# DualGuard: A Parameter Space Transformation Approach for Bidirectional Defense in Split-Based LLM Fine-Tuning

**Zihan Liu[1], Yizhen Wang[1], Rui Wang[1,2*], Sai Wu[1,3]**

[1]Zhejiang University,
[2]Hangzhou High-Tech Zone (Binjiang) Institute of Blockchain and Data Security,
[3]Zhejiang Key Laboratory of Big Data Intelligent Computing

***Correspondence:** rwang21@zju.edu.cn

## Abstract

Integrating split learning with large language model fine-tuning (LLM-FT) enables secure collaboration between a trusted local client and a well-equipped remote server, but it is vulnerable to data reconstruction attacks (DRAs) that exploit transmitted activations and gradients. Current defense methods, like adding noise to activations or gradients, often sacrifice task-specific model performance under strict privacy constraints. This paper introduces DualGuard, a bidirectional defense mechanism against DRAs for split-based LLM-FT. DualGuard proposes a local warm-up parameter space transformation to alter client-side model parameters before training, using multi-task learning to strike a balance between privacy protection and model performance. Additionally, a global fine-tuning parameter space retention strategy prevents the model from reverting to vulnerable states during formal fine-tuning. Experiments show that DualGuard outperforms current defense methods against various DRAs, while maintaining task performance. Our code will be made publicly available.

## 1 Introduction

Large language model fine-tuning (LLM-FT) customizes a pre-trained model (Bommasani et al., 2021; Brown et al., 2020; Touvron et al., 2023) for a specific task by training it further on new data (Gururangan et al., 2020; Lee et al., 2020; Radford, 2018). Combining LLM-FT with split learning (SL) (Gupta and Raskar, 2018; Vepakomma et al., 2018), which splits the model into client-resident head and tail layers and server-resident trunk layers, allows a trusted local client to co-train with a powerful remote server. The client processes the raw data, generates intermediate smashed data, and sends them to the server for further training. By only transmitting intermediate data, data privacy is maintained while the language model is efficiently fine-tuned (Chen et al., 2024b; Wu et al., 2023).

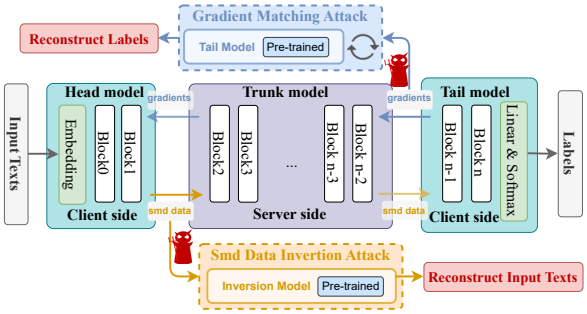



Figure 1: Privacy risk in split-based LLM fine-tuning, where attackers may reconstruct the original data via transmitted intermediate activations or gradients.

While split-based LLM-FT (Cao et al., 2024; Lin et al., 2024; Shen et al., 2023) enhances data privacy by avoiding transmitting raw data, privacy leakage risk remains through transmitted intermediate data (Pasquini et al., 2021; Li et al., 2024). Malicious attackers can reconstruct the original data using data reconstruction attacks (DRAs), as shown in Figure 1. Forward smashed data inversion (He et al., 2019; Zhang et al., 2023; Chen et al., 2024a) can reconstruct input data from intermediate activations. Backward gradient matching attacks (Zhu et al., 2019; Deng et al., 2021; Balunovic et al., 2022) can generate dummy gradients to reconstruct output labels and deduce the original input.

To mitigate DRAs, additional defense measures are needed (Shen et al., 2023; Thapa et al., 2022; Nguyen et al., 2023). Perturbation-based techniques, especially leveraging differential privacy (DP) (Dwork et al., 2006), are commonly used to modify sensitive attributes and enhance privacy protection. For forward smashed data inversion attacks, DP noise is added to either the original data output (Hoory et al., 2021) or intermediate activations (Du et al., 2023b,a) to prevent reconstructing the original data. To address backward gradient matching attacks, introducing noise into gradients can help thwart attempts to reconstruct label information (Abadi et al., 2016).



*Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 17065–17080

However, these defense methods face limitations in split-based LLM-FT scenarios. Firstly, there is a trade-off between privacy protection and task performance. Perturbation-based methods often rely on high noise levels to enhance data security, potentially impacting task performance (Farrand et al., 2020). Secondly, the inherent "auto-regressive" nature of LLMs creates a significant overlap between original inputs and output labels, rendering them susceptible to both forward and backward attacks (Chen et al., 2024a). However, current defense mechanisms primarily target one type of attack, leaving vulnerabilities against the other. Lastly, the "not-too-far" property of LLM-FT, where fine-tuned model parameters remain close to the pre-trained model, allows attackers to acquire extensive prior knowledge. This poses a significant vulnerability that current methods struggle to fully mitigate (Chen et al., 2024a). For a detailed discussion and experimental validation of these limitations, please refer to Appendix E.

To tackle the limitations and bolster defense against DRAs while maintaining downstream task performance, this paper presents DualGuard, a bidirectional defense mechanism tailored for split-based LLM-FT scenarios. DualGuard underscores the importance of prior knowledge in pre-trained models for DRAs and introduces a novel *parameter space transformation* approach. This approach aims to increase the dissimilarity between the client model and the pre-trained model, preventing attackers from exploiting the pre-trained model's prior knowledge. DualGuard provides defense against both forward and backward attacks while maintaining downstream task performance. Our key contributions include:

- We first propose a *local warm-up parameter space transformation strategy* to locally adjust the client-side model parameters to a different parameter space before formal training. This is achieved by connecting the client-resident head and tail models through a projection layer and using multi-task learning to balance privacy protection and task performance.
- During formal fine-tuning, the tail model may revert to the original pre-trained parameter space, leaving it vulnerable to backward gradient-matching attacks. Hence, we further introduce a *global fine-tuning parameter space retention strategy* to locally preserve the modified parameter space of the tail model.
- We implement DualGuard and conduct extensive experiments to demonstrate its effectiveness in defending against various DRAs. Our results show a significant decrease in the attack success rate, with the average RougeL-F score decreasing from 0.752 to 0.167. Importantly, we maintain robust performance on downstream tasks, effectively balancing privacy protection and task performance.

## 2 Background and Related Works

### 2.1 Split-based LLM Fine-Tuning

In resource-constrained environments like small enterprises and research institutions, data owners face challenges in fine-tuning LLMs due to limited computational resources and privacy concerns. Split Learning (Gupta and Raskar, 2018; Vepakomma et al., 2018; Chen et al., 2024a; Han et al., 2021; Wang et al., 2023; Shen et al., 2023) addresses these issues by splitting the model into client-resident and server-resident components, reducing the client's computational load and ensuring privacy by transmitting smashed data instead of raw data. The U-shaped split learning (USL) (Gupta and Raskar, 2018; Lyu et al., 2023) further enhances privacy protection by splitting the model into head, trunk, and tail layers, with the trusted client managing head and tail layers and server handling trunk layers, as shown in Figure 1. During forward computation, the client processes raw data, generates smashed data, and sends it to the server for trunk layer training. The client then trains the tail layers to infer labels. In backward propagation, gradients flow from the tail through the trunk layers to the head layers. This process enables efficient training with a resource-constrained client, all while avoiding the transmission of the raw data.

### 2.2 Data Reconstruction Attacks Vulnerabilities

In split-based LLM-FT, while raw data is not directly transmitted, vulnerabilities persist through transmitted smash data. Data reconstruction attackers (DRAs) on the server typically rely on four types of information to reconstruct the raw data: (1) white-box access to the server-side model (including model structure and all fine-tuned parameters), (2) semi-white-box access to the client-side model (comprising model structure and only pre-trained parameters), (3) public auxiliary datasets with similar data distribution, and (4) transmitted smash data and gradients during fine-tuning.

Several common DRA methods can be applied in split-based LLM-FT scenarios, where the client exchanges smashed data and gradients during forward and backward propagation, respectively. Forward smashed data inversion attacks (He et al., 2019; Zhang et al., 2023; Chen et al., 2024a) involve reconstructing original inputs from smashed data using an inversion attack model trained with an auxiliary dataset and the pre-trained head model. Backward gradient matching attacks (Zhu et al., 2019; Deng et al., 2021; Balunovic et al., 2022; Li et al., 2024) aim to reconstruct output labels, which can deduce the original input in LLM. These attacks utilize the trunk model's output activations and a pre-trained tail model to adjust a dummy gradient that matches real gradients, and reconstruct a dummy label that closely resembles the real output label. Besides, advanced BiSR (Chen et al., 2024a) achieves high-quality bidirectional reconstructions by combining forward smashed data inversion with backward gradient matching, posing a significant privacy risk in split-based LLM-FT scenarios.

### 2.3 Defense Methods against DRA

To thwart attackers attempting to reconstruct original data from forward smashed data and backward gradients, perturbation-based approaches are commonly used (Feyisetan et al., 2020; Chatzikokolakis et al., 2013; Shen et al., 2023; Du et al., 2023b,a; Wang et al., 2024; Mai et al., 2023). These methods obscure sensitive information in private data with differential privacy (DP) (Dwork et al., 2006) noise during forward or backward propagation. In forward propagation, noise is added to intermediate layers, such as embedding vectors (Chatzikokolakis et al., 2013) or activation values (Du et al., 2023b), to prevent attackers from inferring original inputs from perturbed smashed data. For backward propagation, DP noise is introduced into gradients (Abadi et al., 2016) to ensure that the gradients utilized in model updates do not reveal private output labels. However, these methods can only defend against either forward or backward attacks individually, not simultaneously for both.

Additionally, techniques like homomorphic encryption (Lu et al., 2023; Zimerman et al., 2023; Chen et al., 2022) and multi-party secure computation (Dong et al., 2023; Akimoto et al., 2023) offer effective data security guarantees. Nonetheless, their significant computational and communication overhead currently renders them impractical for use in the training process of LLM fine-tuning.

