https://doi.org/10.1007/sxxxxx-yyy-zzzz-1

RESEARCH ARTICLE

BEFI: Balanced and Efficient Federated Inference of Large Language Models

Lulu Zhang[1], Qian Tao[2]✉, Zimu Zhou[3,4], Yuanyuan Zhang[5]✉, Yuxiang Wang[1], Yongxin Tong[1,2]

1. State Key Laboratory of Complex & Critical Software Environment, School of Computer Science and Engineering, Beihang University 100191, Beijing, China
2. Beijing Advanced Innovation Center for Future Blockchain and Privacy Computing, School of Artificial Intelligence, Beihang University 100191, Beijing, China
3. Department of Data Science, City University of Hong Kong 999077, Hong Kong, China
4. City University of Hong Kong Shenzhen Research Institute 518057, Shenzhen, China
5. School of Computer Science and Artificial Intelligence, Beijing Wuzi University 101149, Beijing, China

Received month dd, yyyy; accepted month dd, yyyy

E-mail: qiantao@buaa.edu.cn; zhangyuanyuan@bwu.edu.cn.

© Higher Education Press 2026

Abstract

Large language models have shown significant performance across various tasks. This paper focuses on federated LLM inference, in which the context generated by silos requires being protected, resulting in both inefficiency and imbalance in communication overhead. To address this issue, this paper proposes BEFI, enabling Balanced and Efficient Federated LLM Inference with the equipment of carefully designed mechanisms for information sharing among silos. For important tokens, we propose a fine-grained, fixed-size cache sharing mechanism that enables direct and balanced sharing of the KV cache of these tokens. For less critical tokens, we propose a context-free intermediate states sharing mechanism, which allows for the sharing of tokens of arbitrary length with constant communication overhead. Finally, we evaluate the effectiveness of BEFI across various LLMs and privacy mechanisms, demonstrating that BEFI reduces communication overheads by up to 96.0%, while maintaining balanced communication.

Key words

Large Language Model, Inference Optimization, Federated Learning

1 Introduction

Large language models (LLMs) have demonstrated remarkable performance across a broad spectrum of text-related tasks, including translation [1, 2], text generation [3], and beyond. This success is largely credited to attention mechanism [4], which enables context-aware predictions by computing attention weights for each token based on preceding generated contexts.

In many emerging applications, LLM inference occurs in *federated* settings, where input and output contexts are distributed across multiple *data silos*. These silos collaboratively analyze data using a shared LLM, often an open-source model hosted on platforms like Hugging Face. Such scenarios arise in applications requiring *cross-silo information synthesis* such as federated medical diagnosis [5] and federated transportation prediction [6]. Due to the security requirements and privacy regulations such as GDPR [7], each silo must perform local inference to avoid exposing sensitive context to other silos.

Example 1. Consider a multi-branch company. Each branch uses a shared LLM to summarize its internal operational report and sends it to the parent company to generate aggregated insights using the same model. It is crucial that each branch's context (red portion for each branch company in Fig. 1) is generated locally. Otherwise, the generated context would be immediately exposed to the third party responsible for performing the inference.

To enable privacy-preserving federated LLM inference, a straightforward approach is to directly share information necessary for subsequent inference (*i.e.,* the KV cache) across silos, with shared information perturbed by privacy-preserving techniques such as differential privacy (DP) [8]. However, the large parameter scale of LLMs and their sequence-dependent generation paradigm present two key challenges to straightforward sharing of cached information across silos in federated LLM inference.

(1) Massive Communication Overhead. Modern LLMs typically use a KV cache to accelerate inference without compromising accuracy, albeit at the cost of increased memory usage. However, directly sharing the KV cache across silos in federated LLM inference introduces significant communication overhead due to the large volume of cached data. Table 1 presents the KV cache size compared to the model size of LLAMA-13B with various context lengths. As more tokens are gener-

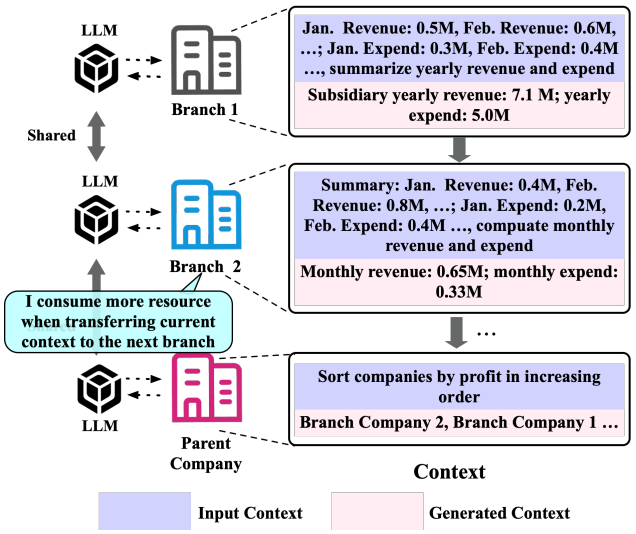


Fig. 1 An example of federated LLM inference.

ated, the KV cache size increases dramatically, leading to unacceptable communication overhead.

(2) Unbalanced Communication Cost and Noise Accumulation. Due to the sequence-dependent generation paradigm of LLMs, the size of the KV cache across silos continues to grow during inference. This results in larger communication overhead and increased noise accumulation for generating a token in the latter silos compared to the former ones, leading to the imbalanced communication overhead in federated LLM inference. As shown in Table 1, assuming each silo generates 400 tokens and the shared KV cache is perturbed using a DP mechanism, the third silo must transfer 40 GB more KV cache compared to the first silo. Furthermore, the perplexity continues to increase as DP noise is added to more KV cache. Here, the PPL score for both cases with and without different privacy when the context length is 400 remains unchanged because the privacy perturbation is added after the inference of the first silo. Thus, PPL with DP is not affected by the DP noise.

Inspired by the above discussion, this paper addresses the issues of imbalanced and inefficient communication overhead in federated LLM inference. To our knowledge, we are the first to highlight the imbalanced concerns in this scenario. Specifically, we propose BEFI, a framework for Balanced and Efficient Federated LLM Inference. To eliminate the inefficiency and imbalance in communication overhead caused by the continuously growing KV cache, BEFI introduces the Fixed-Size KV Cache Sharing mechanism. This approach limits the shared KV cache across silos to a small, fixed size, retaining only the cache of important tokens. Less important tokens are selectively replaced as subsequent silos generate additional tokens. Moreover, to reserve the contributions of other tokens, we propose the Context-Free Intermediate States Sharing mechanism, which enables the sharing of intermediate information of an arbitrary number of tokens in a constant space. This makes the intermediate states sharing mechanism an effective complement for achieving balanced federated LLM inference.

In summary, this paper makes the following key contributions:

- To the best of our knowledge, we are the first to define and study the problem of federated LLM inference with efficient and balanced communication overhead.
- We propose BEFI, a novel framework designed to ensure balanced communication overhead through efficient sharing of limited-size KV cache information.
- For other contexts, we introduce a context-free intermediate states sharing mechanism, which enables compressed intermediate states exchanged among silos, maintaining a constant communication size regardless of context length.
- Extensive experiments across three LLM families demonstrate that BEFI achieves comparable inference performance while reducing communication costs by 19.9% to 96.0%, with well-balanced communication overhead across different silos.

2 Preliminaries and Problem

2.1 Attention and KV Cache

Attention mechanisms occupy a central position in LLMs. Consider embeddings of $t-1$ tokens, represented as $\mathbf{X} \in \mathbb{R}^{(t-1)\times d}$, where d denotes the dimensionality of each token's embedding. An attention mechanism generates the t-th token as follows:

$$\begin{aligned} &\mathbf{Q} = \mathbf{X}\mathbf{W}^Q,\ \mathbf{K} = \mathbf{X}\mathbf{W}^K,\ \mathbf{V} = \mathbf{X}\mathbf{W}^V \in \mathbb{R}^{(t-1)\times h} \\ &\mathbf{A}^w = \mathtt{softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{h}}),\ \mathbf{A}^o = \mathbf{A}^w\mathbf{V} \in \mathbb{R}^{(t-1)\times h} \end{aligned} \tag{1}$$

where h represents the hidden dimension of the attention mechanism, $\mathbf{K}^T$ denotes the transpose of the key matrix $\mathbf{K}$, and $\mathtt{softmax}$ denotes the softmax function, defined as follows.

$$\mathtt{softmax}(\mathbf{X}) = \frac{\mathtt{exp}(\mathbf{X})}{\mathtt{sum}_2(\mathtt{exp}(\mathbf{X}))} \tag{2}$$

In this context, the function $\mathtt{exp}(\mathbf{X})$ denotes the application of the element-wise exponential function to the matrix $\mathbf{X}$, and the summation function $\mathtt{sum}_2(\cdot)$ is performed along the second dimension of $\mathtt{exp}(\mathbf{X})$.

The process of the attention, as described in Equ. 1, reveals that duplicated computations occur during inference of different tokens. For instance, the query matrix $\mathbf{Q} \in \mathbb{R}^{(t-1)\times h}$ used in the inference of the t-th token overlaps with the first $t-1$ rows of $\mathbf{Q}$ used in the inference of the $(t+1)$-th token. This observation motivates the technique known as KV cache [9, 10], which losslessly accelerates the inference process in large language models.

Formally, given the embeddings of the $(t-1)$-th token $\mathbf{x} \in \mathbb{R}^{1\times d}$, an attention utilizing the KV cache infers the t-th token as follows:

$$\begin{aligned} \mathbf{Q}_t = \mathbf{x}\mathbf{W}^Q, \mathbf{K}_t = \mathbf{x}\mathbf{W}^K, \mathbf{V}_t = \mathbf{x}\mathbf{W}^V \in \mathbb{R}^{1\times h} \\ \mathbf{K}^S = \mathtt{cat}(\mathbf{K}^S, \mathbf{K}_t), \mathbf{V}^S = \mathtt{cat}(\mathbf{V}^S, \mathbf{V}_t) \\ \mathbf{A}^w = \mathtt{softmax}(\frac{\mathbf{Q}_t(\mathbf{K}^S)^T}{\sqrt{h}}), \mathbf{A}^o = \mathbf{A}^w\mathbf{V}^S \end{aligned} \tag{3}$$

where $\mathbf{K}^S$ (*resp.*, $\mathbf{V}^S$) maintains the cache of key (*resp.*, value) matrix throughout the inference of different tokens, with $\mathtt{cat}$ representing the

Table 1 Ratio of KV cache size to model size, and perplexity on Wikitext dataset (LLAMA-13B, Batch Size=16, Laplacian mechanism). The Ratio column is computed as the cache size divided by the model size.

