CF-012

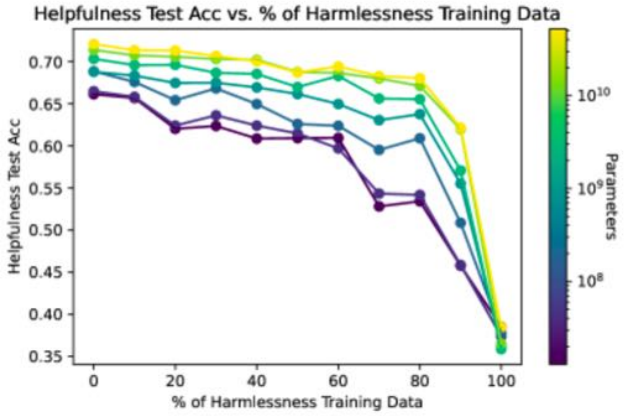
大規模言語モデルのデータセキュリティ
Data privacy and security in Large Language model

張 祺智 † 洪 爵 † 呉 燁 † 韓 東力 ‡
Qizhi Zhang Jue Hong Ye Wu Dongli Han

1. Introduction - Development Status and Risk Insight of Large Language models

The rapid development and widespread application of large language models, represented by OpenAI's GPT, Meta's Llama, and Google's Gemini, marks the entry of a new era of intelligence in the 21st century. Their development is the result of technological innovation, data accumulation, improvement in computing power, and progress in algorithms. Large language models have demonstrated powerful capabilities in multiple fields by processing and learning massive data, and are even redefining all aspects of work and life. However, at the same time, they also bring new challenges such as user privacy protection, ethical security, model transparency and interpretability, and model asset security.

Recently, several disputes have arisen due to security issues of large language models. For example, OpenAI faces legal action related to its data collection and usage, Samsung Electronics experienced a data leak during the use of GPT, and LLM generated and executed code for cross-site scripting (XSS) attacks on the web. The security control of large language models is a recognized challenge in the industry, and there is a natural conflict between security and chat capabilities. Model security is highly valued worldwide. The following figure illustrates the inverse relationship between the usefulness of model results and the harmlessness of training data..



In the lifecycle of a large language model, it can be divided into five stages: data preparation, pre-training, fine-tuning, deployment, and launching a new service. Each stage may encounter various risks. In the data preparation stage, there is a risk of data poisoning; in the pre-training and fine-tuning stages, there is a risk of training data leakage and layer leakage; in the deployment stage, there is a risk of partial model leakage, deployment code leakage, and tampering; in the service stage, there are risks such as user privacy leakage, bias in the large language model, hallucinations, harmful replies, jailbreaking of the large language model, and prompt word injection attacks.

From the perspective of model users and model holders, risks can be divided into those related to data assets security and those related to model assets security. Data assets security risks refer to the fact that model users such as banks, governments, hospitals, and small and medium-sized enterprises deal with a large amount of private data, which may pose unsafe factors when using models for inference. Model asset security risks refer to the fact that model holders such as OpenAI consider the data used for their models and training models as their core assets, which must not be leaked. Therefore, large language model risks are very important and complex from different perspectives, presenting an urgent problem to be tackled in the implementation of large language models.

Taking into account the various risk factors of the large language model lifecycle, the Jeddak team has developed a security capability that encompasses the entire LLM lifecycle from two dimensions: data, environment, and infrastructure, drawing on their previous accumulation of security technology.

In terms of data security, the team has created confusion vector retrieval and inference technology to ensure data security in the RAG process. For the training and fine-tuning processes of large language models, a privacy-preserving training scheme has been implemented. Finally, to address the interaction between users and their private large language models, a large language model guardrail technology has been devised to mitigate privacy leakage risks and AIGC risks. The underlying technologies employed include vector Order Preserving Encryption, secure multi-party computation, text vector privacy, model forgetting and output suppression, scalable Federated Learning, and fine-tuning, among others.

Regarding environmental and infrastructure security, a trusted LLM ecosystem has been established on top of the Confidential Computing infrastructure to safeguard model assets. These security capabilities are built upon the underlying security technology of the Jeddak team, which has accumulated expertise in security algorithms and infrastructure to effectively support the security features of LLM. The technologies involved encompass large-scale confidential containers, confidential databases, remote certification services, trusted key management, and heterogeneous trusted computing power, among others.

2. Internal data security system in Jeddak

We use technologies such as information obfuscation, privacy computing, and model enhancement to mitigate the risk of sensitive information leakage in large-scale Model Training and applications. During the data preprocessing phase, we adopt security measures such as information obfuscation and encryption, mainly including vector encryption and retrieval, and multidimensional data synthesis. During the Model Training phase, we adopt security measures such as privacy computing, including federated training and fine-tuning, and fully encrypted Machine Learning. During the model service phase, we adopt security measures such as model IO control and enhancement, including model privacy security guardrails, model forgetting, and reinforcement.