## 3 The Proposed DualGuard Approach

In this section, we introduce DualGuard, a bidirectional defense mechanism specifically designed for split-based LLM-FT scenarios. We first overview the main idea behind DualGuard and then explore its key design components, i.e., local warm-up parameter space transformation and global fine-tuned parameter space retention.

### 3.1 Overview

DualGuard aims to prevent attackers from exploiting prior knowledge in split-based LLM-FT, defend against both forward and backward attacks, and maintain task performance. To achieve these objectives, DualGuard introduces a novel *parameter space transformation paradigm*. Unlike perturbation-based approaches that involve adding DP noise to smashed data or gradients, DualGuard emphasizes the importance of safeguarding prior knowledge within the client-resident head and tail models, and strategically increases the disparity between the private client model and the public pre-trained model, thereby effectively defending against various data reconstruction attacks (DRAs).

DualGuard comprises two main components: local warm-up parameter space transformation and global fine-tuned parameter space retention, with the design flow illustrated in Figure 2. In the local warm-up phase, before formal fine-tuning, the head and tail models are transformed into a secure parameter space using a projection layer, which thwarts attackers from exploiting prior knowledge. During fine-tuning, the global retention strategy ensures that the client's tail model does not revert to its pre-trained state, providing protection against gradient-based attacks while preserving task performance. Next, we delve into the design details.

### 3.2 Local Warm-up Parameter Space Transformation

To address attacks that exploit prior knowledge from pre-trained models, we propose a *local warm-up parameter space transformation strategy* to decrease the relevance between server-side knowledge and client-side models. This warm-up phase is conducted locally on the client side to prevent data exposure to the server. This strategy involves a projection-based nonlinear transformation that links the head and tail models, transforming their parameters into a new space through a projection layer. This process typically employs a three-layer MLP, chosen for its lightweight architecture and

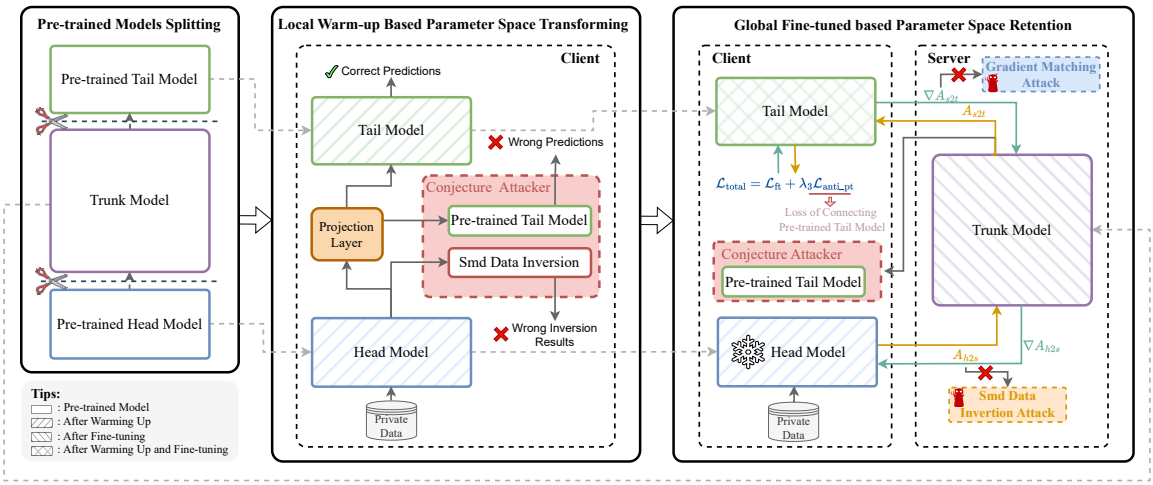


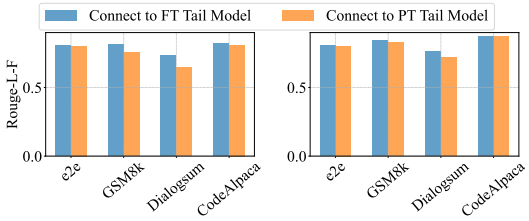

Figure 2: DualGuard utilizes a combination of local warm-up parameter space transformation and global fine-tuning parameter space retention to safeguard client model parameters, effectively balancing privacy protection and task performance. The warm-up phase takes place exclusively on the client side, mitigating data leakage risks with low computational overhead. In the global fine-tuning phase, the head model remains static to extract features, while the tail model undergoes further fine-tuning to adapt to downstream tasks within a secure parameter space.

ability to effectively transform the parameter space. It detaches the model's features from their original pre-trained forms, hindering attackers from reconstructing the model using pre-trained parameters.

To protect privacy while maintaining downstream task performance, we adopt a *multi-task learning method* during the local warm-up phase. This warm-up phase is designed around three main goals: (1) preventing reconstruction from smashed data, (2) deactivating the applicability of the pre-trained tail model, and (3) maintaining downstream task performance. By optimizing these objectives, we can strike a balance between safeguarding privacy and preserving task effectiveness.

### 3.2.1 Prevent reconstruction from smashed data

The risk of smashed data reconstruction arises from semi-white-box inversion models like SIP, which leverage the semantic properties of smashed data to reconstruct original inputs. These models, trained on auxiliary datasets and pre-trained models, can still utilize inherent semantic relationships in smashed data, allowing strong inversion even with real privacy data and fine-tuned models. To counter this, we disrupt these relationships by transforming the head model's parameter space. By inputting smashed data into the pre-trained SIP model to calculate inversion loss ($\mathcal{L}_{\text{inv}}$) and optimizing the head model parameters to maximize this loss, we increase the difficulty for attackers to reconstruct the original input, ensuring better protection against semi-white-box inversion attacks.The anti-inversion loss is formally defined as:




(a) GPT2-large (b) Llama3.2-1b

Figure 3: Comparison of RougeL-F scores between pre-trained and fine-tuned tail models.

$$\mathcal{L}_{\text{anti\_inv}} = \frac{1}{\mathcal{L}_{\text{inv}}} \tag{1}$$

### 3.2.2 Deactivating Pre-trained Model Applicability

In split-based LLM-FT, the 'not-too-far' property allows the server to feed its output into the pre-trained tail model, which can infer label information, potentially leaking private data. This attack, called the Connect to Pre-trained Tail Model Attack (CPTA), is illustrated in Figure 3, which compares RougeL-F scores between original labels and predictions from the pre-trained versus fine-tuned tail model, showing the pre-trained model can effectively reconstruct the original labels.

To mitigate this, we propose to transform the parameters of the client tail model to a different space than the pre-trained tail model to reduce its vulnerability to reconstruction attacks. By passing the head model's output through a projection layer into the pre-trained tail model, we calculate a reconstruction loss and optimize it to prevent the model from generating accurate labels. This reduces the applicability of the tail model for label

reconstruction, enhancing privacy protection.The method is formalized through the anti-pre-trained tail model loss, defined as:

$$\mathcal{L}_{\text{anti\_local\_pt}} = \frac{1}{\mathcal{L}_{\text{local\_pt}}} \tag{2}$$

#### 3.2.3 Maintain downstream task performance

By minimizing the two losses, we prevent attackers from reconstructing input data and labels from activations or gradients. At the same time, we ensure that both the head and tail models remain functional for subsequent tasks, with the downstream task loss defined as the cross-entropy loss.

$$\mathcal{L}_{\text{local\_ft}} = -\frac{1}{N}\sum_{t=1}^{N} \log P_\theta(x_t|x_{<t}) \tag{3}$$

where $N$ denotes the total length of the input sequence, $x_t$ represents the $t$-th token in the sequence, $x_{<t}$ denotes all tokens before the $t$-th token, and $P_\theta(x_t|x_{<t})$ is the probability of predicting $x_t$ based on $x_{<t}$.

#### 3.2.4 Loss Function Design

Based on the three objectives described above, we define the loss function for the warm-up phase as follows:

$$\mathcal{L}_{\text{warm\_up}} = \mathcal{L}_{\text{local\_ft}} + \lambda_1 \mathcal{L}_{\text{anti\_inv}} + \lambda_2 \mathcal{L}_{\text{anti\_local\_pt}} \tag{4}$$

Here, $\mathcal{L}_{\text{local\_ft}}$ maintains the performance of downstream tasks, $\mathcal{L}_{\text{anti\_inv}}$ increases the difficulty for attackers to reconstruct input data from forward-propagated activations, and $\mathcal{L}_{\text{anti\_local\_pt}}$ reduces the applicability of the pre-trained tail model in the attacker's model. $\lambda_1$ and $\lambda_2$ are hyper parameters that balance the weights of different objectives within the loss function.

### 3.3 Global Fine-tuned Parameter Space Retention

After the local warm-up parameter space transformation, the client head and tail models are moved to a secure space. The head model is frozen to ensure safe forward activations, while the trunk and tail models continue fine-tuning for tasks.

**Issue of tail model reversion to unsafe states.** However, during fine-tuning, the client tail model may revert to an unsafe state as the trunk model, with its extensive layers, dominates the process. This reversion enables gradient matching attacks and CPTA, risking private data reconstruction (as shown in subsection 4.4).

**Parameter space retention.** To prevent this, we introduce a global fine-tuned parameter space retention strategy, adding an anti-pre-trained tail model loss term, $\mathcal{L}_{\text{anti\_global\_pt}}$, to keep the tail model in the secure parameter space and prevent it from reverting to the pre-trained model's unsafe state. Formally, the $\mathcal{L}_{\text{anti\_global\_pt}}$ and the complete loss function Loss in formal split-based fine-tuning can be expressed as follows:

$$\mathcal{L}_{\text{anti\_global\_pt}} = \frac{1}{L_{\text{global\_pt}}} \tag{5}$$

$$\mathcal{L}_{\text{global}} = \mathcal{L}_{\text{global\_ft}} + \lambda_3 \mathcal{L}_{\text{anti\_global\_pt}} \tag{6}$$

where $L_{\text{global\_pt}}$ denotes the accuracy of the pre-trained model in reconstructing private data, $L_{\text{global\_ft}}$ represents the client-side loss on downstream tasks, and $\lambda_3$ is the weight factor balancing privacy protection and task performance.

