TextFusion: Privacy-Preserving Pre-trained Model Inference via Token Fusion

Xin Zhou[1]*, Jinzhu Lu[1]*, Tao Gui[2]†, Ruotian Ma[1], Zichu Fei[1]
Yuran Wang[4], Yong Ding[4], Yibo Zhang[4], Qi Zhang[1]†, Xuanjing Huang[1, 3]
[1]School of Computer Science, Fudan University, Shanghai, China
[2]Institute of Modern Languages and Linguistics, Fudan University, Shanghai, China
[3] International Human Phenome Institutes, Shanghai, China
[4] Honor Device Co., Ltd
{xzhou20,tgui,qz}@fudan.edu.cn, lujz21@m.fudan.edu.cn

Abstract

Recently, more and more pre-trained language models are released as a cloud service. It allows users who lack computing resources to perform inference with a powerful model by uploading data to the cloud. The plain text may contain private information, as the result, users prefer to do partial computations locally and upload intermediate representations to the cloud for subsequent inference. However, recent studies have shown that plain text can also be recovered by intermediate representations with reasonable accuracy, thus the risk of privacy leakage still exists. To address this issue, we propose *TextFusion*, a novel method for preserving inference privacy. Specifically, we train a Fusion Predictor to dynamically fuse token representations, which hides multiple private token representations behind an unrecognizable one. Furthermore, a misleading training scheme is employed to privatize these representations. In this way, the cloud only receives incomplete and perturbed representations, making it difficult to accurately recover the complete plain text. The experimental results on diverse classification tasks show that our approach can effectively preserve inference privacy without substantially sacrificing performance in different scenarios.

1 Introduction

Pre-trained language models (PLMs) achieve state-of-the-art performance in many NLP tasks (Devlin et al., 2018; Liu et al., 2019; Brown et al., 2020; Qiu et al., 2020) . In industrial applications, running full PLM locally can be very expensive or even infeasible for most users due to the high computation requirements. Therefore, PLMs are usually released as cloud services, allowing users to access these powerful models by uploading the data to the cloud (DALE, 2015; Pais et al., 2022).

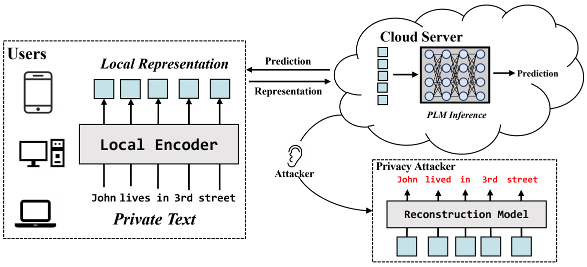


Figure 1: Illustration of privacy attack when using cloud services. Users upload the local representations to the cloud server for subsequent inference. Attacks can recover these representations to plain text, thus steal private user information.

Though inference with cloud models is convenient and powerful, it brings risk to privacy due to the sensitive nature of user data (Chi et al., 2018). For example, plain text data may contain private information about the user, such as name, address, and phone number. It is unacceptable for most users to upload such private data directly to the cloud (Jegorova et al., 2021). A natural way to avoid this problem is to perform affordable computation locally to obtain intermediate representations and upload them to the cloud for subsequent computation (Zhang et al., 2021), as shown in Figure 1. However, the intermediate representation still leaks privacy through recent text reconstruction techniques (Song and Raghunathan, 2020; Pan et al., 2020). The privacy leakage during the inference phase not only prevents users from benefiting from the PLM, but also results in disputes, penalties, and reputation damages to service providers (Chen et al., 2022b).

Most privacy-preserving methods focus on privatizing representation in the training phase. In their methods, differential privacy (Hoory et al., 2021; Yue et al., 2021), noise injection (Lyu et al., 2020a; Plant et al., 2021) and adversarial training (Li et al., 2018; Coavoux et al., 2018) are employed to reduce the privacy information in representations

*Equal contribution.
†Corresponding authors.

during training. However, privacy attackers can train a text reconstruction model that directly uses privatized representations to recover raw words because these learned representations and related raw words are accessible to attackers in inference phase (Song and Raghunathan, 2020; Höhmann et al., 2021). The risk of privacy leakage still exists.

In this paper, we propose TextFusion to preserve inference privacy. Our method directly widens the gap between intermediate representations and plain text in the inference phase. Specifically, we train a fusion predictor to identify which token representations are suitable to be fused into one representation. For each forward inference, we fuse these representations at the target layer and output reorganized token representations that cannot be aligned with the raw words in plain text. Meanwhile, a misleading training scheme is adopted to make both fused and unfused representations not similar to the related words. In this way, we break the one-to-one relationship between token representation and raw words. The incomplete and scrambled representation sequence makes it difficult for an attacker to recover the plain text. Additionally, token fusion cannot be directly applied to token classification tasks such as named entity recognition, because these tasks require predictions of each token. To solve this problem, we draw inspiration from early exiting (Xin et al., 2020; Li et al., 2021), which gets predictions for confident tokens in earlier layers. We only fuse these token representations and keep others unchanged. Thus, we can still preserve privacy while getting predictions of all tokens. Our codes are publicly available at `https://github.com/xzhou20/TextFusion`.

Our contribution can be summarized as follows:

- We propose TextFusion, a novel method for preserving inference privacy with dynamic token fusion.
- We apply token fusion to both sentence and token classification. We train a fusion predictor to ensure token fusion does not affect task completion.
- We conduct experiments on four classification benchmarks. The experimental results show that our method can protect inference privacy without substantially sacrificing performance.