2.1 Private Inference and Vector Retrieval

Consider the application of RAG (Retrieval-Augmented Generation) technology in the following plaintext application

scenario composed of users, computing platforms, and storage parties: The user first provides the inferred Prompt to the platform, and the computing platform segments the Prompt to obtain the embeddings of each word in the Prompt by searching the word embedding table. After inputting LLM, the sentence embedding is obtained, and the sentence embedding is used to initiate a retrieval request to the storage party to obtain the most relevant background knowledge related to the sentence embedding. The computing platform splices the background knowledge with the user Prompt, inputs it into LLM together to complete the inference, and returns the result to the user to improve the effect and prevent "illusions". There are two risks here：

A. The platform directly obtains the user's original plaintext prompt.

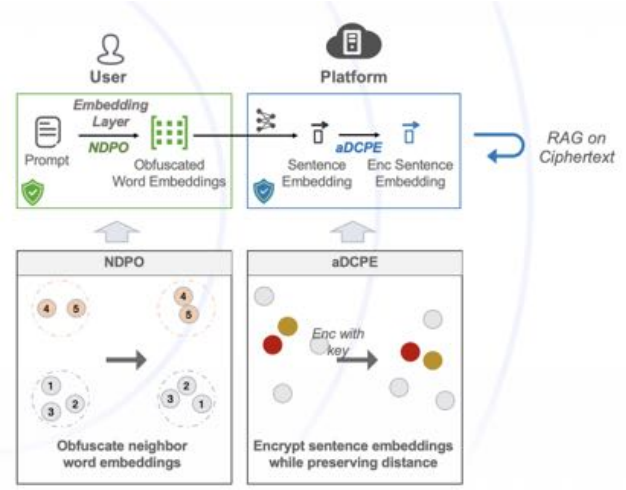
B. The storage party uses embedding inversion attack to restore user prompts from sentence embeddings.

How to make inference performance and model effects meet industrial-grade requirements (almost identical to plaintext) while protecting user privacy is a challenge and a difficult problem of balancing security and usability.

Our method is:

A. Users use Neighbor Distance Preserving Obfuscation (NDPO) to obfuscate word embeddings to prevent the platform from associating plaintext tokens of prompts through embeddings. In order to ensure the effect of the model on the obfuscated word embeddings, the platform needs to fine-tune the model on the obfuscated word embeddings.

B. The platform uses aDCPE to encrypt sentence embeddings to prevent storage parties from restoring user prompts through embedding inversion attacks.

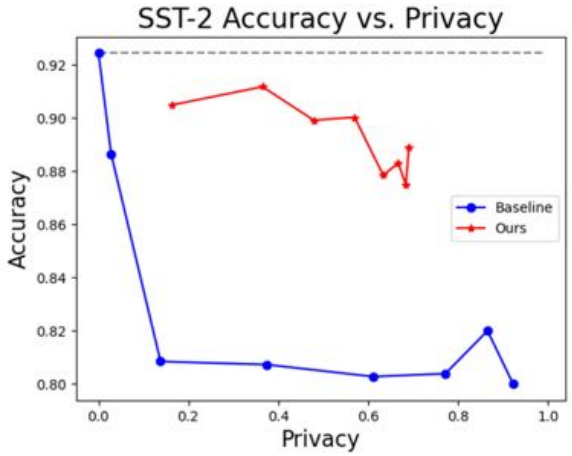


The details of the technical plan are as follows:

1. NDPO: Clusters all word embeddings in the vocabulary and confuses them. By adding DP noise to the embeddings, the embeddings within each cluster are indistinguishable. Compared to directly using DP to add noise to all embeddings, NDPO considers the data distribution of the embeddings themselves and discretely adds noise to the intra-cluster quantities through clustering, which can better preserve the semantic information of the embeddings. At the same time, NDPO also considers the differences between the different component values of the embedding vector and confuses the components with greater noise. For example, in a cluster, if different embeddings only differ significantly in the first 10 components, larger noise is added to these 10 components to confuse the vectors within the cluster.

2. aDCPE: As an encryption algorithm, the platform will input a key to encrypt the sentence embedding, so that the domain of the ciphertext vector and the plaintext vector are completely different, but the approximate distance relationship between the vectors will be preserved.

Our word embedding protection is as follows: In the case of accuracy loss < 3.5%, it resists nearest neighbor attacks and successfully protects about 58% of tokens.



Our sentence embedding protection effect is as follows:

original sentence	One Ring to rule them all, One Ring to find them, One Ring to b bind them.
attack to plaintext embed.	One Ring to rule them all, One Ring to find them, One Ring to b darkness binds them all.
attack to cipher embed.	is syphiliac osteoporosis after rappers''sewage' stage performa