During this stage, the client head model remains frozen, effectively extracting features from the input data. Moreover, the client tail model is maintained within a safe parameter space throughout training and is continuously fine-tuned to adapt to downstream tasks.

## 4 Experiment

### 4.1 Experimental Setup

**Models and Datasets.** We choose three popular large language models, i.e., GPT2-Large (Radford et al., 2019), Llama3.2-1b (Dubey et al., 2024), and Qwen2-1.5b (Yang et al., 2024). We perform experiments on five different datasets. Each dataset corresponds to a different natural language generation task, including structured data text generation (e2e (Novikova et al., 2017)), code generation (CodeAlpaca-20k (Chaudhary, 2023)), mathematical reasoning (GSM8k (Cobbe et al., 2021)), and dialogue summary generation (Dialogsum (Chen et al., 2021)). In addition, the Wikitext dataset (Merity et al., 2016) was used as an auxiliary dataset for the attackers in the experiments.

**Tested Attack Methods.** The considered data reconstruction attacks (DRAs) methods include the forward smashed data inversion attack SIP (Chen et al., 2024a), backward gradient matching attacks TAG (Deng et al., 2021) and LAMP (Balunovic et al., 2022), bidirectional data augmentation attack BiSR (Chen et al., 2024a), and the directly-connecting attack CPTA (refer to §3.2).

**Baseline and Defense Methods.** We compare the performance of DualGuard's mechanism with a

| Attack Method | Defense Method | GPT2-large | | | | Llama3.2-1b | | | | Qwen2-1.5b | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | GSM8k | Dialogsum | CodeAl | e2e | GSM8k | Dialogsum | CodeAl | e2e | GSM8k | Dialogsum | CodeAl | e2e |
| SIP | w/o defense | 87.63 | 89.78 | 77.92 | 89.33 | 84.32 | 91.08 | 68.04 | 85.59 | 84.20 | 88.02 | 64.49 | 85.75 |
| | DP-forward | 36.37 | 27.90 | 36.39 | 38.92 | **0.39** | 14.75 | 0.24 | 0.12 | 12.94 | 14.75 | 4.93 | 12.85 |
| | $d_\chi$P | 44.48 | 41.84 | 43.59 | 60.30 | 61.23 | 68.97 | 53.19 | 69.22 | 18.76 | 13.38 | 15.94 | 22.47 |
| | DP-SGD* | 87.37 | 90.31 | 77.95 | 88.30 | 86.24 | 91.43 | 70.49 | 87.20 | 84.57 | 87.52 | 67.80 | 85.82 |
| | DualGuard | **0.00** | **0.88** | **0.00** | **0.00** | 0.49 | **0.18** | **0.05** | **0.00** | **9.04** | **2.02** | **6.47** | **0.30** |
| TAG* | w/o defense | 81.05 | 80.93 | 77.91 | 84.91 | 94.91 | 93.86 | 72.14 | 95.78 | 81.90 | 93.81 | 77.70 | 93.35 |
| | DP-forward | 86.84 | 50.20 | 79.86 | 80.44 | 83.03 | 89.69 | 88.85 | 71.94 | 80.70 | 89.00 | 83.10 | 91.44 |
| | $d_\chi$P | 87.30 | 73.40 | 78.62 | 71.24 | 78.36 | 69.13 | 69.54 | 92.96 | 67.99 | 67.40 | 66.86 | 80.08 |
| | DP-SGD* | **0.00** | **0.24** | **0.04** | **0.00** | **1.21** | **0.94** | **1.40** | **0.40** | **1.23** | **1.27** | **0.74** | **0.00** |
| | DualGuard | 20.96 | 18.94 | 8.24 | 15.28 | 21.77 | 24.02 | 10.34 | 3.61 | 23.55 | 5.25 | 4.91 | 13.05 |
| LAMP* | w/o defense | 79.50 | 80.35 | 78.93 | 85.31 | 94.78 | 96.02 | 72.72 | 96.02 | 83.96 | 93.13 | 81.13 | 92.77 |
| | DP-forward | 85.51 | 47.30 | 79.67 | 81.81 | 84.07 | 89.73 | 89.25 | 71.98 | 81.03 | 87.30 | 85.46 | 92.25 |
| | $d_\chi$P | 87.73 | 67.75 | 78.31 | 67.02 | 79.94 | 66.75 | 67.28 | 95.21 | 68.02 | 66.85 | 66.86 | 81.08 |
| | DP-SGD* | **0.35** | **0.09** | **0.14** | **0.00** | **1.31** | **1.13** | **1.84** | **0.52** | **1.47** | **2.14** | **0.92** | **0.44** |
| | DualGuard | 25.35 | 22.66 | 10.63 | 19.14 | 22.32 | 33.11 | 11.28 | 4.72 | 26.31 | 5.62 | 5.78 | 12.28 |
| BiSR | w/o defense | 79.54 | 78.05 | 71.32 | 82.88 | 84.22 | 83.45 | 56.31 | 80.03 | 82.28 | 86.29 | 76.80 | 86.38 |
| | DP-forward | 79.02 | 44.86 | 62.98 | 77.30 | 35.08 | 29.42 | 28.78 | 22.60 | 64.01 | 74.12 | 62.98 | 91.04 |
| | $d_\chi$P | 57.56 | 40.16 | 52.50 | 57.16 | 66.95 | 56.23 | 56.28 | 81.66 | 40.89 | 34.67 | 58.53 | 68.91 |
| | DP-SGD* | 12.20 | 7.96 | 11.22 | 6.65 | 7.25 | 6.44 | 6.12 | 1.48 | 7.92 | 5.47 | 5.88 | 2.31 |
| | DualGuard | **1.58** | **6.77** | **5.63** | **1.89** | **2.46** | **2.25** | **3.70** | **0.00** | **5.35** | **3.68** | **4.09** | **1.56** |
| CPTA | w/o defense | 75.52 | 64.80 | 80.98 | 80.19 | 82.56 | 71.58 | 86.96 | 80.23 | 85.88 | 71.92 | 89.21 | 80.28 |
| | DP-forward | 78.48 | 63.66 | 69.23 | 78.62 | 60.51 | 51.23 | 47.29 | 59.59 | 73.02 | 55.93 | 65.17 | 77.63 |
| | $d_\chi$P | 66.43 | 55.35 | 68.95 | 73.78 | 73.81 | 63.94 | 73.27 | 71.88 | 43.87 | 34.78 | 33.23 | 44.28 |
| | DP-SGD* | 71.77 | 53.60 | 69.55 | 48.73 | 76.55 | 58.79 | 78.83 | 51.86 | 81.46 | 61.36 | 80.86 | 65.69 |
| | DualGuard | **0.00** | **0.00** | **0.98** | **0.00** | **0.38** | **0.45** | **0.71** | **0.01** | **5.20** | **5.67** | **0.76** | **0.55** |
| **Optimal Attack** | w/o defense | 87.63 | 89.78 | 80.98 | 89.33 | 94.91 | 96.02 | 86.96 | 96.02 | 85.88 | 93.81 | 89.21 | 93.35 |
| | DP-forward | 86.84 | 63.66 | 79.86 | 81.81 | 84.07 | 89.73 | 89.25 | 71.98 | 81.03 | 89.00 | 85.46 | 92.25 |
| | $d_\chi$P | 87.73 | 73.40 | 78.62 | 73.78 | 79.94 | 69.13 | 73.27 | 95.21 | 68.02 | 67.40 | 66.86 | 81.08 |
| | DP-SGD* | 87.37 | 90.31 | 77.95 | 88.30 | 86.24 | 91.43 | 78.83 | 87.20 | 84.57 | 87.52 | 80.86 | 85.82 |
| | DualGuard | **25.35** | **22.66** | **10.63** | **19.14** | **22.32** | **33.11** | **11.28** | **4.72** | **26.31** | **5.62** | **6.47** | **13.05** |

Table 1: Defense performance, measured in ROUGE-L F1 Score % (↓), of various defense methods against five attackers. The optimal attack among the five attackers against a particular defense method, are listed in the final row and highlighted with a gray background, represents the comprehensive defensive capability of each defense method. To counter gradient matching attacks, we adapt the original DP-SGD method, which clips and adds noise to model weight gradients, to be applied to the server's output activation gradients. TAG and LAMP, designed for white-box settings, are modified for semi-white-box SL by replacing white-box model sections with pre-trained weights. Results for these variants are marked with an asterisk *.

baseline no-defense approach and three advanced defense mechanisms: two designed to defend against forward attacks (DP-forward (Du et al., 2023b), d$\chi$P (Chatzikokolakis et al., 2013)) and one tailored for defending against backward attacks (DP-SGD (Abadi et al., 2016)). More setup details and evaluations on split segment analysis, defense overhead, and convergence performance can be found in Appendices A, B, C, and D, respectively.

### 4.2 Defense Performance against DRAs

In this subsection, we test the defensive capabilities of DualGuard against the five attack methods, compared with existing defense methods. We measure their effectiveness using text similarity scores RougeL-F and Meteor, with lower values indicating better defense performance. The results measured by RougeL-F are shown in Table 1, with the results measured by Meteor shown in Appendix F. We also present a real example from the GSM8k dataset showcasing attack outcomes under various defense methods in Appendix G. Next, we discuss defense performance against different attackers.

