# CIPHERGENOME: HOMOMORPHIC INFERENCE FOR GENOMIC MIXTURE-OF-EXPERTS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Genome foundation models are growing into sparse mixture-of-experts (MoE) networks whose expert weights no longer fit on the machines that hold the sequences, yet sending a private genome to rented accelerators exposes it: we show that a single server hosting one expert recovers the input nucleotides with 99.8% top-1 accuracy. We present CipherGenome, a protocol that keeps the embedding, attention and router of a 15.1B-parameter MoE genome model on a trusted thin client and outsources every expert projection, 95.8% of the parameters, to untrusted and possibly colluding GPU servers under module-LWE encryption. The design exploits three structural facts: expert layers are linear between two SwiGLU gates, expert weights are public, and GPU integer tensor cores can evaluate a ciphertext–weight product exactly modulo $2^{48}$ in a single GEMM. The client evaluates the nonlinearity exactly and re-encrypts with fresh secrets, so no polynomial approximation or bootstrapping is ever needed. On 72 windows from 12 bacterial genomes, encryption adds $2.54 \times 10^{-4}$ nats per token of KL divergence (95% CI upper bound $3.95 \times 10^{-4}$), below a pre-registered non-inferiority margin and indistinguishable from bf16 inference, while the same inversion attack falls to chance level. A reusable public hint cuts end-to-end latency by 3.54 times, wire compression reduces traffic 6.8 times, per-layer padding reduces routing leakage from 54.9% to 8.9% accuracy, and HE-compatible int4 experts remain non-inferior to their plaintext counterparts. Per expert and token, the server-side cost is more than six orders of magnitude below a CKKS baseline.

## 1 INTRODUCTION

Genome foundation models are becoming both larger and sparser. Dense models such as DNABERT-2, HyenaDNA, the Nucleotide Transformer and Evo 2 (Zhou et al., 2024; Nguyen et al., 2023; Dalla-Torre et al., 2025; Brixi et al., 2025) have given way to mixture-of-experts (MoE) designs such as GenomeOcean-MoE (Zhou et al., 2025b), following the trajectory of language models from sparsely gated experts (Shazeer et al., 2017) through Switch and GShard to Mixtral, DeepSeek-MoE and DeepSeek-V3 (Fedus et al., 2022; Lepikhin et al., 2021; Jiang et al., 2024; Dai et al., 2024; DeepSeek-AI, 2024). In a 15.1B-parameter GenomeOcean-MoE, 95.8% of the parameters and 85.7% of the per-token linear FLOPs sit in the experts. The laboratories and hospitals that hold patient or strain sequences therefore increasingly lack the accelerator memory to serve such models, whose efficient deployment already relies on expert parallelism across many GPUs (Rajbhandari et al., 2022; Kwon et al., 2023). The natural remedy, renting GPU servers or joining a collaborative inference swarm (Borzunov et al., 2023), sends the sequence to machines the data owner does not control.

Genomic data is a particularly unforgiving thing to leak. It is permanent, it identifies individuals and their relatives (Homer et al., 2008), and recent work shows that hidden representations of DNA language models can be inverted back to the sequence (Ouaari et al., 2026), mirroring the text setting (Morris et al., 2023). MoE adds a second channel: the pattern of expert selections itself reveals the input (Yona et al., 2024; Ding et al., 2025; Nuriyev & Kulp, 2026). We confirm both risks directly on GenomeOcean-MoE. A server that hosts a single expert and trains a small decoder on the activations it receives recovers 99.8% of the input tokens at layer 0 (Figure 2c), and per-expert routing counts alone identify the source organism among 16 taxa with 54.9% accuracy, against 6.25% chance (Section 4.4).



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

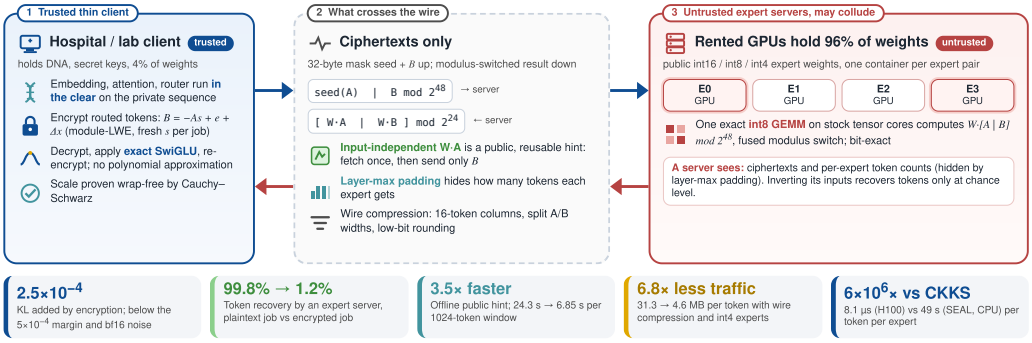



Figure 1: CipherGenome at a glance. A trusted thin client keeps the sequence, the keys and the 4% of weights outside the experts. Routed tokens leave only as module-LWE ciphertexts. Untrusted, possibly colluding servers holding the public expert weights compute one exact integer GEMM per job. The client evaluates SwiGLU exactly and re-encrypts. Headline numbers are from Section 4.

Cryptographic inference offers protection but has not been practical at this scale. Non-interactive homomorphic schemes such as CKKS (Cheon et al., 2017) must approximate softmax and gated activations with high-degree polynomials and bootstrap repeatedly (Zhang et al., 2025; Moon et al., 2025). Two-party protocols replace this with interaction but require a second party with server-class compute and large communication (Huang et al., 2022; Lu et al., 2025; Dong et al., 2025), and existing MoE-specific protocols target private weights rather than private inputs (Zhou et al., 2025a; Shen et al., 2026).