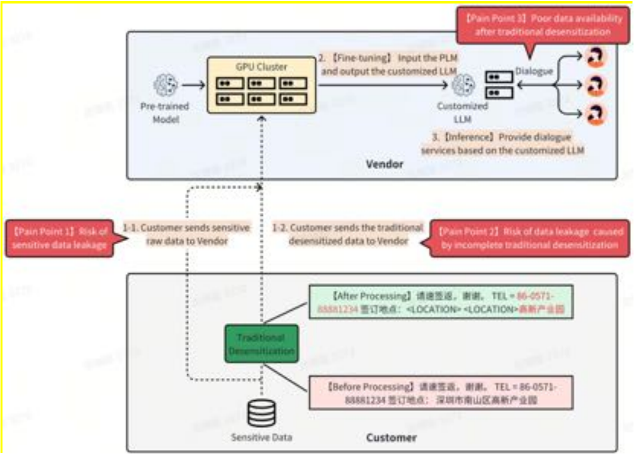
The search effect of the protected sentence embedding is as follows:

dataset	**index**	Recall		speed	
		plaintext	ciphertext	plaintext	ciphertext
sift-128-euclidean	HNSW	93.92%	93.51%	3,581,328	3,536,657
	IVF	99.99%	98.33%	18,624	16,791
glove-200-angular	HNSW	92.64%	90.83%	193,476	184,109
	IVF	100%	96.07%	9,083	8,998

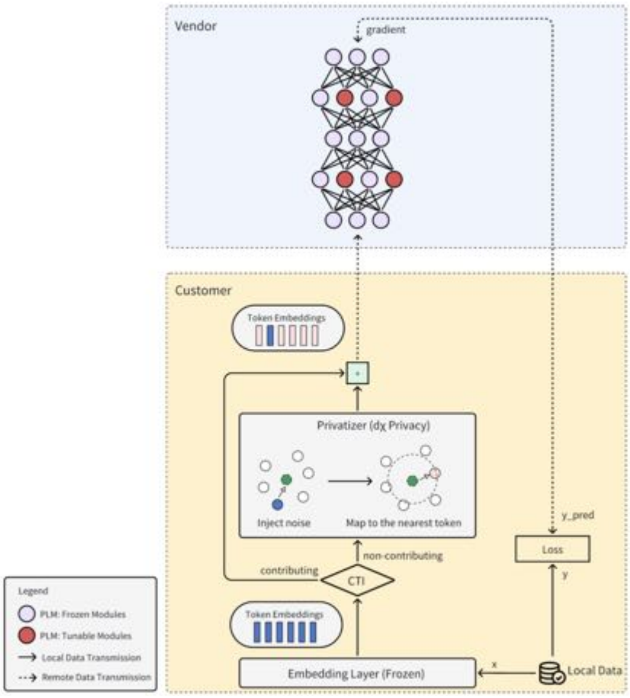
There is almost no loss in recall rate and performance while protecting privacy.

2.2 Federated fine tuning

The target scenario here is that the pre-trained model needs to be fine-tuned on downstream task data to improve usability, but the data provider and the Model Training party belong to different departments of the same company or different companies, and the sensitive information in the data causes circulation barriers. The pain point here is that if the original data source is directly given to the Model Training party, there is a risk of sensitive original data source leakage. If traditional desensitization methods are used, there will be the following two problems: 1. Incomplete coverage, incomplete desensitization leading to the risk of data leakage; 2. Poor usability affecting the subsequent Model Training effect.



We use model segmentation and text privacy methods to protect the privacy of both parties' privacy data/models. Model segmentation refers to: segmenting pre-trained models and deploying the bottom model on the client side to avoid direct disclosure of the original data source; text privacy refers to: the client side uses LDP, PAC and other privacy mechanisms to disturb the vectorized text representation and strengthen privacy protection. Among them, we also introduce important lexical recognition technology, which measures the contribution of lexical elements to utility goals to achieve Self-Adaptation perturbation mechanism and improve utility privacy trade-offs.

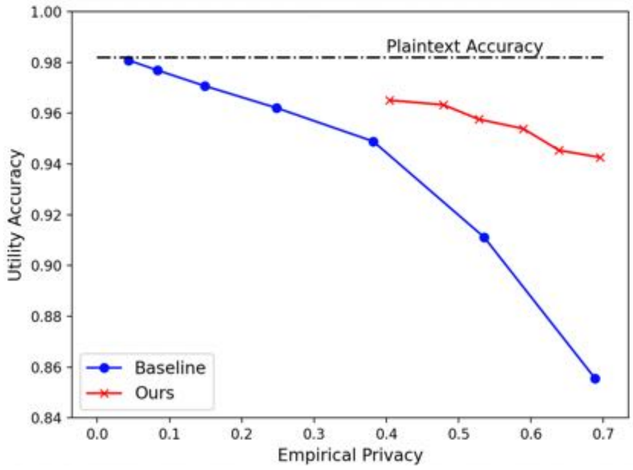


Our system can support multiple pre-trained models, such as Skylark, ChatGLM, etc.; support multiple efficient parameter tuning methods, such as LoRA, Prompt Tuning, etc.; support custom privacy enhancement parameters to ensure controllable model performance loss.

Note 1: LDP refers to local differential privacy.

Note 2: PAC refers to "probably approximately correct".

