EAI Endorsed Transactions
on Scalable Information Systems
Research Article EAI.EU

A Communication-Reduced Privacy-Preserving ViT Inference Framework for Distributed Edge Intelligence

Tingting Chen[1,*]

[1]School of Information Engineering, Jiangsu Maritime Institute, Nanjing, 211170, China

Abstract

To address privacy leakage from user images, intermediate representations, and outputs during Vision Transformer (ViT) inference in distributed edge services, this paper presents SViT, a two-server secret-sharing framework with offline correlated randomness. The revised design specifies fixed-point arithmetic over the ring Z_(2^64) with 16 fractional bits, fresh one-time masks, probabilistic truncation, numerical ranges, and complete input-output procedures for SExp, SDiv, SSqrt, SVar, SLayerNorm, SSoftmax, and SGeLU. The protocols use fixed-depth range reduction, lookup-assisted initialization, and a constant number of Newton updates, so their online depth is independent of numerical convergence tolerances. Under the semi-honest, non-colluding-server model, the revised security analysis defines approximate ideal functionalities, public leakage, simulator inputs, and sequential composition. Existing microbenchmarks show 2.28-6.50 times lower runtime and 4.00-14.20 times lower online communication than CrypTen for the reported core operators; for SDiv, runtime decreases from 9.1 ms to 1.4 ms and communication from 10.8337 MB to 0.7629 MB. Synthetic Q16 numerical checks report maximum absolute errors of 5.10×10−5 for SExp and 4.73×10−4 for SGeLU, maximum relative errors of 1.27×10−5 for reciprocal and 3.72×10−5 for square root, and 100% top-1 consistency over 10,000 randomly generated Softmax vectors. The reported end-to-end result remains 3799.744 ms and 2.85 GB per inference; therefore, SViT is described as communication-reduced relative to the evaluated MPC baselines rather than universally lightweight, and its present practical scope is primarily high-bandwidth LAN or provider-edge deployments.

Keywords: privacy-preserving inference, secret sharing, Vision Transformer, fixed-point arithmetic, distributed edge intelligence, communication reduction

Received on 06 June 2026, accepted on 03 August 2026, published on 20 August 2026

doi: 10.4108/eetsis.13362

*Corresponding author. Email: ctt@jmi.edu.cn

1. Introduction

1.1 Research Background and Significance

With the integrated development of distributed networks, edge computing, and artificial intelligence services, visual perception models are gradually moving from centralized cloud deployment to edge nodes close to data sources. The Transformer architecture has promoted the development of large-scale vision and language models [1]. ViT divides images into serialized patches and uses self-attention for global feature modeling, achieving strong performance in image classification and visual understanding [2]. Transformer-based models, including BERT and foundation models, have consequently enabled a broad range of language and vision inference services [3-4].

In these scenarios, edge computing offloads computational tasks to locations close to users and devices, thereby reducing network latency, relieving cloud-side load, and improving real-time responsiveness [5]. Edge intelligence further deploys model inference, data analysis, and intelligent decision-making at the edge, enabling complex visual perception tasks to be performed in

resource-constrained, network-heterogeneous, and data-distributed environments [6-7]. However, edge visual inference often requires users to upload raw images, feature representations, or interaction data, which may contain faces, identity information, geographical locations, industrial defect images, medical images, and business-sensitive information. Once edge nodes or transmission links are attacked, user privacy, business data, and inference results may be leaked. Therefore, building a secure, efficient, and lightweight privacy-preserving ViT inference framework for distributed edge intelligence systems has important research significance and practical value.

1.2 Existing Problems

Although secure multi-party computation, homomorphic encryption, and secret sharing have been applied to private inference, ViT deployment at the edge remains difficult for three measurable reasons. First, Softmax, LayerNorm, GeLU, reciprocal, and square-root evaluation dominate interactive secure computation. Second, communication rather than arithmetic can become the bottleneck in geographically distributed deployments. Third, preserving the published ViT-B/16 topology does not by itself guarantee numerical or classification equivalence: fixed-point encoding, clipping, truncation, and nonlinear approximation must be quantified. Accordingly, this revision separates protocol correctness, numerical fidelity, model-level accuracy, and deployment practicality instead of treating them as a single “lightweight” claim.

The revised study therefore asks four questions: (i) which operator constructions differ from prior private-Transformer systems; (ii) under what fixed-point and leakage assumptions the protocols are correct and simulatable; (iii) which costs are shifted offline and which remain online; and (iv) under what bandwidth and latency conditions the reported 2.85 GB end-to-end communication is practically usable. The term communication-reduced is used whenever the evidence is relative to CrypTen or MP-SPDZ, while broader edge-lightweight claims are avoided.

1.3 Limitations of Existing Schemes

Existing private-inference studies can be grouped into general secret-sharing/MPC systems, hybrid HE-MPC systems, and Transformer-specific systems. SecureML, MiniONN, GAZELLE, XONN, SecureNN, FALCON, Delphi, CrypTFlow2, SiRNN, CrypTen, ABY2.0, and MP-SPDZ provide reusable arithmetic, comparison, truncation, and mixed-protocol components [8-22]. Transformer-oriented frameworks such as MPCFormer, THE-X, Iron, SecFormer, PUMA, BOLT, BumbleBee, and BLB optimize nonlinear functions, matrix multiplication, HE-MPC conversion, or model transformation [23-28,31-32]. Complementary recent work studies activation-function co-design and maliciously secure PPML [29-30], while SecViT explores communication- and latency-efficient ViT adaptation [33]. Together, these systems establish a substantially stronger baseline than a general MPC framework alone.

The closest comparison is operator-level rather than merely architectural. Iron, BOLT, BumbleBee, and BLB protect both inputs and proprietary model parameters in a two-party client-model-owner setting and optimize Transformer matrix multiplication and nonlinear functions [25,28,31-32]. SViT instead assumes a public or jointly authorized ViT-B/16 model and distributes each activation tensor between two non-colluding edge servers; its specific contribution is the combination of an offline-dealer sharing pattern with Q16 fixed-depth protocols for the ViT nonlinear pipeline. This distinction narrows the novelty claim: SViT does not introduce the first private Transformer inference framework, and it does not provide model privacy in the current configuration. Non-interactive HE-based inference such as NEXUS further illustrates the rapidly changing communication frontier [34].

1.4 Contributions of This Paper

To address the above problems, the revised manuscript presents SViT as a communication-reduced privacy-preserving ViT inference framework. Its claims are restricted to the semi-honest, non-colluding-server model and to the specific fixed-point/operator configuration described in Sections 3-6. The main contributions are as follows.

(1) A distributed inference and leakage model is specified for a user, two non-colluding edge servers, and an offline trusted dealer. Input, intermediate-feature, and output shares are protected, while tensor dimensions, message lengths, timing, and the public model configuration are explicitly treated as leakage.

(2) Complete fixed-point procedures are provided for SExp, SDiv, SSqrt, SVar, SLayerNorm, SSoftmax, and SGeLU, including input domains, range reduction, truncation, fresh masks, output scaling, protocol composition, and symbolic online-cost expressions.

(3) The published ViT-B/16 topology is retained, but the revised manuscript no longer equates structural preservation with zero accuracy loss. Numerical fidelity is evaluated separately, and plaintext-versus-secure top-1/top-5 validation is identified as a required model-level experiment when prediction traces and checkpoints are available.

(4) The existing protocol microbenchmarks and end-to-end overhead are retained, supplemented by Q16 numerical checks and bandwidth-constrained lower-bound analysis. The results show relative communication reduction against the tested MPC baselines while also identifying the 2.85 GB communication cost as a material deployment limitation.