**Defense against forward smashed data inversion.** We chose to use Wikitext as an auxiliary dataset to train a GRU model as a smashed data inversion model (ref to (Chen et al., 2024a)) to perform the SIP attack because of its rigorous linguistic structure, high-quality content, and coverage of different topics and domains. The SIP row in Table 1 shows that without any defense mechanism, the SIP attack can reconstruct most of the original inputs, as seen in RougeL-F scores exceeding 85% across models and datasets. Forward defense mechanisms like DP-forward and d$\chi$P can reduce privacy risks. However, backward defenses like DP-SGD* fail to prevent original data reconstruction. In contrast, DualGuard effectively protects private data, reducing RougeL-F scores to below 10%. The benefits of our DualGuard mainly come from the proposed local warm-up parameter space transformation strategy, which transforms the client head model into a different parameter space before the formal split-based LLM fine-tuning. Conse-

| Model | Defense Method | GSM8k | | Dialogsum | | CodeAlpaca | | e2e | |
|---|---|---|---|---|---|---|---|---|---|
| | | RougeL-F | Meteor | RougeL-F | Meteor | RougeL-F | Meteor | RougeL-F | Meteor |
| GPT2-large | w/o defense | 81.422 | 75.688 | 73.549 | 70.985 | 81.837 | 76.719 | 80.633 | 76.884 |
| | DP-forward | 76.447 | 65.466 | 64.139 | 58.768 | 58.926 | 58.257 | 76.682 | 72.405 |
| | $d_\chi$P | 70.267 | 58.072 | 60.690 | 52.860 | 72.262 | 62.066 | 67.160 | 57.602 |
| | DP-SGD* | 77.545 | 65.826 | 66.921 | 64.166 | 78.584 | 72.306 | 76.380 | 71.349 |
| | DualGuard | **80.046** | **73.107** | **71.520** | **67.762** | **79.624** | **74.326** | **79.596** | **74.737** |
| Llama 3.2-1b | w/o defense | 84.352 | 82.712 | 76.163 | 73.068 | 87.108 | 85.128 | 80.642 | 76.858 |
| | DP-forward | 69.250 | 57.130 | 58.039 | 51.739 | 56.684 | 45.191 | 63.963 | 57.325 |
| | $d_\chi$P | 77.528 | 68.872 | 68.561 | 63.162 | 76.603 | 68.130 | 75.887 | 70.914 |
| | DP-SGD* | 81.976 | 78.036 | 71.562 | 66.422 | **85.088** | **82.985** | 73.109 | 66.985 |
| | DualGuard | **82.163** | **79.152** | **73.950** | **69.688** | 83.231 | 80.708 | **77.191** | **73.473** |
| Qwen 2-1.5b | w/o defense | 86.055 | 88.444 | 76.787 | 74.256 | 89.286 | 87.652 | 80.637 | 76.629 |
| | DP-forward | 78.262 | 73.834 | 66.169 | 62.661 | 72.335 | 64.029 | 77.907 | 73.357 |
| | $d_\chi$P | 51.946 | 35.287 | 41.797 | 30.831 | 51.946 | 35.284 | 49.116 | 37.109 |
| | DP-SGD* | **86.016** | **88.340** | 73.413 | 70.580 | **87.949** | **85.181** | 77.679 | 73.072 |
| | DualGuard | 83.658 | 83.982 | **74.225** | **72.166** | 85.536 | 83.153 | **79.389** | **74.968** |

Table 2: Task performance (ROUGE-L F1 Score and Meteor Score % ↑) of models under different methods and datasets. Higher scores indicate better performance. The scores of no-defense method are labeled in underline and the scores of the optimal defense method are highlighted in bold.

quently, during formal split-based fine-tuning, the SIP attack model trained on the original pre-trained model completely loses its ability to reconstruct data when confronted with our defense strategy.

**Defense against backward gradient matching attacks.** We study the effectiveness of defense mechanisms against backward gradient matching attacks TAG and LAMP, as shown in the corresponding TAG* and LAMP* rows in Table 1. The results reveal that the TAG and LAMP attackers achieve significant reconstructions of the original private sequences without defense mechanisms. Forward defenses like DP-forward and d$\chi$P are ineffective against these backward gradient matching attacks. Conversely, the backward defense mechanism DP-SGD*, successfully defends against gradient matching attacks. However, it does not fully prevent attackers from reconstructing original inputs using smashed data. DualGuard proves to be effective in defending against gradient matching attacks, with RougeL-F scores ranging from 3.61% to 33.11%. This level of recovery is inadequate for reconstructing coherent sentences or maintaining valid semantics (refer to Appendix G for a specific example). This success results from the synergy of local warm-up parameter space transformation and global fine-tuning parameter space retention. These mechanisms collectively transform the client tail model into a distinct parameter space from the pre-trained model, preventing activation gradients from being exploited during backward propagation.

**Defense against bidirectional and direct-connecting attacks.** The advanced DRA method BiSR combines optimized SIP and TAG attacks for bidirectional data reconstruction in split-based LLM-FT. DualGuard consistently outperforms other defenses against BiSR, keeping RougeL-F scores below 5.63%, thanks to its parameter space transformation paradigm, which counters both forward smashed data inversion and backward gradient matching attacks. We also evaluated the CPTA attack, which directly connects the fine-tuned server-side trunk model to the public pre-trained tail model. Due to the 'not-too-far' property of LLM-FT, perturbation-based defenses struggle, with CPTA achieving RougeL-F scores between 20% and 90%. DualGuard mitigates this risk by transforming the client's fine-tuned tail model into a secure parameter space, reducing CPTA attack effectiveness and keeping RougeL-F below 27% across all models and datasets.

**Defense against optimal attacks.** It is essential to simultaneously evaluate defense mechanisms based on their effectiveness in defending against various attacks in both forward and backward propagation processes. The maximum reconstruction success rate, as the attacker's optimal attack strategy, is a key and real indicator of a defense method's efficacy in split-based LLM-FT. For a clearer illustration, see a real example in Appendix G. As shown in the last row of Table 1, DualGuard shows robust defense capabilities against optimal attacks, with an average RougeL-F score of 1.67% across models and datasets. This outperforms existing methods, with average scores of 8.21% for DP-forward, 8.67% for d$\chi$P, and 7.52% for DP-SGD*, which illustrates that the existing methods can only defend against one type of attack but not against bidirectional attacks. DualGuard effectively thwarts attacks from both forward and backward directions, offering better protection than unidirectional defense methods vulnerable to optimal attacks.

### 4.3 Downstream Tasks Performance.

To study the impact of our method on the performance in downstream tasks, we compare DualGuard with the baseline no-defense approach and the four defense methods, with the same models and datasets above. The results are shown in Table 2. The results reveal that the downstream task performance of our DualGuard defense mechanism is well maintained compared to the scenario without defense, with difference scores of RougeL-F averaging at 2.4%. This performance surpasses other privacy defense methods in most test cases, with the average difference scores of DP-forward, d$\chi$P, and DP-SGD* being 15.2%, 21.6%, and 4.2%, respectively, across different models and datasets. Although DP-SGD* shows slightly better performance in a few test cases, the improvement is minimal. However, DP-SGD* entirely fails to defend against the reconstruction of original input data by forward smashed data inversion attackers, as shown in Table 1. In comparison, our method demonstrates robust bidirectional defense capabilities while preserving the performance of large language models in downstream tasks.

### 4.4 Ablation Experiments

In this subsection, we analyze the impact of two proposed strategies on the above five attackers using the Qwen2-1.5b model and the two large datasets CodeAlpaca20K and e2e.

**Impact of local warm-up parameter space transformation.** The warm-up phase aims to transfer the parameters to a secure space capable of withstanding various attack methods. After this phase is completed, we connect the head and tail models to the pre-trained trunk model (marked as **Warm-up Only**) and compare the results against a baseline approach without defense mechanisms (marked as **Naive USL**) to assess effectiveness. As shown in Figure 4, our method shows almost complete resistance to SIP attacks, with RougeL-F scores dropping from more than 70% to less than 10%. It also significantly reduces the RougeL-F scores of TAG*, LAMP*, BiSR, and CPTA attacks, confirming that the warm-up head and tail model parameters diverge from the pre-trained model parameters.

**Impact of global fine-tuned parameter space retention.** The global fine-tuned parameter space retention strategy prevents the tail model parameters from reverting to the original vulnerable space during formal fine-tuning after the warm-up phase. We conduct experiments comparing RougeL-F scores with and without this strategy (marked as **Warm-up+USL** and **DualGuard**, respectively). In Figure 5, without the strategy, the RougeL-F scores for the TAG*, LAMP*, and BiSR attacks are significantly higher compared to when the strategy is used. Moreover, under the CPTA attack, defense capabilities are completely lost without this strategy, as the parameters revert to the vulnerable space during formal fine-tuning. This issue is significantly alleviated when the strategy is applied.

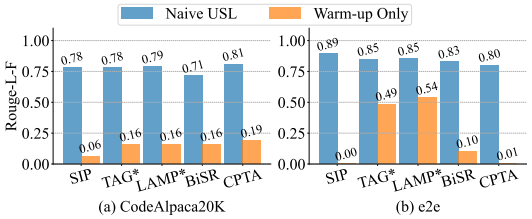



Figure 4: Impact of local warm-up parameter space transformation by comparing the defense performance for naive USL and our warm-up-only version (↓).

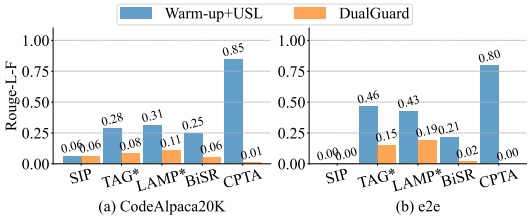



Figure 5: Impact of global fine-tuned parameter space retention by comparing the defense performance for warm-up driven USL and our complete DualGuard (↓).

### 4.5 Hyper-parameter Experiment

**Impact of $\lambda_1$, $\lambda_2$ and $\lambda_3$.** To evaluate the effects of hyperparameters $\lambda_1$, $\lambda_2$, and $\lambda_3$, we conducted experiments using the GSM8K dataset and the GPT-2 model, testing various hyperparameter combinations, as illustrated in Figure 6. Our results highlight the influence of these hyperparameters on model task performance and defense performance. Specifically, as $\lambda_1$ increasing from 0 to 40, the SIP attack success rate from 86.64% to 0.00% while task performance remains stable, slightly decreasing from 81.43% to 80.95%. Similarly, increasing $\lambda_2$ from 0 to 80 decreases the TAG attack success rate from 67.98% to 8.78%, with task performance remaining consistent. For $\lambda_3$, increasing its value from 0 to 4 significantly lowers the TAG attack success rate from 98.68% to 19.24%. However, further increases in $\lambda_3$ lead to a rebound in the attack success rate, likely due to excessive transformation of the tail model's parameters, which disrupts the synergy among the tail, head, and trunk models.