We federated fine turn the Roberta-large language model under the financial sentiment analysis dataset financial phrasebank [1], and the experimental results are as follows:

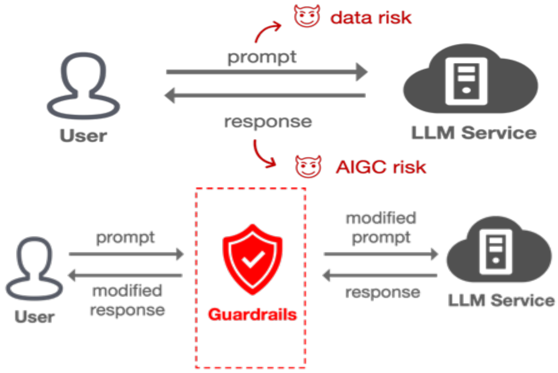


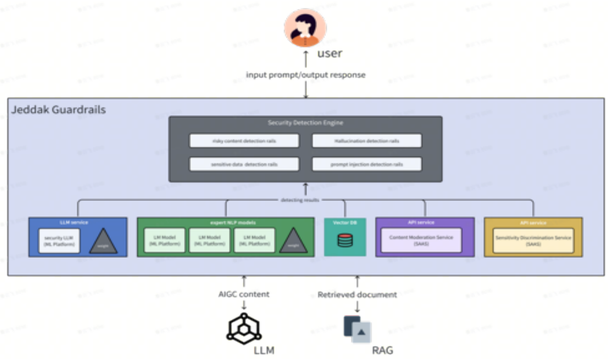
In the case of accuracy loss < 2%, the effect of resisting Embedding Inversion attacks: about 50% of the words cannot be correctly restored.

- Plaintext：5月9日消息，近日，北京海阔天空投资管理有限公司南昌负责人张三涉嫌非法吸收公众存款罪，被南昌市东湖区检察院依法提起公诉，涉及非法吸存金额达2000万元。
- Attack to cipher：夫特adiumlas，options[，北京ழுறdefin海ulos 꿰ཀ ଢାব人称被臣涉嫌非法吸收公众存款罪。被 Hyp सालमा ದ್ವಜ坐了对称检察院依法提起公िটির ரட非法吸存金额ดள far_fields。

2.3 Privacy Guardrail

In this section we consider two risks in large language model applications: one is AIGC risk, including harmful content/model illusion/prompt attacks. The other is data risk, mainly referring to the risk of privacy leakage.





Harmful content refers to illegal and irregular content, prejudice, hatred, discrimination, and insults; hallucinations refer to model output content that does not match context or facts, common sense; hint attacks refer to malicious attackers using jailbreak and other hint words to induce the model to output risky content. Data risks include personal data privacy leaks, mainly caused by the model remembering sensitive privacy information.

We have developed a model data privacy guardrail system to

resist these two risks. The functions of the model data privacy guardrail system include:

• Content security compliance: filtering harmful content, content compliance legal and non-toxic

• The content is true and credible: to prevent the output of misinformation and ensure that the answer is true and credible

• Prompt attack protection: jailbreak and prompt injection detection, protection model is not attacked

• Privacy leak protection: privacy sensitive data detection and desensitization to prevent leakage

The core technologies used in the model data privacy guardrail system include four aspects:

• Hybrid expert strategy: including rule matching + vector recall + safety large language model

• Security agent: CoT-based security prompt words, combined with RAG, Agent to achieve controllable behavior

• Active learning: automatic data exploration and utilization based on reinforcement learning, and model forgetting

• Privacy Enhancement Technology: Text vector perturbation based on differential privacy, parallel replacement can restore desensitization

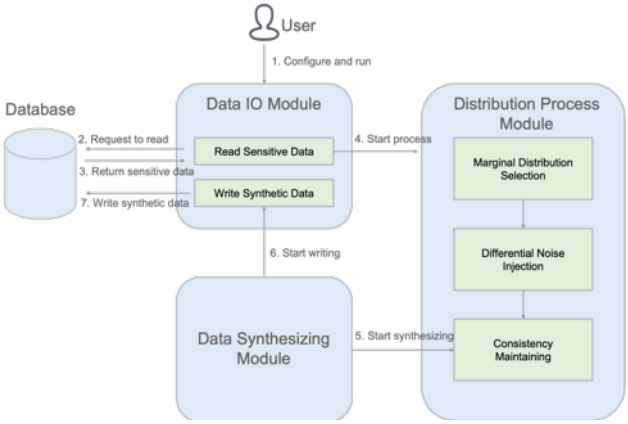
We report on the effectiveness of our Privacy Guardrail system in two applications:

The protection effect for a certain advertising model within the company is as follows: the success rate of blocking harmful content is 81% - > 95%, and the success rate of blocking prompt attacks is 82% - > 96%.

For a large language model of the company's internal real estate protection effect is as follows: illusion output reduced by 60%, mobile phone number and other PII privacy protection rate of 95%.