2. Related Work and Problem Analysis

2.1 Privacy-Preserving Machine Learning Inference

Privacy-preserving machine learning inference aims to enable the server to compute model outputs without obtaining users' plaintext inputs, while allowing users to obtain inference results. SecureML was among the early works that applied secret sharing to scalable privacy-preserving machine learning and proposed fixed-point arithmetic computation and secure approximations for common nonlinear functions [8]. MiniONN transformed neural networks into forms suitable for secure two-party computation and implemented privacy-preserving prediction based on secret sharing and mixed protocols [9]. DeepSecure used garbled circuits to achieve provable security for deep learning inference, but its communication and computation overheads are high for large-scale visual models [10]. GAZELLE combined homomorphic encryption and garbled circuits for low-latency secure neural network inference [11]. XONN constructed efficient oblivious inference protocols for binarized neural networks and has advantages in reducing communication, but it relies on network binarization and has limited adaptability to general ViT structures [12].

Overall, early privacy-preserving inference schemes mainly focused on CNNs or shallower models and provided good support for operators such as convolution, fully connected layers, ReLU, and Maxpool. However, self-attention, Softmax, GeLU, and LayerNorm in ViT impose higher requirements on protocol design. Especially in edge scenarios, inference services require not only security but also low latency, low communication, and high throughput. Therefore, lightweight secure protocols need to be redesigned for Transformer structures.

2.2 Secure Multi-Party Computation and Secret-Sharing Protocols

Secure multi-party computation provides a fundamental basis for privacy-preserving inference. SecureNN and FALCON extend neural network training and inference capabilities from the perspectives of three-party secure computation and malicious security, respectively [13-14]. Delphi reduces inference costs through the co-design of cryptographic protocols and machine learning models, demonstrating the importance of joint optimization between protocols and models [15]. CrypTFlow and CrypTFlow2 further optimize comparison, division, and large-scale DNN inference in secure inference, supporting more realistic deep neural network scenarios [16-17]. SIRNN designs dedicated secure protocols for mathematical functions in RNNs, such as exponentials, sigmoid, tanh, and inverse square root, showing that protocol-level optimization for complex functions can significantly reduce communication overhead [18]. CrypTen provides a secure multi-party computation framework for machine learning and facilitates the construction of privacy-preserving model inference systems [19].

At the lower protocol layer, ABY provides an efficient mixed-protocol framework among arithmetic sharing, Boolean sharing, and garbled circuits [20]. ABY2.0 further optimizes the online communication overhead of semi-honest two-party computation and provides an important basis for lightweight secret-sharing inference [21]. MP-SPDZ provides a general implementation platform for various secure multi-party computation protocols [22]. However, general-purpose frameworks usually require conversion among multiple protocols or iterative approximation of complex functions, which still leads to large online communication when directly applied to ViT inference. Therefore, this paper further constructs dedicated secure protocols for key ViT operators to improve edge inference efficiency.

2.3 Privacy-Preserving Inference for Transformers and ViT

With the rise of BERT, ViT, and foundation models, private Transformer inference has progressed rapidly [2-4]. MPCFormer uses MPC with knowledge distillation; THE-X combines homomorphic encryption with nonlinear approximation; Iron designs private protocols for Transformer linear and nonlinear operators; and SecFormer and PUMA optimize secure inference for larger Transformer models [23-27]. BOLT reduces Transformer-inference cost through operator-oriented protocols, BumbleBee emphasizes communication-friendly two-party inference, and BLB fuses fine-grained operators in a hybrid CKKS-MPC pipeline [28,31-32]. SecViT further studies communication- and latency-efficient private inference for Vision Transformers [33]. Importantly, these systems make different model and protocol trade-offs: MPCFormer employs knowledge distillation, THE-X introduces HE-compatible nonlinear approximations, and SecFormer and PUMA retain standard Transformer components more closely while still realizing expensive nonlinear operations through secure protocols or numerical approximations [23-27]. These differences motivate protocol-level comparison rather than a broad claim that preserving the original architecture is unique.

The comparison also shows that “preserving the model” has multiple meanings. SViT retains the ViT-B/16 layer topology but uses fixed-point representations and secure approximations; MPCFormer changes the model through distillation; THE-X and several hybrid systems approximate nonlinear functions; Iron, BOLT, BumbleBee, and BLB focus on protocol or representation changes without necessarily preserving identical floating-point execution. Therefore, the revised evaluation distinguishes topology preservation, numerical error, prediction consistency, model privacy, and communication cost.

2.4 Edge Intelligence and Distributed Privacy Protection

Edge computing and edge intelligence move computing, storage, and inference capabilities closer to data sources, offering advantages in reducing latency and saving bandwidth [5-7]. However, the openness, heterogeneity, and

distributed nature of edge nodes also introduce more complex data security risks. Compared with centralized cloud inference, edge inference involves more cross-node data flows, more intermediate states, and more communication interfaces. Attackers may infer user privacy from a single node, local cache, or communication messages. Therefore, privacy-preserving inference in edge intelligence systems is not only an algorithmic security issue but also a data security problem in distributed network systems.

Related studies in scalable information systems have examined privacy-preserving data sharing through multi-layer access control [35] and secure edge-based data processing in distributed infrastructures [36]. These studies support the joint consideration of privacy, communication efficiency, and distributed deployment, but they do not provide SMPC protocols for ViT nonlinear operators or the two-non-colluding-server inference model studied here.

Existing edge intelligence studies mostly focus on model compression, task offloading, and inference scheduling, while privacy-preserving mechanisms are often treated as additional modules. For large models such as Vision Transformers, directly using high-overhead secure protocols weakens the real-time capability of edge deployment; excessive compression or replacement of key functions may also harm inference accuracy. Therefore, this paper combines secret sharing with optimization of key ViT operators and performs secure inference collaboratively through two edge servers, balancing privacy protection, communication efficiency, and model accuracy.

A synthesis of existing studies shows that privacy-preserving machine learning inference has extended from early CNN models to Transformer models, but distributed edge ViT inference still faces the following limitations. First, general-purpose secure computation frameworks provide insufficient support for key nonlinear functions in ViT, and direct reuse leads to high computation and communication overhead. Second, representative Transformer schemes make different trade-offs: MPCFormer uses knowledge distillation, THE-X uses HE-compatible function approximation, and other systems retain the Transformer structure more closely but still incur secure-nonlinear and communication costs [23-27]. Third, existing studies focus more on cloud or client-server models and provide insufficient discussion of collaborative inference between two independently administered, non-colluding edge servers. Fourth, online communication, inference latency, and end-to-end deployability in edge environments still require further optimization. Therefore, this paper presents a communication-reduced privacy-preserving ViT inference framework for distributed edge intelligence.

To further highlight the differences between the proposed method and representative privacy-preserving inference schemes, this paper compares them in terms of core techniques, applicable models, preservation of the original Transformer/ViT structure, and edge adaptability, as shown in Table 1.

Table 1. Comparison of representative privacy-preserving inference schemes

Method	Security / parties	Protected assets & model privacy	Nonlinear strategy / model change	Reported deployment limitation
SecureML / MiniONN	Semi-honest; 2P/3P variants	Input; model privacy depends on setting	Mixed sharing/GC; CNN-oriented	High overhead for ViT nonlinear operators
CrypTen / MP-SPDZ	General MPC; configurable	Input/model depend on sharing policy	General arithmetic and conversions; no ViT specialization	Requires operator-specific tuning
MPCFormer / THE-X	2P; MPC or HE	Input and usually model	Distillation or function approximation	Accuracy or conversion overhead
Iron / BOLT	2P client-model owner	Input + proprietary model	Dedicated Transformer protocols	Cloud/client-server focus; substantial bandwidth
BumbleBee / BLB	2P; semi-honest	Input + model	Optimized MM/nonlinear or CKKS-MPC fusion	Large-model focus and trusted assumptions
SViT (revised)	2 non-colluding edge servers + offline dealer	Input/intermediate/output; model public	Q16 fixed-depth nonlinear protocols; topology retained	2.85 GB/inference; semi-honest; LAN-oriented

Table 1 now uses verifiable comparison criteria. SViT differs from Iron, BOLT, and BumbleBee primarily in deployment ownership: activations are split across two non-colluding edge servers, whereas several closest systems protect a client input and a model owner's parameters. This choice reduces the model-privacy scope and introduces a trusted-dealer assumption, but supports cross-provider edge execution when both servers are independently administered.

