[22] Jin, S., Pang, X., Wang, Z., Wang, H., Du, J., Hu, J., Ren, K.: Safeguarding LLM Embeddings in End-Cloud Collaboration via Entropy-Driven Perturbation (2025)

[23] Harel, R., Gilboa, N., Pinter, Y.: Token-Level Privacy in Large Language Models (2025)

[24] Harel, R., Elboher, Y., Pinter, Y.: Protecting privacy in classifiers by token manipulation. In: Proceedings of the Fifth Workshop on Privacy in Natural Language Processing, pp. 29–38. Association for Computational Linguistics, Bangkok, Thailand (2024). https://aclanthology.org/2024.privatenlp-1.4/

[25] Shi, W., Cui, A., Li, E., Jia, R., Yu, Z.: Selective differential privacy for language modeling. In: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 2848–2859. Association for Computational Linguistics, Seattle, United States (2022). https://doi.org/10.18653/v1/2022.naacl-main.205

[26] Zhang, K., Tsai, P.-W., Tian, J., Zhao, W., Cai, X., Gao, L., Chen, J.: Towards privacy in decentralized iot: a blockchain-based dual response dp mechanism. Big Data Mining and Analytics **7**(3), 699–717 (2024)

[27] Vepakomma, P., Singh, A., Gupta, O., Raskar, R.: NoPeek: Information leakage reduction to share activations in distributed deep learning. In: 2020 International Conference on Data Mining Workshops (ICDMW), pp. 933–942 (2020). https://doi.org/10.1109/ICDMW51313.2020.00134

[28] Mireshghallah, F., Taram, M., Ramrakhyani, P., Jalali, A., Tullsen, D.M., Esmaeilzadeh, H.: Shredder: Learning noise distributions to protect inference privacy. In: Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 3–18 (2020). https://doi.org/10.1145/3373376.3378522

[29] Mai, P., Yan, R., Huang, Z., Yang, Y., Pang, Y.: Split-and-denoise: Protect large language model inference with local differential privacy. In: Proceedings of the 41st International Conference on Machine Learning (2024)

[30] Wan, Z., Cheng, A., Wang, Y., Wang, L.: Information Leakage from Embedding in Large Language Models (2024)

[31] Dong, T., Meng, Y., Li, S., Chen, G., Liu, Z., Zhu, H.: Depth gives a false sense of privacy: LLM internal states inversion. In: 34th USENIX Security Symposium (USENIX Security 25), pp. 1629–1648 (2025)

[32] Erdoğan, E., Küpçü, A., Çiçek, A.E.: UnSplit: Data-Oblivious Model Inversion, Model Stealing, and Label Inference Attacks Against Split Learning (2021)

[33] Zhu, X., Luo, X., Wu, Y., Jiang, Y., Xiao, X., Ooi, B.C.: Passive Inference Attacks on Split Learning via Adversarial Regularization (2023)

[34] Luo, X., Yu, T., Xiao, X.: Prompt inference attack on distributed large language model inference frameworks. In: Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security, pp. 1739–1753 (2025). https://doi.org/10.1145/3719027.3744820

[35] Kan, Z., Qiao, L., Yu, H., Peng, L., Gao, Y., Li, D.: Protecting user privacy in remote conversational systems: A privacy-preserving framework based on text sanitization. arXiv preprint arXiv:2306.08223 (2023)

[36] Xu, Q., Qu, L., Xu, C., Cui, R.: Privacy-aware text rewriting. In: Proceedings of the 12th International Conference on Natural Language Generation, pp. 247–257 (2019)

[37] Wu, X., Gong, L., Xiong, D.: Adaptive differential privacy for language model training. In: Proceedings of the First Workshop on Federated Learning for Natural Language Processing (FL4NLP 2022), pp. 21–26 (2022)

[38] Chen, Y., Lent, H., Bjerva, J.: Text embedding inversion security for multilingual language models. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 7808–7827. Association for Computational Linguistics, Bangkok, Thailand (2024)

[39] Gu, R., Wang, S., Dai, H., Chen, X., Wang, Z., Bao, W., Zheng, J., Tu, Y., Huang, Y., Qi, L., *et al.*: Fluid-shuttle: efficient cloud data transmission based on serverless computing compression. IEEE/ACM Transactions on Networking **32**(6), 4554–4569 (2024)

[40] Schler, J., Koppel, M., Argamon, S., Pennebaker, J.W.: Effects of age and gender on blogging. In: Computational Approaches to Analyzing Weblogs: Papers from the 2006 AAAI Spring Symposium, pp. 199–205 (2006)