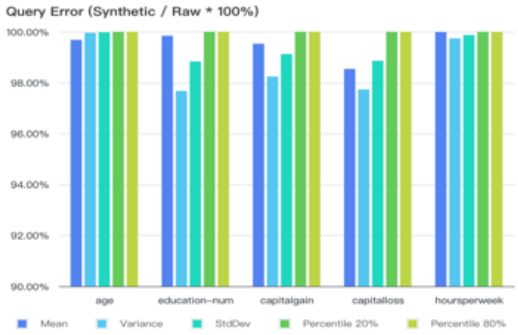
2.4 Data Synthesis

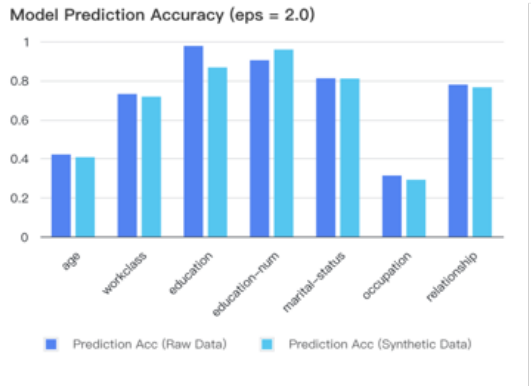
In order to improve the ability of large language models and downstream task performance, a large amount of high-quality sample data is needed for Model Training and fine-tuning. However, there are often two pain points: on the one hand, the privacy and legality of sample data are difficult to guarantee; on the other hand, the limited sample size in real-world scenarios will affect the model effect.



In order to solve these two pain points, we provide data synthesis technology to generate training samples. Through techniques such as distributed sampling and generative networks, we obtain "newly created" data that is close to the original sample distribution, solving the problem of insufficient samples. We also apply differential privacy in the generation process to protect sensitive information and mitigate risks. We can support multidimensional and MultiModal Machine Learning data synthesis, which is both applicable and usable.

We report the application effect of our data synthesis technology as follows: In the process of training the user portrait attribute prediction model, we sample and synthesize the original samples, then perform statistical analysis on the synthesized data, and perform DNN Model Training prediction. The attribute statistical indicators and model effects of the generated data are close to the performance of the original data source.





2.5 Domain large language model: JeddakLLM

In order to provide a professional intelligent assistant for the development, use and operation of privacy computing products. We rely on high-quality domain data, efficient information retrieval capabilities and a highly professional prompt vocabulary to develop JeddakLLM. At present, JeddakLLM has been applied to the following scenarios:

Scenario 1: Jeddak product deployment Copilot

JeddakLLM can provide pre-deployment consultation for products (such as network strategy, machine resource requirements); provide product deployment and upgrade steps; assist customers in data backup and restoration.

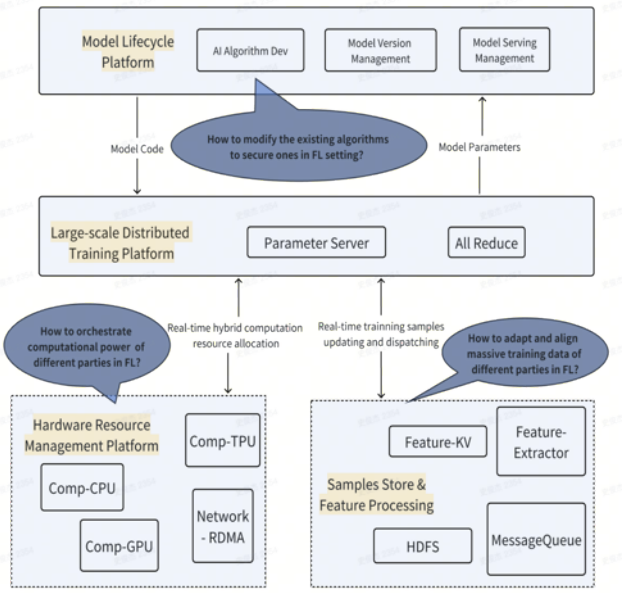
Scenario 2: privacy computing consulting assistant

JeddakLLM can provide knowledge consultation in the field of privacy computing.

2.6 AI Native FL

In the industrial world, when implementing privacy computing technologies such as FL and federated fine-tuning into production operations, there are challenges in large-scale engineering. Taking ByteDance as an example, various AI models, including large language models, use the company's unified AI infra to manage massive training data sources, dynamically distribute computing resources, and manage Model Iteration. Business parties facing specific security and privacy issues deeply rely on using these public AI infrastructures to complete business processes. So, when we actually solve security issues in training and inference in various business scenarios, we need to make our solutions easy to integrate into the existing system processes of the business, and strictly ensure performance

and stability while ensuring functionality and security.

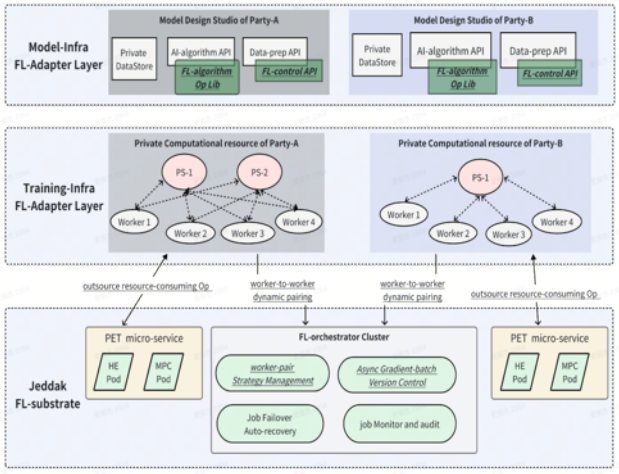


We propose an AI-native large-scale federated training architecture, Native-FL, to address this issue. Under the Native-FL architecture, business parties can easily federalize existing large language models and Recommender system models with one click on the existing model management platform. At the same time, the transformed federated training and federated fine-tuning can be processed on a large scale based on the training mode of the AI platform's parameter server. It is fully compatible with inherent components such as computing power scheduling, massive data preprocessing, and feature processing on the existing AI platform. At the same time, federated training and fine-tuning can still meet the extreme requirements for performance and reliability in actual online business, supporting the update training of billions of new training samples every day.