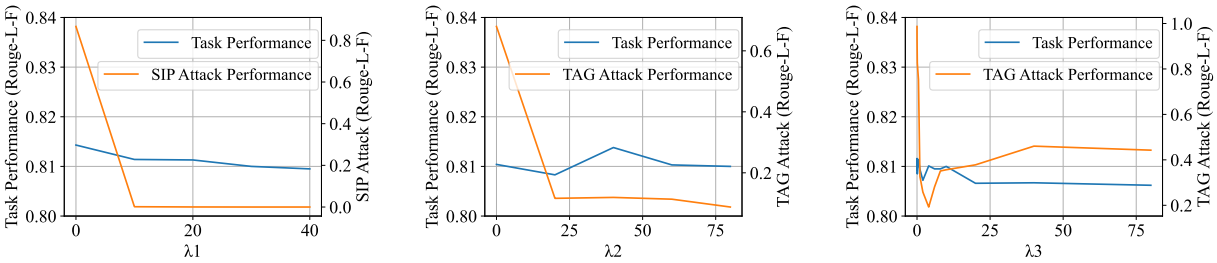



Figure 6: Task performance (ROUGE-L F1 score ↑) and attack performance (ROUGE-L F1 score ↓) of GPT-2 Large on GSM8K across $\lambda_1$, $\lambda_2$ and $\lambda_3$ configurations.

This disruption may cause gradient information leakage, exploitable by the adaptive TAG attack strategy. Consequently, the optimal value of $\lambda_3$ is critical and may vary depending on the specific model and dataset employed. In this paper, we adopt default hyperparameter settings of $\lambda_1 = 40$, $\lambda_2 = 80$, and $\lambda_3 = 10$, which consistently achieve near-optimal results across most scenarios. In future work, we plan to investigate adaptive strategies for optimizing these parameter settings to further enhance performance.

**Impact of $L_{\text{local\_ft}}$.** To evaluate the impact of the proposed loss term $L_{\text{local\_ft}}$ in the warm-up phase, We conducted experiments on multiple datasets using GPT-2 Large, measuring task performance using the ROUGE-L F1 score (%). The results are presented in Table 3, comparing the task performance with and without $L_{\text{local\_ft}}$. As shown in Table 3, when $L_{\text{local\_ft}}$ is applied, the ROUGE-L F1 scores range from 73.55% to 81.84%. However, removing this loss term leads to a drastic decline in performance, with scores dropping to a range of 28.89% to 69.32%. On average, the absence of $L_{\text{local\_ft}}$ results in a performance degradation of approximately 29.64%, underscoring its critical role in maintaining model accuracy. These results demonstrate that $L_{\text{local\_ft}}$ is essential for achieving high task performance, as its absence severely disrupts feature extraction in the head model and reduces the tail model's adaptability to downstream tasks during the warm-up phase.

| Dataset | w/ $L_{\text{local\_ft}}$ | w/o $L_{\text{local\_ft}}$ |
|---|---|---|
| GSM8k | 81.42 | 42.81 |
| Dialogsum | 73.55 | 43.21 |
| CodeAlpaca | 81.84 | 28.89 |
| e2e | 80.63 | 69.32 |

Table 3: Task performance (ROUGE-L F1 Score % ↑) with and without Loss term $L_{\text{local\_ft}}$.

## 5 Conclusion

This paper introduces DualGuard, a bidirectional defense mechanism designed to combat data reconstruction attacks in split-based LLM-FT. DualGuard combines local warm-up parameter space transformation and global fine-tuned parameter space retention to increase the divergence between client models and their pre-trained counterparts, thwarting attacks with priority knowledge. Experimental results show that DualGuard surpasses current defenses, offering superior privacy protection while maintaining task performance.

## Limitations

Although DualGuard has proven effective in countering DRAs, there are areas for further research to overcome its limitations. One potential improvement is to explore alternative projection strategies beyond the current three-layer MLP, aiming to enhance privacy protection with minimal overhead. Additionally, optimizing the weights of the loss function for multiple goals could be enhanced by implementing dynamic weight adjustments instead of fixed empirical values. Furthermore, to improve scalability, distributing the head and tail models of DualGuard to clients using a distributed GPU cluster could support larger language models.

## Acknowledgments

This research is supported by the "Pioneer" R&D Program of Zhejiang (Grant No.2024C01019) and the Hangzhou Joint Fund of the Zhejiang Provincial Natural Science Foundation of China (Grant No.LHZSD24F020001). The author gratefully acknowledges the support of Zhejiang University Education Foundation Qizhen Scholar Foundation.

## References


Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318.

Yoshimasa Akimoto, Kazuto Fukuchi, Youhei Akimoto, and Jun Sakuma. 2023. Privformer: Privacy-preserving transformer with mpc. In *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*, pages 392–410. IEEE.

Mislav Balunovic, Dimitar Dimitrov, Nikola Jovanović, and Martin Vechev. 2022. Lamp: Extracting text from gradients with language model priors. *Advances in Neural Information Processing Systems*, 35:7641–7654.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, and 1 others. 2021. On the opportunities and risks of foundation models. *Preprint*, arXiv:2108.07258.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Linxiao Cao, Yifei Zhu, and Wei Gong. 2024. Sfprompt: Communication-efficient split federated fine-tuning for large pre-trained models over resource-limited devices. *Preprint*, arXiv:2407.17533.

Konstantinos Chatzikokolakis, Miguel E Andrés, Nicolás Emilio Bordenabe, and Catuscia Palamidessi. 2013. Broadening the scope of differential privacy using metrics. In *Privacy Enhancing Technologies: 13th International Symposium, PETS 2013, Bloomington, IN, USA, July 10-12, 2013. Proceedings 13*, pages 82–102. Springer.

Sahil Chaudhary. 2023. Code alpaca: An instruction-following llama model for code generation. https://github.com/sahil280114/codealpaca. GitHub repository, accessed: 2023-12-21.

Guanzhong Chen, Zhenghan Qin, Mingxin Yang, Yajie Zhou, Tao Fan, Tianyu Du, and Zenglin Xu. 2024a. Unveiling the vulnerability of private fine-tuning in split-based frameworks for large language models: A bidirectionally enhanced attack. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 2904–2918.

Tianyu Chen, Hangbo Bao, Shaohan Huang, Li Dong, Binxing Jiao, Daxin Jiang, Haoyi Zhou, Jianxin Li, and Furu Wei. 2022. The-x: Privacy-preserving transformer inference with homomorphic encryption. *Preprint*, arXiv:2206.00216.

Yi-Qiang Chen, Teng Zhang, Xin-Long Jiang, Qian Chen, Chen-Long Gao, and Wu-Liang Huang. 2024b. Fedbone: Towards large-scale federated multi-task learning. *Journal of Computer Science and Technology*, 39(5):1040–1057.

Yulong Chen, Yang Liu, Liang Chen, and Yue Zhang. 2021. Dialogsum: A real-life scenario dialogue summarization dataset. *Preprint*, arXiv:2105.06762.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Jieren Deng, Yijue Wang, Ji Li, Chao Shang, Hang Liu, Sanguthevar Rajasekaran, and Caiwen Ding. 2021. Tag: Gradient attack on transformer-based language models. *Preprint*, arXiv:2103.06819.

Ye Dong, Wen-jie Lu, Yancheng Zheng, Haoqi Wu, Derun Zhao, Jin Tan, Zhicong Huang, Cheng Hong, Tao Wei, and Wenguang Chen. 2023. Puma: Secure inference of llama-7b in five minutes. *Preprint*, arXiv:2307.12533.

Minxin Du, Xiang Yue, Sherman SM Chow, and Huan Sun. 2023a. Sanitizing sentence embeddings (and labels) for local differential privacy. In *Proceedings of the ACM Web Conference 2023*, pages 2349–2359.

Minxin Du, Xiang Yue, Sherman SM Chow, Tianhao Wang, Chenyu Huang, and Huan Sun. 2023b. Dp-forward: Fine-tuning and inference on language models with differential privacy in forward pass. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 2665–2679.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pages 265–284. Springer.

Tom Farrand, Fatemehsadat Mireshghallah, Sahib Singh, and Andrew Trask. 2020. Neither private nor fair: Impact of data imbalance on utility and fairness

in differential privacy. In *Proceedings of the 2020 workshop on privacy-preserving machine learning in practice*, pages 15–19.

Oluwaseyi Feyisetan, Borja Balle, Thomas Drake, and Tom Diethe. 2020. Privacy-and utility-preserving textual analysis via calibrated multivariate perturbations. In *Proceedings of the 13th international conference on web search and data mining*, pages 178–186.

Otkrist Gupta and Ramesh Raskar. 2018. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116:1–8.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. *Preprint*, arXiv:2004.10964.

Dong-Jun Han, Hasnain Irshad Bhatti, Jungmoon Lee, and Jaekyun Moon. 2021. Accelerating federated learning with split learning on locally generated losses. In *ICML 2021 workshop on federated learning for user privacy and data confidentiality. ICML Board*.

Zecheng He, Tianwei Zhang, and Ruby B Lee. 2019. Model inversion attacks against collaborative inference. In *Proceedings of the 35th Annual Computer Security Applications Conference*, pages 148–162.

Shlomo Hoory, Amir Feder, Avichai Tendler, Sofia Erell, Alon Peled-Cohen, Itay Laish, Hootan Nakhost, Uri Stemmer, Ayelet Benjamini, Avinatan Hassidim, and 1 others. 2021. Learning and evaluating a differentially private pre-trained language model. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1178–1189.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Weijun Li, Qiongkai Xu, and Mark Dras. 2024. Seeing the forest through the trees: Data leakage from partial transformer gradients. *Preprint*, arXiv:2406.00999.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Zheng Lin, Xuanjie Hu, Yuxin Zhang, Zhe Chen, Zihan Fang, Xianhao Chen, Ang Li, Praneeth Vepakomma, and Yue Gao. 2024. Splitlora: A split parameter-efficient fine-tuning framework for large language models. *Preprint*, arXiv:2407.00952.