We observe that the genome MoE setting admits a much simpler and exact design. First, expert weights are public: the model owner releases them, so only the input must be hidden. Second, an expert is a pair of linear maps separated by one gated nonlinearity, and those linear maps carry almost all of the model's parameters. Third, a product of a public integer matrix with a learning-with-errors (LWE) ciphertext is itself an integer matrix product, which stock GPU int8 tensor cores evaluate exactly if the ciphertext is split into bytes. CipherGenome combines these facts (Figure 1). A thin client runs the 4% of the model outside the experts, encrypts the tokens routed to each expert under module-LWE with a shared, seed-compressed mask, and sends them to the server hosting that expert. The server multiplies by its weights in one exact GEMM and switches the result to a smaller modulus. The client decrypts, evaluates SwiGLU (Shazeer, 2020) exactly, re-encrypts under fresh secrets, and submits the down projection. Because nonlinearities never leave the client, no approximation or bootstrapping is ever needed, and because every ciphertext uses a fresh secret, security holds even if all servers pool their views.

**Contributions.**

- **An MoE-shaped trust boundary.** We identify the experts' linear maps as the outsourcing unit for public-weight MoE models and give a protocol in which servers see only fresh-secret module-LWE ciphertexts, so that confidentiality survives collusion among all servers (Section 3).
- **Exact encrypted expert evaluation on stock tensor cores.** We map the ciphertext–weight product modulo $2^{48}$ onto a single int8 GEMM with a fused modulus switch. A Cauchy–Schwarz bound proves that per-token scaling makes the computation wrap-free (Proposition 1), and all kernels are bit-exact against an FP64 reference.
- **Measured fidelity and security.** Under pre-registered protocols with known-bad controls, encryption is non-inferior to exact inference on a 15.1B genome MoE, and an inversion attack that succeeds on plaintext jobs falls to chance on encrypted jobs (Section 4.1).
- **A practical deployment toolkit.** A public reusable hint, wire compression, HE-compatible int8 and int4 experts and per-layer padding reduce latency by 3.54 times, traffic by 6.8 times, and routing leakage to near chance, each at pre-registered fidelity (Sections 4.2 to 4.5).

## 2 RELATED WORK

**Privacy of genome models and MoE routing.** Genomic data enables re-identification from aggregate statistics (Homer et al., 2008), which has motivated homomorphic encryption for genome-wide association studies (Blatt et al., 2020), and genome language models memorize and leak training sequences (Nemecek et al., 2026). At inference time, embeddings of DNA models can be inverted (Ouaari et al., 2026), as can text embeddings (Morris et al., 2023). MoE routing is a separate side channel: expert choices leak prompts (Yona et al., 2024), timing and cache behavior expose routing (Ding et al., 2025), and expert selections alone reconstruct much of the input text (Nuriyev & Kulp, 2026). We quantify this channel for a genome MoE and give a cheap mitigation.

**Homomorphic and two-party transformer inference.** Early private inference combined HE with garbled circuits or secret sharing (Mohassel & Zhang, 2017; Juvekar et al., 2018; Huang et al., 2022). For transformers, THE-X approximates nonlinearities for HE (Chen et al., 2022); MPC-Former, PUMA, BOLT and BumbleBee reduce two-party cost (Li et al., 2023; Dong et al., 2025; Pang et al., 2024; Lu et al., 2025); NEXUS and THOR make CKKS inference non-interactive (Zhang et al., 2025; Moon et al., 2025). These systems protect both weights and inputs and pay for polynomial approximation, bootstrapping or heavy interaction. CryptoMoE and SecMoE extend two-party inference to MoE layers with private weights (Zhou et al., 2025a; Shen et al., 2026). Closest to our boundary, Frery et al. (2025) outsource linear layers of an open-weight model under LWE-style encryption for LoRA fine-tuning of a dense network, and Galanopoulos et al. (2026) study client-assisted CKKS for a small dense genome model. We target a 15B MoE, many possibly colluding servers, exact integer arithmetic on tensor cores, and routing leakage.

**Lattice primitives and systems.** Our encryption is module-LWE (Regev, 2009; Lyubashevsky et al., 2013; Langlois & Stehlé, 2015), the hardness assumption underlying BGV and CKKS (Brakerski et al., 2014; Cheon et al., 2017) and libraries such as OpenFHE (Al Badawi et al., 2022), with parameters chosen against the lattice estimators of Albrecht et al. (2015) and the HE standard (Albrecht et al., 2021). The reusable input-independent hint follows the idea of SimplePIR (Henzinger et al., 2023), which precomputes a database times a public LWE matrix. Unlike trusted-hardware outsourcing (Tramèr & Boneh, 2019), whose one-time masks must be recomputed per query, our hint is public and reusable. Our weight quantization builds on RTN, AWQ and GPTQ (Dettmers et al., 2022; Lin et al., 2024; Frantar et al., 2023).

## 3 METHOD

### 3.1 SETTING AND THREAT MODEL

The model is a Mixtral-style MoE (Jiang et al., 2024) with $L = 24$ layers, $E = 8$ experts per layer, top-2 routing, hidden size $d = 3072$ and expert width $f = 8192$. Expert $e$ of layer $\ell$ computes

$$y = W_2\left(\mathrm{SiLU}(W_1 x) \odot W_3 x\right), \qquad W_1, W_3 \in \mathbb{R}^{f\times d},\ W_2 \in \mathbb{R}^{d\times f}. \tag{1}$$

The *client* holds the sequence, all secret keys and every non-expert weight (0.6B parameters). Each *server* holds the public weights of some experts; an *orchestrator* forwards jobs. Servers and the orchestrator are honest-but-curious and may collude arbitrarily. Their joint view must reveal nothing about the sequence beyond public job metadata: layer, expert, stage, and a token count that padding can hide (Section 4.4). Integrity against malicious servers and protection of the weights are out of scope.