Behind this architecture, we have done a series of innovative work. In our non-invasive "embedded" architecture design, we do not modify most of the existing AI native components, but instead implement the privacy technologies we need, such as PSI and DP noise addition, in the native extension of the AI platform. At the same time, we do not modify the existing training scheduling method of the platform itself but introduce lightweight additional federated training coordination components to adapt to single-party distributed training models such as parameter servers. For example, for two self-scheduled trainers, they are paired and associated in real time with a certain trainer of the collaborating party through the federated scheduler after startup, and jointly complete the federated training or fine-tuning of a batch of training samples. This achieves horizontal scalability for massive training computing power.

In addition, we use a series of algorithms and engineering innovations to achieve performance comparable to plaintext unilateral training. For example, we draw on the multi-machine asynchronous training model under the parameter server architecture, and version control the federated training layer for interaction between the two parties, thereby achieving configurable two-party paired trainer asynchronous training.

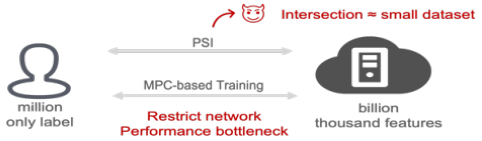
In the cross-business federated Recommender system modeling application, Business A requires compliance to use the user characteristics of Business B to fine-tune and optimize the recommendation recall model. Using our Native-FL framework, federated training of daily 200 million users and 1 billion samples is achieved, and the online Recommender system model is updated in real-time. After AB-TEST, the click-through rate of the federated model content significantly increased by + 1.45% , and the overall 3-day retention of Business A significantly increased by + 0.11% (meeting business expectations).



2.7 Full-Encrypted Machine Learning

The background of Full-Encrypted Machine Learning is the joint modeling of ByteDance and Financial Institution. This scenario has the following characteristics: **high security**: Financial Institution has particularly high requirements for data security and does not want to leak any information; **unbalance**: ByteDance has 1 billion-level users and thousand-dimensional characteristics, while Financial Institution only has millions of users and only labels; **WAN**: bandwidth is only 100Mbps, and latency is 20ms.

The security risk of this scenario is that if the traditional PSI federated/MPC training method is used for joint modeling, the intersection will be approximately equal to the small data sample set due to the large data sample set being close to the complete set.



Traditional Federated ML training

The performance challenge brought by this scenario is that if traditional ECDH-based PSI with payload is used, the full data ciphertext needs to be sent at least once. However, under 100Mbps internet bandwidth, it takes 7.4 days to send the full plaintext of 1 billion x 1000-dimensional data once.

In response to the security risks and performance challenges of this scenario, we present our "Full Encrypted Machine Learning" solution. The features of this solution are high security: no leakage of raw input, intersection ID, features and labels; high performance: million vs 1 billion, thousand dimensional features, running time from weeks to days.

The technical principle of our " Full Encrypted Machine Learning" solution is mainly:

In response to security risks, we use [PSI to Share] to replace traditional PSI, so that both parties cannot know the intersection ID, but obtain the share of the payload (feature, label) corresponding to the intersection.

In response to performance challenges, we use a protocol that high computation complexity in ByteDance and low computation and communication complexity in Financial Institution and

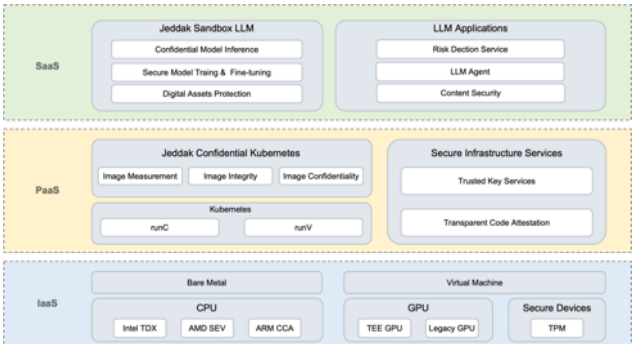
utilize the Spark cluster in ByteDance.

The experimental results show that the running time of our PSI to share protocol is 17h50min for a million VS 1 billion + thousand-dimensional datasets in WAN network environment.