Wen-jie Lu, Zhicong Huang, Zhen Gu, Jingyu Li, Jian Liu, Cheng Hong, Kui Ren, Tao Wei, and WenGuang Chen. 2023. Bumblebee: Secure two-party inference framework for large transformers. *Cryptology ePrint Archive*.

Song Lyu, Zheng Lin, Guanqiao Qu, Xianhao Chen, Xiaoxia Huang, and Pan Li. 2023. Optimal resource allocation for u-shaped parallel split learning. In *2023 IEEE Globecom Workshops (GC Wkshps)*, pages 197–202. IEEE.

Peihua Mai, Ran Yan, Zhe Huang, Youjia Yang, and Yan Pang. 2023. Split-and-denoise: Protect large language model inference with local differential privacy. *Preprint*, arXiv:2310.09130.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *Preprint*, arXiv:1609.07843.

Khoa Nguyen, Tanveer Khan, and Antonis Michalas. 2023. Split without a leak: Reducing privacy leakage in split learning. In *International Conference on Security and Privacy in Communication Systems*, pages 321–344. Springer.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. *Preprint*, arXiv:1706.09254.

Dario Pasquini, Giuseppe Ateniese, and Massimo Bernaschi. 2021. Unleashing the tiger: Inference attacks on split learning. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 2113–2129.

Alec Radford. 2018. Improving language understanding by generative pre-training. https://openai.com/research/language-unsupervised.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Xicong Shen, Yang Liu, Huiqi Liu, Jue Hong, Bing Duan, Zirui Huang, Yunlong Mao, Ye Wu, and Di Wu. 2023. A split-and-privatize framework for large language model fine-tuning. *Preprint*, arXiv:2312.15603.

Chandra Thapa, Pathum Chamikara Mahawaga Arachchige, Seyit Camtepe, and Lichao Sun. 2022. Splitfed: When federated learning meets split learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8485–8493.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. 2018. Split learning for health: Distributed deep learning without sharing raw patient data. *Preprint*, arXiv:1812.00564.

Teng Wang, Lindong Zhai, Tengfei Yang, Zhucheng Luo, and Shuanggen Liu. 2024. Selective privacy-preserving framework for large language models fine-tuning. *Information Sciences*, page 121000.

Yiming Wang, Yu Lin, Xiaodong Zeng, and Guannan Zhang. 2023. Privatelora for efficient privacy preserving llm. *Preprint*, arXiv:2311.14030.

Maoqiang Wu, Guoliang Cheng, Dongdong Ye, Jiawen Kang, Rong Yu, Yuan Wu, and Miao Pan. 2023. Federated split learning with data and label privacy preservation in vehicular networks. *IEEE Transactions on Vehicular Technology*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, and 1 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Zongshun Zhang, Andrea Pinto, Valeria Turina, Flavio Esposito, and Ibrahim Matta. 2023. Privacy and efficiency of communications in federated split learning. *IEEE Transactions on Big Data*, 9(5):1380–1391.

Ligeng Zhu, Zhijian Liu, and Song Han. 2019. Deep leakage from gradients. *Advances in neural information processing systems*, 32.

Itamar Zimerman, Moran Baruch, Nir Drucker, Gilad Ezov, Omri Soceanu, and Lior Wolf. 2023. Converting transformers to polynomial form for secure inference over homomorphic encryption. *Preprint*, arXiv:2311.08610.

## A Experimental Setup Details

**Hyper Parameter Settings.** To access DualGuard within a split-based LLM-FT framework, the models are divided into a head model, a trunk model, and a tail model to ensure a manageable computational load on the client's end. The hyperparameters used for each LLM model are optimized for performance, as detailed in Table 4. It is important to notice that three numbers in split segments means the layers count of the head model, the trunk model, and the tail model respectively. In addition, in our experiments, we fixed the weights for the loss functions of the local warm-up and global fine-tuning phases, based on the empirical optimum determined during the experiments.

We employ LoRA (Hu et al., 2021) to fine-tune the head and tail models to adapt a real-world resource-constrained environment, and LoRA configurations are listed in Table 5. All experiments are conducted on a Tesla V100-SXM2 GPU (32GB) to simulate the split-based LLM-FT scenario, similar to existing split learning based works (Chen et al., 2024a; Lin et al., 2024; Thapa et al., 2022).

| Model | Split Segments | $\lambda_1$ | $\lambda_2$ | $\lambda_3$ | Warm-up Epc |
|---|---|---|---|---|---|
| GPT2-large | 3-30-3 Layers | 40 | 80 | 10 | 2 |
| Llama3.2-1b | 3-26-3 Layers | 40 | 80 | 10 | 2 |
| Qwen2-1.5b | 2-12-2 Layers | 40 | 80 | 10 | 2 |

Table 4: DualGuard hyper parameters for LLMs.

| Model | Task Type | r | LoRA Alpha | LoRA Dropout | Target Modules |
|---|---|---|---|---|---|
| GPT2-large | CAUSAL-LM | 2 | 32 | 0.1 | c-proj , c-attn |
| Llama3.2-1b | CAUSAL-LM | 2 | 32 | 0.1 | q-proj , v-proj |
| Qwen2-1.5b | CAUSAL-LM | 2 | 32 | 0.1 | q-proj , v-proj |

Table 5: DualGuard LoRA hyper parameters for LLMs. This is implemented by **peft** package in Python.

Besides, We need to compare the performance of our DualGuard with three advanced defense mechanisms, DP-forward, d$\chi$P, and DP-SGD. It is important to note that the $\epsilon$ values, which indicate the noise level in perturbation-based methods (the higher the $\epsilon$ value, the less noise is added), are set as $\epsilon = 2$ for DP-forward and DP-SGD, and $\epsilon = 0.15$ for d$\chi$P. These settings are optimized to strike a balance between defense performance and task performance, with the corresponding task performance shown in Appendix E.

**Evaluation metrics.** To comprehensively evaluate the effectiveness of the proposed defense mechanism and its impact on downstream task performance, we adopt two widely used metrics for assessing natural language generation quality: RougeL-F (Lin, 2004) and Meteor (Banerjee and Lavie, 2005). RougeL-F evaluates the longest common subsequence (LCS) matching between the generated text and reference text. Meteor complements this by capturing multi-level similarity between the generated and reference texts, including lexical, semantic, and word-order matching.

**Details of Applied Datasets.** We performed experiments on five different datasets. Each dataset corresponds to a different natural language generation task, including structured data text generation (e2e (Novikova et al., 2017)), code generation (CodeAlpaca-20k (Chaudhary, 2023)), mathematical reasoning (GSM8k (Cobbe et al., 2021)), and dialogue summary generation (Dialogsum (Chen et al., 2021)). In addition, the Wikitext dataset (Merity et al., 2016) was used as an auxiliary dataset for the attackers in the experiments.

| Dataset | Task | Avg. length | Total samples |
|---|---|---|---|
| e2e | structured data text generation | 61.5 | 42061 |
| CodeAlpaca | code generation | 112.4 | 18019 |
| GSM8k | mathematical reasoning | 157.1 | 7473 |
| Dialogsum | dialogue summary generation | 248.1 | 12460 |

Table 6: Summary of datasets used in the experiments

## B Impact of Split Segments

In the split learning framework, the default split segments for the head, trunk, and tail models are 3, 30, and 3 layers, respectively. This subsection investigates the task and defense performance with different split segments. We test three attacker methods: the forward SIP attack, the backward TAG attacker, and the bidirectional BiSR attacker. Table **??** illustrates the effectiveness of DualGuard in downstream tasks and defense against various attacks at different split segments, with **Naive** indicating USL without any defense mechanisms. As the split segments of the head and tail models increase from 1 to 11 layers, the performance of downstream tasks experiences a slight reduction in RougeL-F from 80.91% to 78.73%. This reduction is attributed to the increasing number of model layers in the frozen head model. Hence, it is recommended to allocate fewer layers to the head model and more layers to the tail model if there are more computing and storage resources available on the local client to avoid this issue. In contrast, the defense against the backward attack TAG* is strengthened, with the RougeL-F scores reduced from 39.44% to 3.16% as the split segments of the head and tail models increase from 1 to 7 layers. This improvement is due to the increased number of model layers in the tail model, leading to more divergence in activation gradients from the pre-trained tail model and neutralizing the gradient matching attacks. Besides, DualGuard consistently demonstrates robust effectiveness against SIP and BiSR attacks, with the RougeL-F scores staying below 6% benefiting from our parameter space transformation method.

## C Defense Overhead

To assess the time and memory overhead of DualGuard compared to other methods, we conduct experiments on GPT2-large using various datasets. Figure 7 illustrates the time and memory overhead of DualGuard and other defense methods. The time cost of DualGuard is slightly higher than

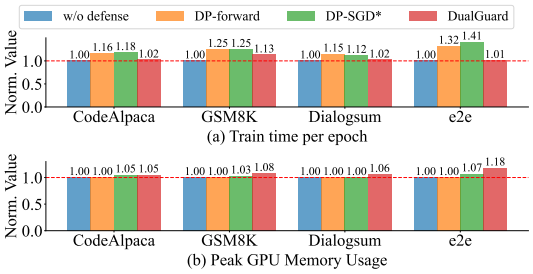



Figure 7: Comparison of defense overhead (↓).