Context	Cache	Model	Ratio	PPL w/o DP	PPL w/ DP
400	19.6GB	48.9GB	40.1%	6.65	6.65
800	39.0GB	48.9GB	79.8%	7.37	822
1200	60.0GB	48.9GB	123%	7.11	3795

concatenation function. Notably, an LLM contains multiple attention layers, each of which should store an independent KV cache. Therefore, the total size of the KV cache is $O(lth)$, where l is the number of attention layers in the LLM. For time complexity, considering naive matrix multiplication as a context [11,12], an attention with KV cache takes $O(dh)$ for calculating $\mathbf{K}$ and $\mathbf{V}$, $O(th)$ for calculating $\mathbf{A}^w\mathbf{V}^S$, leading to a time complexity $O((t+d)h)$.

Takeaways. (1) Inefficiency of the Attention for Federated LLM Inference. The size of the KV cache increases dramatically as the context grows, as shown in Table 1. Sharing the whole KV cache leads to a large communication overhead. (2) Imbalanced Communication Overhead. The size of KV cache is related to the length of the context, which results in the imbalanced communication cost.

2.2 Problem Definition

This paper focuses on balanced and efficient federated LLM inference where the context is distributed across multiple silos. These silos together employ a shared LLM for collaborative analysis. Formally, consider n silos, denoted as $\{S_1, ..., S_n\}$, where each silo S_i aims to generate p_i tokens. Given the inference order $O = \langle S_1, S_2, ..., S_n \rangle$, the silos are permitted to generate context sequentially. In particular, each silo S_i generates its p_i tokens, denoted as T_i, based on the shared information $\mathcal{L}$ from the preceding silo S_{i-1}. $\mathcal{L}$ contains the information of all previously generated tokens, *i.e.,* $\cup_{1\le j<i} T_j$. After S_i generates its context T_i, it produces the shared information $\mathcal{L}_i$, which represents the context within the current silo, and merges it into $\mathcal{L}$ via $\mathcal{L} = \texttt{comb}(\mathcal{L}, \mathcal{L}_i)$, where the function `comb` merges $\mathcal{L}_i$ into $\mathcal{L}$. The updated information $\mathcal{L}$ is then transferred to the next silo for the subsequent inference.

In the context of federated LLM inference, we focus on a privacy-preserving approach that simultaneously achieves balanced and efficient communication overhead while the inference across silos.

Definition 1 (Balanced and Efficient Federated LLM Inference)**.** Given n silos $S_1, ..., S_n$, the problem requires a solution for federated LLM inference that satisfies the following requirements:

1. Balanced and Efficient Communication Among Silos: For any silo, the size of the shared information should be relatively small and independent of the number of tokens previously generated by the former silos.
2. Shared Information Privacy Protection: the shared information among silos must be protected by differential privacy, ensuring no silo's context can be leaked to other silos.

2.3 CacheShare: A Straightforward Solution

Inspired by the attention process with KV cache as illustrated in Equ. 3, a straightforward approach for federated LLM inference is to share the KV cache among silos, which we refer to as CacheShare. Formally, suppose the first p tokens have been generated by previous silos. The corresponding key and value caches of these tokens, denoted as $\mathbf{K}^{glo} \in \mathbb{R}^{p\times h}$ and $\mathbf{V}^{glo} \in \mathbb{R}^{p\times h}$ respectively, have been perturbed with DP noise and then shared to the current silo. Let $\mathbf{K}^{loc}, \mathbf{V}^{loc} \in \mathbb{R}^{(t-1)\times h}$ denote the locally generated $t-1$ tokens in the current silo. CacheShare constructs the key and value matrices $\mathbf{K}$ and $\mathbf{V}$ by replacing the second line of Equ. 3 with the following:

$$\mathbf{K} = \texttt{cat}(\mathbf{K}^{glo}, \mathbf{K}^{loc}, \mathbf{K}_t),\ \mathbf{V} = \texttt{cat}(\mathbf{V}^{glo}, \mathbf{V}^{loc}, \mathbf{V}_t) \tag{4}$$

CacheShare suffers from inefficient and imbalanced communication overhead due to the large-volume KV cache with increasing size. For the i-th silo, it must transfer the KV cache of all previously generated tokens to the subsequent silo, leading to a total communication cost of $O(lh\sum_{j\le i} p_j)$, where l denotes the number of attention layers, h is the hidden dimension, and $\sum_{j\le i} p_j$ represents the total number of tokens generated by the first i silos.

3 Methods

3.1 BEFI Overview

In general, the essence of a framework for privacy-preserving federated LLM inference lies in the design of the information to be shared, *i.e.,* $\mathcal{L}$, and how this information is utilized during the inference. In BEFI, we facilitate the sharing of two distinct types of information across data silos, based on the importance of tokens.

(1) Fixed-Size KV Cache for Important Tokens. In BEFI, a small and fixed-size number of tokens can be shared across silos by transmitting their KV cache. This information is represented as $\mathbf{K}^{glo}$ and $\mathbf{V}^{glo}$, denoting the global key matrices and value matrices, respectively. Each silo can utilize this mechanism to select and share important tokens along with their full KV cache information.

(2) Context-Free Compressed States for Remaining Tokens. For other tokens, we compress the intermediate states from the softmax function as the shared information, defined as $\mathbf{I}$. This shared information is independent of context length, thereby enabling a balanced and efficient communication cost in federated LLM inference.

As shown in Fig. 2, BEFI supports the sharing of two types of information, *i.e.,* the fixed-size KV cache for important tokens and the compressed intermediate states for remaining tokens. A silo S_i first generates local tokens based on the shared global KV cache,

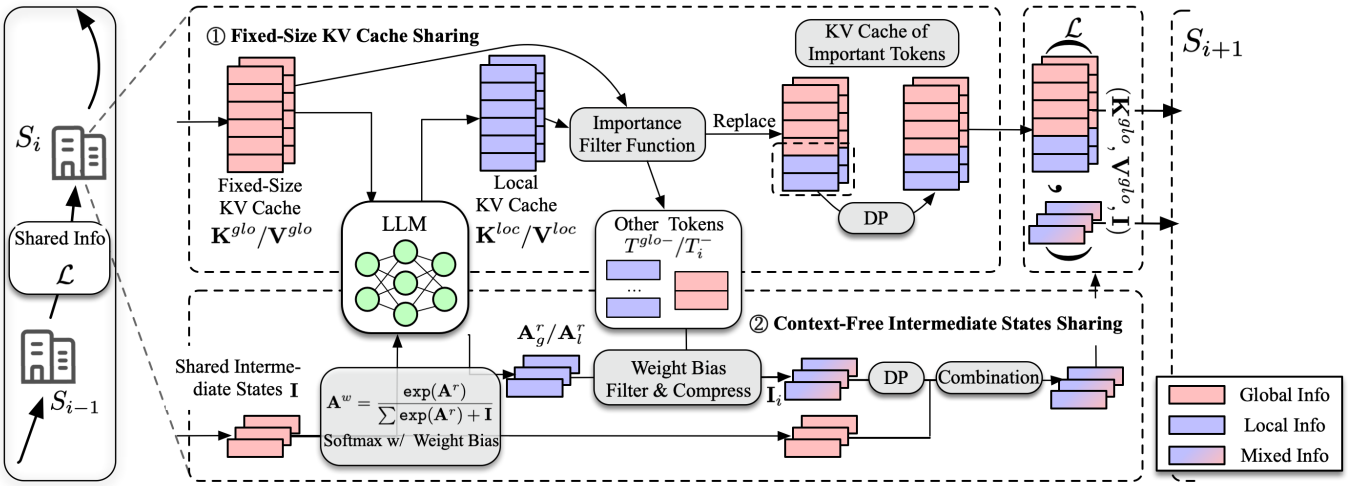


Fig. 2 Overview of BEFI Framework.

denoted as $\mathbf{K}^{glo}$ and $\mathbf{V}^{glo}$, as well as the shared intermediate states $\mathbf{I}$. After the inference, S_i filters a small set of locally important tokens. The information of these tokens is shared by replacing a part of the global KV cache with their perturbed cache (the upper part ① in Fig. 2). Meanwhile, the information of the remaining tokens is shared via the compressed intermediate states (the lower part ② in Fig. 2), independent of context length.

3.2 Fixed-Size KV Cache Sharing

Basic Idea. BEFI enables the sharing of a *fixed-size KV cache* across silos, ensuring that the communication overhead remains independent of the number of previously generated tokens. Specifically, a KV cache of fixed-size c is shared among all silos, denoted by $\mathbf{K}^{glo}$ and $\mathbf{V}^{glo}$, representing the global key and value matrices, respectively. Each time a silo completes its inference and obtains the local token set T_i, it chooses $\frac{c}{n}$ tokens from T_i and adds noise to their KV cache entries (assuming c is always divisible by n). These perturbed KV cache entries are then utilized to replace a portion of the KV cache corresponding to $\frac{c}{n}$ tokens in $\mathbf{K}^{glo}$ and $\mathbf{V}^{glo}$, forming the updated KV cache to be passed on to the subsequent silo.

Algorithm 1 illustrates pseudocode of the fixed-size KV cache sharing mechanism. For the i-th silo S_i, it receives the perturbed cache from the previous silo, denoted as $\mathbf{K}^{glo}$ and $\mathbf{V}^{glo}$, and initializes the global token set T^{glo} and local token set T^{loc}. S_i first generates local tokens using the context provided by the global cache $\mathbf{K}^{glo}$ and $\mathbf{V}^{glo}$ following Equ. 3 (lines 1-6). S_i then applies the `filter` function to select $\frac{(n-1)c}{n}$ global important tokens, denoted as T^{glo+}, and $\frac{c}{n}$ local important tokens, denoted as T_i^+ (lines 7-8). Given a token set T, `filter`(T, m) identifies m important tokens and $|T| - m$ remaining tokens based on predefined criteria. Subsequently, S_i selects globally (*resp.* locally) key cache of important tokens and unimportant tokens, denoted as $\mathbf{K}^{glo+}$(*resp.* $\mathbf{K}^{loc+}$) and $\mathbf{K}^{glo-}$(*resp.* $\mathbf{K}^{loc-}$), respectively (lines 9-10). The same filtering process is applied to value matrices (lines 11-12). Notably, we use the concatenation function `cat` as the function `comb` for merging the shared information into local information, as mentioned in Sec. 2.2. Finally, the KV cache of the selected local important tokens is perturbed and merged into the global cache (lines 13-14). The remaining tokens are instead shared via intermediate states, as detailed in Sec. 3.3.