3. Environment and Infrastructure Security System in Jeddak

3.1 Jeddak Confidential Computing Infrastructure

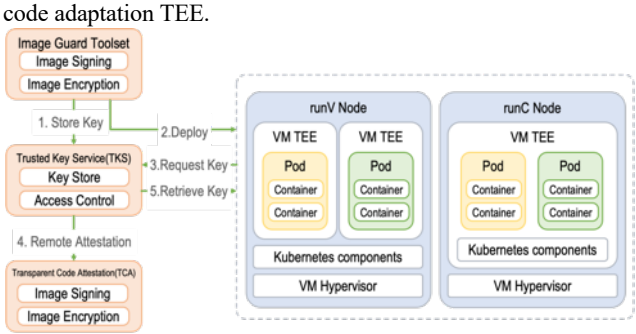
The Jeddak confidential computing infrastructure is divided into three tiers:



1. IaaS layer: integrates TEE hardware with multiple architectures, supports heterogeneous secure computing power; adapts to multiple cloud computing resources, fully leverages TEE security capabilities, and supports complex computing scenarios on the user side.
2. PaaS layer: Ensure container confidentiality, integrity, and trustworthiness, support multiple container runtimes; and seamlessly integrate with existing Cloud Native systems based on K8s security design.
3. SaaS layer: Ensure the security and verification of user asset code and data; and provide large language model services based on TEE design, providing security reinforcement for the entire lifecycle of large language models.

3.2 Key Technology 1: Jeddak Confidential Kubernetes, JCK

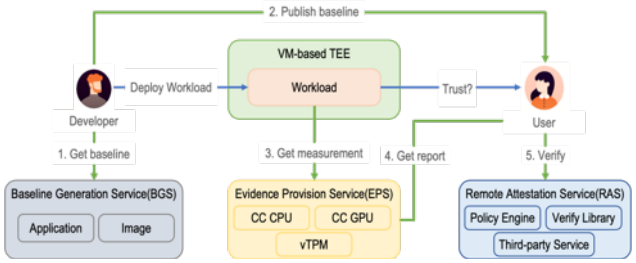
Based on the most widely used container orchestration tool Kubernetes, we designed a secure container solution based on the virtual machine TEE. Containerization reduces deployment complexity and minimizes the transformation cost of business code adaptation TEE.



Our secure container solution supports multiple Lines of Business, deploys secure high-performance service clusters through JCK, and has no significant difference in computing performance compared to non-TEE environments.

3.3 Key Technology 2: Transparent Code Attestation (TCA)

The problem to be solved in this part is how developers can prove to users that the code running in VM-Based TEE is consistent with the public Baseline.



The TCA system we developed consists of three sub-services:

1. Baseline Generation System (BGS),
2. Evidence Provision Service (EPS)
3. Remote Attestation Service (RAS)

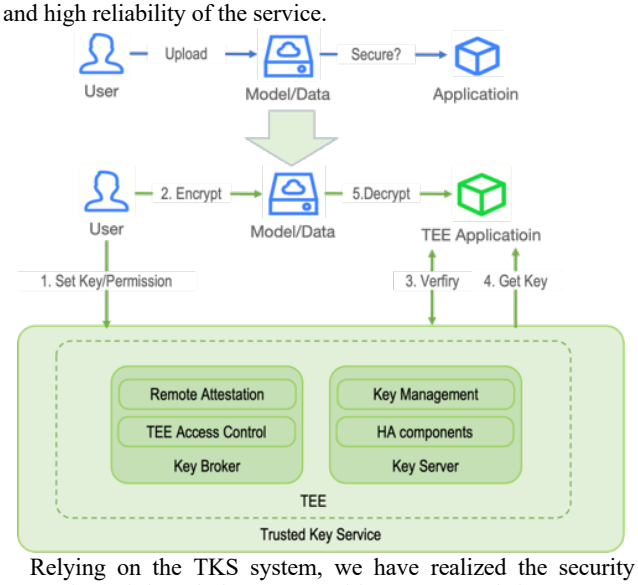
By using these three sub-services, the entire code authentication process has been connected, and a production-usable remote proof system has been realized. The entire process of R & D-deployment-launch has been adapted, and it will soon be launched and support the company's full confidential computing business.

3.4 Key Technology 3: Trusted Key Service (TKS)

The problem we need to solve here is trusted CDKEY management. Generally, the requirements for our CDKEY management system are as follows:

1. Ensure that only legitimate services can obtain the key
2. Ensure the trustworthiness of the entire process environment of key management
3. Ensure high availability of key access

Our TKS system consists of a Key Brocker subsystem and a Key Server subsystem. The Key Brocker subsystem implements access control based on TEE, and the Key Server subsystem uses TEE security mechanism to protect the key. And use clustering and leader/follower architecture design to ensure high throughput and high reliability of the service.



Relying on the TKS system, we have realized the security protection of the whole process of key usage and data storage, and the service response time is less than 10ms.

3.5 Key Technology 4: TEE with Legacy GPU

When applying confidential computing to provide large-scale model services for business parties, an unavoidable obstacle is the GPU. The actual needs in the business require us to build secure heterogeneous computing power. We have collected and summarized the requirements for confidential computing in large-scale model application scenarios and emphasized the pain

points of existing heterogeneous confidential computing.