2.5 Protocol-Level Distinction and Scope of Novelty

The revised novelty claim is limited to the following integration. First, SViT fixes a common arithmetic representation for all nonlinear operators and explicitly connects SExp, SDiv, SSqrt, and SVar to SSoftmax, SLayerNorm, and SGeLU. Second, lookup-assisted initialization and a fixed number of Newton updates replace convergence-dependent iterations; the protocol depth is therefore input-size dependent but tolerance independent. Third, the trusted dealer generates fresh correlated randomness offline so that online execution between the two edge servers uses only shares and preprocessed masks. These components are not claimed individually as first inventions; the contribution is their ViT-B/16-specific composition, quantified operator implementation, and distributed edge ownership model.

The present framework does not protect model parameters, does not tolerate collusion, and does not provide malicious security. Moreover, retaining the layer topology does not prove unchanged classification accuracy. These boundaries are stated explicitly to avoid overlap with

broader claims made for model-private Transformer systems [25,28,31-33].

3. System Model and Threat Model for Distributed Edge Intelligence

3.1 System Model

The SViT system consists of a user U, two independently administered edge servers P0 and P1, and an offline trusted dealer T. In the default configuration evaluated in this paper, the trained ViT-B/16 parameters are public to both edge servers or jointly authorized for deployment; model confidentiality is not provided. The user splits the image tensor into additive shares and sends one share to each server. T generates Beaver triples, truncation masks, lookup masks, and other correlated randomness before inference, distributes one share to each server, and then goes offline. P0 and P1 execute the embedding, Transformer encoder, and classifier on secret-shared activations. The user reconstructs either the top-1 label or, if explicitly requested, a probability vector from two output shares. The use of Beaver-triple preprocessing follows circuit-randomization-based multiparty protocols [37].

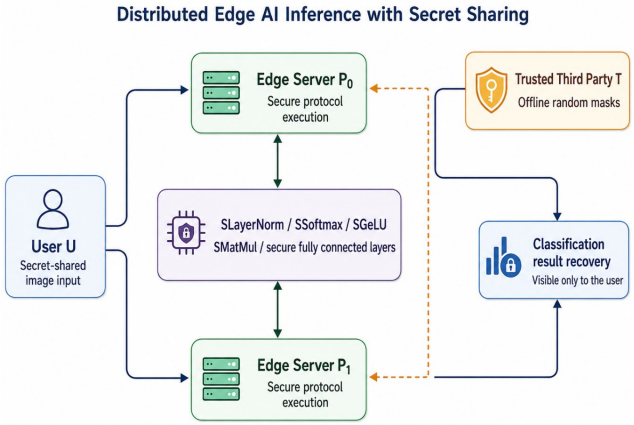

Figure 1. SViT system model for distributed edge intelligence

Figure 1 illustrates the ownership and trust boundaries: no single edge server receives a complete input or activation tensor, but the model structure, tensor dimensions, execution schedule, and message sizes are public. Returning only a top-1 label minimizes output leakage; returning a complete class-probability vector may reveal more information about the model and training distribution and is therefore treated as an optional policy choice rather than a default guarantee.

Let the arithmetic ring be $R = \square_2 \ell$, with $\ell = 64$ in the reported configuration. A real value x is encoded as $\mathrm{Enc}_f(x) = \mathrm{round}\left(2^f x\right) \bmod 2^\ell$ with f=16 fractional bits. For an encoded image tensor X, the user samples $[X]_0$ uniformly in R and sets $[X]_1$=X-$[X]_0$ mod 2^ℓ. Server P_b holds only $[X]_b$. Intermediate tensors and outputs use the same additive-sharing convention unless a protocol explicitly converts to Boolean or multiplicative shares.

$$\boldsymbol{X} = [\boldsymbol{X}]_0 + [\boldsymbol{X}]_1 \quad (mod \quad 2^l) \tag{1}$$

A single share is uniformly distributed conditioned on the public tensor shape and therefore hides the encoded value. Correct reconstruction requires both shares. Fresh preprocessing is mandatory: reusing a mask or Beaver triple across executions can expose linear relations between private values and is prohibited by the protocol identifier and monotonic mask counter described in Section 4.1.

3.2 Threat Model

The formal model is static semi-honest security with at most one corrupted edge server. P0 and P1 follow the protocol but may inspect their local shares, correlated-randomness shares, transcript, timing, and intermediate states. They are assumed not to collude. T is trusted to sample correct, independent, one-time randomness and erase its generation state after distribution. The user input is well formed and lies within the declared numerical domain. These assumptions are consistent with cross-provider deployments only when administrative and legal separation makes collusion unlikely; they are not a substitute for malicious security.

The public leakage function L contains the model identifier and topology, public model parameters, tensor dimensions, batch size, sequence length, fixed-point parameters (ell,f), protocol identifiers, prescribed message lengths, and execution timing. L excludes input pixels, activation values, output shares, random masks, and reconstructed labels. A corrupted server receives its own input share, preprocessing share, output share, and transcript; the security goal is that these values can be simulated from that local information and L.

SViT protects input confidentiality, activation confidentiality, and unreconstructed output confidentiality against one semi-honest server. It does not protect public model weights. If the deployment instead secret-shares model parameters, additional secure matrix-multiplication costs and a separate model-privacy proof are required and are outside the current measurements. Output confidentiality ends when the user reconstructs the result, and repeated probability-vector queries may enable black-box extraction or membership inference; rate limiting, top-k restriction, and output auditing are recommended.

Collusion between P0 and P1 immediately reveals all additive shares. Compromise of T can reveal or bias preprocessing correlations; reuse of offline masks can reveal differences between private values; maliciously selected inputs can trigger out-of-range arithmetic or model-extraction attacks; and active protocol deviation can corrupt results or create selective-failure leakage. These cases remain outside the proof, but their consequences are now explicit. Practical extensions include authenticated secret sharing, sacrifice checks for triples, verifiable preprocessing,

range proofs, message authentication codes, redundant execution, and dealer-free distributed preprocessing.

Table 2. SViT Threat Model and Privacy Protection Objectives

Item	Revised specification
Adversary	Static semi-honest corruption of at most one edge server
Non-collusion	P0 and P1 never combine shares outside the prescribed protocol
Trusted dealer	Offline-only; fresh one-time correlated randomness; erasure after distribution
Protected assets	Input images, intermediate activations, unreconstructed output shares
Model privacy	Not provided in the evaluated public-model configuration
Public leakage L	Topology, dimensions, batch/sequence length, message sizes, timing,
Numerical domain	Q16 in $\mathbb{Z}_{2^{64}}$; range checks and clipping as specified in Table 3
Out of scope	Collusion, malicious deviation, dealer compromise, mask reuse, malicious inputs
Recommended extensions	Authenticated sharing, verifiable/dealer-free preprocessing, output controls

Table 2 separates protected assets from public leakage and states the non-collusion and dealer assumptions. Accordingly, the security result in Section 5 is conditional: it proves privacy against one semi-honest server with fresh preprocessing, not against collusion or active attacks.