### 3.2 SHARED-MASK MODULE-LWE FOR EXPERT JOBS

Let $R = \mathbb{Z}[X]/(X^N + 1)$ with $N = 256$ and module rank $r = 8$, so the LWE dimension is $n = rN = 2048$, and let $q = 2^Q$. For a job whose $T_e$ routed tokens are stacked as a matrix $M \in \mathbb{Z}_q^{d\times T_e}$, the client groups tokens into $k = \lceil T_e/N \rceil$ ring slots and samples a fresh ternary secret $s_j \in R^r$ per slot. All slots share one public mask $A \in R_q^{d\times r}$, expanded from a 32-byte seed with ChaCha20. The ciphertext is

$$B_{:,j} = -A\, s_j + e_j + M_{:,j} \pmod q, \qquad e_j \sim \mathrm{CBD}(21), \tag{2}$$

and only the seed and $B$ are sent. The server, holding an integer weight matrix $W \in \mathbb{Z}^{m \times d}$, computes

$$Y = W\,[\,A \mid B\,] \pmod q, \qquad Y_A = WA, \quad Y_B = WB = -WA\,s + We + WM. \tag{3}$$

It switches $Y$ to $q' = 2^{Q'}$ by rounding and returns it. The client recovers $WM + We$ as $Y_B + Y_A s$, where the ring products are exact negacyclic convolutions evaluated with float64 FFTs on 24-bit limbs. The server never uses the ring structure: Equation (3) is a plain integer matrix product, which is what makes it fast.

**Exact integer GEMM on int8 tensor cores.** Weights are quantized per output row to int16 and stored as two balanced int8 limbs, and $[A|B]$ is split into $\lceil Q/8 \rceil$ balanced base-256 digits. All limb–digit products are computed by one `torch._int_mm` call on the stacked operands, and a fused Triton epilogue shifts, sums, reduces modulo $2^Q$ and applies the modulus switch. Limb pairs whose shift exceeds $Q$ are pruned, so the cost is 11 int8 GEMM units for int16 weights at $Q = 48$, 5 for int8 at $Q = 40$, and 4 for int4 at $Q = 32$. Every configuration is bit-exact against an FP64 reference (Appendix D).

### 3.3 Wrap-free scaling and exact nonlinearity

Real activations must be scaled to integers without the product $WM$ wrapping around modulo $q$. We scale each token individually.

**Proposition 1** (No wraparound)**.** *Let $\beta = \max_i \|W_{i,:}\|_2$ be the largest row norm of the integer weight, published with the model. If token $x_t$ is encoded as $m_t = \mathrm{round}(\Delta_t x_t)$ with $\Delta_t = 2^{Q-2}/(\beta\|x_t\|_2)$, then $|(Wm_t)_i| < 2^{Q-1}$ for every row $i$, so $WM$ is recovered exactly modulo $q$ up to the LWE noise term $We$.*

The proof, by Cauchy–Schwarz, is in Appendix A. The error of a decrypted expert output decomposes into weight quantization, rounding of $m_t$, the noise $We$, and the modulus switch; with int16 weights, $Q = 48$ and $Q' = 24$ it is dominated by quantization (Section 4.1). After decrypting the gate and up projections, the client evaluates SiLU and the gate product exactly in float64, folds in the routing weight, and encrypts the result under a fresh secret for the down projection. The client therefore computes the model's nonlinearities with no approximation at all.

### 3.4 Security

Every job uses a fresh secret and fresh noise; masks are either fresh per job or, in hint mode, a fixed public matrix per (layer, expert, stage). In both cases each ciphertext is a module-LWE sample, and the joint view of all servers is a collection of independent-secret samples, which is indistinguishable from uniform under the decision module-LWE assumption (Langlois & Stehlé, 2015); a multi-secret hybrid argument covers the fixed-mask case (Appendix B). Modulus switching and low-bit rounding are post-processing of ciphertexts and cannot reduce security. With $n = 2048$, ternary secrets and centered-binomial noise of standard deviation 3.24, our estimate for $Q = 48$ is 147.9 bits under the BKZ-sieve cost model used by the HE standard tables, and 149.8 and 201.2 bits under the more conservative core-SVP model for the $Q = 40$ and $Q = 32$ quantized deployments (Appendix B).

### 3.5 Deployment optimizations

**Public hint.** The term $WA$ in Equation (3) does not depend on the input. With a fixed public mask per (layer, expert, stage), the client fetches all $WA$ once, and online jobs carry only $B$ and $WB$. This is the SimplePIR idea (Henzinger et al., 2023) applied to experts; secrets stay fresh per job.

**Wire compression.** Three post-processing steps shrink messages further: token columns are sent at a granularity of 16 rather than full 256-slot ring blocks, since unused slots never need to leave the client; the $A$ and $B$ parts are switched to separate widths (24 and 22 bits); and the low bits of $B$ are rounded away on the uplink (8 bits at $Q = 48$, 4 at $Q = 32$). Batching several windows lets them share one $A$ part.