2 Background

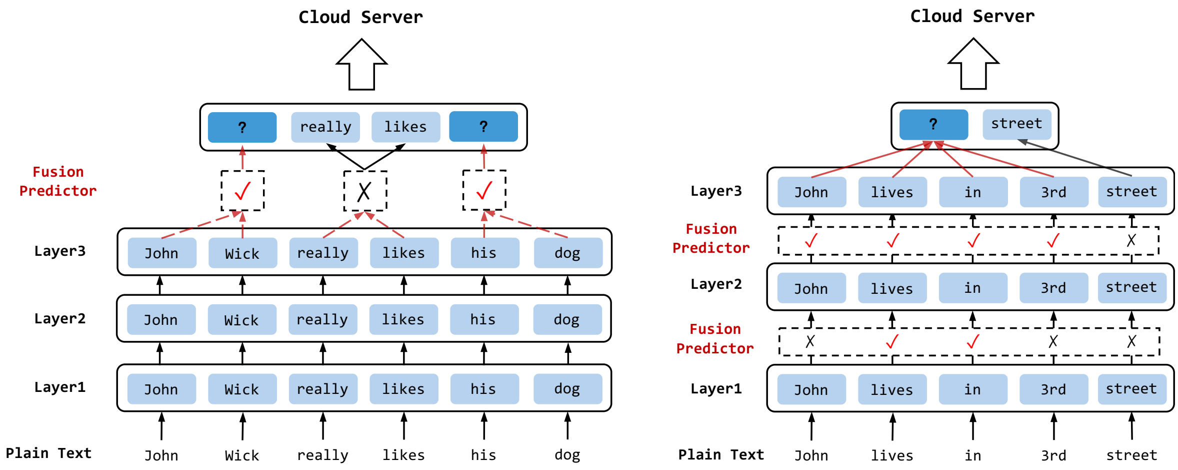
2.1 Inference with Cloud Model

Inference of PLMs requires significant computation resources, which is infeasible on resource-limited devices like mobile phones and smart chips. To make these large and powerful models benefit more users, PLMs are usually deployed as cloud services. Suppose a user wants to use cloud services to analyze the sentiment of text $X = [x_1, x_2, ..., x_n]$. For privacy reasons, the user does not upload plain text X directly to the cloud but first performs an affordable local computation to obtain the intermediate representation $\mathbf{H} = Enc_c(X)$, where Enc_c consists a few PLM layers and is deployed on a local device, $\mathbf{H} \in \mathbb{R}^{n \times d}$ and d is the dimension of representation. Then the user uploads intermediate representation $\mathbf{H}$ to the cloud server for the subsequent inference $Y = Enc_s(\mathbf{H})$, where Enc_s is the remaining PLM layers deployed on a cloud server and Y is the prediction that will be sent back to the user. In this scenario, the privacy attackers are not accessible to the private plain text. However, the intermediate representation can still leak the privacy under text reconstruction attacks.

2.2 Text Reconstruction Attack

Text reconstruction attack uses the token representation $\mathbf{h}_i$ to predict its original word x_i. Even in the strictest case, where the attacker can only obtain intermediate representations, Feyisetan et al. (2020) shows intermediate representations can be still recovered to the original word via finding the nearest word in the word embedding matrix. A more critical scenario is that attackers can obtain data representations shared by users and have query access to the public encoder Enc_c. Although attackers have no knowledge about the architecture and parameters of Enc_c, they can still exploit query feedback of Enc_c to carry out reconstruction attacks (Song and Raghunathan, 2020; Höhmann et al., 2021). For instance, attackers can generate intermediate representations by querying Enc_c for many times. Then they can build a reconstruction model Rec: $\mathcal{H} \rightarrow \mathcal{X}$, where $\mathcal{H}$ means the representation space and $\mathcal{X}$ means the input space. Rec can be optimized by minimizing the objective between representation and raw text:

$$\Theta_{\text{Rec}} = \underset{\Theta_{\text{Rec}}}{\arg\min} - \sum_{i=1}^{n} \log P\left(V\left(x_i\right) | \mathbf{h_i}\right) \quad (1)$$



(a) Inference with token fusion for sentence classification

(b) Inference with token fusion for token classification

Figure 2: A comparison of token fusion in different classification tasks. For sentence classification, the fusion predictor decides whether adjacent non-[CLS] representations can be fused. The cloud server can still make a prediction based on [CLS]. For token classification, fusion predictors predict the label probability distribution of each token at each layer. We collect tokens with low-uncertainty distribution and save their labels locally. Token fusion only occurs in these confident tokens to ensure unlabeled tokens can get predictions in the cloud server. In this way, the fusion predictors can hide private tokens without affecting task completion.

where V maps the plain text to the vocabulary space. This attack can invalidate the privacy-preserving methods in the training phase because attackers can directly establish the relationship between the privatized token representation and its raw word.

3 Methodology

In this section, we present TextFusion, which adopts dynamic token fusion to preserve privacy directly in the inference phase. We use an example to illustrate our idea. As shown in Figure 2 (b), when processing the plain text "*John lives in 3rd street*", the local encoder equipped with TextFusion takes five raw words as input but only outputs two token representations (special tokens like [CLS] are not counted). The token fusion naturally hides the private words *John* and *3rd* and breaks the one-to-one relation between token representations and raw words. Then these two privatized token representations are shared with third parties. The incomplete representation sequence still keeps sufficient information for downstream tasks but hinders privacy attackers from performing text reconstruction attacks.