Algorithm 1: Fixed-Size Cache Sharing Mechanism

Input: The shared information $\mathcal{L} = (\mathbf{K}^{glo}, \mathbf{V}^{glo}, \mathbf{I})$ from previous silo;

1 $\mathbf{K}^{loc}, \mathbf{V}^{loc} \leftarrow$ empty vector; $T^{glo} \leftarrow$ global token set; $T_i \leftarrow \emptyset$;
2 **for** $t = 1, 2, ..., p_i$ **do**
3 Update local cache $\mathbf{K}^{loc}, \mathbf{V}^{loc}$ via lines 1-3 of Equ. 3;
4 $\mathbf{K} \leftarrow$ `cat`$(\mathbf{K}^{glo}, \mathbf{K}^{loc})$, $\mathbf{V} \leftarrow$ `cat`$(\mathbf{V}^{glo}, \mathbf{V}^{loc})$;
5 Infer next token based on $\mathbf{K}$, $\mathbf{V}$, and line 4 of Equ. 3;
  Append the inferred token to T_i;
6 $T^{glo+}, T^{glo-} \leftarrow$ `filter`$(T^{glo}, \frac{(n-1)c}{n})$;
7 $T_i^+, T_i^- \leftarrow$ `filter`$(T_i, \frac{c}{n})$;
8 $\mathbf{K}^{glo+}, \mathbf{K}^{glo-} \leftarrow \mathbf{K}^{glo}[T^{glo+}], \mathbf{K}^{glo}[T^{glo-}]$;
9 $\mathbf{K}^{loc+}, \mathbf{K}^{loc+} \leftarrow \mathbf{K}^{loc}[T_i^+], \mathbf{K}^{loc}[T_i^-]$;
10 $\mathbf{V}^{glo+}, \mathbf{V}^{glo-} \leftarrow \mathbf{V}^{glo}[T^{glo+}], \mathbf{V}^{glo}[T^{glo-}]$;
11 $\mathbf{V}^{loc+}, \mathbf{V}^{loc-} \leftarrow \mathbf{V}^{loc}[T_i^+], \mathbf{V}^{loc}[T_i^-]$;
12 $\mathbf{K}^{glo} \leftarrow$ `cat`$(\mathbf{K}^{glo+},$ `dp`$(\mathbf{K}^{loc+}))$; //`comb` for $\mathbf{K}^{glo}$
13 $\mathbf{V}^{glo} \leftarrow$ `cat`$(\mathbf{V}^{glo+},$ `dp`$(\mathbf{V}^{loc+}))$; //`comb` for $\mathbf{V}^{glo}$
14 Update $\mathbf{I}$ using T^{glo-}, T_i^-;
15 **return** $(\mathbf{K}^{glo}, \mathbf{V}^{glo}, \mathbf{I})$;

Complexity Analysis. Regarding communication cost, Algorithm 1 ensures that the size of the perturbed KV cache transmitted across silos $O(clh)$, which is independent of the context length. Here, l denotes the number of attention layers. As for time complexity, as discussed in Sec. 2.1, performing inference based on $t-1$ local tokens and p global tokens' KV cache requires $O((c+t+d)h)$ time.

Token Filtering Function. The proposed fixed-size KV cache sharing mechanism allows for different strategies for token filtering.

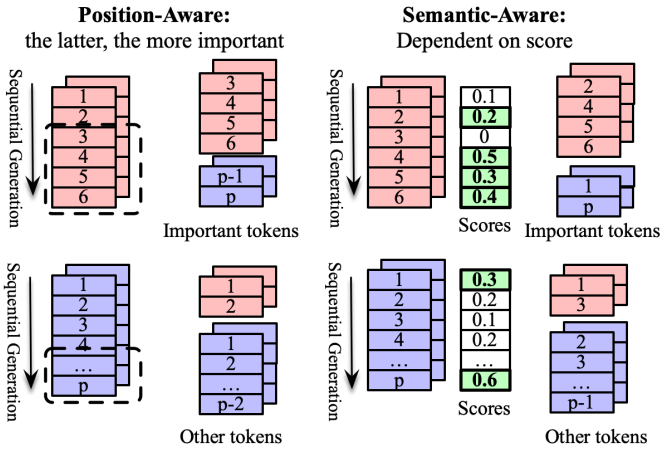


Fig. 3 Position-Aware vs. Semantic-Aware Filtering Functions (Red: global KV cache; Blue: local KV cache).

Position-Aware Filtering Function. Previous studies [13–15] indicate that recently generated tokens often play a more crucial role in generating the current token. Based on this insight, we define `filter` as a function that prioritizes the selection of more recently generated tokens. Formally, given a set of r sequentially generated tokens T and a number q, the filtering function is defined as:

$$\texttt{filter}(T, q) = T_{r-q+1:r}, T_{1:r-q}$$

where $T_{i:j}$ denotes the subset of tokens from the i-th to the j-th position in T. The subset $T_{r-q+1:r}$ consists of the q most recent tokens, designated as important tokens for sharing.

Semantic-Aware Filtering Function. Another line of research [16, 17] leverages the historical attention scores to dynamically evaluate token importance. We refer to such strategies as semantic-aware filtering functions. The function is defined as:

$$\begin{aligned}&\texttt{filter}(T, q) = \\ &\texttt{argmax}_q(\texttt{score}(\mathbf{A}^w)), \texttt{argmin}_{r-q}(\texttt{score}(\mathbf{A}^w))\end{aligned} \tag{5}$$

where $\mathbf{A}^w$ represents the attention scores as illustrated in Equ. 3. Here, $\texttt{argmax}_q$ (*resp.* $\texttt{argmin}_q$) returns tokens with the highest (*resp.* lowest) q scores, and `score` defines various importance metrics based on $\mathbf{A}^w$. To ensure efficiency, these functions typically exhibit a time complexity that scales linearly with the number of tokens.

Fig. 3 illustrates difference between the position-aware and semantic-aware filtering functions. The position-aware filtering function always chooses the latest generated tokens as important tokens, while the semantic-aware filtering function uses importance scores to filter the important tokens.

Impact of Noise. We are interested in how noise added to KV cache affects the attention output $\mathbf{A}^o$. formally, define the DP noise added to the key and value cache as $\delta\mathbf{K}$ and $\delta\mathbf{V}$, respectively. Notably, the following analysis also fits for CacheShare.

Theorem 1. Consider inference process in Equ. 3. We are interested in $\Delta\mathbf{A}^o = \widetilde{\mathbf{A}}^o - \mathbf{A}^o$, the difference between the attention outputs before and after the perturbation. Let λ_Q, λ_V, and λ_W represent the largest singular values of $\mathbf{Q}_t$, $\mathbf{V}$, and $\mathbf{A}^w$, respectively. Given $\delta\mathbf{K}_i (resp.\ \delta\mathbf{V}_i) \in \mathbb{R}^{1\times h}$ the DP noise added to the key (*resp.* value) cache of a global token $i \in T^{glo}$, we have

$$||\Delta\mathbf{A}^o||_2 \leq \lambda_W \delta_V + \frac{1}{\sqrt{h}} \lambda_Q \delta_K (\lambda_V + \delta_V)$$

We use $\delta_K = \sum_{i\in T^{glo}} ||\delta\mathbf{K}_i||$ and $\delta_V = \sum_{i\in T^{glo}} ||\delta\mathbf{V}_i||$ to denote the sum of the norm of the DP noise added to the key and value caches of the global tokens T^{glo}, respectively.

The proof is provided in Appendix A.1.1. Theorem 1 characterizes how the noise injected into shared KV cache influences $\mathbf{A}^o$. In general, the larger the number of tokens shared in the form of KV cache, the greater the upper bound on the norm of the attention output $\mathbf{A}^o$. Specifically, since the DP noise is typically sampled independently from a given distribution, we assume that the expected magnitude of each noise term is bounded above by a constant $\widetilde{d}$, *i.e.,* $\mathbb{E}[||\delta\mathbf{K}_i||_2]$ and $\mathbb{E}[||\delta\mathbf{V}_i||_2] < \widetilde{d}$.

3.3 Context-Free Intermediate States Sharing

For the remaining tokens not selected in Sec. 3.2, we require each silo to share the intermediate states of these tokens. The selection of states should satisfy the following criteria:

- *Context-Free Information Size*: The size of the shared states must be independent of the number of the remaining tokens to ensure balanced communication cost among silos.
- *Ease of Utilization*: The selected intermediate states should capture essential information of remaining tokens and be readily compatible with the inference process of LLMs.

Choice of Intermediate States. The weight distribution derived from the `softmax` function reflects the importance of preceding tokens in predicting the current token [4]. We propose recording the sum of the exponential attention weights of unimportant tokens as intermediate states. This information serves as an *attention weight bias* in the softmax function, enabling subsequent silos to incorporate the influence of unimportant tokens.

$$\begin{aligned}\textbf{Inference}:\ \mathbf{A}^r_g &= \frac{1}{\sqrt{h}}\mathbf{Q}(\mathbf{K}^{glo})^T,\ \mathbf{A}^r_l = \frac{1}{\sqrt{h}}\mathbf{Q}(\mathbf{K}^{loc})^T \\ \mathbf{A}^r &= \texttt{cat}(\mathbf{A}^r_g, \mathbf{A}^r_l) \\ \mathbf{A}^w &= \texttt{exp}(\mathbf{A}^r)/(\Sigma\ \texttt{exp}(\mathbf{A}^r) + \mathbf{I}) \\ \mathbf{A}^o &= \mathbf{A}^w\mathbf{V}\end{aligned} \tag{6}$$

$$\begin{aligned}\textbf{Update}:\ \mathbf{I}_i &= \texttt{cat}(\mathbf{A}^r_g[T^{glo-}], \mathbf{A}^r_l[T^-_i]) \\ \mathbf{I}_i &= \texttt{sum}_2(\texttt{exp}(\mathbf{I}_i));\ \mathbf{I} = \mathbf{I} + \texttt{dp}(\mathbf{I}_i)\end{aligned} \tag{7}$$

Equ. 6 illustrates the *inference stage* of our proposed context-free intermediate states sharing mechanism. During the inference stage, given the global (*resp.* local) key cache $\mathbf{K}^{glo}$ (*resp.* $\mathbf{K}^{loc}$), each silo

first calculates the global (*resp.* local) component of the input to softmax function, denoted by $\mathbf{A}_g^r$ (*resp.* $\mathbf{A}_l^r$). These two components are then concatenated to form the complete input to the softmax function for subsequent inference. Unlike the standard softmax function presented in Equ. 2, the shared intermediate states, denoted by $\mathbf{I}$, are used as a weight bias in the computation of the attention weights $\mathbf{A}^w$, as illustrated in the third line of Equ. 6. As for the *update stage*, as illustrated in Equ. 7, we have introduced how to obtain the global and local unimportant tokens, denoted by T^{glo-} and T_i^- respectively, in Sec. 3.2. The intermediate states corresponding to these tokens in $\texttt{exp}(\mathbf{A}^r)$ are first summed, then perturbed, and finally added to the global weight bias.