**Algorithm 1** One CipherGenome MoE layer (client side). Server calls are lines 4 and 9.

**Require:** hidden states $H \in \mathbb{R}^{T\times d}$; public manifest (row scales, $\beta_{\text{up}}$, $\beta_{\text{down}}$, AWQ input scales)
1: route: $(\text{sel}, w) \leftarrow \text{top2}(\text{softmax}(HW_g))$; group token indices $\mathcal{T}_e$ per expert $e$
2: **for** each expert $e$ with $\mathcal{T}_e \neq \emptyset$ (all experts, in parallel) **do**
3: $M \leftarrow \text{round}(\Delta \odot H_{\mathcal{T}_e})$ with $\Delta_t = 2^{Q-2}/(\beta_{\text{up}}\|h_t\|)$; pad $|\mathcal{T}_e|$ to the layer maximum if layermax
4: send (seed, $B$) with $B = -As + e + M$, fresh $s$; receive $Y = \text{ModSwitch}(W_{1,3}[A|B])$
5: $G, U \leftarrow \text{Dec}_s(Y) \odot \text{scale}$ (use the offline hint $W_{1,3}A$ in hint mode)
6: $Z \leftarrow \text{SiLU}(G) \odot U \odot w_{\mathcal{T}_e,e}$ (exact, float64)
7: $M' \leftarrow \text{round}(\Delta' \odot Z)$ with $\Delta'_t = 2^{Q-2}/(\beta_{\text{down}}\|z_t\|)$
8: draw a fresh secret $s'$
9: send (seed$'$, $B'$) with $B' = -A's' + e' + M'$; receive $Y' = \text{ModSwitch}(W_2[A'|B'])$
10: $O_{\mathcal{T}_e} \mathrel{+}= \text{Dec}_{s'}(Y') \odot \text{scale}'$
11: **end for**
12: **return** $O$

**Routing-count padding.** The orchestrator must know how many tokens each expert receives. Padding every expert in a layer to the layer's maximum count, which we call layermax, leaves only one number per layer visible.

**HE-compatible quantization.** Weights can be quantized to int8 or int4 per output row with RTN, AWQ (Lin et al., 2024), GPTQ (Frantar et al., 2023) or AWQ followed by GPTQ. Row scales are applied after decryption, and AWQ input scales are folded into the client's plaintext, so the server still sees a pure integer matrix. Fewer weight bits allow a smaller modulus and fewer GEMM units.

Algorithm 1 summarizes one MoE layer. Only lines 4 and 9 involve the servers, and both are single exact GEMMs; everything else is local and cheap.

**Where the bytes go.** For a job with $T$ routed tokens, input width $d_{\text{in}}$ and output width $d_{\text{out}}$, the uplink carries $d_{\text{in}}T$ coefficients of $Q$ bits and the downlink carries $d_{\text{out}}(n+T)$ coefficients of $Q'$ bits:

$$\text{bytes}(T) = \tfrac{1}{8}\big(Q\, d_{\text{in}}T + Q'_A\, d_{\text{out}}\, n + Q'\, d_{\text{out}}T\big). \tag{4}$$

The middle term is independent of the input and of $T$. It is exactly the part that the public hint moves offline and that batching amortizes, and it explains why hint mode and batching give the two largest reductions in Sections 4.2 and 4.3. The two remaining terms scale with the tokens actually routed, so the online cost per token is independent of how many experts the model has; this is the property that lets the design scale with sparsity.

## 4 Experiments

**Setup.** We use GenomeOcean-MoE (Zhou et al., 2025b) (15.1B parameters, 24 layers, 8 experts, top-2). Evaluation uses 72 windows of 1024 tokens drawn from 12 complete bacterial genomes spanning four phyla, with three seeds (44, 142, 256); four further genomes serve only for quantization calibration. Every experiment follows a protocol frozen before data collection, with a single primary endpoint, a pre-registered prediction and, where applicable, a known-bad configuration that must fail. Confidence intervals are cluster bootstraps over genomes (10,000 resamples). Systems run on H100 GPUs with the client, an orchestrator and four server containers, each holding two experts, as separate processes communicating over TCP (Appendix C).

### 4.1 Fidelity and security

**Encryption is non-inferior to exact inference.** With int16 weights, $Q = 48$ and $Q' = 24$, the KL divergence of CipherGenome to the fp32 model is 0.0032 nats per token, against 0.0030 for exact fp64 experts and 0.0033 for standard bf16 inference (Figure 2a). The primary endpoint, the KL added by encryption, is $2.54 \times 10^{-4}$ with a 95% CI of $[1.21 \times 10^{-4}, 3.95 \times 10^{-4}]$, below the pre-registered margin of $5 \times 10^{-4}$ (Figure 2b). Relative to bf16, CipherGenome differs by $-1.09 \times 10^{-4}$ with a CI that includes zero, and top-1 agreement is 94.3% against 94.4%, so encrypted outsourcing is as faithful as standard inference. On an fp32 backbone, where bf16 routing flips are absent, the

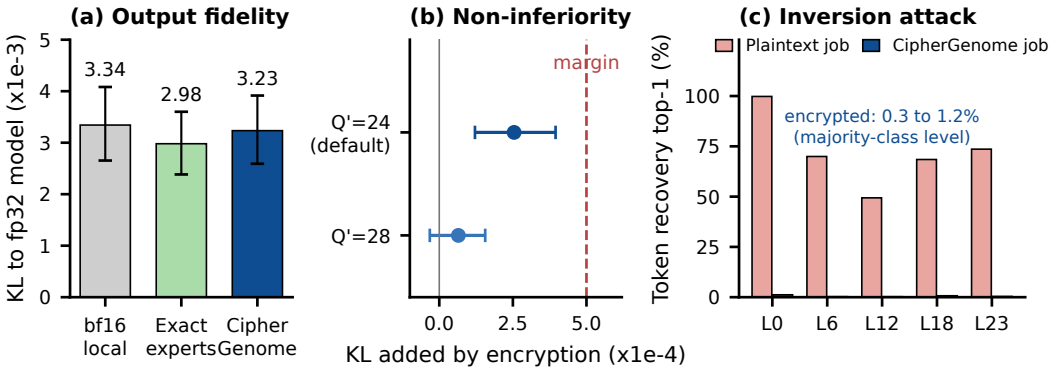



Figure 2: **Fidelity and confidentiality.** (a) KL divergence of next-token distributions to the fp32 model, 72 windows; bf16 is standard local inference, exact evaluates experts in fp64. (b) KL added by encryption relative to exact experts, with 95% cluster-bootstrap CIs and the pre-registered $5 \times 10^{-4}$ margin. (c) Top-1 token recovery by a server that hosts one expert and trains a decoder on its inputs, mean over three seeds; plaintext jobs versus CipherGenome jobs.