3.1 Model Overview

TextFusion keeps the basic architecture as PLM. Differently, we train a fusion predictor to determine which token should be fused, based on the user's config and token representation itself. These suitable token representations are fused in the privacy-preserving layer, usually the last layer of local encoder. To further generate private representation, we conduct misleading training to make both fused and unfused token representations not similar to related words. The details are shown in the following subsections.

3.2 Token Fusion Mechanism

There are two critical points of token fusion mechanism. First, the fusion strategy should be dynamic and hard to be broken by an attacker. Second, the fusion should have as little influence as possible on the target NLP task. We first describe how a fusion predictor works in sentence classification and extend it to token classification.

Sentence Classification. Given a plain text $X = [x_1, x_2, ..., x_n]$, the goal of sentence classification is to assign a label y to X. Generally, PLM prepends a special token [CLS] to the input text and takes the final representation of [CLS] for classification. Fusion of non-CLS token representations will not make sentence classification unworkable. Therefore, we apply a simple but effective fusion strategy for sentence classification, which fuses adjacent token representations directly.

Given the representations $\mathbf{H} = [\mathbf{h}_1, ..., \mathbf{h}_n]$ in the target layer, we consider every two adjacent representations as a region. The fusion predictor determines whether the region can be fused to $\hat{\mathbf{h}}_\mathbf{i} = \sum_{2i}^{2i+1} \mathbf{h}_i$ by predicting a score:

$$s(\hat{\mathbf{h}}_i) = \sigma(\mathbf{W}_2(\mathrm{Tanh}(\mathbf{W}_1\hat{\mathbf{h}}_\mathrm{i} + \mathrm{b}_1)) + \mathrm{b}_2), \tag{2}$$

where i means the i-th region in the sequence, $\sigma(.)$ is sigmoid activation function to ensure the score is between 0 and 1, $\{\mathbf{W}_1, \mathbf{W}_2, b_1, b_2\}$ are trainable parameters of fusion predictor and $s(.) \in \mathbb{R}$. We apply the score as the weight of the fused representation and get final fused representation $\mathbf{h}' = s(\hat{\mathbf{h}}) * \hat{\mathbf{h}}$ for subsequent inference.

Token Classification. The goal of token classification is to predict the labels $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_n]$ with the equal length n as plain text X. The token fusion reduces the sequence length, which makes it impossible to get predictions for fused tokens. As the result, we cannot perform token fusion directly in token classification tasks. To avoid this problem, we draw inspiration from early exiting mechanism (Xin et al., 2020; Li et al., 2021), which assumes that the representations at an earlier layer of PLMs are adequate to make a correct prediction. We can only fuse the token representations that are confident to get labels locally, leaving unlabeled token representations to the cloud server. In this way, token fusion is still be done dynamically and does not affect the completion of token classification.

In token classification, the fusion predictor aims to predict whether the current token representation can get a correct label. For the l-th layer of local encoder, we calculate the entropy $s_i^{(l)}$ for each representation $\mathbf{h}_i^{(l)}$ to indicate how confident it is to get the label:

$$\mathbf{p}_i^{(l)} = \mathrm{Softmax}(\mathbf{W}\mathbf{h}_\mathrm{i}^{(\mathrm{l})}), \tag{3}$$

$$s_i^{(l)} = \frac{-\mathbf{p}_i^{(l)} * \log \mathbf{p}_i^{(l)}}{\log C}, \tag{4}$$

where $\mathbf{W}$ is the parameters of a fusion predictor at l-th layer, C is the number of label set and $\mathbf{p}_i^{(l)} \in \mathbb{R}^C$ is the label probability distribution for the i-th token. The smaller the $s_i^{(l)}$ is, the more confident for $\mathbf{h}_i^{(l)}$ to make a correct prediction.

To collect as many predictions as possible, we insert a fusion predictor into each layer of local encoder and train them with golden labels:

$$\mathcal{L}_{fp} = \frac{1}{L} \sum_{l=1}^{L} \sum_{i=1}^{n} CE(\mathbf{y}_i, \mathbf{p}_i^{(l)}), \tag{5}$$

where L is the number of fusion predictor, CE is the cross-entropy loss function.

We identify confident token representations with uncertainty below a predefined threshold at each layer and record their predictions. The confident and adjacent token representations will be fused into one in the target layer and other tokens can get predictions in the cloud server. As such, we can still preserve privacy in token classification while getting predictions of all tokens.

3.3 Misleading Training

Achilles' Heel of Token Fusion. With the obstacle of token fusion, a privacy attacker cannot get ideal training data, i.e., the one-to-one token representation and its original word. This makes it difficult for an attacker to train a specific text reconstruction model for TextFusion. Despite the privacy of the fused token being guaranteed, the unfused token representations are still at risk of attack. To further protect privacy, we need to promote irrelevance in the token representation with respect to the original word.

Misleading Loss. The representations we deal with come from very shallow layers of the PLM and usually represent the foundational features of the data (Jawahar et al., 2019). A good shallow representation is decisive for the subsequent model inference, as the result, unconstrained perturbations on these representations can be fatal to performance. Therefore, we propose to mislead the attacker by making the token representation more predictable to a similar but different word. Specifically, we calculate the Euclidean distance between each representation and the embedding matrix in the target layer, and select the word with the closest distance and not included in the input as the misleading word. We involve training a secondary objective to predict the misleading words for both fused and unfused token representations:

$$\mathcal{L}_{mislead} = -\sum_{i=1}^{m} \frac{1}{k} \sum_{j=1}^{k} P\left(x_j | \mathbf{h}_i'; \theta_{mis}\right), \tag{6}$$

where x_j is the misleading word, m is the sequence length, k means that $\mathbf{h}_i'$ is a fusion of m token representations ($\mathbf{h}_i'$ is a unfused token when $k = 1$). θ_{mis} is the parameters of a linear classifier, which is initialized by embedding matrix. This objective constrains the perturbation direction through the Euclidean distance, which is helpful for performance and misleads the attacker effectively.