Complexity Analysis. In terms of communication cost, each attention layer must independently store its intermediate states, with each layer requiring $O(1)$ storage space. Consequently, the total communication cost is $O(l)$. Regarding computation cost, every silo computes the sum of embeddings in $\mathbf{A}^r$ corresponding to unimportant tokens and adds this sum to the global weight bias $\mathbf{I}$. This operation takes $O(p_i h)$ time, where p_i represents the number of tokens in current silo. Specifically, $p_i - \frac{c}{n}$ tokens are drawn from T_i^-, and the remaining $\frac{c}{n}$ tokens are obtained from T^{glo-}.

Impact of Noise. Similarly, we analyze how the DP noise added to the weight bias affects the attention output. Recall that $\mathbf{A}^r$ denotes the input to the softmax function in Equ. 6.

Theorem 2. Denote the perturbed weight bias as $\widetilde{\mathbf{I}} = \mathbf{I} + \Delta I$ and the largest singular values of $\mathbf{W}^V$ and $\mathbf{A}^w$ as λ_V and λ_W, respectively. If the perturbation ΔI satisfies $\Delta I \geq 0$ or $\sum \mathbf{A}^r \geq \ln(2|\Delta I|)$, then the difference of the attention outputs, denoted by $\Delta\mathbf{A}^o$, is bounded by

$$||\Delta\mathbf{A}^o||_2 \leq \lambda_V \lambda_W$$

The proof of Theorem 2 is provided in Appendix A.1.2. Theorem 2 establishes an upper bound on the error in the attention output introduced by the DP noise in the context-free intermediate states sharing mechanism. From this result, we observe that the noise ΔI does not directly appear in the derived error bound. Instead, it affects the condition under which the bound holds, *i.e.,* whether $\Delta I \geq 0$ or $\sum \mathbf{A}^r \geq \ln(2|\Delta I|)$. Moreover, many commonly used additive DP mechanisms, such as the Laplacian and Gaussian mechanisms, employ noise distributions around 0. The symmetry implies that the condition required for inequality in Theorem 2 to hold is satisfied with high probability.

Corollary 1. Suppose ΔI is generated by a DP mechanism that employs a noise distribution symmetric about 0, such as the Laplacian or Gaussian mechanism. Then, we have $P(||\Delta\mathbf{A}^o||_2 \leq \lambda_W \lambda_V) \geq 0.5$, where $P(\cdot)$ denotes the probability of the corresponding event.

Corollary 1 holds because the condition $\Delta I \geq 0$ in Theorem 2 is satisfied with probability at least 0.5, *i.e.,* $P(\Delta I \geq 0) \geq 0.5$, when ΔI is sampled from a distribution symmetric about 0.

3.4 Put Them Together

By integrating the fixed-size KV cache sharing and context-free intermediate states sharing mechanisms, the entire process of BEFI could be represented as shown in Algorithm 1 with the replacement of line 5 with Equ. 6 and the replacement of line 15 with Equ. 7. BEFI ensures the balanced and efficient communication overhead and the privacy of context in federated LLM inference.

Balanced and Efficient Communication Cost. The shared information $\mathcal{L}$ contains two components: $\mathbf{K}^{glo}$ and $\mathbf{V}^{glo}$ for the fixed-size cache sharing mechanism, and $\mathbf{I}$ for the intermediate states sharing mechanism. The size of both $\mathbf{K}^{glo}$ and $\mathbf{K}^{loc}$ is $O(clh)$, while $\mathbf{I}$ has a size $O(l)$. Therefore, the total size of $\mathcal{L}$ is $O(clh + l)$, which is independent of the number of previously generated tokens. In practice, a relatively small c, like 16 for 4 silos, could already ensure a considerable performance, as our evaluation shows in Sec. 4.

Privacy Protection of BEFI. In BEFI, the shared information from a silo, whether through the fixed-size cache sharing mechanism (lines 13-14 in Algorithm 1), or the context-free intermediate states sharing mechanism (Equ. 7), is perturbed before being merged into $\mathcal{L}$. This perturbation ensures that BEFI can effectively protect the privacy of the shared information from leakage.

3.5 Discussion

In this section, we discuss the adaptability of BEFI to the various distributions of p_i and other variants of attention mechanisms.

Adaptability of BEFI to Different Distributions of p_i. In Algorithm 1, the number of generated tokens p_i for the i-th silo S_i is determined by the demand of S_i. BEFI accommodates different distributions of p_i by maintaining a stable number of local tokens (*i.e.,* $\frac{c}{n}$) whose KV caches are shared with other silos. This design reflects the adaptability of BEFI to varying p_i distributions: even in the extreme case where silos simultaneously have very short and long contexts, BEFI ensures that all silos share the same number of tokens through the fixed-size KV cache sharing mechanism. Meanwhile, silos with long contexts can still share their contextual information via the context-free intermediate states sharing mechanism. This mechanism enables the sharing of context regardless of length with a constant sharing size, making it adaptable to extreme distributions of p_i.

Adaptability of BEFI to Other Variants of Attention Mechanisms. Recent efforts have proposed several variants of attention mechanisms, including sparse attentions [18–20] and linear attentions, to achieve more efficient inference. We emphasize that BEFI is orthogonal to *sparse attentions*, and thus could be adapted to them: BEFI defines how to filter those important tokens to be shared among silos, while sparse attentions determine the inference process of each silo. The proposed Context-Free Intermediate States Sharing mechanism is still adaptable because the sparse attentions also consider the softmax function as a component. Similarly, for *linear attentions*, the Fixed-Size KV Cache Sharing mechanism is adaptable. However, the Context-Free Intermediate States Sharing mechanism cannot be directly applied to

linear attentions due to the elimination of softmax function in the linear attentions. We leave this as future work.

4 Evaluation

In this section, we aim to address the following questions to validate the efficacy of BEFI. **Q1**: Does BEFI ensure balanced communication cost and comparable computation costs? **Q2**: Can BEFI achieve performance comparable to the baselines achieving federated LLM inference? **Q3**: How do variations in parameters, such as the DP privacy settings and the size of the fixed-size KV cache, affect BEFI? **Q4**: For ablation analysis, how do the proposed information sharing mechanisms affect the performance of BEFI?

4.1 Experimental Setup

Language Models and Datasets. To evaluate our framework, we select three families of language models: LLAMA-2 [21] with $7B$ and $13B$ parameter variants, MISTRAL [22] with $7B$, and FALCON [23] with $7B$. For datasets, we use Wikitext [24] and PG19 [25] to assess the performance of the proposed BEFI framework, and test the capability of BEFI on long context using LongBench [26]. We report the perplexity scores of the selected baselines and our framework on both datasets to validate the effectiveness of BEFI.

Baselines. To the best of our knowledge, no prior work has specifically addressed the problem of balanced federated LLM inference. Therefore, we compare BEFI against the following two representative frameworks: CacheShare and TokenShare.

- CacheShare: Each silo generates context by the utilization of KV cache, as described in Sec. 2.3. The entire KV cache is then perturbed and shared across silos.
- TokenShare: Silos generate context following the vanilla attention, *i.e.,* Equ. 1, after which the token embeddings are perturbed and shared. This method represents prior works that applied perturbations to input embeddings of language models, like [27–29].

All three frameworks *i.e.,*TokenShare, CacheShare, and BEFI, support the integration of different DP techniques for privacy protection. We evaluate the following two representative DP techniques:

- Laplacian Mechanism: A classic DP technique that adds independent, element-wise noise to a matrix, based on the Laplacian distribution. This mechanism is widely used to achieve ϵ-DP in scalar and vector-valued queries.
- MVG [30]: A mechanism that applies structured noise to matrices based on multivariate Gaussian distribution.

Benchmarks. We set the generated context length to 400 tokens per silo during evaluation and use 4 silos by default. For experiments with varying parameters, we evaluate different fixed KV cache sizes $c \in \{4, 8, 16, 32\}$, with 8 the default, and privacy budget $\epsilon \in \{2, 4, 8\}$ with 2 as the default. To address **Q1**, we report the communication cost per silo and computation cost per token of BEFI and the baselines. To answer **Q2**, we compare the perplexity scores of BEFI and baselines on the first 1600 concatenated tokens from Wikitext and PG19 under identical experimental settings, and partition the input question into 4 pieces for answering the question in the dataset LongBench. For **Q3**, we evaluate communication and computation costs by varying the KV cache size c, and model performance by varying c and the privacy budget ϵ. To investigate **Q4**, we separately enable the proposed fixed-size KV cache sharing and the context-free intermediate states sharing mechanisms, and compare their performance against BEFI.

DP Noise Injection. the privacy perturbation of all three algorithms are generated by either Laplacian mechanism or MVG [30]. For TokenShare, privacy perturbation is applied to the token embeddings shared among silos, without utilizing any KV cache. The inference process follows Equation 1. For CacheShare, perturbation is injected directly into the shared KV caches, as defined in Equation 3. For BEFI, noise is added to both the filtered tokens' KV cache and the shared intermediate states. In all cases, the privacy perturbation is generated using identical parameters (ϵ and δ) and without clipping.

Implementation. BEFI and the baseline methods are implemented using PyTorch and built upon the Hugging Face Transformers library. All experiments are conducted on a machine equipped with 200GB of CPU memory and an A800 GPU with 80GB of memory. Each experiment is repeated 3 times and the average is reported.

4.2 Experimental Results

4.2.1 Evaluation on Communication Cost

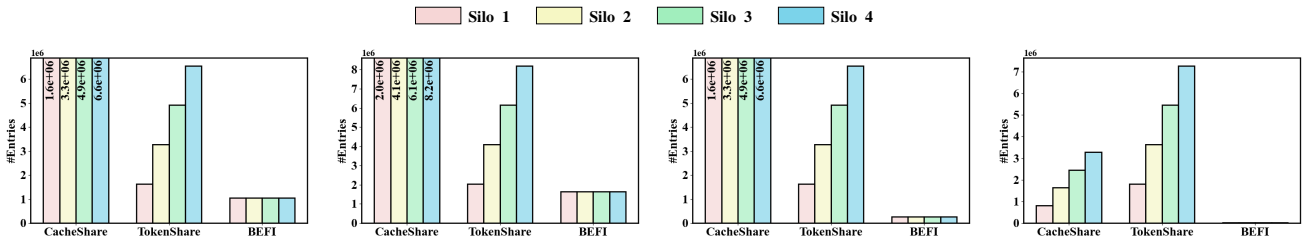
Fig. 4 presents the communication cost per silo when transmitting information to the next silo, evaluated on the Wikitext dataset across four types of LLMs. Results on the PG19 dataset are omitted due to similar trends.