KL of CipherGenome to the fp32 model is only $1.6 \times 10^{-4}$ with 99.7% top-1 agreement. A known-bad setting ($Q' = 16$ on int8 weights) fails with a KL increase of $2.79 \times 10^{-2}$, which confirms that the endpoint detects real degradation. Raising $Q'$ to 28 lowers the added KL to $6.5 \times 10^{-5}$ at the cost of more download.

**Where the residual error comes from.** Replaying the fp32 model's routing decisions isolates the experts from bf16 routing flips. With gold routing, the KL to the fp32 model falls to $4.0 \times 10^{-4}$ for both exact and encrypted experts, and the two differ by only $2.2 \times 10^{-6}$ (Table 5). Almost all of the residual divergence of any bf16 deployment therefore comes from routing flips caused by bf16 attention, not from encryption. At the level of a single expert call, the mean relative error of a decrypted output is $2.8 \times 10^{-4}$ over 13,824 calls, dominated by int16 weight quantization.

**Encrypted jobs defeat inversion.** We give an attacker the strongest single-expert position: it hosts one expert, sees every job for it, and trains a decoder on 21,000 to 27,000 labelled tokens from training genomes to predict each input nucleotide token. On plaintext jobs it recovers 99.8% of tokens at layer 0 and 42 to 74% at layers 6 to 23 (Figure 2c). On CipherGenome jobs, accuracy in all ten (layer, expert) cells lies at the majority-class baseline, 0.19 to 1.20%, and the pre-registered endpoint passes. Positive controls with broken cryptography, a reused secret or a zero secret, restore recovery to 98.8% and 98.8%, so the attack is strong and the protection comes from encryption itself.

### 4.2 End-to-end system

Figure 3a reports end-to-end latency. In compact mode, which downloads $WA$ with every job, a 1024-token window takes 24.28 s. The public hint removes this term and brings latency to 6.85 s, a 3.54-times speedup (95% CI 3.29 to 3.80), and cuts online traffic from 31.3 to 8.94 MB per token. Adding layermax padding costs 1.2 times the hint latency (8.23 s). With int8 AWQ+GPTQ experts at $Q = 32$, a configuration that is non-inferior to its own plaintext model (Section 4.5), latency drops to 5.03 s, 4.76 times faster than compact mode and within 3.8 times of unprotected expert parallelism (1.33 s). The client does not store the 29 GB of expert weights; its GPU footprint during the run peaks at 6.3 GiB.

Against a standard homomorphic baseline the gap is far larger. Evaluating one GenomeOcean expert on one token with CKKS in Microsoft SEAL (Benaissa et al., 2021) takes at least 49 to 57 seconds on a CPU, whereas the CipherGenome server spends 8.1 microseconds per token per expert on an H100 at 4096 tokens per job (Figure 3b). The hardware differs, but a gap of more than six orders of

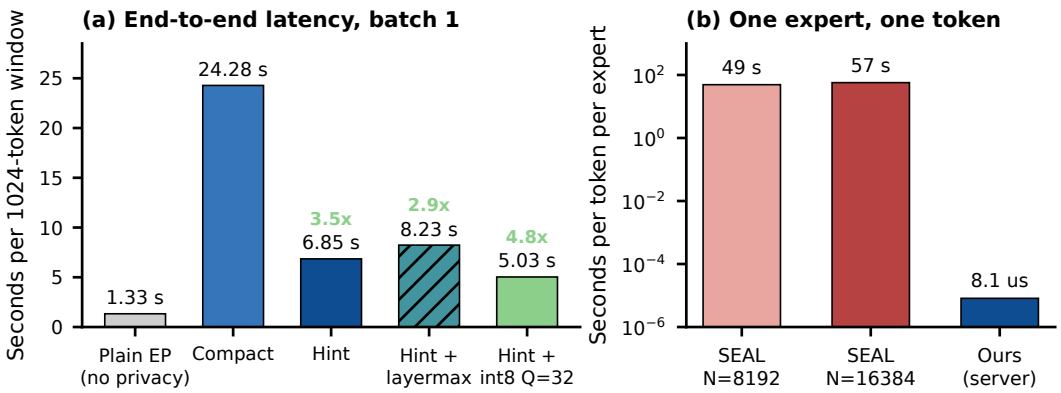



Figure 3: **Latency.** (a) Median seconds per 1024-token window at batch size 1 for the multi-process system on the same two H100 GPUs, 12 windows per arm; green labels give the speedup over compact mode. Plain EP is expert parallelism without encryption. (b) Seconds per token for one expert on a CKKS baseline (Microsoft SEAL via TenSEAL, CPU, lower bound by tile counting) and for the CipherGenome server job (H100).

Table 1: Design comparison with representative private transformer inference systems. CipherGenome is the only design that keeps nonlinearities exact, needs no trusted second party, and tolerates collusion of all servers; the price is that weights must be public, which is the norm for released genome models.

| System | Primitive | Nonlinearity | Client interaction | Colluding servers | MoE |
|---|---|---|---|---|---|
| THE-X, NEXUS, THOR | CKKS | polynomial approx. | none | n/a (one server) | no |
| BumbleBee, PUMA, BOLT | 2PC | approx. + protocols | many rounds | no | no |
| CryptoMoE, SecMoE | 2PC | approx. + protocols | many rounds | no | yes |
| Two-server sharing | additive shares | exact (client) | 2 per layer | no | any |
| CipherGenome (ours) | module-LWE | exact (client) | 2 per layer | yes | yes |

magnitude is not explained by hardware: CipherGenome avoids ciphertext rotations, relinearization and bootstrapping entirely. Table 1 places the design among prior private-inference approaches.