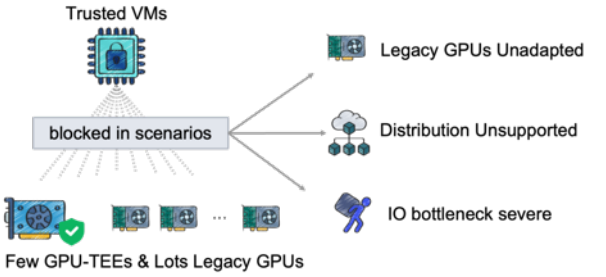
First, heterogeneous confidential computing requires the computing power of a wide range of legacy GPU cards, and limiting specific CPU and GPU models will greatly increase computing power costs.

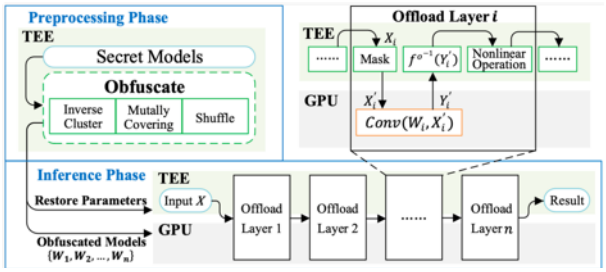
Secondly, large-scale businesses urgently require heterogeneous confidential computing to support large-scale distributed training and pushing, providing a relatively complete cluster ecosystem.

Finally, the current heterogeneous secret computing performs poorly in IO-intensive scenarios (training, etc.), and optimization needs to be carried out at the system level.

Targeting these pain points, we trained and developed a heterogeneous confidential computing framework at the algorithm level for large language models. First, we evaluated the work in this field and broke through the cutting-edge solutions based on confusion protection in the traditional DNN model field for the past two years. Then, we proposed a new protection method based on the attack method. [2]

Following the preliminary exploration results, we are designing a large-model heterogeneous security training and inference scheme based on transformer, while conducting engineering optimization at the system level. The current implementation method relies on a distributed framework to achieve communication between TD VM and GPU, with inference latency of about 4x of pure GPU and throughput of 70%. To improve efficiency, we are exploring the direct connection between TD VM and legacy GPU, and using DPDK to improve the IO efficiency of TD VM and the outside world, including network optimization, with an expected latency of about 2x.



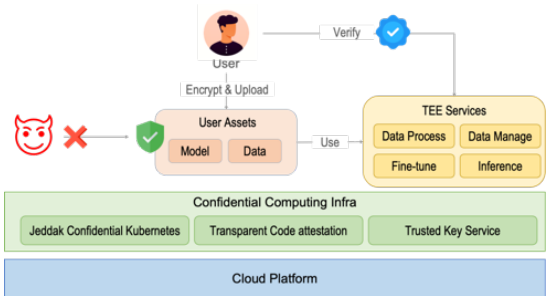


3.6 Asset protection scenarios

Consider the model and data security when using Cloud Computing Platform. This involves complex calculations and data flow processes.

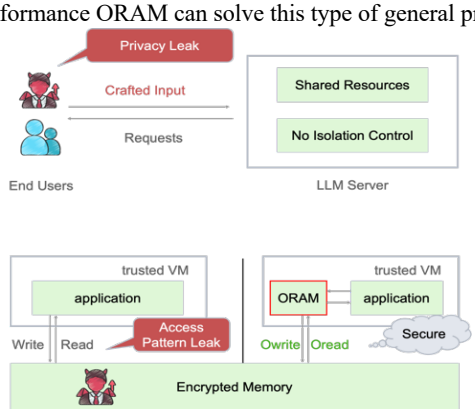
We build secure services based on confidential computing infrastructure, allowing user assets to be used only in TEE and verifying the credibility of asset processing in TEE.

Based on this, the full-process protection of user assets is realized, and the performance impact of pure CPU computing tasks is less than 5%.



3.7 Prospective problems

We are also committed to exploring the cutting-edge security issues faced by large language models in the real world and in theory. For example, in real applications, in order to pursue performance, large language models generally adopt multi-tenant services, serving multiple users on the same infrastructure. This efficiency-first operation brings privacy leaks. We have conducted research on high-performance LLM multi-tenant frameworks represented by vLLM and SGLang, with the goal of accurately recovering the original input of other tenants from shared caches. Similarly, we focus on the security issues designed by TEE itself. For example, TEE did not consider protecting the access mode of memory read and write during design, which brought certain privacy leaks. Designing feasible high-performance ORAM can solve this type of general problem.



4. Summary and Outlook

Security and trustworthiness is the core proposition of LLM technology development, and data privacy and model asset security are the key. We have built an endogenous data security system and the Jeddak large-scale model security capability system of the environment and infrastructure security system is used to ensure the data privacy and model asset security in the application of LLM technology.

In the next stage, we will promote a variety of technology cross-fusion innovation, enrich endogenous security mechanisms, conduct forward-looking research in researching and perceiving new security risks and designing protection plans; and continuously improve end-to-end security capabilities to escort large language model applications.

参考文献

[1] P. Malo and A. Sinha and P. Korhonen and J. Wallenius and P. Takala. Good debt or bad debt: Detecting semantic orientations in economic texts. Journal of the Association for Information Science and Technology. 2014, volume 65..

[2] GroupCover: A Secure, Efficient and Scalable Inference Framework for On-device Model Protection based on TEEs. Accepted by ICML2024.