3.4 Training and Inference

Training consists of two stages: performance guarantee and privacy guarantee. In the first stage, we insert fusion predictors into the fine-tuned PLM and train them jointly on the target task. In this stage, we use the soft fusion trick, which fuses all-region in sentence classification and does not fuse tokens in token classification. The task objective will guide fusion predictors to make decisions beneficial to classification performance. In the second stage, we enable the real token fusion, based on a threshold to decide which words are fused and which remain unchanged in the target layer. We train the model and fusion predictor with the task and misleading loss and set multiple fusion thresholds to reduce the gap between training and inference. The sandwich rule (Yu and Huang, 2019), ℓ_1 regularization (Zheng et al., 2022) and window-based uncertainty (Li et al., 2021) are used to stabilize training.

At the inference time, the trained PLM is split into a local encoder for a user and a cloud encoder for a cloud service. Based on fusion predictors and a threshold, the token representations are fused dynamically in the last layer of the local encoder. Then the reorganized representations with privacy guarantee are uploaded to the cloud server.

4 Experiment Setup

4.1 Datasets

We conduct our experiments on four benchmarks, covering three widely used text classification tasks and two common languages. In sentence classification, we choose **SST-2** (Socher et al., 2013), a sentiment analysis dataset for single-sentence classification and **MRPC** (Dolan and Brockett, 2005) , a paraphrase dataset for sentence-pair classification. In token classification, we choose two name entity recognition (NER) datasets, including **CoNLL2003** (Tjong Kim Sang and De Meulder, 2003) for English and **resume** (Zhang and Yang, 2018) for Chinese. The statistics of datasets are shown in Appendix A.1.

4.2 Baselines

For a thorough comparison, we select three privacy-preserving methods, including noise injection, adversarial training and data augmentation. The standard fine-tuning is also used to show the privacy risk in the inference phase.

Fine-tune (Devlin et al., 2018) follows the standard fine-tuning process without privacy guarantee.

DPNR (Lyu et al., 2020b) utilizes differential privacy to provide privacy guarantee, and masks words via dropout to further enhance privacy.

CAPE (Plant et al., 2021) injects calibrated Laplace noise to perturb representations and adopts adversarial learning to reduce private variables.

SanText+ (Yue et al., 2021) replace the sensitive words in plain text with other words in vocabulary. The word selection is based on differential privacy and word frequency.

4.3 Privacy Attack Methods

Three text reconstruction attack methods are used to recover the raw words from intermediate token representations in the inference phase.

KNN (Qu et al., 2021) assumes attackers are only accessible to representations from users. It computes the Euclidean distance between each token representation and public word embedding matrix. The word of the nearest distance is selected as the reconstruction results.

InvBert (Höhmann et al., 2021) assumes attackers have access to query the public local encoder using plain text and get the corresponding representations. Then they can train a text reconstruction model to directly recover a token representation to the raw word.

MLC (Song and Raghunathan, 2020) follows the same access as InvBert but does not train a one-to-one reconstruction model. It builds a multi-label classification (MLC) model for the whole sequence, not for each representation.

4.4 Evaluation Metrics

Privacy metrics evaluate how much information can be recovered from representations, the higher the metric, the more privacy is leaked. A good approach should be high in performance metrics, but low in privacy metrics. We list the privacy metrics adopted in our experiments as follows and show details in the Appendix A.3.

Token-Hit is a coarse-grained privacy metric that measures the accuracy of recovered words. It does not consider the word order, treats raw and recovered words as two sets, and calculates the percentage of recovered words in the raw words.

Rouge-L (Lin, 2004) is used as a fine-grained privacy metric to measure the readability of the

Dataset	Methods	Task ↑	KNN Attack			InvBert Attack			MLC Attack
			Token-hit ↓	Rouge-L ↓	Ent-hit ↓	Token-hit ↓	Rouge-L ↓	Ent-hit ↓	Token-hit ↓
SST-2	Fine-tune	92.20	80.94	89.37	–	100	100	–	54.64
	DPNR	89.56	2.89	0.69	–	94.66	69.74	–	53.79
	CAPE	87.84	1.36	0.00	–	12.31	9.29	–	**19.01**
	SanText+	82.99	62.64	16.91	–	72.33	20.6	–	49.41
	TextFusion	**90.36**	**0.00**	**0.00**	–	**2.58**	**0.06**	–	20.23
MRPC	Fine-tune	90.06	81.80	36.09	–	100	68.87	–	45.26
	DPNR	82.24	2.00	0.00	–	79.80	58.32	–	18.04
	CAPE	82.05	1.95	0.14	–	77.66	55.62	–	17.33
	SanText+	81.56	75.96	23.93	–	81.40	36.04	–	49.57
	TextFusion	**88.17**	**0.00**	**0.00**	–	**1.56**	**0.72**	–	**9.36**
CoNLL2003	Fine-tune	91.42	87.31	95.92	95.94	100	96.92	100	44.46
	DPNR	82.80	1.93	0.66	1.79	78.66	54.01	87.11	23.52
	CAPE	81.08	1.27	0.00	1.48	36.16	22.27	50.72	28.54
	SanText+	63.48	67.35	14.6	13.4	76.04	13.79	16.27	59.72
	TextFusion	**89.78**	**0.01**	**0.00**	**0.01**	**2.05**	**1.09**	**0.00**	**6.07**
Resume	Fine-tune	94.23	91.59	48.42	76.63	99.97	47.49	98.33	44.74
	DPNR	87.86	2.97	5.64	0.00	90.97	46.53	70.45	39.76
	CAPE	85.41	**0.00**	0.00	0.00	24.38	18.35	5.56	32.91
	SanText+	89.80	72.00	29.32	21.23	92.08	23.46	70.61	80.81
	TextFusion	**93.10**	0.02	**0.00**	**0.00**	**3.04**	**0.00**	**0.00**	**10.05**