### 4.3 Reducing bandwidth

Because the nonlinearity runs on the client, every expert job returns its output, and the protocol's traffic is dominated by message sizes rather than rounds. Wire compression shrinks those messages without touching the arithmetic (Figure 4a). For int16 experts in hint mode, 16-token granularity, split $A/B$ widths and 8-bit uplink rounding lower online traffic from 8.92 to 5.43 MB per token, a 39% reduction, with a KL of $1.45 \times 10^{-4}$ to the plaintext model (CI upper bound $1.89 \times 10^{-4}$) and an unchanged expert error. Without any client-side hint, batching eight windows shares the $A$ part and lowers compact-mode traffic from 31.34 to 8.09 MB per token (74% less). For int4 experts at $Q = 32$, where the lattice dimension can drop to 1536 while keeping 138 bits of core-SVP security, hint-mode traffic falls to 4.59 MB per token and batched compact traffic to 6.85 MB per token, again non-inferior (KL $9.8 \times 10^{-5}$). Relative to the original compact protocol this is a 6.8-fold reduction. Known-bad settings with more aggressive rounding fail the same endpoint, which shows where the safe region ends.

### 4.4 Routing leakage and its mitigation

The orchestrator must learn how many tokens each expert receives, and these counts carry information. From the 192 per-(layer, expert) block counts of a 1024-token window, a logistic-regression classifier identifies the source genome among 16 taxa with 54.9% accuracy (chance 6.25%, Figure 4b), and leakage grows with sequence length (Appendix G). Layermax padding reveals only one

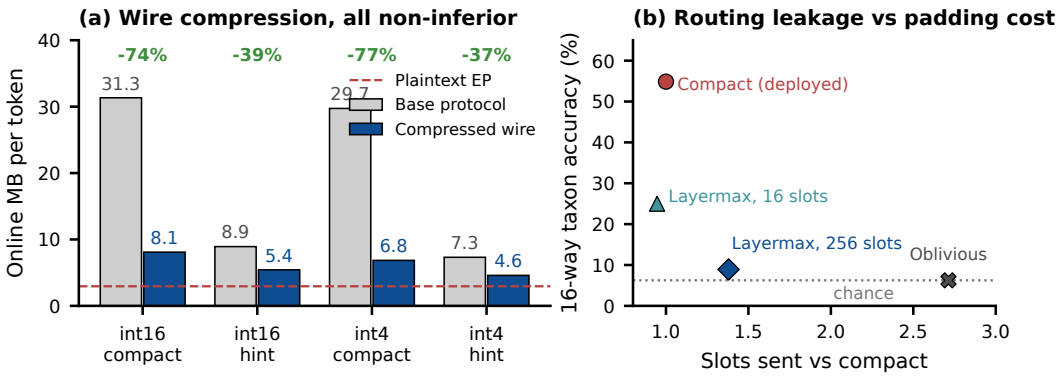



Figure 4: **Traffic and routing leakage.** (a) Online megabytes per token before and after wire compression for int16 and int4 experts, in compact mode with batch 8 and in hint mode; every compressed configuration is non-inferior to its plaintext model on 72 windows. The dashed line is unprotected expert parallelism with the same client-side SwiGLU. (b) Accuracy of a 16-way taxon classifier that sees only per-expert token counts, against the number of slots sent relative to compact mode, $T = 1024$.

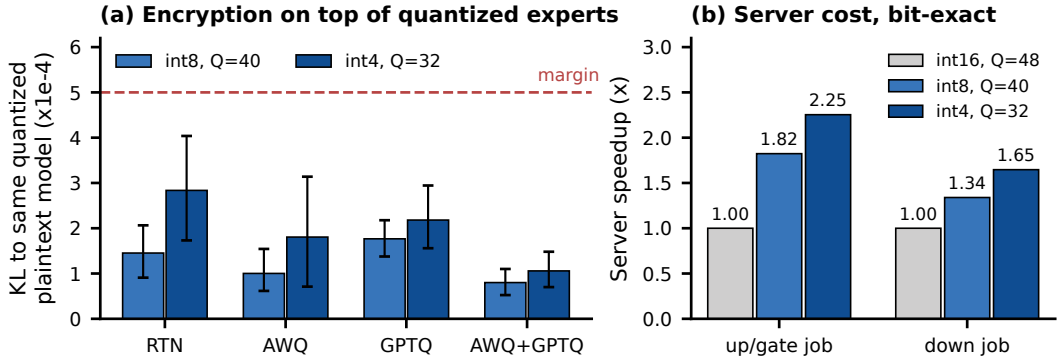



Figure 5: **Quantized deployment.** (a) KL divergence between the CipherGenome model and the plaintext model with the same public quantized experts, fp32 backbone, 72 windows, with 95% CIs and the non-inferiority margin. (b) Server job speedup over int16 at $Q = 48$ for the up/gate and down projections at 4096 tokens; all kernels are bit-exact.

number per layer and reduces the accuracy to 8.9%, near chance, at 1.38 times the slots of compact mode. Oblivious padding removes the channel entirely at 2.71 times the cost. Combining layermax with the 16-token granularity of Section 4.3 sends 0.95 times the slots of compact mode while lowering accuracy to 25.0%, so bandwidth and privacy improve together.