Overall, BEFI reduces the communication overhead by 19.9% to as much as 96.0% (incurred for FALCON-7B). As the context length increases, both TokenShare and CacheShare incur higher communication costs in the former silos. Moreover, CacheShare results in significantly higher communication cost compared to the other methods (except for FALCON-7B), due to the sharing of the entire KV cache across silos, which is substantially larger in size. In contrast, BEFI maintains a stable and low communication cost across silos. This stability stems from the fixed-size KV cache and the consistent size of the weight bias. Notably, CacheShare incurs less overhead than TokenShare in FALCON-7B, as it employs a multi-query attention with only 1 header for key/value cache.

4.2.2 Evaluation on Computation Cost

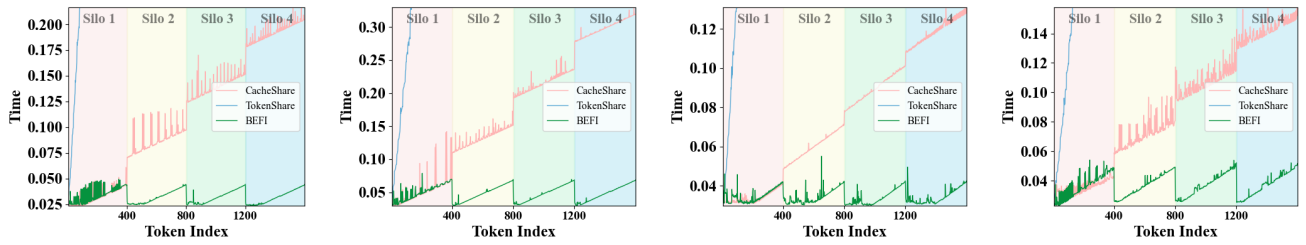
Fig. 5 presents the computation cost of inferring one token for three frameworks, using a batch size of 32 and the Laplacian mechanism on Wikitext. Results for other mechanisms are omitted, as they exhibit similar trends.

As the number of generated tokens increases, the time required to generate each token grows for both TokenShare and CacheShare. This trend reflects the unbalanced and increasing computational cost of TokenShare and CacheShare. In contrast, BEFI exhibits a stable computation time per token. This stability arises as each silo generates tokens based on a fixed-size cache, and the intermediate states sharing mechanism incurs a constant computational cost.



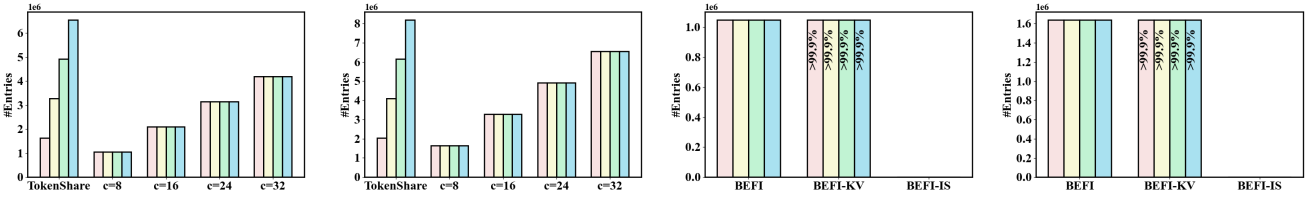
(a) LLAMA-7B **(b)** LLAMA-13B **(c)** MISTRAL-7B **(d)** FALCON-7B

Fig. 4 Computation overhead for different silos.



(a) LLAMA-7B **(b)** LLAMA-13B **(c)** MISTRAL-7B **(d)** FALCON-7B

Fig. 5 Computational cost for inferring one token.



(a) LLAMA-7B (Varying c). **(b)** LLAMA-13B (Varying c). **(c)** LLAMA-7B (Ablation Study). **(d)** LLAMA-13B (Ablation Study).

Fig. 6 Communication overhead for varying parameters and ablation study.

Table 2 Experimental Results on performance (Token: TokenShare, Cache: CacheShare). Underline indicates the best performance among the three frameworks. The ↑ column indicates the improvement ratio achieved by BEFI compared to TokenShare.

Dataset	Model	Perplexity							
		Laplacian				MVG			
		Cache	Token	BEFI	↑	Cache	Token	BEFI	↑
Wikitext	LLAMA-7B	12620	171.8	**52.2**	69.6%	8749	118.2	**54.6**	53.8%
	LLAMA-13B	9167	85.2	**79.8**	6.3%	8253	74.6	**63.9**	14.3%
	MISTRAL-7B	6393	117.2	**19.7**	83.2%	7531	93.5	**20.9**	77.6%
	FALCON-7B	536.5	**10.1**	14.0	-	756.3	**13.7**	16.8	-
	QWEN3-8B	7893	136.9	**16.2**	88.2%	7293	108.5	**18.8**	82.7%
PG19	LLAMA-7B	12814	57.5	**35.3**	38.6%	7955	116.9	**43.8**	62.5%
	LLAMA-13B	9589	49.2	**47.1**	4.1%	6411	73.9	**59.1**	20.0%
	MISTRAL-7B	6611	89.0	**15.9**	82.1%	6235	72.4	**13.8**	80.9%
	FALCON-7B	486.8	**10.9**	18.7	-	372.7	**9.4**	12.9	-
	QWEN3-8B	5792	133.7	**19.7**	85.3%	9605	74.7	**17.1**	77.1%

4.2.3 Evaluation on Performance

As shown in Table 2, we compare the performance of various baselines with BEFI across two datasets and different LLMs. BEFI outperforms CacheShare and TokenShare in most cases, demonstrating its superior effectiveness. This result validates the strong performance of BEFI, which not only achieves balanced and efficient communication overhead but also ensures high-quality LLM inference. In contrast, CacheShare exhibits the poorest performance among all frameworks, with at

Table 3 Experimental Results of LLAMA-7B on long-context dataset LongBench. Underline indicates the best performance among the three frameworks.

Model	lcc	multi_news	qasper	qmsum	repo-p	samsum	trec	triviaqa
Origin	66.7	3.68	9.6	21.3	59.8	41.8	66.0	87.7
TokenShare	1.2	0.08	0.2	0.53	1.4	0.13	0.0	0.6
CacheShare	29.1	**6.7**	1.9	6.89	30.52	1.97	15.0	5.5
BEFI	**41.5**	5.98	**5.7**	**17.1**	**41.9**	**33.3**	**34.5**	**74.8**

least a 12.4× degradation compared to BEFI. This performance drop is likely due to the accumulation of injected DP noise in the shared KV cache over longer inference sequences, as analyzed in Sec. 3.2. Besides, we observe that TokenShare outperforms BEFI slightly for FALCON-7B. This is probably because FALCON-7B utilizes a multi-query attention mechanism, and the key and value matrices have only 1 header. As a result, the fixed-size KV cache shared among silos carries less information, leading to a worse performance.

Table 3 further presents the experimental results on the dataset LongBench under LLAMA-7B, where Origin represents the original model without differential privacy. BEFI outperforms TokenShare and CacheShare in most cases, and achieves close performance compared to the original model without privacy perturbation. In contrast, TokenShare and CacheShare perform badly with several cases, achieving results close to 0. The results validate the flexibility of BEFI for long-context tasks.

4.2.4 Evaluation on Various Parameters.

Fig. 6a and Fig. 6b present the communication overhead of BEFI as the KV cache size varies for LLAMA-7B and LLAMA-13B, respectively. For different c, BEFI consistently maintains a balanced communication overhead. As c increases, the communication overhead per silo in BEFI also increases. Notably, when $c = 24$, BEFI incurs a communication overhead lower than that of the third silo in CacheShare. These results demonstrate that the size of the cache is the primary factor determining the communication overhead, highlighting the necessity of selecting tokens based on their importance in BEFI.

Table 4 presents the perplexity of BEFI under varying privacy budget ϵ. We observe that as the privacy requirement becomes stricter (*i.e.,* smaller ϵ), the performance of BEFI degrades slightly. However, this degradation remains within a reasonable range, especially when compared to CacheShare, where perplexity values reach into the thousands. These results demonstrate that BEFI is robust and flexible across a wide range of privacy settings.

4.2.5 Evaluation on Ablation Study

Fig. 6c and Fig. 6d present the ablation study on communication cost for LLAMA-7B and LLAMA-13B, respectively. We compare BEFI with two ablated variants: one implementing only the fixed-size KV cache sharing mechanism (namely BEFI-KV), and another with only the context-free intermediate states sharing mechanism (namely BEFI-IS). We observe that both mechanisms result in constant communication costs. Furthermore, the communication of BEFI-KV accounts for more than 99% of that of BEFI. This is because the fixed-size KV cache constitutes the majority of the shared information.

Table 4 Performance of LLAMA-7B varying ϵ. BEFI always outperforms CacheShare and TokenShare.

Framework		Cache		Token		BEFI	
ϵ		4.0	8.0	4.0	8.0	4.0	8.0
Wikitext	Lap	10k	7k	153	136	36.0	35.3
	MVG	7k	7k	107	102	66.9	42.7
PG19	Lap	10k	10k	49.4	48.9	22.8	24.7
	MVG	7k	6k	111	105	33.0	28.2

Table 5 Ablation study for LLAMA-7B (Underline: outperforms TokenShare).

Dataset	Wikitext		PG19	
DP	Laplace	MVG	Laplace	MVG
BEFI-KV	**64.2**	**69.7**	329	**28.1**
BEFI-IS	**126**	**115.3**	**67.2**	**70.9**
BEFI	**52.2**	**54.6**	**35.3**	**43.8**

Table 5 presents the ablation study of BEFI in terms of perplexity scores for LLAMA-7B. On different datasets and DP mechanisms, both BEFI-KV and BEFI-IS outperform TokenShare in most cases, while slightly underperforming compared to BEFI. This indicates that both the proposed fixed-size cache sharing and intermediate states sharing mechanisms contribute positively to BEFI.

5 Related Work

Differential Privacy. Differential privacy (DP) has emerged as a leading paradigm for protecting data privacy [8, 31–33]. Recently work has focused on various data types, including user-level differential privacy [34,35], geometric differential privacy [36,37], and matrix-valued differential privacy [30, 38]. In particular, matrix-valued differential privacy is widely used to protect the value of embeddings.

[39] proves that the Johnson-Lindenstrauss transform inherently satisfies matrix-valued differential privacy. Blum *et al.* [40] and Upadhyay *et al.* [41] further improve the efficiency by efficiently simulating the Johnson-Lindenstrauss transform. [30] formally defines matrix-valued differential privacy and propose utilizing the matrix-variate Gaussian distribution [42] to meet the privacy requirement. Yang *et al.* [43] optimize the matrix-variate Gaussian under dimension-aware utility constraints. In comparison, our work can employ any of these methods as a base mechanism to protect the information.