Table 1: Main results of TextFusion and baselines on four datasets under different text reconstruction attacks. **Task** represents the metric of downstream task, accuracy for SST-2 while F1 for MRPC, CoNLL2003 and Resume, respectively. ↑ means higher is better. **Token-hit**, **Rouge-L** and **Ent-hit** represent the metric of privacy leakage, ↓ means lower is better. The best results among all methods except Fine-tune are marked in bold. Ent-hit is not support on sentence-level task, thus filled with – in SST-2 and MRPC.

recovered text. It evaluates the similarity between the recovered text and the raw text based on the longest common subsequence.

Ent-hit is a NER-specific privacy metric to measure the leakage of key information such as name and location. It achieves this by calculating the percentage of recovered entities.

4.5 Implementation Details

To simulate cloud inference scenarios, we split bert-base series models (Devlin et al., 2018) to local encoder with 3 layers and server encoder with 9 layers. The final representations of local encoder will be shared with third parties such as service providers and thus under the risk of privacy leakage. All the privacy-preserving (except SanText+) and privacy attack baselines are conducted in the output representations of local encoder. SanText+ directly replaces the raw text. For KNN, we use the PLM embedding without fine-tuning to calculate the Euclidean distance. For InvBert and MLC, we use the training set to query the local encoder and get the representations to train a bert-base model. Since attackers cannot train a InvBert specifically for TextFusion, we use the InvBert trained on fine-tuned PLM to attack TextFusion.

AdamW optimization algorithm and linear decaying schedule are used for all methods. We conduct a comprehensive hyper-parameter search to reproduce the baselines in our setting. For all baselines, we save the model with the best performance on the validation set. More details about hyper-parameters of baselines and TextFusion are shown in the Appendix A.2.

5 Results and Analysis

5.1 Main Results

Overall Comparison. We demonstrate the experimental results of TextFusion and the baselines across 4 datasets in Table 1. From the table, we can observe that (1) The proposed TextFusion substantially outperforms most baselines in task performance and privacy metrics under different attacks. (2) The privacy-preserving baselines effectively protect privacy under the KNN attack, but cannot defend InvBert attack as effectively as TextFusion. This is not surprising, since they do not **privatize token representations during inference**. An attacker can still train a powerful text reconstruction model to recover the raw text. Our method hides the token representation dynamically during inference, which hinders the training of text reconstruction model. (3) The task performance of privacy-preserving baselines drops substantially. We speculate that this is due to the **vulnerability of the shallow representation**. Take adversarial training as an example, we perform adversarial training for all token representations in the 3rd layer, while supervising the main task with the first token representation from the 12th layer for sentence classification tasks. The huge gap between these two objectives makes it difficult to achieve a balance, which eventually leads to the destruction of the shallow representation and thus affects the task performance. TextFusion uses misleading training to mitigate this problem and achieves better performance.

Detailed Comparison. From Table 1 we can find that: (1) Equipped with adversarial training, CAPE generally achieved better results under InvBert's attack than DPNR which only injects noise. This indicates that adversarial training can provide a certain degree of resistance to InvBert's attack. (2) From the unsatisfactory results of SanText+, we speculate that even if we replace the original words with perturbated ones, they may still be recovered by the attacker from the intermediate representation since the **one-to-one relation still exist between the replaced words and the original ones**. (3) MLC takes the entire representation sequence as input and performs multilabel classification to identify the raw words contained in the sequence, which does not require one-to-one token representation and original word as training data. But experimental results show that TextFusion can still protect privacy under such attack, which indicates that **incomplete representation sequences can impede attackers to train a powerful reconstruction model**.

5.2 Ablation Study

In this section, we conduct a series of experiments to verify the effectiveness of the two major components of TextFusion.

How does token fusion affect the results? First, we demonstrate the privacy-preserving capability of the token fusion mechanism. We attack our model before the misleading training stage under different token fusion ratios. Results in Figure 3 show that the proposed token fusion mechanism can substantially reduce the Token-hit rate with the increase of the token fusion ratio, suggesting that the fusion operation is effective in preserving privacy. Note that the task performance is unaffected even if we increase the fusion ratio in text classification. However, in token classification, the fusion rate has a greater impact on the performance because the early exiting mechanism makes unconfident tokens exit at shallow layers, leading to wrong predictions.

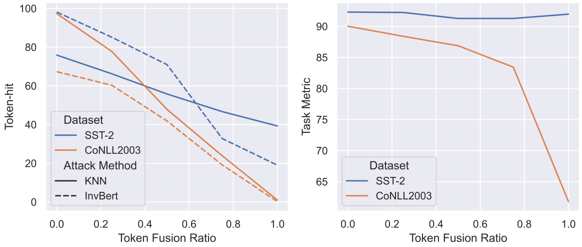


Figure 3: Privacy and performance under different token fusion ratio. The task metrics for SST-2 and CoNLL2003 are accuracy and F1, respectively. The lower Token-hit is, the more privacy is protected.