### 4.5 Quantized Experts

A deployer may prefer smaller experts. When the model is released with HE-compatible int8 or int4 experts, the right question is whether encryption preserves that model. It does (Figure 5a). With int8 experts at $Q = 40$, all four quantization methods are non-inferior to their plaintext counterparts, with KL between $8.0 \times 10^{-5}$ and $1.8 \times 10^{-4}$. With int4 experts at $Q = 32$, all four are again non-inferior, with KL between $1.1 \times 10^{-4}$ and $2.8 \times 10^{-4}$ and at least 99.6% top-1 agreement. A known-bad downlink width fails. Fewer weight bits make the server job cheaper, up to 2.25 times faster than int16 for int4 at $Q = 32$, while remaining bit-exact (Figure 5b). Smaller moduli also raise the security margin to 201.2 bits of core-SVP hardness.

## 5 DISCUSSION AND CONCLUSION

CipherGenome shows that exact homomorphic outsourcing of a 15B genome model is practical once the protocol follows the model's structure. Keeping nonlinearities on the client removes approximation and bootstrapping; public expert weights reduce the server's work to one exact integer GEMM; fresh-secret LWE makes the design robust to collusion among all servers; and the MoE router, which creates a new leakage channel, can be padded cheaply.

**Scaling with sparsity.** The client's work per token is the embedding, attention and router, plus encryption and decryption of the two routed expert inputs and outputs; none of it grows with the number of experts. The servers' work and the online traffic scale with the routed tokens, not with total model size (Equation (4)). Larger and sparser MoE models, which place an ever larger share of parameters in experts, therefore move an ever larger share of the model to untrusted hardware at a constant per-token client cost. GenomeOcean-MoE already places 95.8% of its parameters in experts; frontier language MoEs place more.

**Scope.** The protocol is best suited to prefill-style workloads such as scoring, embedding and variant-effect prediction over long sequences, which dominate genomic applications, and to datacenter links where a few megabytes per token are affordable. Its main cost is the round trip for the client-side SwiGLU. Moving parts of that nonlinearity to the server with exact encrypted comparison, adding integrity checks against malicious servers, and extending the protocol to token-by-token generation are natural next steps. The same boundary applies to any public-weight MoE, including language models, where it offers exact outputs with a security guarantee that does not depend on trusting any single server.

## AI USE STATEMENT

We used generative AI tools for two purposes. First, we used them to aid and polish the writing of this paper, improving wording, grammar and readability. Second, we used generative AI tools to generate synthetic datasets: the random activation matrices, random integer weight matrices and random ciphertexts used in the kernel correctness tests and microbenchmarks of Appendix D are synthetic data produced with AI-assisted scripts. We have reviewed all AI-assisted content and take full responsibility for the final content of this work.

## ETHICS STATEMENT

This work protects the confidentiality of genomic sequences during model inference. All sequences we use are public reference genomes of bacterial strains from NCBI RefSeq; no human data or human subjects are involved. The inversion attack of Section 4.1 is evaluated only on public data and demonstrates a risk that our protocol removes. The routing-leakage analysis quantifies a side channel so that deployments can mitigate it. CipherGenome protects inputs against honest-but-curious servers; it does not protect against a compromised client or against servers that return incorrect results, and deployments handling human genomes must combine it with the usual governance and consent processes.

## REPRODUCIBILITY STATEMENT

Section 3 and Appendices A and B specify the protocol, parameters and security argument. Appendix C lists the model, genomes, window sampling, seeds and hardware, and Appendix J summarizes the frozen protocol, primary endpoint and known-bad control of every experiment. The supplementary material contains the implementation of the client, server kernels and orchestrator, unit tests for bit-exactness, the scripts that produce every table and figure from per-run records, and the per-window results behind each number reported here.

## REFERENCES


Ahmad Al Badawi, Jack Bates, Flavio Bergamaschi, David Bruce Cousins, Saroja Erabelli, Nicholas Genise, Shai Halevi, Hamish Hunt, Andrey Kim, Yongwoo Lee, Zeyu Liu, Daniele Micciancio, Ian Quah, Yuriy Polyakov, Saraswathy R.V., Kurt Rohloff, Jonathan Saylor, Dmitriy Suponitsky, Matthew Triplett, Vinod Vaikuntanathan, and Vincent Zucca. OpenFHE: Open-source fully homomorphic encryption library. In *Proceedings of the 10th Workshop on Encrypted Computing & Applied Homomorphic Cryptography (WAHC)*, pp. 53–63, 2022. doi: 10.1145/3560827.3563379.

Martin Albrecht, Melissa Chase, Hao Chen, Jintai Ding, Shafi Goldwasser, Sergey Gorbunov, Shai Halevi, Jeffrey Hoffstein, Kim Laine, Kristin Lauter, Satya Lokam, Daniele Micciancio, Dustin Moody, Travis Morrison, Amit Sahai, and Vinod Vaikuntanathan. Homomorphic encryption standard. In *Protecting Privacy through Homomorphic Encryption*, pp. 31–62. Springer, 2021. doi: 10.1007/978-3-030-77287-1_2.

Martin R. Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015. doi: 10.1515/jmc-2015-0016.

Ayoub Benaissa, Bilal Retiat, Bogdan Cebere, and Alaa Eddine Belfedhal. TenSEAL: A library for encrypted tensor operations using homomorphic encryption. In *ICLR 2021 Workshop on Distributed and Private Machine Learning (DPML)*, 2021. arXiv:2104.03152.

Marcelo Blatt, Alexander Gusev, Yuriy Polyakov, and Shafi Goldwasser. Secure large-scale genome-wide association studies using homomorphic encryption. *Proceedings of the National Academy of Sciences*, 117(21):11608–11613, 2020. doi: 10.1073/pnas.1918257117.