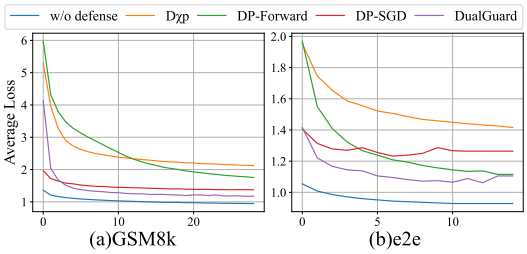
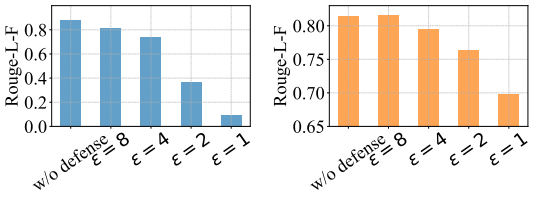



Figure 8: Convergence curves of datasets GSM8k and e2e under different methods and model GPT2-large.

the method without defense (1.01 to 1.13 times) but remains lower than some other defense methods. This increase is due to DualGuard's multi-task learning strategy, which involves additional computation of the loss of the pre-trained tail model, leading to a moderate increase in time overhead. Nonetheless, DualGuard is notably more efficient than perturbation-based defense methods that require frequent cropping and noise addition during training. Regarding memory overhead, DualGuard exhibits a slight increase (1.05 to 1.18 times) compared to the method without defense and other defense methods. Although the head model is frozen in DualGuard, reducing the need for optimizer and gradients for this part during training, an extra copy of the pre-trained tail model must be stored by the client, resulting in additional memory usage. However, since the pre-trained tail model comprises a small number of layers and does not require training, the additional memory overhead is minimal and remains within acceptable limits.

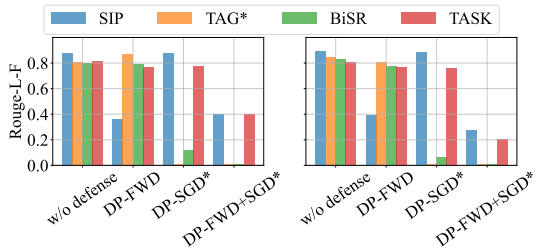



(a) Defense performance (↓) (b) Task performance (↑)

Figure 9: Trade-off between defense and task performances under different privacy budgets, with ↑ indicating higher values are better and ↓ indicating lower values are better.

| Split Segments | TASK (↑) | | SIP (↓) | | TAG* (↓) | | BiSR (↓) | |
|---|---|---|---|---|---|---|---|---|
| | RougeL-F | Meteor | RougeL-F | Meteor | RougeL-F | Meteor | RougeL-F | Meteor |
| Naive | 81.84 | 76.72 | 89.78 | 93.42 | 80.93 | 85.68 | 78.05 | 81.45 |
| 1-34-1 | 80.91 | 75.60 | 0.16 | 0.03 | 39.44 | 17.00 | 3.86 | 1.39 |
| 3-30-3 | 80.60 | 75.06 | 0.00 | 0.00 | 15.28 | 4.42 | 1.89 | 0.88 |
| 5-26-5 | 80.10 | 74.67 | 0.00 | 0.00 | 3.74 | 1.00 | 5.73 | 3.31 |
| 7-22-7 | 79.74 | 74.40 | 0.00 | 0.00 | 3.16 | 0.65 | 5.92 | 3.03 |
| 9-18-9 | 78.90 | 73.38 | 1.03 | 0.15 | 9.39 | 2.55 | 3.94 | 1.00 |
| 11-14-11 | 78.73 | 73.03 | 5.61 | 1.07 | 4.43 | 1.43 | 4.33 | 0.80 |

Table 7: Task performance (ROUGE-L F1 Score and Meteor Score % ↑) and defense performance (ROUGE-L F1 Score and Meteor Score % ↓) across split segments.




(a) GSM8k (b) e2e

Figure 10: Defense efficacy of diverse attack-defense pairs.

## D Convergence performance

Since the methods change the head model and tail model parameter space during local warm-up parameter space transformation phase, which may causes a slowdown in convergence during formal fine-tuning, so we use the GPT2-large model and two datasets, GSM8k and e2e, to compare the convergence of the various methods. The convergence curves are shown in Figure 8.

In this experiment, the horizontal coordinate is the number of validations, it should be noted that each epoch will be validated 5 times, and the average loss of the whole validation dataset will be taken each time, i.e., the vertical coordinate. Besides, due to the different epochs required for the completion of the fine-tuning of the different methods, this truncated the loss data with only a selected number of validation times, 28 for GSM8k and 14 for e2e, respectively. We can find that our method does cause a slowdown in convergence compared to the method without defense, but converges a little faster than perturbation-based methods DP-Forward, D$\chi$p, and DP-SGD.

## E Limitations of Defense in Split-Based LLM-FT

Although the perturbation-based method provides effective protection for general deep learning and LLM-FT scenarios, it faces several constraints when applied to split-based LLM-FT situations.

**Trade-off between privacy preservation and task performance.** One major limitation of introducing random noise is the potential performance degradation in downstream tasks. Strengthening privacy protection often requires higher noise levels, which can adversely hamper task-specific model performance (Zhu et al., 2019; Farrand et al., 2020). To validate it, we run the GPT2-large (Radford et al., 2019) model on the GSM8k (Cobbe et al., 2021) dataset using the SIP reconstruction (Chen et al., 2024a) for attack and the state-of-the-art DP-forward (Du et al., 2023b) for defense, and show the defense performance and the task performance under different privacy budgets $\epsilon$ in Figure 9(a) and Figure 9(b), respectively, with the y-axis representing the text similarity metric RougeL-F (Lin, 2004). We can see that, as $\epsilon$ decreases (indicating increased noise levels), defense performance improves as the similarity between attacker-generated sequences and original input decreases. However, this improvement comes at the cost of decreased task-specific model performance. For instance, reducing $\epsilon$ from 8 to 1 results in a drop in RougeL-F score for SIP reconstruction from 86% to 9%, along with a decrease in the performance of downstream tasks from 81.9% to 69.8%.

**Inadequate defense against bidirectional attacks.** Existing defense mechanisms often focus on either forward or backward propagation, overlooking the interconnectedness of LLMs. The auto-regressive nature of the LLMs results in substantial overlap between input and labels during training, requiring simultaneous protection for both. Defenses against forward activations alone do not suffice against gradient matching attacks, and defenses against backward gradients leave models vulnerable to smashed data inversion attacks. To verify this, we test the defense performance, measured by the RougeL-F metric, of different defense mech-

anisms (DP-forward (Du et al., 2023b) and DP-SGD (Abadi et al., 2016)) against different attacks (forward smashed data inversion (SIP) (Chen et al., 2024a), backward gradient matching (TAG) (Deng et al., 2021), and bidirectional(BiSR) attack (Chen et al., 2024a)) using the GPT2-large (Radford et al., 2019) model on the GSM8k (Cobbe et al., 2021) and e2e (Novikova et al., 2017) datasets. The results, shown in Figure 10, indicate that defenses targeting only one direction of attack are ineffective against the other. For example, on the GSM8k dataset, DP-SGD effectively defends against TAG attacks but fails to prevent SIP reconstruction. Conversely, DP-Forward can resist SIP attacks but is ineffective against TAG attacks. Besides, an attempt to merge DP-SGD and DP-forward resulted in a notable decline in downstream task performance, with RougeL-F dropping to below 40%, based on our experiments.

**Significant prior knowledge in split-based LLM-FT.** Additionally, the not-too-far property of LLM-FT indicates that fine-tuning involves minimal updates to model parameters, keeping the features embedded in the pre-trained weights largely intact. These retained features serve as crucial prior knowledge for DRAs, granting attackers extensive insight. When coupled with bidirectional attack strategies, attackers can exploit this prior knowledge to launch sophisticated attacks, challenging existing defense mechanisms and hindering effective risk mitigation (Chen et al., 2024a). For instance, applying the advanced defense method DP-Forward with the noise level set as $\epsilon = 2$, to the GPT-2 model on the GSM8K dataset only marginally reduces the RougeL-F score of the BiSR attacker from 79.54% to 79.02%, as depicted in Figure 10.

## F Defense Performance against DRAs in Meteor Metric

To more comprehensively assess DualGuard's ability to defend against various data reconstruction attacks (DRAs), we evaluated the Meteor metrics for DualGuard and other defense methods across different attack strategies. These metrics are used to measure the semantic similarity between the reconstructed text and the original text, as shown in Table 8. Forward defense methods such as DP-forward and D$\chi$p are effective against forward-direction attacks like SIP but are unable to defend against backward attacks such as TAG and LAMP, and the meteoric ratio of the reconstruction result to the original text is greater than 66.75% and up to 92.25%. Meanwhile, DP-SGD is able to resist the reverse attack but not the forward attack, and the meteoric ratio of the reconstruction result to the original text is greater than 67.80% and up to 91.41%. When facing the attacker's optimal attack, the reconstruction rate with methods like DP-forward, DXP, and DP-SGD exceeds 63.66%, reaching up to 96.02%. In contrast, the reconstruction rate for the optimal attack using the DualGuard defense remains below 33.11%. Thus, DualGuard demonstrates bidirectional defense capabilities, which other methods lack, and the experimental results align with those discussed in §4.2.

## G Illustrative Example of Attack Results

To provide a clearer visualization of data reconstruction attacks (DRAs), we utilize the GPT2-large model and a single input sample from the GSM8k dataset to showcase the real outcomes of different defense mechanisms against DRAs. We select forward smashed data inversion (SIP) and backward gradient matching (TAG) as our attack methods for comparison. Additionally, we evaluate DualGuard against a no-defense method and perturbation-based methods DP-Forward and DP-SGD. Note that, since DP-Forward and D$\chi$p both introduce noise to smashed data during forward propagation, we only show the results of more superior DP-Forward in this comparison.