4. Communication-Reduced SViT Inference Framework Based on Secret Sharing

4.1 Fixed-Point Representation and Preprocessing

All secure arithmetic is performed in $R = \mathbb{Z}_{2^{64}}$ using signed two's-complement interpretation and f=16 fractional bits. A multiplication of two Q16 values produces Q32 and is returned to Q16 by probabilistic truncation TruncPr, which uses a fresh random mask and has an unbiased error bounded by one least-significant fixed-point unit except with negligible wraparound probability. Values are range checked before nonlinear evaluation. Inputs outside the declared domain are securely clipped, and denominators smaller than $\varepsilon=2^{-12}$ are replaced by epsilon through a secure comparison and selection. The fixed-point representation and probabilistic-truncation design follow established secure fixed-point computation principles [38].

The offline dealer produces authenticated protocol identifiers, Beaver multiplication triples, truncation masks, comparison/lookup masks, and reciprocal or inverse-square-root seed-table shares. Every preprocessing item is consumed once. The online phase contains no random-number generation by T. Table 3 summarizes the representation, numerical range, and stability rules used by all composite operators.

Table 3. Fixed-point and cryptographic configuration

Item	Setting	Valid domain	Stability / security rule
Ring / precision	$\mathbb{Z}_{2^{64}}$, f=16	Signed magnitude below 2^{47}	Abort or securely clip before wraparound
SExp	4th-order range-reduced polynomial	$x\in[-8,0]$	Values below -8 mapped to zero
SDiv	8-bit seed + 2 Newton updates	$\lvert b\rvert \geq 2^{-12}$	Secure sign handling; epsilon floor
SSqrt	8-bit inverse-sqrt seed + 2 updates	$x \in [2^{-12}, 2^{12}]$	Negative input rejected
SSoftmax	max subtraction + SExp/SDiv	logit gap clipped at -8	Top-1 default output policy
SGeLU	tanh form via SExp/SDiv	$x\in[-6,6]$	Secure clipping outside domain
Preprocessing	Fresh triples/masks per call	Unique session/counter	No reuse; dealer erases seed state

4.2 Basic Secure Primitives

The protocol notation [x] denotes an additive share pair over R.SAdd and multiplication by a public constant are local. SMul uses one Beaver triple; SCompare returns a shared bit; SSelect([b],[x],[y]) returns [b]x+(1-[b])y; TruncPr restores Q16 scaling after multiplication; and SLookup retrieves a secret-shared table entry without revealing the secret index. These primitives determine the round and communication expressions reported in Section 5.3.

4.3 Complete Protocol Specifications

The complete operator definitions are given below. Each protocol receives shares with the scale and domain in Table 3 and returns Q16 shares. Public constants are encoded once. All SLookup, SMul, SCompare, and TruncPr calls consume fresh preprocessing identified by (session,layer,operator,counter). The attention, Softmax, LayerNorm, and GeLU equations retained below define the target floating-point operators; Algorithms 1 and 2 define their secure fixed-point realization.

$$\mathrm{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathrm{Softmax}\left(\mathbf{Q}\mathbf{K}^{\mathrm{T}} / \sqrt{d_k}\right)\mathbf{V} \qquad (2)$$

Here, Q, K, V, and the attention output all participate in computation in secret-shared form. SSoftmax, SMatMul, and secure fully connected layers jointly support the implementation of secure multi-head attention. For classification probability computation, Softmax can be written as:

$$\mathrm{Softmax}\left(x_i\right) = \frac{\exp\left(x_i - \max(x)\right)}{\sum_j \exp\left(x_j - \max(x)\right)} \qquad (3)$$

The normalization operation LayerNorm can be expressed as:

$$\mathrm{LayerNorm}(\mathbf{X}) = \frac{\mathbf{X}-\mu}{\sqrt{\sigma^2+\varepsilon}}\boldsymbol{\gamma}+\boldsymbol{\beta} \quad (4)$$

The GeLU activation function in the feed-forward network adopts the commonly used approximation:

$$\mathrm{GeLU}(x) = 0.5x\left[1+\tanh\left(\sqrt{\frac{2}{\pi}}\left(x+0.044715x^3\right)\right)\right] \quad (5)$$

Equations (2)-(5) define the target operators. SExp, SDiv, SSqrt, and SVar are implemented as fixed-depth protocols in Algorithm 1; SLayerNorm, SSoftmax, and SGeLU compose these primitives as shown in Algorithm 2. The use of a fixed polynomial degree and exactly two Newton updates changes the complexity claim from convergence-dependent O(Nm) to O(N) for fixed public accuracy parameters, not to constant total work.

Algorithm 1. Fixed-depth basic nonlinear protocols

Protocol	Input-output specification and executable steps
$\mathrm{SExp}([x]) \to [e^x]$	1. Securely clip x to [-8,0]. 2. Compute shared k=round(x/ln2) and r=x-kln2. 3. Evaluate $p_4(r)=1+r+\frac{r^2}{2}+\frac{r^3}{6}+\frac{r^4}{24}$ with Horner SMul+TruncPr. 4 Obtain $[2^k]$ by SLookup over the finite exponent table. 5 Return $\mathrm{TruncPr}([2^k]\cdot[p4(r)])$; return 0 when x<-8.
$\mathrm{SDiv}([a],[b]) \to [a/b]$	1. SCompare enforces |b|>=epsilon and extracts sign. 2. Normalize |b| and privately select an 8-bit reciprocal seed y0. 3. For j=0,1 compute $y_{j+1}=\mathrm{TruncPr}(y_j(2-by_j))$. 4. Restore scale/sign and return TruncPr(a·y2).
$\mathrm{SSqrt}([x]) \to [\sqrt{x}]$	1 Reject x<0 and floor x at epsilon. 2 Normalize x and privately select an 8-bit inverse-square-root seed y0. 3 For j=0,1 compute y_(j+1)=TruncPr(0.5 y_j·(3-x y_j^2)). 4 Return TruncPr(x·y2) with exponent rescaling.
$\mathrm{SVar}([x_1,\ldots,x_N]) \to S[var]$	1. Compute $[\mu]=N^{-1}\sum[x_i]$ locally. 2. Compute $[d_i]=[x_i]-[\mu]$. 3. Compute $[s_i]=\mathrm{TruncPr}(\mathrm{SMul}([d_i],[d_i]))$. 4. Return $N^{-1}\sum[s_i]+\varepsilon$.

Algorithm 2. Composite ViT nonlinear protocols

Protocol	Input-output specification and executable steps
SLayerNorm ([X],γ,β)	1. [v]=SVar([X]). 2. [s]=SSqrt([v]). 3. For each element compute $[z_i]=\mathrm{SDiv}([x_i]-[\mu],[s])$. 4. Return $\gamma\cdot[z_i]+\beta$. γ/β are public in the evaluated model policy.
SSoftmax([x1 ...xN])	1 Compute shared maximum [M] by a comparison tree. 2 $[u_i]=[x_i]-[M]$. 3 $[e_i]=\mathrm{SExp}([u_i])$. 4. Compute $[D]=\sum[e_i]$. 5 Return $[p_i]=\mathrm{SDiv}([e_i],[D])$.
SGeLU([x])	1. $[z]=\sqrt{\frac{2}{\pi}}([x]+0.044715\cdot\mathrm{TruncPr}([x]^3))$. 2. Securely clip |z| to 3. 3 Compute [q]=SExp(-2|z|). 4. [t]=sign(z)·SDiv(1-[q],1+[q]). 5 Return TruncPr(0.5[x]·(1+[t])).

4.4 Secure ViT Inference Workflow

SViT retains the ViT-B/16 layer topology: patch embedding, Transformer encoder blocks, and a classifier. "Retains" refers to graph structure only. Floating-point operators are replaced by the Q16 protocols in Algorithms 1 and 2, so numerical and prediction fidelity must be measured separately. Linear layers use secret-shared matrix multiplication; the Transformer encoder invokes SLayerNorm, SSoftmax, and SGeLU and remains the dominant online cost.