Is misleading training necessary? Another question is whether misleading training plays a dominant role in privacy protection rather than token fusion. We emphasize that a major effect of token fusion hinders the training of the reconstruction model. Without token fusion, the attacker can get ideal data to train a powerful text reconstruction model. To verify this, we apply the misleading training to the standard fine-tuning and use InvBert to attack this model. We also use the InvBert, trained on misleading data, to attack the complete TextFusion model. From the results in Table 2, we find that neither can misleading training alone defend against InvBert attack, nor can token fusion protects unfused tokens. Only when combined with both two components, our proposed TextFusion can defend against all attacks.

Dataset	Method	Task	KNN	InvBert
SST-2	Fine-tune	**92.20**	80.94	100
	+ Misleading	92.02	00.00	100
	+ Token fusion	91.97	39.29	19.02
	+ TextFusion	90.36	**00.00**	**02.58**

Table 2: Ablation Study on TextFusion. Task's metric is accuracy, KNN and InvBert's metric is Token-hit.

5.3 Case Study

We take one example from SST-2 to show the attack results of KNN and InvBert on the fine-tuned model and TextFusion. As shown in Figure 4, we can observe that the recovered words are basically irrelevant to the input text, indicating that our

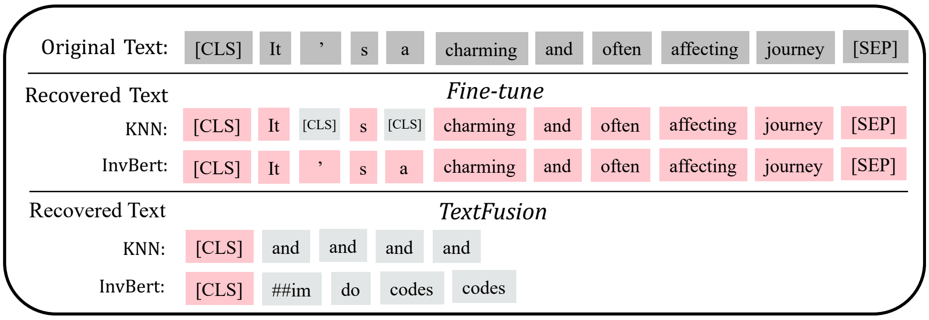

Original Text: [CLS] It ' s a charming and often affecting journey [SEP]
Recovered Text
Fine-tune
KNN: [CLS] It [CLS] s [CLS] charming and often affecting journey [SEP]
InvBert: [CLS] It ' s a charming and often affecting journey [SEP]
Recovered Text
TextFusion
KNN: [CLS] and and and and
InvBert: [CLS] ##im do codes codes

Figure 4: Illustration of privacy attack results on TextFusion for a input example from SST-2. The input-related words are highlighted as red.

method can effectively remove privacy attributes. Besides, TextFusion reduces the sequence length, as a result, even if an attacker could recover the original word, its position cannot be aligned with the original input because word fusion occurs, which not only makes the text less readable, but also makes it difficult for attackers to obtain high-quality training data.

5.4 Privacy in Different Layers

In our experiments, the local encoder consists of 3 transformer layers. However, different users have different computing resources, so it is necessary to verify the effectiveness of our method in the lower layer. We extend TextFusion to the first and second layer and show the results of all three layers. Experimental restults in Table 3 show that the TextFusion can still protect privacy at shallower layers. However, fusing and misleading representations at these layers inevitably degrade performance, there is a trade-off between the degree of privacy protection and performance.

Method	Layer	Task	KNN	InvBert
Fine-tune	layer 1	92.20	93.47	100
	layer 2		85.69	100
	layer 3		80.94	100
TextFusion	layer 1	89.22	2.19	2.45
	layer 2	90.02	0.00	2.08
	layer 3	90.36	0.00	2.58

Table 3: Task performance and privacy using TextFusion at different layers on the SST-2.

6 Related Work

Cloud-based PLM inference enables users who lack computing resources to benefit from large models (Pais et al., 2022) by uploading data to the cloud. Many efforts have been made to keep this process from leaking user privacy from uploaded data. A kind of method reduce the private information in representation during training, such as differential privacy (Habernal, 2021; Hoory et al., 2021), noise injection (Xu et al., 2020; Ponomareva et al., 2022) and adversarial training (Coavoux et al., 2018; Plant et al., 2021). These methods do not fully protect the privacy under the text reconstruction attack during inference(Song and Raghunathan, 2020) . Homomorphic encryption (Feng et al., 2020; Chen et al., 2022a) encrypts the computation of the model, it introduces time-consuming computation and additional computational cost, which is contrary to our low computational resource scenario, and is not considered by us. Yue et al. (2021); Xu et al. (2020) propose to replace the sensitive words in text, but the reasonableness of the replaced words is difficult to guarantee, especially in token classification. These methods cannot preserve inference privacy effectively and efficiently.

Token reduction is similar to token fusion but with different motivations. These works use attention (Goyal et al., 2020), gradients (Modarressi et al., 2022) and reinforcement learning (Ye et al., 2021) to remove redundant representations to accelerate inference. We leave the combination of token fusion and token reduction to future work.

7 Conclusion

In this paper, we propose TextFusion, a privacy-preserving approach for on-cloud pre-trained model inference. The key idea of TextFusion is to fuse the token representations during local inference and shares incomplete and perturbed token representations to the third parties. These reorganized representations make it hard for privacy attackers to recover the token representations back to raw words that contain the private user information. To this end, we train a fusion predictor to fuse token dynamically and employ the misleading training to mislead the attacker in both fused and unfused representations. The experimental results on four sentence and token classification datasets show that our method can protect privacy effectively while achieving the comparable performance with fine-tuning.