As shown in Figure 11, the original input (which is the same as the output labels in LLM situations due to the self-regressive nature) is represented by the blue text, while the outcomes of the data reconstruction attackers after applying various defense methods are displayed, with the red content indicating the overlap with the origin input data. We can see that, Without any defense, SIP and TAG successfully recover over 90% of the original data. The DP-Forward defense effectively combats forward attacker SIP, recovering approximately 40% of the origin input data, but backward attacker TAG still manages to recover most of the output labels, which can also expose the original data. In contrast, DP-SGD successfully prevents TAG recovery but fails against SIP. Our DualGuard method provides robust defense against both SIP and TAG, with no data recovered after SIP and only a few after TAG. These findings highlight that in the face of simultaneous bidirectional attacks, our method outperforms single-directional defense strategies.

| Attack Method | Defense Method | GPT2-large | | | | Llama3.2-1b | | | | Qwen2-1.5b | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | GSM8k | Dialogsum | CodeAl | e2e | GSM8k | Dialogsum | CodeAl | e2e | GSM8k | Dialogsum | CodeAl | e2e |
| SIP | w/o defense | 87.63 | 89.78 | 77.92 | 89.33 | 84.32 | 91.08 | 68.04 | 85.59 | 84.20 | 88.02 | 64.49 | 85.75 |
| | DP-forward | 36.37 | 27.90 | 36.39 | 38.92 | **0.39** | 14.75 | 0.24 | 0.12 | 12.94 | 14.75 | 4.93 | 12.85 |
| | $d_\chi$P | 44.48 | 41.84 | 43.59 | 60.30 | 61.23 | 68.97 | 53.19 | 69.22 | 18.76 | 13.38 | 15.94 | 22.47 |
| | DP-SGD* | 87.37 | 90.31 | 77.95 | 88.30 | 86.24 | 91.43 | 70.49 | 87.20 | 84.57 | 87.52 | 67.80 | 85.82 |
| | DualGuard | **0.00** | **0.88** | **0.00** | **0.00** | 0.49 | **0.18** | **0.05** | **0.00** | **9.04** | **2.02** | **6.47** | **0.30** |
| TAG* | w/o defense | 81.05 | 80.93 | 77.91 | 84.91 | 94.91 | 93.86 | 72.14 | 95.78 | 81.90 | 93.81 | 77.70 | 93.35 |
| | DP-forward | 86.84 | 50.20 | 79.86 | 80.44 | 83.03 | 89.69 | 88.85 | 71.94 | 80.70 | 89.00 | 83.10 | 91.44 |
| | $d_\chi$P | 87.30 | 73.40 | 78.62 | 71.24 | 78.36 | 69.13 | 69.54 | 92.96 | 67.99 | 67.40 | 66.86 | 80.08 |
| | DP-SGD* | **0.00** | **0.24** | **0.04** | **0.00** | **1.21** | **0.94** | **1.40** | **0.40** | **1.23** | **1.27** | **0.74** | **0.00** |
| | DualGuard | 20.96 | 18.94 | 8.24 | 15.28 | 21.77 | 24.02 | 10.34 | 3.61 | 23.55 | 5.25 | 4.91 | 13.05 |
| LAMP* | w/o defense | 79.50 | 80.35 | 78.93 | 85.31 | 94.78 | 96.02 | 72.72 | 96.02 | 83.96 | 93.13 | 81.13 | 92.77 |
| | DP-forward | 85.51 | 47.30 | 79.67 | 81.81 | 84.07 | 89.73 | 89.25 | 71.98 | 81.03 | 87.30 | 85.46 | 92.25 |
| | $d_\chi$P | 87.73 | 67.75 | 78.31 | 67.02 | 79.94 | 66.75 | 67.28 | 95.21 | 68.02 | 66.85 | 66.86 | 81.08 |
| | DP-SGD* | **0.35** | **0.09** | **0.14** | **0.00** | **1.31** | **1.13** | **1.84** | **0.52** | **1.47** | **2.14** | **0.92** | **0.44** |
| | DualGuard | 25.35 | 22.66 | 10.63 | 19.14 | 22.32 | 33.11 | 11.28 | 4.72 | 26.31 | 5.62 | 5.78 | 12.28 |
| BiSR | w/o defense | 79.54 | 78.05 | 71.32 | 82.88 | 84.22 | 83.45 | 56.31 | 80.03 | 82.28 | 86.29 | 76.80 | 86.38 |
| | DP-forward | 79.02 | 44.86 | 62.98 | 77.30 | 35.08 | 29.42 | 28.78 | 22.60 | 64.01 | 74.12 | 62.98 | 91.04 |
| | $d_\chi$P | 57.56 | 40.16 | 52.50 | 57.16 | 66.95 | 56.23 | 56.28 | 81.66 | 40.89 | 34.67 | 58.53 | 68.91 |
| | DP-SGD* | 12.20 | 7.96 | 11.22 | 6.65 | 7.25 | 6.44 | 6.12 | 1.48 | 7.92 | 5.47 | 5.88 | 2.31 |
| | DualGuard | **1.58** | **6.77** | **5.63** | **1.89** | **2.46** | **2.25** | **3.70** | **0.00** | **5.35** | **3.68** | **4.09** | **1.56** |
| CPTA | w/o defense | 75.52 | 64.80 | 80.98 | 80.19 | 82.56 | 71.58 | 86.96 | 80.23 | 85.88 | 71.92 | 89.21 | 80.28 |
| | DP-forward | 78.48 | 63.66 | 69.23 | 78.62 | 60.51 | 51.23 | 47.29 | 59.59 | 73.02 | 55.93 | 65.17 | 77.63 |
| | $d_\chi$P | 66.43 | 55.35 | 68.95 | 73.78 | 73.81 | 63.94 | 73.27 | 71.88 | 43.87 | 34.78 | 33.23 | 44.28 |
| | DP-SGD* | 71.77 | 53.60 | 69.55 | 48.73 | 76.55 | 58.79 | 78.83 | 51.86 | 81.46 | 61.36 | 80.86 | 65.69 |
| | DualGuard | **0.00** | **0.00** | **0.98** | **0.00** | **0.38** | **0.45** | **0.71** | **0.01** | **5.20** | **5.67** | **0.76** | **0.55** |
| **Optimal Attack** | w/o defense | 87.63 | 89.78 | 80.98 | 89.33 | 94.91 | 96.02 | 86.96 | 96.02 | 85.88 | 93.81 | 89.21 | 93.35 |
| | DP-forward | 86.84 | 63.66 | 79.86 | 81.81 | 84.07 | 89.73 | 89.25 | 71.98 | 81.03 | 89.00 | 85.46 | 92.25 |
| | $d_\chi$P | 87.73 | 73.40 | 78.62 | 73.78 | 79.94 | 69.13 | 73.27 | 95.21 | 68.02 | 67.40 | 66.86 | 81.08 |
| | DP-SGD* | 87.37 | 90.31 | 77.95 | 88.30 | 86.24 | 91.43 | 78.83 | 87.20 | 84.57 | 87.52 | 80.86 | 85.82 |
| | DualGuard | **25.35** | **22.66** | **10.63** | **19.14** | **22.32** | **33.11** | **11.28** | **4.72** | **26.31** | **5.62** | **6.47** | **13.05** |

Table 8: Defense performance (measured by Meteor Score % ↓) of different defense method against various DRAs.

| | |
|---|---|
| **Original Input (=Labels)** | ### Question: Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?\n### Answer: Janet sells 16 - 3 - 4 = <<16-3-4=9>>9 duck eggs a day.\nShe makes 9 * 2 = $<<9*2=18>>18 every day at the farmer's market.\n#### 18 |

| Defense Method | Attack Method | Attack Result |
|---|---|---|
| w/o defense | SIP | ei Question : Janet 's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes mixtureins for her friends every day with four. She sells the remainder at the farmers ' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers ' market ?\nhhh answering : Janet sells 16 - 3 - 4 = <16-3-4 =9 ]9 duck eggs a day.\n She makes 9 * 2 = $ <9>2 =18 ]18 every day at the farmer 's market.\niveness 18 |
| | TAG | Covers Question: Janet‹s ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?\nSolution Answer: Janet sells 16 - 3 - 4 = $16-3-4=9>>9 duck eggs a day.\nShe makes 9 * 2 = $29*2=18.18 every day at the farmer's market.\nShe 18 |
| DP-forward | SIP | fungusulations> Janet 's stick laid 16 eggs per Day bill she fed three three breakfaststs wake, bons bombingersé her Buddy towns day lasted mile common She bought introduced remainder each on farmer 17 Market 2014emies budget2 per fresh duck egg arms telling much mostly dollars does she Union every day minute his farmeristic market ? Serious step solution turn Janet sell 16 20043 - 4 = <16-3 164ifying9 µves duck eggs a day discovery strategy She made 9 types 2 = $ <93 *iaia18 mythology 18 any day run pressed farmer 's market church toolís Revolution |
| | TAG | cycle Question: Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?\n### Answer: Janet sells 16 - 3 - 4 = <<16-3-4=9>>9 duck eggs a day.\nShe makes 9 * 2 = $<<9*2=18>>18 every day at the farmer's market.\n#### 18 |
| DP-SGD* | SIP | Zh Question : Janet 's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes mixtureins for her friends every day with four. She sells the remainder at the farmers ' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers ' market ?\ninfo answering : Janet sells 16 - 3 - 4 = <16-3-4 =9 ]9 duck eggs a day.\n She makes 9 * 2 = $ trig9 *2 =18 ]18 every day at the farmer 's market.\nGT 18 |
| | TAG | JrbreadVeh Olderatorial•ional necessity CunninghamHom subscribed skewed devs Replacement Sinnvan Acadalian covering SeasonsOPSross\x17maryachineibel Kingdomsxt masqualantryAvailabilityessee WD randomizedinventory Bergerestinebspval cerv mantleeniriftguewagen gloveswiadra Puzz guest paranormal CorinthIPS Oaksfal Sebastlua fosterrollersshireonge Troparia lengths FitzgeraldNY TRE.– Zur calmingedom \u200ewingsluck BurstumbersbuffCVE RELEshoreimov DH itars strideabloWCathyintendRESULTS anywayagar Finder Jorge RED SAFabor snail Qiao Lahansk Lindsey fadesattaynskiordingrgSON Hitch Mim\\",voicerackQUIgivingWF Tennixelrowd Josérusifice |
| Dual-Guard | SIP | \n\n\n\