During inference, the input image is encoded and shared, the two servers consume fresh preprocessing, and every activation remains shared. The model parameters are public in the evaluated setting. A session aborts if a range check fails, a preprocessing identifier is repeated, or a transcript length deviates from the public schedule. Figure 2 shows the online data path; dealer communication occurs only before the online timing boundary.

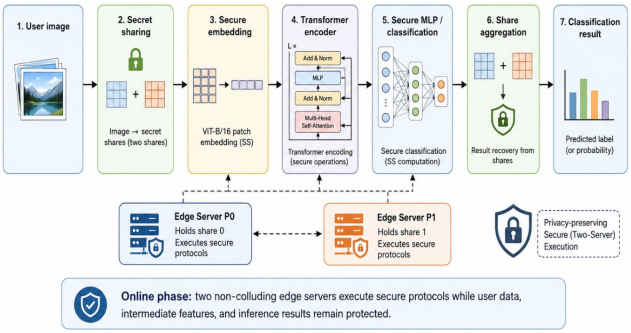

Figure 2. SViT secure inference workflow in distributed edge networks

As shown in Figure 2, the online path consists of secure embedding, 12 Transformer blocks, secure classification, and output-share return. A single server sees its local shares and the public schedule but not the reconstructed tensors. The figure does not imply malicious security or model privacy.

For reproducibility, Algorithm 3 specifies the end-to-end sequence and explicitly separates dealer preprocessing from online execution.

Algorithm 3. End-to-End Privacy-Preserving ViT Inference Workflow

Step	Operation description
Input	User image X, trained ViT-B/16 model parameters, and secure protocol set Π={SExp, SDiv, SSqrt, SVar, SLayerNorm, SSoftmax, SGeLU}
1	The user splits X into secret shares [X]0 and [X]1 and sends them to edge servers P0 and P1, respectively.
2	The trusted third party T generates random masks and auxiliary randomness in the offline phase and distributes them to P0 and P1.
3	P0 and P1 execute the secure embedding layer based on the shared values and obtain patch embeddings in secret-shared form.
4	P0 and P1 iteratively execute SLayerNorm, SMatMul, SSoftmax, and SGeLU to complete secure Transformer encoding.
5	P0 and P1 execute the secure fully connected layer and secure classification layer to generate classification result shares [Y]0 and [Y]1.
6	The user receives [Y]0 and [Y]1, recovers the classification result Y, and obtains the predicted label or class probabilities.
Output	ViT classification result Y under privacy-preserving conditions

Algorithm 3 separates cost accounting into offline preprocessing, user sharing, online server execution, and user reconstruction. Offline random-mask generation and storage are not free; Section 6.6 defines the quantities that must be measured in a complete artifact evaluation.

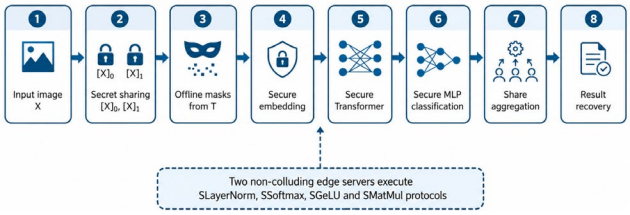

Figure 3. SViT secure inference algorithm workflow

5. Correctness, Security, and Complexity Analysis

5.1 Functional Correctness and Numerical Stability

For a real-valued operator F and public domain D, define the approximate ideal functionality $\bar{F}_{\ell,f,D}$ to reconstruct encoded inputs internally, apply the clipping and fixed-depth approximation in Tables 3 and Algorithms 1-2, and return additive shares of the Q16 result. Functional correctness means that the real protocol reconstructs exactly the same ring element as $\bar{F}$ except with negligible preprocessing or wraparound failure probability. Numerical fidelity measures the difference between decode($\bar{F}$) and the floating-point target F.

Proposition 1 (Ring correctness). If every intermediate signed magnitude is below 2^{47} and every preprocessing item is fresh and correctly generated, *SAdd*, *SMul*, *SSelect*, *SLookup*, and *TruncPr* reconstruct the ring operations specified by Algorithms 1-2. Proposition 2 (Composite correctness). Under the domain conditions in Table 3, SLayerNorm, SSoftmax, and SGeLU securely compose the basic functionalities and reconstruct the corresponding Q16 approximate operators.

A deterministic Q16 reference implementation was used to assess operator-level numerical fidelity independently of the unavailable model checkpoint. The test used round-to-nearest Q16 encoding, quantization after each multiplication, an 8-bit midpoint seed table, exactly two Newton updates, and random seed 20260726 for Softmax vectors. Over dense scalar grids, the maximum absolute error was 5.10×10^{-5} for SExp on [-8, 0]. The maximum relative errors of reciprocal on [0.5,1) and square root on [0.5, 2] were 1.27×10^{-5} and 3.72×10^{-5}, respectively. The standard tanh-form GeLU differed from exact erf-based GeLU by at most 4.73×10^{-4} on [-6, 6]. For 10,000 Gaussian 197-dimensional logit vectors (standard deviation 2), Q16 SSoftmax preserved the floating-point top-1 class in 100% of trials; its 99.9th-percentile absolute probability error was 6.11×10^{-4}. These synthetic checks validate the stated arithmetic configuration but do not replace dataset-level classification accuracy.

Table 4. Synthetic Q16 numerical-fidelity checks

Operator	Test domain / sample	Error statistic	Result
SExp	200,001 points; [-8,0]	Max absolute error	5.10×10^{-5}
SDiv reciprocal	200,001 points; [0.5,1)	Max relative error	1.27×10^{-5}
SSqrt	200,001 points; [0.5,2]	Max relative error	3.72×10^{-5}
SGeLU	240,001 points; [-6,6]	Max absolute error vs exact GeLU	4.73×10^{-4}
SSoftmax	10,000 vectors; length 197; seed 20260726	Top-1 consistency / P99.9 error	100% / 6.11×10^{-4}

5.2 Simulation-Based Security

Let L be the public leakage defined in Section 3.2. For each protocol Pi, the ideal functionality receives the parties' local input shares and returns local output shares of $\bar{F}$. A simulator Sim_b for corrupted server Pb receives its input share, preprocessing share, output share, security parameter, and L. It samples uniformly distributed messages with the prescribed lengths and invokes the simulators of SMul, SCompare, SLookup, and TruncPr. Because unused shares are uniform and fresh preprocessing is independent across calls, the simulated transcript is computationally indistinguishable from the real transcript under the security of the underlying primitives.

Theorem 1 (Semi-honest privacy). Assuming secure Beaver multiplication, secure comparison/lookup/truncation, a non-colluding pair of servers, and a correctly behaving offline dealer that never reuses correlations, Algorithms 1-3 securely realize their approximate ideal functionalities against one static semi-honest corruption, leaking only L. The proof follows by sequential replacement of each primitive call with its ideal execution and the standard sequential-composition theorem [39]. The theorem does not cover model privacy, collusion, malicious deviations, selective failures, or a compromised dealer.

5.3 Computational and Communication Complexity Analysis

Let N denote the number of scalar inputs to an operator, d the feature dimension, ell the ring bit length, p the public polynomial degree, r the public number of Newton updates, and R_{mul}, R_{cmp}, R_{lookup}, and R_{trunc} the online round costs of the corresponding primitives. The earlier symbol m was ambiguous because it was used both for input size and iteration count; it is no longer used in the complexity statement. SViT fixes p=4 and r=2, so scalar nonlinear work

is O(N), while attention matrix multiplication remains $O(N^2d)$. Offline triple/mask generation is accounted for separately and is not included in online rounds.