8 Limitations

Although the proposed token fusion strategy is simple and effective, it's better to propose a unified fusion method for both token and sentence classification tasks. Besides, token fusion relies on getting the predictions for confident representations on early layer for token classification. It will limit the applications of TextFusion when a very large fusion ratio is required. These two problems will be explored in our future work.

Acknowledgements

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by National Natural Science Foundation of China (No. 61906176, 62206057, 62076069, 61976056), Program of Shanghai Academic Research Leader (No. 22XD1401100), and Beijing Academy of Artificial Intelligence (BAAI). This work was sponsored by Program of Shanghai Academic Research Leader, Grant No. 22XD1401100 and CCF-Tencent Open Fund.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

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

Tianyu Chen, Hangbo Bao, Shaohan Huang, Li Dong, Binxing Jiao, Daxin Jiang, Haoyi Zhou, Jianxin Li, and Furu Wei. 2022b. THE-X: Privacy-preserving transformer inference with homomorphic encryption. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3510–3520, Dublin, Ireland. Association for Computational Linguistics.

Jianfeng Chi, Emmanuel Owusu, Xuwang Yin, Tong Yu, William Chan, Patrick Tague, and Yuan Tian. 2018. Privacy partitioning: Protecting user data during the deep learning inference phase. *arXiv preprint arXiv:1812.02863*.

Maximin Coavoux, Shashi Narayan, and Shay B Cohen. 2018. Privacy-preserving neural representations of text. *arXiv preprint arXiv:1808.09408*.

ROBERT DALE. 2015. Nlp meets the cloud. *Natural Language Engineering*, 21(4):653–659.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Bo Feng, Qian Lou, Lei Jiang, and Geoffrey C Fox. 2020. Cryptogru: Low latency privacy-preserving text analysis with gru. *arXiv preprint arXiv:2010.11796*.

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.

Saurabh Goyal, Anamitra Roy Choudhury, Saurabh M Raje, Venkatesan T Chakaravarthy, Yogish Sabharwal, and Ashish Verma. 2020. Power-bert: accelerating bert inference via progressive word-vector elimination. In *Proceedings of the 37th International Conference on Machine Learning*, pages 3690–3699.

Ivan Habernal. 2021. When differential privacy meets nlp: The devil is in the detail. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1522–1528.

Johannes Höhmann, Achim Rettinger, and Kai Kugler. 2021. Invbert: Text reconstruction from contextualized embeddings used for derived text formats of literary works. *arXiv preprint arXiv:2109.10104*.

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

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Marija Jegorova, Chaitanya Kaul, Charlie Mayor, Alison Q O'Neil, Alexander Weir, Roderick Murray-Smith, and Sotirios A Tsaftaris. 2021. Survey: Leakage and privacy at inference time. *arXiv preprint arXiv:2107.01614*.

Xiaonan Li, Yunfan Shao, Tianxiang Sun, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2021. Accelerating bert inference for sequence labeling via early-exit. *arXiv preprint arXiv:2105.13878*.

Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018. Towards robust and privacy-preserving text representations. *arXiv preprint arXiv:1805.06093*.

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

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Lingjuan Lyu, Xuanli He, and Yitong Li. 2020a. Differentially private representation for nlp: Formal guarantee and an empirical study on privacy and fairness. *arXiv preprint arXiv:2010.01285*.

Lingjuan Lyu, Xuanli He, and Yitong Li. 2020b. Differentially private representation for NLP: Formal guarantee and an empirical study on privacy and fairness. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2355–2365, Online. Association for Computational Linguistics.

Ali Modarressi, Hosein Mohebbi, and Mohammad Taher Pilehvar. 2022. Adapler: Speeding up inference by adaptive length reduction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1–15.

Sebastião Pais, João Cordeiro, and M Luqman Jamil. 2022. Nlp-based platform as a service: a brief review. *Journal of Big Data*, 9(1):1–26.

Xudong Pan, Mi Zhang, Shouling Ji, and Min Yang. 2020. Privacy risks of general-purpose language models. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1314–1331. IEEE.

Richard Plant, Dimitra Gkatzia, and Valerio Giuffrida. 2021. CAPE: Context-aware private embeddings for private language learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7970–7978, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Natalia Ponomareva, Jasmijn Bastings, and Sergei Vassilvitskii. 2022. Training text-to-text transformers with privacy guarantees. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2182–2193.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897.

Chen Qu, Weize Kong, Liu Yang, Mingyang Zhang, Michael Bendersky, and Marc Najork. 2021. Natural language understanding with privacy-preserving bert. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 1488–1497.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Congzheng Song and Ananth Raghunathan. 2020. Information leakage in embedding models. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 377–390.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. Deebert: Dynamic early exiting for accelerating bert inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2246–2251.

Zekun Xu, Abhinav Aggarwal, Oluwaseyi Feyisetan, and Nathanael Teissier. 2020. A differentially private text perturbation method using a regularized mahalanobis metric. *arXiv preprint arXiv:2010.11947*.

Deming Ye, Yankai Lin, Yufei Huang, and Maosong Sun. 2021. Tr-bert: Dynamic token reduction for accelerating bert inference. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5798–5809.

Jiahui Yu and Thomas S Huang. 2019. Universally slimmable networks and improved training techniques. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1803–1811.