Federated Learning for LLMs. Numerous studies have been proposed to protect the privacy of training data in federated settings. These works focus on privacy-preserving techniques for federated training or

finetuning, addressing aspects such as privacy guarantees [44–47], and computational efficiency [48–50], among others. In contrast, relatively few works have explored federated inference for LLMs [51,52]. These works mainly consider scenarios where models are distributed among silos and investigate how to perform inference on local data. In comparison, we consider a novel yet practical scenario where the context is distributed across silos.

KV Cache Compression. To avoid the large space of KV cache [9, 10], a research line proposes distinguishing the important tokens and only storing the KV cache of these tokens. Preliminary works showed that the recently generated tokens tend to place more importance, thus proposing only storing the KV cache of recent tokens [53,54]. Beltagy *et al.* [13] and Xiao *et al.* [55] consider preserving the privacy of the KV cache of both the recent tokens and initially tokens.

In contrast, more recent works propose dynamically evicting KV cache based on the context. H2O [16] proposes determining the tokens based on their contribution scores. InfiniGen [56] improves the inference efficiency by overlapping the process of token selection and cache loading. Scissorhand [17] treats KV cache eviction as a budget-limited problem and selects the cache based on the attention weights. Other works include reducing the dimensionality of cache channel [57], mining diversity among heads [58], layer-wise cache eviction [59], and the mixture of various policies [60,61].

6 Conclusion

This paper focuses on a novel yet practical scenario in federated LLM inference, where all silos employ an identical LLM and the context is distributed among silos. The sequential generation paradigm of attention and the commonly used KV cache techniques lead to an inefficient and unbalanced communication overhead in such a scenario. To this end, we propose BEFI, a framework that not only preserves the privacy of individual context but also ensures balanced and efficient communication costs while inferring across silos. BEFI incorporates a fixed-size KV cache sharing mechanism for important tokens, as well as a context-free intermediate states mechanism that enables compressed representation of the other tokens of arbitrary length. Extensive experiments on different datasets validate the effectiveness of BEFI in achieving both efficient and balanced federated LLM inference.

Acknowledgements

This research is supported by National Key Research and Development Program of China (No. 2023YFF0725103), National Science Foundation of China (No. 62425202, 62336003, 62572412, 62572061, 62532002), Beijing Natural Science Foundation (No. Z230001), the State Key Laboratory of Complex & Critical Software Environment (SKLCCSE), City University of Hong Kong Shenzhen Research Institute (CityUHKSRI), and Beijing Advanced Innovation Center for Future Blockchain and Privacy Computing.

Competing Interests

The authors declare that they have no competing interests or financial conflicts to disclose.

Appendix

A.1 Proofs of Theorems

A.1.1 Proof of Theorem 3

Recall that the perturbed key cache and value are defined as $\widetilde{\mathbf{K}} = \mathbf{K} + \delta\mathbf{K}$ and $\widetilde{\mathbf{V}} = \mathbf{V} + \delta\mathbf{V}$, respectively. We first reclaim the theorem in following.

Theorem 3. Consider Equ. 3. We are interested in $\Delta\mathbf{A}^o = \widetilde{\mathbf{A}}^o - \mathbf{A}^o$, the difference between the attention outputs before and after the perturbation. Let λ_Q, λ_V, and λ_W represent the largest singular values of $\mathbf{Q}_t$, $\mathbf{V}$, and $\mathbf{A}^w$, respectively. Given $\delta\mathbf{K}_i (resp.\ \delta\mathbf{V}_i) \in \mathbb{R}^{1\times h}$ the noise added to the key (*resp.* value) cache of a global token $i \in T^{glo}$, we have

$$||\Delta\mathbf{A}^o||_2 \leq \lambda_W \delta_V + \frac{1}{\sqrt{h}}\lambda_Q \delta_K (\lambda_V + \delta_V)$$

where $||\mathbf{x}||$ denotes the norm of embeddings of current token. $\delta_K = \sum_{i\in T^{glo}} ||\delta\mathbf{K}_i||$ and $\delta_V = \sum_{i\in T^{glo}} ||\delta\mathbf{V}_i||$ are the sum of the norm of the noise added to the global tokens' key and value caches, respectively.

We establish two lemmas that support the proof of Theorem 3.

Lemma 1. The L2 norm of the concatenation of two matrices $\mathbf{A} \in \mathbb{R}^{a\times l}$ and $\mathbf{B} \in \mathbb{R}^{b\times l}$ is upper bounded as follows:

$$||\texttt{cat}(\mathbf{A}, \mathbf{B})||_2 \leq \lambda_A + \lambda_B$$

where λ_A and λ_B are the largest singular values of $\mathbf{A}$ and $\mathbf{B}$, respectively, and `cat` denotes the concatenation of two matrices.

Proof. The concatenation of two matrices is represented as

$$\texttt{cat}(\mathbf{A}, \mathbf{B}) = \texttt{cat}(\mathbf{A}, \mathbf{0}_{b\times l}) + \texttt{cat}(\mathbf{0}_{a\times l}, \mathbf{B})$$

where $\mathbf{0}_{a\times l}$ (*resp.* $\mathbf{0}_{b\times l}$) denotes the all-zero matrix of size $a \times l$ (*resp.* $b \times l$). Observe that all elements in the last b columns and corresponding rows of the matrix product $\texttt{cat}(\mathbf{A}, \mathbf{0}_{b\times l}) \cdot \texttt{cat}(\mathbf{A}, \mathbf{0}_{b\times l})^T$ are zeros. The remaining elements are identical to those in $\mathbf{A} \cdot \mathbf{A}^T$, which implies that the eigenvalues of $\texttt{cat}(\mathbf{A}, \mathbf{0}_{b\times l}) \cdot \texttt{cat}(\mathbf{A}, \mathbf{0}_{b\times l})^T$ are the same as those of $\mathbf{A} \cdot \mathbf{A}^T$. This indicates that $\texttt{cat}(\mathbf{A}, \mathbf{0}_{b\times l}$ shares the same singular values of $\mathbf{A}$. Thus, we have

$$\begin{aligned} ||\texttt{cat}(\mathbf{A}, \mathbf{B})||_2 &= ||\texttt{cat}(\mathbf{A}, \mathbf{0}_{b\times l}) + \texttt{cat}(\mathbf{B}, \mathbf{0}_{a\times l})^T||_2 \\ &\leq ||\texttt{cat}(\mathbf{A}, \mathbf{0}_{b\times l})||_2 + ||\texttt{cat}(\mathbf{B}, \mathbf{0}_{a\times l})^T||_2 \\ &= ||\mathbf{A}||_2 + ||\mathbf{B}||_2 \\ &= \lambda_A + \lambda_B \end{aligned}$$

□

Lemma 1 illustrates how the concatenation of two matrices affects the L2 norm. We then analyze the impact of DP noise on the attention weights $\mathbf{A}^w$, as illustrated in Lemma 2. Part of the proof of Lemma 2 is inspired by the work of [17].

Lemma 2. Given the noise added to the key matrix $\delta\mathbf{K}$ and the value matrix $\delta\mathbf{V}$, the difference in the attention weight satisfies

$$||\Delta\mathbf{A}^w||_2 \leq \frac{1}{\sqrt{h}}\lambda_Q ||\delta\mathbf{K}||_2$$

Proof. When a silo generates the current token, its key (*resp.* value) cache consists of two components: (1) the global cache $\widetilde{\mathbf{K}}^{glo} = \mathbf{K}^{glo} + \delta\mathbf{K}$ (*resp.* $\widetilde{\mathbf{V}}^{glo} = \mathbf{V}^{glo} + \delta\mathbf{V}$) which incorporates DP noise, and (2) the local key (*resp.* value) cache $\mathbf{K}^{loc}$ (*resp.* $\mathbf{V}^{loc}$) which remains uncorrupted by the DP noise.

$$
\begin{aligned}
&||\Delta\mathbf{A}^w||_2 = ||\widetilde{\mathbf{A}}^w - \mathbf{A}^w||_2 \\
&= ||\texttt{softmax}(\frac{\mathbf{Q}^t\,\texttt{cat}(\widetilde{\mathbf{K}}^{glo}, \mathbf{K}^{loc})^T}{\sqrt{h}}) \\
&\quad - \texttt{softmax}(\frac{\mathbf{Q}^t\,\texttt{cat}(\mathbf{K}^{glo}, \mathbf{K}^{loc})^T}{\sqrt{h}})||_2 \\
&\le ||\frac{\mathbf{Q}^t\,\texttt{cat}(\widetilde{\mathbf{K}}^{glo}, \mathbf{K}^{loc})^T}{\sqrt{h}} - \frac{\mathbf{Q}^t\,\texttt{cat}(\mathbf{K}^{glo}, \mathbf{K}^{loc})^T}{\sqrt{h}}||_2 \\
&= \frac{1}{\sqrt{h}}||\mathbf{Q}^t\,\texttt{cat}(\delta\mathbf{K}, \mathbf{0}_{i\times h})^T||_2 \\
&= \frac{1}{\sqrt{h}}\lambda_Q \cdot ||\texttt{cat}(\delta\mathbf{K}, \mathbf{0}_{i\times h})^T||_2 \\
&\le \frac{1}{\sqrt{h}}\lambda_Q \cdot ||\delta\mathbf{K}||_2
\end{aligned}
$$

where the fourth line follows from the Lipschitz continuity of the softmax function [17, 62], and the seventh line follows from Lemma 1 and $||\mathbf{0}_{i\times h}||_2 = 0$. The lemma is proved. □

Based on the above lemma, we then analyze the error of the attention output $\mathbf{A}^o$, *i.e.,* $\Delta\mathbf{A}^o = \widetilde{\mathbf{A}}^o - \mathbf{A}^o$.

Lemma 3. Consider the inference process in Equ. 3 and let $\Delta\mathbf{A}^o = \widetilde{\mathbf{A}}^o - \mathbf{A}^o$ denote the difference between the attention outputs before and after the perturbation. Let λ_Q, λ_V, and λ_W represent the largest singular values of $\mathbf{Q}_t$, $\mathbf{V}$, and $\mathbf{A}^w$, respectively. We have

$$
||\Delta\mathbf{A}^o||_2 \le \lambda_W||\delta\mathbf{V}||_2 + \frac{1}{\sqrt{h}}\lambda_Q||\delta\mathbf{K}||_2 \cdot (\lambda_V + ||\delta\mathbf{V}||_2)
$$

.