Table 5. Symbolic online cost of revised SViT protocols

Protocol	Secure mult.	Compare/lookup	Online rounds	Transmitted elements
SExp	p+1 per scalar	1 clip + 1 lookup	$R_{cmp}+R_{lookup}+(p+1)(R_{mul}+R_{trunc})$	$\Theta(N(p+1))$
SDiv	2r+1 per scalar	1 floor + 1 seed lookup	$R_{cmp}+R_{lookup}+(2r+1)(R_{mul}+R_{trunc})$	$\Theta(Nr)$
SSqrt	3r+1 per scalar	1 sign + 1 seed lookup	$R_{cmp}+R_{lookup}+(3r+1)(R_{mul}+R_{trunc})$	$\Theta(Nr)$
SVar	N squares	none	$R_{mul}+R_{trunc}$	$\Theta(N)$
SSoftmax	SExp + N divisions	max tree + lookups	$ceil(log_2N)R_{cmp}+R_{SExp}+R_{SDiv}$	$\Theta(N(p+r))$
SLayerNorm	SVar + N div.	sqrt seed lookup	$R_{SVar}+R_{SSqrt}+R_{SDiv}$	$\Theta(Nr)$
SGeLU	SExp + SDiv + 4 mult.	clip/sign	$R_{cmp}+R_{SExp}+R_{SDiv}+4(R_{mul}+R_{trunc})$	$\Theta(N(p+r))$

Table 5 reports symbolic costs rather than conflating input length with convergence iterations. With fixed p and r, the nonlinear protocols are linear in the number of elements; however, constants include secure multiplication, truncation, comparison, and lookup costs, and the Transformer attention matrices remain quadratic in sequence length.

For the implementation benchmark, Table 6 retains the measured communication-round formulas reported for the basic protocols. The symbols are now defined consistently: n is the number of scalar elements and ell is the bit length of one ring element. Offline dealer-to-server communication, preprocessing storage, and mask generation are excluded from these online formulas and are discussed in Section 6.6.

Table 6. Online communication formulas for basic protocols

Protocol	CrypTen communication rounds	CrypTen communication overhead	SViT communication rounds	SViT communication overhead
SExp	8	16nl	2	4nl
SDiv	33+lb l	142nl	6	10nl
SSqrt	18	50nl	4	7nl
SVar	1	2nl	1	2nl
SLayerNorm	—	—	11	19nl
SGeLU	—	—	15	27nl

Table 6 shows the implementation's relative reduction for SExp, SDiv, and SSqrt. These formulas do not establish low end-to-end communication by themselves, because SSoftmax and repeated Transformer blocks amplify per-element costs. Therefore, the revised manuscript reports the complete 2.85 GB result and analyzes bandwidth limits in Section 6.5.

6. Experimental Evaluation and Practicality Analysis

6.1 Experimental Configuration and Reproducibility Boundary

The reported experimental configuration uses two Intel Xeon Platinum 8255C servers with 38 GB RAM and NVIDIA Tesla V100 GPUs, a Python/PyTorch implementation, ViT-B/16, 224 x 224 inputs, batch size 1, and a flower-image dataset recorded as 3,817 training and 500 test images. The exact dataset name, repository or version, license, original sample count, and split procedure were not preserved in the available experimental record; therefore, a formal dataset citation cannot be added without author confirmation, and these provenance fields must be supplied before final publication. The revised protocol configuration is ell=64 and f=16. Any reproducible rerun should keep tensor dimensions, thread count, security level, preprocessing-cache policy, and network-emulation settings identical across baselines. The current experimental record also does not specify software versions, commit identifiers, compiler flags, per-protocol tensor shapes, CPU/GPU utilization, or public source-code availability; these items are required reproducibility metadata and limitations of the present evaluation.

6.2 Numerical Fidelity and Model-Level Accuracy Claim

The previous wording suggested that retaining the ViT-B/16 topology implied no classification-accuracy loss. That claim is withdrawn. Table 4 evaluates operator-level numerical fidelity only. A complete model-level evaluation should report plaintext FP32, plaintext Q16, and secure-SViT top-1/top-5 accuracy on identical test samples, together with prediction agreement, maximum and mean logit deviation, and class-probability deviation. Because the present experiment did not archive model checkpoints, secure prediction traces, or per-sample outputs, these dataset-level metrics cannot be reconstructed retrospectively and are not claimed in this paper.

The recommended acceptance criterion is that SViT's secure output exactly matches the plaintext Q16 implementation and that the plaintext-Q16 accuracy change relative to FP32 is separately disclosed. This separates cryptographic correctness from quantization/approximation error and permits an auditable attribution of any prediction difference.

6.3 Performance of Secure Computing Protocols

Table 7 retains the available operator microbenchmarks. The archived experimental record does not preserve sufficient logs to determine whether every CrypTen and MP-SPDZ value was rerun under a fully matched configuration; consequently, the entries are treated as internal historical measurements rather than as a cross-paper benchmark. They establish only relative runtime and online-

communication differences for the recorded tensor configurations and do not constitute an end-to-end comparison with BOLT, BumbleBee, BLB, Iron, or SecViT. Such a comparison requires executable implementations under matched model, precision, hardware, preprocessing, and network settings.

Table 7. Recorded runtime and online communication of secure computing protocols

Protocol	MP-SPDZ runtime/ms	CrypTen runtime/ms	SViT runtime/ms	MP-SPDZ communication/MB	CrypTen communication/MB	SViT communication/MB
SExp	—	38.4	6.1	—	1.2207	0.3052
SDiv	885.8	9.1	1.4	176	10.8337	0.7629
SSqrt	1982.9	266.4	116.8	358.08	3.8146	0.5341
SVar	158.12	9.4	2.5	0.6624	0.1526	0.1526
SSoftmax	2326.79	703.2	234.5	161.347	14.1372	1.5661
SLayerNorm	—	—	16.4	—	—	1.4496
SGeLU	—	—	8.1	—	—	1.9073

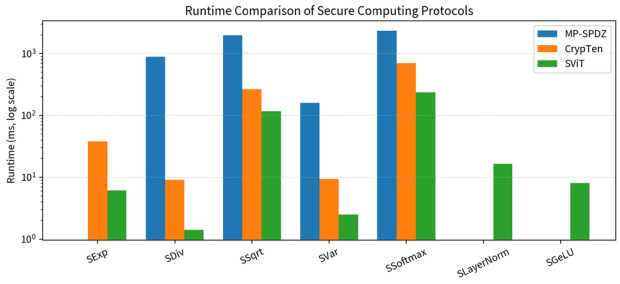

Figure 4. Recorded runtime of secure computing protocols

To further compare the online communication overhead of different protocols, the online communication comparison of secure computing protocols is shown in Figure 5.

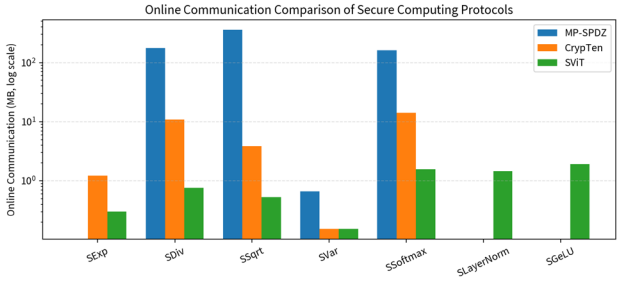

Figure 5. Recorded online communication of secure computing protocols

Figures 4 and 5 summarize the recorded runtime and online communication of the reported core operators relative to the available CrypTen and MP-SPDZ measurements. The strongest recorded example is SDiv (1.4 ms and 0.7629 MB). Because the archived baseline logs are incomplete, these results support only an internal operator-level communication-reduction claim; they do not establish a fully matched comparison or superiority over recent end-to-end private Transformer systems [28,31-33].