Xiang Yue, Minxin Du, Tianhao Wang, Yaliang Li, Huan Sun, and Sherman S. M. Chow. 2021. Differential privacy for text analytics via natural text sanitization. In *Findings, ACL-IJCNLP 2021*.

Xiaoyu Zhang, Chao Chen, Yi Xie, Xiaofeng Chen, Jun Zhang, and Yang Xiang. 2021. Privacy inference attacks and defenses in cloud-based deep neural network: A survey. *arXiv preprint arXiv:2105.06300*.

Yue Zhang and Jie Yang. 2018. Chinese NER using lattice LSTM. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1554–1564, Melbourne, Australia. Association for Computational Linguistics.

Rui Zheng, Bao Rong, Yuhao Zhou, Di Liang, Sirui Wang, Wei Wu, Tao Gui, Qi Zhang, and Xuanjing Huang. 2022. Robust lottery tickets for pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2211–2224, Dublin, Ireland. Association for Computational Linguistics.

A Appendix

A.1 Dataset Statistic

We follow the official dataset split for CoNLL2003 and resume. The test sets for SST-2 and MRPC are not publicly available, so we use the original validation set directly as the test set. The statistics of four datasets used in our experiments is shown in Table 4.

Category	Dataset	#Train	#Test	Labels
Single-sentence	SST-2	67k	0.9k	2
Sentence-pair	MRPC	3.7k	0.4k	3
Token	CoNLL2003	51.5k	46.7k	9
Token	resume	124.1k	15.1k	17

Table 4: Statistics of the datasets. For SST-2 and MRPC, # means the number of sentences. For CoNLL2003 and resume, # means the number of tokens.

A.2 Hyperparameters

In this section, we show the hyperparameters used for the baselines and how we search these hyperparameters. As for TextFusion, we search the learning rate from 1e-5 and 5e-5, the misleading loss weights is 0.05. In the second stage, the thresholds were randomly selected from [0.1, 0.2, 0.3] for the token classification task and from [0,0.5,1] for the sentence classification task. For all privacy-preserving baselines, we train 10 epochs and search the learning rate from [1e-3, 1e-4, 1e-5, 1e-6]. For CAPE (Plant et al., 2021), we search the adversarial training weights λ from [0.001, 0.01, 0.1, 1, 5] and noise rate ϵ from [0.01, 0.1, 0.5, 1, 5]. For DPNR (Lyu et al., 2020b), we search the noise rate ϵ from [0.01, 0.1, 0.5, 1, 5] and the word dropout rate μ from [0,0.1, 0.3, 0.5]. For SanText+(Yue et al., 2021), we set the privacy parameter ϵ as 12 to maximize privacy protection performance. As for the InvBert and MLC, we search the learning rate from [1e-4, 1e-5, 1e-6]. We take the closest word as the attack result of KNN, the predicted word with the highest probability as the attack result of InvBert (Höhmann et al., 2021), and the word with prediction probability greater than 0.5 as the prediction result of MLC (Song and Raghunathan, 2020). Our experiments are conducted on NVIDIA GeForce RTX 2080 TI.

A.3 Privacy Metrics

As stated in Section 4.4, we adapt three metrics to evaluate degree of privacy leakage: Token-hit, Ent-hit and Rouge-L. The Rouge-L (Lin, 2004) is a widely used text generation metric, so we do not describe it here. We focus on Token-hit and Ent-hit and give a formulaic form to them.

Token-hit is a common privacy metric that measures the accuracy of recovered words. Given a original input text $X^{ori} = [x_1^{ori}, ..., x_{n_{ori}}^{ori}]$ with length n_{ori}, we convert it to a set $S^{ori} = \{s_1^{ori}, ..., s_{m_{ori}}^{ori}\}$ with m_{ori} different words. The attacker use the intermediate representations to get recovered text $X^{rec} = [x_1^{rec}, ..., x_{n_{rec}}^{rec}]$, we also convert it to a set $S^{rec} = \{x_1^{rec}, ..., x_{m_{rec}}^{rec}\}$ with m_{rec} different words. The Token-hit calculate the percentage of words in set S^{rec} to the words in set S^{ori}, which can be formulated as follow:

$$\text{Token-hit} = \frac{|S^{rec} \cap S^{ori}|}{|S^{ori}|}, \tag{7}$$

where $|.|$ means the length of the set. Suppose the $|S^{rec} \cap S^{ori}|$ is k, where $k < min\{|S^{rec}|, |S^{ori}|\}$, the Token-hit for input text X and recovered text X^{rec} is $\frac{k}{m_{ori}}$. The advantage of Token-hit is its universality. This metric is task-independent and word order-independent, and therefore applicable to all tasks, attacks and privacy-preserving methods.

Ent-hit is a NER-specifically task metric that counts how many entities in the input text are accurately recovered. Given a original input text X^{ori} and its labels $Y = [y_1, ..., y_n]$. We take the entity words based the Y to get a entitiy set $E = e_1, ..., e_m$ where m is the number of entities. The entity may contain more than one word, only when the recovered text X^{rec} contains each word in the entity and the word order is correct can it be counted as one hit. We formulate this process as:

$$\text{Ent-hit} = \frac{\sum_{i=1}^{m} \mathbf{I}(e_i; X^{ori})}{m} \tag{8}$$

where $\mathbf{I}(e_i; X^{ori}) \rightarrow \{0, 1\}$ is the indicator function indicating whether each word of entity e_i is in X^{ori} in order. Our settings are strict because we believe that small changes in the entities may point to others and thus mislead the privacy attacker instead, and thus do not reflect the true privacy leakage.