Proof. Define $\widetilde{\mathbf{V}} = \texttt{cat}(\widetilde{\mathbf{V}}^{glo}, \mathbf{V}^{loc})$ as the concatenation of $\widetilde{\mathbf{V}}^{glo}$ and $\mathbf{V}^{loc}$. Define $\mathbf{V} = \texttt{cat}(\mathbf{V}^{glo}, \mathbf{V}^{loc})$.

$$
\begin{aligned}
&||\Delta\mathbf{A}^o||_2 = ||\widetilde{\mathbf{A}}^o - \mathbf{A}^o||_2 \\
&= ||\widetilde{\mathbf{A}}^w\widetilde{\mathbf{V}} - \mathbf{A}^w\mathbf{V}||_2 \\
&\le ||\widetilde{\mathbf{A}}^w\widetilde{\mathbf{V}} - \widetilde{\mathbf{A}}^w\mathbf{V}||_2 + ||\widetilde{\mathbf{A}}^w\mathbf{V} - \mathbf{A}^w\mathbf{V}||_2 \\
&= ||\widetilde{\mathbf{A}}^w||_2||\widetilde{\mathbf{V}} - \mathbf{V}||_2 + ||\widetilde{\mathbf{A}}^w - \mathbf{A}^w||_2||\mathbf{V}||_2 \\
&\le (||\mathbf{A}^w||_2 + ||\Delta\mathbf{A}^w||_2)||\widetilde{\mathbf{V}} - \mathbf{V}||_2 + ||\Delta\mathbf{A}^w||_2||\mathbf{V}||_2 \\
&= ||\mathbf{A}^w||_2||\delta\mathbf{V}||_2 + ||\Delta\mathbf{A}^w||_2(||\delta\mathbf{V}||_2 + ||\mathbf{V}||_2) \\
&\le \lambda_W||\delta\mathbf{V}||_2 + \frac{1}{\sqrt{h}}\lambda_Q||\delta\mathbf{K}||_2(\lambda_V + ||\delta\mathbf{V}||_2)
\end{aligned} \tag{8}
$$

Here, the fifth line follows from the triangle inequality for matrix norms, which gives $||\widetilde{\mathbf{A}}^w||_2 = ||\mathbf{A}^w + \Delta\mathbf{A}^w||_2 \le ||\mathbf{A}^w||_2 + ||\Delta\mathbf{A}^w||_2$. The seventh line is derived from the upper bound of $||\Delta\mathbf{A}^w||_2$ obtained from Lemma 2. λ_Q, λ_V, and λ_W represent the largest singular values of $\mathbf{Q}_t$, $\mathbf{V}$, and $\mathbf{A}_w$, respectively. □

Based on the above lemmas, we are ready to prove the theorem. Recall that the inference of the current token relies on two types of tokens. For tokens T^{glo} whose KV caches are perturbed and transferred, their noises $\delta\mathbf{V}_i$ formulate $\delta\mathbf{V}$ by the concatenation. For tokens T^{loc}, which are generated locally and have no perturbation on their KV caches, their noises are always 0. We have

$$
\begin{aligned}
||\delta\mathbf{V}||_2 &= ||\texttt{cat}(\{\mathbf{V}_i \text{ for } i \in T^{glo}\}, \{\mathbf{V}_i \text{ for } i \in T^{loc}\})||_2 \\
&\le \sum_{i\in T^{glo}} ||\delta\mathbf{V}_i||_2 + \sum_{i\in T^{loc}} ||\delta\mathbf{V}_i||_2 \\
&= \sum_{i\in T^{glo}} ||\delta\mathbf{V}_i||_2 \\
&= \delta_V
\end{aligned}
$$

where the second line is derived from Lemma 1. Similarly, we have $||\delta\mathbf{K}||_2 \le \delta_K$. Substituting these results into Lemma 3, we obtain

$$
||\Delta\mathbf{A}^w||_2 \le \lambda_W\delta_V + \frac{1}{\sqrt{h}}\lambda_Q\delta_K(\lambda_V + \delta_V)
$$

which proves the theorem. Notice that the largest singular values of matrices, *i.e.,* λ_W, λ_Q, and λ_V, remain no less than 0.

A.1.2 Proof of Theorem 2

We first reclaim Theorem 2. Recall that $\mathbf{A}^r$ is the input to the softmax function in Equ. 6.

Theorem 4. Given the perturbed weight bias $\widetilde{\mathbf{I}} = \mathbf{I} + \Delta\mathbf{I}$, if the perturbation ΔI satisfies $\Delta I \ge 0$ or $\sum\mathbf{A}^r \ge \ln(2|\Delta I|)$, then the difference of the attention outputs, denoted by $\Delta\mathbf{A}^o$, is bounded by

$$
||\Delta\mathbf{A}^o||_2 \le \lambda_W\lambda_V
$$

.

Proof. We first compute the bound on $||\Delta\mathbf{A}^w||_2$. Let $\mathbf{A}^r_e = \texttt{exp}(\mathbf{A}^r)$ denote the element-wise exponential of $\mathbf{A}^r$.

$$
\begin{aligned}
||\Delta\mathbf{A}^w||_2 &= ||\widetilde{\mathbf{A}}^w - \mathbf{A}^w||_2 \\
&= ||\frac{\mathbf{A}^r_e}{\sum\mathbf{A}^r_e + \mathbf{I} + \Delta I} - \frac{\mathbf{A}^r_e}{\sum\mathbf{A}^r_e + \mathbf{I}}||_2 \\
&= \frac{|\Delta I|}{|\sum\mathbf{A}^r_e + \mathbf{I} + \Delta I|}||\mathbf{A}^w||_2
\end{aligned} \tag{9}
$$

Specifically, considering that $\texttt{sum}(\mathbf{A}^r) \ge \ln(2|\Delta I|)$, we have

$$
\begin{aligned}
\sum\mathbf{A}^r_e &= \sum\texttt{exp}(\mathbf{A}^r_{1,i}) \ge e^{\sum\mathbf{A}^r_{1,i}} \\
&\ge e^{\ln(2|\Delta I|)} = 2|\Delta I|
\end{aligned} \tag{10}
$$

where $\mathbf{A}^r_{1,i}$ represents the $(1,i)$-th element of $\mathbf{A}^r$, and the first line follows from the convexity of the exponential function. Moreover, note that $\mathbf{I}$, being the summation of exponentials, is strictly positive.

- If $\Delta I \ge 0$, since $\sum\mathbf{A}^t_e + \mathbf{I} > 0$, we have $|\Delta I| \le |\sum\mathbf{A}^r_e + \mathbf{I} + \Delta I|$.
- If $\mathbf{A}^t \ge \ln(2|\Delta I|)$, we have $\sum\mathbf{A}^t_e + \mathbf{I} > \sum\mathbf{A}^t_e \ge 2|\Delta I|$, and thus $|\Delta I| \le |\sum\mathbf{A}^t_e + \mathbf{I} + \Delta I|$.

For either case, it always holds $|\Delta I| \le |\sum\mathbf{A}^t_e + \mathbf{I} + \Delta I|$. Substituting this result into Equ. 10, we have $||\Delta\mathbf{A}^w||_2 \le ||\mathbf{A}^w||_2$. Finally, consider the attention output $||\Delta\mathbf{A}^o||_2$:

$$
\begin{aligned}
||\Delta\mathbf{A}^o||_2 &= ||\widetilde{\mathbf{A}}^o - \mathbf{A}^o||_2 \\
&= ||\widetilde{\mathbf{A}}^w\mathbf{V} - \mathbf{A}^w\mathbf{V}||_2 \\
&= ||\Delta\mathbf{A}^w||_2||\mathbf{V}||_2 \\
&\le ||\mathbf{A}^w||_2||\mathbf{V}||_2 \\
&= \lambda_W\lambda_V
\end{aligned} \tag{11}
$$

which completes the proof. □

References

[1] Zhu W, Liu H, Dong Q, Xu J, Huang S, Kong L, Chen J, Li L. Multilingual machine translation with large language models: Empirical results and analysis. In: NAACL. 2024, 2765–2781

[2] He Z, Liang T, Jiao W, Zhang Z, Yang Y, Wang R, Tu Z, Shi S, Wang X. Exploring human-like translation strategy with large language models. Trans. Assoc. Comput. Linguistics, 2024, 12: 229–246

[3] Wu J, Yang S, Zhan R, Yuan Y, Chao L S, Wong D F. A survey on llm-generated text detection: Necessity, methods, and future directions. Comput. Linguistics, 2025, 51(1): 275–338

[4] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser L, Polosukhin I. Attention is all you need. In: NIPS. 2017, 5998–6008

[5] Ali M, Naeem F, Tariq M, Kaddoum G. Federated learning for privacy preservation in smart healthcare systems: A comprehensive survey. IEEE J. Biomed. Health Informatics, 2023, 27(2): 778–789

[6] Torrepadula d F R, Fisichella M, Martino S D, Mazzocca N. Fedflow: a personalized federated learning framework for passenger flow prediction. Mach. Learn., 2025, 114(7): 166

[7] Voigt P, Bussche V. d A. The EU General Data Protection Regulation (GDPR): A Practical Guide. Springer, 2017

[8] Dwork C. Differential privacy. In: Bugliesi M, Preneel B, Sassone V, Wegener I, eds, ICALP. 2006, 1–12

[9] Pope R, Douglas S, Chowdhery A, Devlin J, Bradbury J, Heek J, Xiao K, Agrawal S, Dean J. Efficiently scaling transformer inference. In: ICML. 2023

[10] Mohtashami A, Jaggi M. Landmark attention: Random-access infinite context length for transformers. arXiv preprint arXiv:2305.16300, 2023

[11] Coppersmith D, Winograd S. Matrix multiplication via arithmetic progressions. J. Symb. Comput., 1990, 9(3): 251–280

[12] Bläser M. Fast matrix multiplication. Theory Comput., 2013, 5: 1–60

[13] Beltagy I, Peters M E, Cohan A. Longformer: The long-document transformer. CoRR, 2020

[14] Xiao G, Tian Y, Chen B, Han S, Lewis M. Efficient streaming language models with attention sinks. In: ICLR. 2024

[15] Zaheer M, Guruganesh G, Dubey K A, Ainslie J, Alberti C, Ontañón S, Pham P, Ravula A, Wang Q, Yang L, Ahmed A. Big bird: Transformers for longer sequences. In: NeurIPS. 2020

[16] Zhang Z, Sheng Y, Zhou T, Chen T, Zheng L, Cai R, Song Z, Tian Y, Ré C, Barrett C W, Wang Z, Chen B. H2O: heavy-hitter oracle for efficient generative inference of large language models. In: NeurIPS. 2023

[17] Liu Z, Desai A, Liao F, Wang W, Xie V, Xu Z, Kyrillidis A, Shrivastava A. Scissorhands: Exploiting the persistence of importance hypothesis for LLM KV cache compression at test time. In: NeurIPS. 2023

[18] Child R, Gray S, Radford A, Sutskever I. Generating long sequences with sparse transformers. arXiv preprint arXiv:1904.10509, 2019

[19] Roy A, Saffar M, Vaswani A, Grangier D. Efficient content-based sparse attention with routing transformers. Transactions of the Association for Computational Linguistics, 2021, 9: 53–68

[20] Tay Y, Bahri D, Yang L, Metzler D, Juan D C. Sparse sinkhorn attention. In: International conference on machine learning. 2020, 9438–9447