6.4 End-to-End Performance and Cost Boundary

Table 8 presents the reported end-to-end online runtime and server-to-server online communication on the recorded flower-image workload. The measured total is 3799.744 ms and 2.85 GB per inference. The timing boundary excludes offline dealer generation, dealer-to-server transfer, preprocessing storage, user-side sharing and reconstruction, and network propagation delay; the dataset-provenance limitation stated in Section 6.1 also applies.

Table 8. Reported online runtime and communication of the SViT framework

Sublayer	Runtime/ms	Online communication/GB	Description
Embedding layer	18.791	0	Mainly local secure linear computation
Transformer encoder	313.551×12	2.83	Consists of multi-head attention, SLayerNorm, SSoftmax, and SGeLU
Classification layer	18.34	0.02	Completes final secure classification
Total	3799.744	2.85	End-to-end privacy-preserving inference overhead

The Transformer encoder accounts for approximately 99.0% of reported runtime and 99.3% of online communication. Preserving all 12 encoder blocks therefore retains a substantial systems cost. The revised interpretation is that SViT reduces communication relative to the evaluated operator baselines while remaining expensive at the full-model level.

Figure 6 visualizes the online breakdown in Table 8. It motivates future optimization of attention, SLayerNorm, SSoftmax, and SGeLU, but does not include dealer preprocessing or network delay.

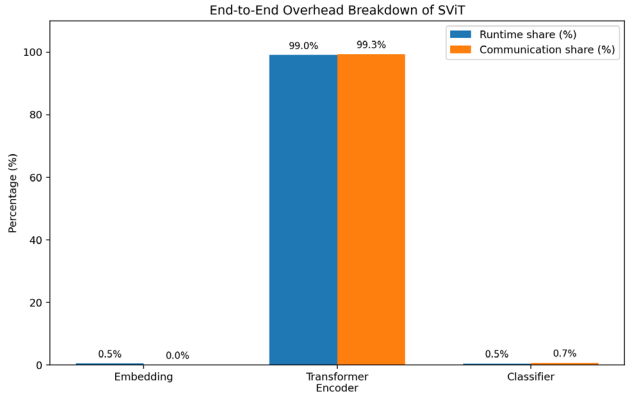

Figure 6. End-to-end inference overhead breakdown of SViT

6.5 Bandwidth- and Latency-Constrained Deployment Analysis

To evaluate practical edge conditions without inventing unreported measurements, Table 9 gives a deterministic lower bound obtained from the measured 2.85 GB online traffic. Transmission time is C/B, where C=22.8 Gbit and B is the effective bidirectional application-layer throughput; total time adds the reported 3.799744 s computation and excludes RTT amplification. Thus, even ideal 1 Gbit/s transport gives at least 26.60 s per inference, while 100 Mbit/s gives at least 231.80 s. The current design is therefore most plausible on a co-located 10 Gbit/s LAN, not on a narrowband mobile edge link.

Table 9. Communication-only lower bounds under representative bandwidths

Effective bandwidth	Transfer lower bound	Compute + transfer	Upper-bound throughput
10 Gbit/s LAN	2.28 s	6.08 s	0.164 inference/s
1 Gbit/s LAN	22.80 s	26.60 s	0.0376 inference/s
100 Mbit/s WAN	228.00 s	231.80 s	0.00431 inference/s
20 Mbit/s constrained	1140.00 s	1143.80 s	0.000874 inference/s

6.6 Offline Cost, Scalability, and Required Ablations

A complete deployment report should separately measure dealer mask/triple generation time, offline communication, preprocessing storage, peak server memory, user-side sharing and reconstruction time, online runtime, online communication, throughput, and concurrent-user scaling. It should also vary batch size, sequence length, round-trip latency, and available bandwidth. The aggregate measurements available in the present study do not permit these quantities to be separated; this limitation defines the scope of the practical conclusions.

Three ablations are required. A1 disables offline preprocessing and generates correlations online to quantify the dealer's contribution. A2 replaces each fixed-depth SExp/SDiv/SSqrt with the configured CrypTen iterative primitive while keeping all other components fixed. A3 measures public-model dual-edge execution against a single co-located secure server pair to isolate network topology. Additional evaluation should include at least one medical or industrial dataset and a lightweight ViT backbone. Because no code, checkpoints, or raw logs accompanied the manuscript, the revision specifies these experiments but does not create unsupported numerical results.

Distributed execution increases the privacy boundary but also exposes communication cost. With 2.85 GB per inference, SViT is not appropriate for most mobile or weak-wireless links. Its current practical niche is a high-bandwidth provider LAN or cross-institution environment in which P0 and P1 are independently administered but connected by fast infrastructure. Edge gateways may submit shares, while the secure Transformer computation remains on server-grade edge clusters.

Within this restricted scope, SViT provides three benefits: no single semi-honest server sees a complete input or activation; fixed-depth nonlinear protocols reduce online interaction relative to the tested general MPC implementations; and the ViT-B/16 topology can be reused without retraining a distilled architecture. The limitations are equally important: the model is public, the dealer and non-collusion assumptions are strong, dataset-level accuracy has not yet been rerun, and full-model communication remains high.

Future work should prioritize maliciously secure and dealer-free preprocessing, authenticated shares, communication compression, sparse/private attention, secure model ownership, multi-user batching, and validation on heterogeneous edge hardware and multiple visual tasks. These are required to extend the framework beyond the present semi-honest high-bandwidth deployment envelope.

In summary, the evidence supports a protocol-level communication-reduction claim, not a universal lightweight claim. The revised manuscript makes this distinction explicit and provides a reproducibility checklist for the remaining accuracy, edge-device, ablation, and recent-system comparisons.

7. Conclusion

This paper revises SViT as a communication-reduced privacy-preserving ViT inference framework for two non-colluding edge servers with offline correlated randomness. The revised method specifies Q16 arithmetic over $\mathbb{Z}_{2^{64}}$, one-time preprocessing, numerical domains, truncation, and complete procedures for SExp, SDiv, SSqrt, SVar, SLayerNorm, SSoftmax, and SGeLU. A simulation-based theorem formalizes security against one static semi-honest server and states the public leakage and excluded attack

classes. Synthetic numerical checks show small scalar approximation errors and 100% top-1 consistency on 10,000 random Softmax vectors, but do not substitute for dataset-level plaintext-versus-secure accuracy. Existing microbenchmarks show relative operator improvements over CrypTen and MP-SPDZ, while the reported end-to-end online cost remains 3799.744 ms and 2.85 GB. Bandwidth analysis demonstrates that this communication is practical mainly on high-speed LAN infrastructure and can be prohibitive on conventional WAN or constrained edge links. Accordingly, the revised contribution is a reproducible operator composition and distributed ownership model with measured relative communication reduction, subject to public-model, semi-honest, non-collusion, trusted-dealer, and high-bandwidth limitations. Model-level accuracy, recent end-to-end baseline comparison, heterogeneous edge hardware, multiple datasets, and controlled ablations remain necessary author-run evaluations before final publication.

References

[1] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Advances in Neural Information Processing Systems. 2017, 30: 5998-6008.

[2] Dosovitskiy A, Beyer L, Kolesnikov A, et al. An image is worth 16x16 words: Transformers for image recognition at scale[C]//International Conference on Learning Representations. 2021.

[3] Devlin J, Chang M W, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding[C]//Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics. 2019: 4171-4186.

[4] Zhou C, Li Q, Li C, et al. A comprehensive survey on pretrained foundation models: A history from BERT to ChatGPT[J]. International Journal of Machine Learning and Cybernetics, 2025, 16(12): 9851-9915. doi:10.1007/s13042-024-02443-6.

[5] Shi W, Cao J, Zhang Q, Li Y, Xu L. Edge computing: Vision and challenges[J]. IEEE Internet of Things Journal, 2016, 3(5): 637-646.