[21] Touvron H, Martin L, Stone K, Albert P, Almahairi A, Babaei Y, Bashlykov N, Batra S, Bhargava P, Bhosale S, Bikel D, Blecher L, Canton-Ferrer C, Chen M, Cucurull G, Esiobu D, Fernandes J, Fu J, Fu W, Fuller B, Gao C, Goswami V, Goyal N, Hartshorn A, Hosseini S, Hou R, Inan H, Kardas M, Kerkez V, Khabsa M, Kloumann I, Korenev A, Koura P S, Lachaux M, Lavril T, Lee J, Liskovich D, Lu Y, Mao Y, Martinet X, Mihaylov T, Mishra P, Molybog I, Nie Y, Poulton A, Reizenstein J, Rungta R, Saladi K, Schelten A, Silva R, Smith E M, Subramanian R, Tan X E, Tang B, Taylor R, Williams A, Kuan J X, Xu P, Yan Z, Zarov I, Zhang Y, Fan A, Kambadur M, Narang S, Rodriguez A, Stojnic R, Edunov S, Scialom T. Llama 2: Open foundation and fine-tuned chat models. CoRR, 2023, abs/2307.09288

[22] Jiang A Q, Sablayrolles A, Mensch A, Bamford C, Chaplot D S, Las Casas d D, Bressand F, Lengyel G, Lample G, Saulnier L, Lavaud L R, Lachaux M, Stock P, Scao T L, Lavril T, Wang T, Lacroix T, Sayed W E. Mistral 7b. CoRR, 2023, abs/2310.06825

[23] Almazrouei E, Alobeidli H, Alshamsi A, Cappelli A, Cojocaru R, Debbah M, Goffinet E, Heslow D, Launay J, Malartic Q, Noune B, Pannier B, Penedo G. Falcon-40B: an open large language model with state-of-the-art performance. 2023

[24] Merity S, Xiong C, Bradbury J, Socher R. Pointer sentinel mixture models. In: ICLR. 2017

[25] Rae J W, Potapenko A, Jayakumar S M, Hillier C, Lillicrap T P. Compressive transformers for long-range sequence modelling. In: ICLR. 2020

[26] Bai Y, Lv X, Zhang J, Lyu H, Tang J, Huang Z, Du Z, Liu X, Zeng A, Hou L, Dong Y, Tang J, Li J. Longbench: A bilingual, multitask benchmark for long context understanding. In: ACL. 2024, 3119–3137

[27] Lyu L, He X, Li Y. Differentially private representation for NLP: formal guarantee and an empirical study on privacy and fairness. In: EMNLP. 2020, 2355–2365

[28] Meehan C, Mrini K, Chaudhuri K. Sentence-level privacy for document embeddings. In: ACL. 2022, 3367–3380

[29] Yue X, Du M, Wang T, Li Y, Sun H, Chow S S M. Differential privacy for text analytics via natural text sanitization. In: ACL/IJCNLP, Findings of ACL. 2021, 3853–3866

[30] Chanyaswad T, Dytso A, Poor H V, Mittal P. MVG mechanism: Differential privacy under matrix-valued query. In: CCS. 2018, 230–246

[31] Abadi M, Chu A, Goodfellow I J, McMahan H B, Mironov I, Talwar K, Zhang L. Deep learning with differential privacy. In: SIGSAC. 2016, 308–318

[32] Dwork C. Differential privacy: A survey of results. In: Agrawal M, Du D, Duan Z, Li A, eds, TAMC. 2008, 1–19

[33] Du M, Yue X, Chow S S M, Wang T, Huang C, Sun H. Dp-forward: Fine-tuning and inference on language models with differential privacy in forward pass. In: CCS. 2023, 2665–2679

[34] Zhao P, Shen L, Fan R, Li Q, Wu H, Wu J, Liu Z. Learning with user-level local differential privacy. CoRR, 2024, abs/2405.17079

[35] Xu Z, Collins M D, Wang Y, Panait L, Oh S, Augenstein S, Liu T, Schroff F, McMahan H B. Learning to generate image embeddings with user-level differential privacy. In: CVPR. 2023, 7969–7980

[36] Chatzikokolakis K, Andrés M E, Bordenabe N E, Palamidessi C. Broadening the scope of differential privacy using metrics. In: PETS. 2013

[37] Liang Y, Yi K. Concentrated geo-privacy. In: Meng W, Jensen C D, Cremers C, Kirda E, eds, SIGSAC. 2023, 1934–1948

[38] Ji T, Li P, Yilmaz E, Ayday E, Ye Y, Sun J. Differentially private binary- and matrix-valued data query: An XOR mechanism. Proc. VLDB Endow., 2021, 14(5): 849–862

[39] Blocki J, Blum A, Datta A, Sheffet O. The johnson-lindenstrauss

transform itself preserves differential privacy. In: FOCS. 2012, 410–419
[40] Blum A, Roth A. Fast private data release algorithms for sparse queries. In: RANDOM. 2013
[41] Upadhyay J. Randomness efficient fast-johnson-lindenstrauss transform with applications in differential privacy and compressed sensing. arXiv preprint arXiv:1410.2470, 2014
[42] Dawid A P. Some matrix-variate distribution theory: notational considerations and a bayesian application. Biometrika, 1981, 68(1): 265–274
[43] Yang J, Xiang L, Chen R, Li W, Li B. Differential privacy for tensor-valued queries. IEEE Trans. Inf. Forensics Secur., 2022, 17: 152–164
[44] Han S, Buyukates B, Hu Z, Jin H, Jin W, Sun L, Wang X, Wu W, Xie C, Yao Y, Zhang K, Zhang Q, Zhang Y, Joe-Wong C, Avestimehr S, He C. Fedsecurity: A benchmark for attacks and defenses in federated learning and federated llms. In: SIGKDD. 2024, 5070–5081
[45] Zhang Z, Zhang J, Huang J, Qu L, Zhang H, Xu Z. Fedpit: Towards privacy-preserving and few-shot federated instruction tuning. CoRR, 2024, abs/2403.06131
[46] Wu C, Li X, Wang J. Vulnerabilities of foundation model integrated federated learning under adversarial threats. CoRR, 2024, abs/2401.10375
[47] Sun Y, Li Z, Li Y, Ding B. Improving lora in privacy-preserving federated learning. In: ICLR. 2024
[48] Qin Z, Chen D, Qian B, Ding B, Li Y, Deng S. Federated full-parameter tuning of billion-sized language models with communication cost under 18 kilobytes. In: ICML. 2024
[49] Khalid U, Iqbal H, Vahidian S, Hua J, Chen C. CEFHRI: A communication efficient federated learning framework for recognizing industrial human-robot interaction. In: IROS. 2023, 10141–10148
[50] Charles Z, Mitchell N, Pillutla K, Reneer M, Garrett Z. Towards federated foundation models: Scalable dataset pipelines for group-structured learning. In: NeurIPS. 2023
[51] Liu Z, Li D, Fernández-Marqués J, Laskaridis S, Gao Y, Dudziak L, Li S Z, Hu S X, Hospedales T M. Federated learning for inference at anytime and anywhere. CoRR, 2022, abs/2212.04084
[52] Ma T, Hoang T N, Chen J. Federated inference through aligning local representations and learning a consensus graph. 2022
[53] Kovaleva O, Romanov A, Rogers A, Rumshisky A. Revealing the dark secrets of BERT. In: EMNLP-IJCNLP. 2019, 4364–4373
[54] Child R, Gray S, Radford A, Sutskever I. Generating long sequences with sparse transformers. CoRR, 2019, abs/1904.10509
[55] Xiao G, Tian Y, Chen B, Han S, Lewis M. Efficient streaming language models with attention sinks. In: ICLR. 2024
[56] Lee W, Lee J, Seo J, Sim J. Infinigen: Efficient generative inference of large language models with dynamic KV cache management. In: OSDI. 2024
[57] Xu Y, Jie Z, Dong H, Wang L, Lu X, Zhou A, Saha A, Xiong C, Sahoo D. Think: Thinner key cache by query-driven pruning. In: ICLR. 2025
[58] Tang H, Lin Y, Lin J, Han Q, Ke D, Hong S, Yao Y, Wang G. Razorattention: Efficient KV cache compression through retrieval heads. In: ICLR. 2025
[59] Yang D, Han X, Gao Y, Hu Y, Zhang S, Zhao H. Pyramidinfer: Pyramid KV cache compression for high-throughput LLM inference. In: ACL. 2024, 3258–3270
[60] Ge S, Zhang Y, Liu L, Zhang M, Han J, Gao J. Model tells you what to discard: Adaptive KV cache compression for llms. In: ICLR. 2024
[61] Adnan M, Arunkumar A, Jain G, Nair P J, Soloveychik I, Kamath P. Keyformer: KV cache reduction through key tokens selection for efficient generative inference. In: MLSys. 2024
[62] Gao B, Pavel L. On the properties of the softmax function with application in game theory and reinforcement learning. CoRR, 2017, abs/1704.00805

Lulu Zhang is currently a Ph.D. candidate in Computer Science and Technology at Beihang University, China. She received the M.S. degree in Computer Science from The Hong Kong Polytechnic University. She has industry experience in artificial intelligence technology at leading companies including Alibaba Group, Didi Chuxing and Baidu, where she contributed to the development and application of AI-driven big data systems. Her research focuses on big data analytics and data mining.

Qian Tao received the Ph.D. degree in Computer Science and Technology from Beihang University, China, in 2021, and the B.S. degree from the same institution. Currently, he serves as an Associate Professor at the School of Artificial Intelligence, Beihang University, and has been an Algorithm Engineer at Alibaba Tongyi Lab. His research interests primarily focus on federated large language models and federated spatial-temporal data analysis.

Zimu ZHOU received the BE from the Department of Electronic Engineering, Tsinghua University, China in 2011 and the PhD from the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, China in 2015. His research focuses on mobile, ubiquitous computing, and federated learning.

Yuanyuan Zhang received the PhD degree in computer science and technology from Beihang University, in 2023. She is currently an associate professor with the School of Computer Science and Artificial Intelligence, Beijing Wuzi University. Her major research interests include federated learning,big spatiotemporal data mining and privacy preserving data analytics.

Yuxiang Wang received the B.E. degree in computer science and technology from Beihang University in 2022. He is currently working toward the Ph.D degree in the School of Computer Science and engineering, Beihang University. His major research interests include vector data management and federated data analytics.

Yongxin Tong received the PhD degree in computer science and engineering from the Hong Kong University of Science and Technology, China in 2014. He is currently a professor in the School of Computer Science and Engineering, Beihang University, China. His research interests include federated large language model, federated learning, spatio-temporal data analytics, and reinforcement learning. He received the championship of KDD Cup 2020 RL track.