[6] Zhou Z, Chen X, Li E, Zeng L, Luo K, Zhang J. Edge intelligence: Paving the last mile of artificial intelligence with edge computing[J]. Proceedings of the IEEE, 2019, 107(8): 1738-1762.

[7] Teerapittayanon S, McDanel B, Kung H T. Distributed deep neural networks over the cloud, the edge and end devices[C]//IEEE International Conference on Distributed Computing Systems. 2017: 328-339.

[8] Mohassel P, Zhang Y. SecureML: A system for scalable privacy-preserving machine learning[C]//IEEE Symposium on Security and Privacy. 2017: 19-38.

[9] Liu J, Juuti M, Lu Y, Asokan N. Oblivious neural network predictions via MiniONN transformations[C]//ACM SIGSAC Conference on Computer and Communications Security. 2017: 619-631.

[10] Rouhani B D, Riazi M S, Koushanfar F. DeepSecure: Scalable provably-secure deep learning[C]//Proceedings of the 55th Annual Design Automation Conference. New York: ACM, 2018: 2:1-2:6. doi:10.1145/3195970.3196023.

[11] Juvekar C, Vaikuntanathan V, Chandrakasan A. GAZELLE: A low latency framework for secure neural network inference[C]//USENIX Security Symposium. 2018: 1651-1669.

[12] Riazi M S, Samragh M, Chen H, et al. XONN: XNOR-based oblivious deep neural network inference[C]//USENIX Security Symposium. 2019: 1501-1518.

[13] Wagh S, Gupta D, Chandran N. SecureNN: 3-party secure computation for neural network training[J]. Proceedings on Privacy Enhancing Technologies, 2019(3): 26-49.

[14] Wagh S, Tople S, Benhamouda F, et al. FALCON: Honest-majority maliciously secure framework for private deep learning[J]. Proceedings on Privacy Enhancing Technologies, 2021(1): 188-208.

[15] Mishra P, Lehmkuhl R, Srinivasan A, Zheng W, Popa R A. Delphi: A cryptographic inference service for neural networks[C]//USENIX Security Symposium. 2020: 2505-2522.

[16] Kumar N, Rathee M, Chandran N, Gupta D, Rastogi A, Sharma R. CrypTFlow: Secure TensorFlow inference[C]//IEEE Symposium on Security and Privacy. 2020: 336-353.

[17] Rathee D, Rathee M, Kumar N, et al. CrypTFlow2: Practical 2-party secure inference[C]//ACM SIGSAC Conference on Computer and Communications Security. 2020: 325-342.

[18] Rathee D, Rathee M, Goli R K K, Gupta D, Sharma R, Chandran N, Rastogi A. SiRnn: A math library for secure RNN inference[C]//2021 IEEE Symposium on Security and Privacy. IEEE, 2021: 1003-1020. doi:10.1109/SP40001.2021.00086.

[19] Knott B, Venkataraman S, Hannun A, et al. CrypTen: Secure multi-party computation meets machine learning[C]//Advances in Neural Information Processing Systems. 2021, 34: 4961-4973.

[20] Demmler D, Schneider T, Zohner M. ABY - A framework for efficient mixed-protocol secure two-party computation[C]//Network and Distributed System Security Symposium. 2015.

[21] Patra A, Schneider T, Suresh A, Yalame H. ABY2.0: Improved mixed-protocol secure two-party computation[C]//USENIX Security Symposium. 2021: 2165-2182.

[22] Keller M. MP-SPDZ: A versatile framework for multi-party computation[C]//ACM SIGSAC Conference on Computer and Communications Security. 2020: 1575-1590.

[23] Li D, Shao R, Wang H, Guo H, Xing E P, Zhang H. MPCFormer: Fast, performant and private Transformer inference with MPC[C]//International Conference on Learning Representations. 2023.

[24] Chen T, Bao H, Huang S, et al. THE-X: Privacy-preserving Transformer inference with homomorphic encryption[C]//Findings of the Association for Computational Linguistics: ACL 2022. 2022: 3510-3520.

[25] Hao M, Li H, Chen H, Xing P, Xu G, Zhang T. Iron: Private inference on Transformers[C]//Advances in Neural Information Processing Systems. 2022, 35: 15718-15731.

[26] Luo J, Zhang Y, Zhang Z, et al. SecFormer: Fast and accurate privacy-preserving inference for Transformer models via SMPC[C]//Findings of the Association for Computational Linguistics: ACL 2024. 2024: 13333-13348. doi:10.18653/v1/2024.findings-acl.790.

[27] Dong Y, Lu W J, Zheng Y, et al. PUMA: Secure inference of LLaMA-7B in five minutes[J]. Security and Safety, 2025, 4: 2025014. doi:10.1051/sands/2025014.
[28] Pang Q, Zhu J, Möllering H, Zheng W, Schneider T. BOLT: Privacy-preserving, accurate and efficient inference for Transformers[C]//2024 IEEE Symposium on Security and Privacy. San Francisco, CA, USA: IEEE, 2024: 4753-4771. doi:10.1109/SP54263.2024.00130.
[29] Diaa A, Fenaux L, Humphries T, et al. Fast and private inference of deep neural networks by co-designing activation functions[C]//33rd USENIX Security Symposium. 2024: 2191-2208.
[30] Yuan B, Yang S, Zhang Y, Ding N, Gu D, Sun S F. MD-ML: Super fast privacy-preserving machine learning for malicious security with a dishonest majority[C]//33rd USENIX Security Symposium. 2024: 2227-2244.
[31] Lu W J, Huang Z, Gu Z, et al. BumbleBee: Secure two-party inference framework for large Transformers[C]//Network and Distributed System Security Symposium. 2025.
[32] Xu T, Lu W J, Yu J, et al. Breaking the layer barrier: Remodeling private Transformer inference with hybrid CKKS and MPC[C]//34th USENIX Security Symposium. 2025: 2653-2672.
[33] Pang Z, Feng B, Luo M, Liu C, Xu S, Zhao K, Wu Y, Zeng B. Privacy-friendly adaptation of Vision Transformers for communication and latency-efficient private inference[C]//Proceedings of the ACM Web Conference 2026. New York: ACM, 2026: 2695-2706. doi:10.1145/3774904.3792211.
[34] Zhang J, Yang X P, He L, Chen K, Lu W J, Wang Y, Hou X, Liu J, Ren K, Yang X H. Secure Transformer inference made non-interactive[C]//Network and Distributed System Security Symposium. San Diego, CA, USA: Internet Society, 2025.
[35] Chenthara S, Ahmed K, Whittaker F. Privacy-Preserving Data Sharing using Multi-layer Access Control Model in Electronic Health Environment[J]. EAI Endorsed Transactions on Scalable Information Systems, 2019, 6(22): e6. doi:10.4108/eai.13-7-2018.159356.
[36] Zeng Y, Kang Z, Shi Z. Secure Data Processing Technology of Distribution Network OPGW Line with Edge Computing[J]. EAI Endorsed Transactions on Scalable Information Systems, 2023, 10(3): e7. doi:10.4108/eetsis.v10i3.2837.
[37] Beaver D. Efficient multiparty protocols using circuit randomization[C]//Advances in Cryptology-CRYPTO '91. Lecture Notes in Computer Science, vol. 576. Springer, 1992: 420-432. doi:10.1007/3-540-46766-1_34.
[38] Catrina O, Saxena A. Secure computation with fixed-point numbers[C]//Financial Cryptography and Data Security-FC 2010. Lecture Notes in Computer Science, vol. 6052. Springer, 2010: 35-50. doi:10.1007/978-3-642-14577-3_6.
[39] Canetti R. Security and composition of multiparty cryptographic protocols[J]. Journal of Cryptology, 2000, 13(1): 143-202. doi:10.1007/s001459910006.