Alexander Borzunov, Dmitry Baranchuk, Tim Dettmers, Max Ryabinin, Younes Belkada, Artem Chumachenko, Pavel Samygin, and Colin Raffel. Petals: Collaborative inference and fine-tuning of large models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 558–568, 2023. doi: 10.18653/v1/2023.acl-demo.54.

Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory*, 6(3):1–36, 2014. doi: 10.1145/2633600.

Garyk Brixi, Matthew G. Durrant, Jerome Ku, Michael Poli, Greg Brockman, Daniel Chang, Gabriel A. Gonzalez, Samuel H. King, David B. Li, Aditi T. Merchant, et al. Genome modeling and design across all domains of life with Evo 2. *bioRxiv*, 2025. doi: 10.1101/2025.02.18.638918. Preprint 2025.02.18.638918.

Tianyu Chen, Hangbo Bao, Shaohan Huang, Li Dong, Binxing Jiao, Daxin Jiang, Haoyi Zhou, Jianxin Li, and Furu Wei. THE-X: Privacy-preserving transformer inference with homomorphic encryption. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 3510–3520, 2022. doi: 10.18653/v1/2022.findings-acl.277.

Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. In *Advances in Cryptology (ASIACRYPT 2017)*, volume 10624 of *Lecture Notes in Computer Science*, pp. 409–437, 2017. doi: 10.1007/978-3-319-70694-8_15.

Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. DeepSeekMoE: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 1280–1297, 2024. doi: 10.18653/v1/2024.acl-long.70.

Hugo Dalla-Torre, Liam Gonzalez, Javier Mendoza-Revilla, Nicolas Lopez Carranza, Adam Henryk Grzywaczewski, Francesco Oteri, Christian Dallago, Evan Trop, Bernardo P. de Almeida, Hassan Sirelkhatim, Guillaume Richard, Marcin Skwark, Karim Beguir, Marie Lopez, and Thomas Pierrot. Nucleotide transformer: building and evaluating robust foundation models for human genomics. *Nature Methods*, 22(2):287–297, 2025. doi: 10.1038/s41592-024-02523-z.

DeepSeek-AI. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. LLM.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 30318–30332, 2022.

Ruyi Ding, Tianhong Xu, Xinyi Shen, Aidong Adam Ding, and Yunsi Fei. MoEcho: Exploiting side-channel attacks to compromise user privacy in mixture-of-experts LLMs. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 2159–2173, 2025. doi: 10.1145/3719027.3765174.

Ye Dong, Wen-jie Lu, Yancheng Zheng, Haoqi Wu, Derun Zhao, Jin Tan, Zhicong Huang, Cheng Hong, Tao Wei, Wenguang Chen, and Jianying Zhou. PUMA: Secure inference of LLaMA-7B in five minutes. *Security and Safety*, 2025. doi: 10.1051/sands/2025014.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. OPTQ: Accurate quantization for generative pre-trained transformers. In *International Conference on Learning Representations (ICLR)*, 2023. Preprint title: GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers.

Jordan Frery, Roman Bredehoft, Jakub Klemsa, Arthur Meyre, and Andrei Stoian. Private LoRA fine-tuning of open-source LLMs with homomorphic encryption. *arXiv preprint arXiv:2505.07329*, 2025.

Christos Galanopoulos, Kimon Antonios Provatas, and Ilias Georgakopoulos-Soares. Feasibility of homomorphic inference for a genomic foundation model. *arXiv preprint arXiv:2609.16211*, 2026.

Alexandra Henzinger, Matthew M. Hong, Henry Corrigan-Gibbs, Sarah Meiklejohn, and Vinod Vaikuntanathan. One server for the price of two: Simple and fast single-server private information retrieval. In *32nd USENIX Security Symposium*, pp. 3889–3905, 2023.

Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V. Pearson, Dietrich A. Stephan, Stanley F. Nelson, and David W. Craig. Resolving individuals contributing trace amounts of DNA to highly complex mixtures using high-density SNP genotyping microarrays. *PLoS Genetics*, 4(8):e1000167, 2008. doi: 10.1371/journal.pgen.1000167.

Zhicong Huang, Wen-jie Lu, Cheng Hong, and Jiansheng Ding. Cheetah: Lean and fast secure two-party deep neural network inference. In *31st USENIX Security Symposium*, pp. 809–826, 2022.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha Chandrakasan. GAZELLE: A low latency framework for secure neural network inference. In *27th USENIX Security Symposium*, pp. 1651–1669, 2018.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles (SOSP)*, pp. 611–626, 2023. doi: 10.1145/3600006.3613165.

Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Designs, Codes and Cryptography*, 75(3):565–599, 2015. doi: 10.1007/s10623-014-9938-4.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. GShard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations (ICLR)*, 2021.

Dacheng Li, Hongyi Wang, Rulin Shao, Han Guo, Eric P. Xing, and Hao Zhang. MPCFormer: fast, performant and private transformer inference with MPC. In *International Conference on Learning Representations (ICLR)*, 2023.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In *Proceedings of Machine Learning and Systems (MLSys)*, volume 6, pp. 87–100, 2024.

Wen-jie Lu, Zhicong Huang, Zhen Gu, Jingyu Li, Jian Liu, Cheng Hong, Kui Ren, Tao Wei, and WenGuang Chen. BumbleBee: Secure two-party inference framework for large transformers. In *Network and Distributed System Security Symposium (NDSS)*, 2025. doi: 10.14722/ndss.2025.230057.

Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *Journal of the ACM*, 60(6):1–35, 2013. doi: 10.1145/2535925.

Payman Mohassel and Yupeng Zhang. SecureML: A system for scalable privacy-preserving machine learning. In *IEEE Symposium on Security and Privacy (S&P)*, pp. 19–38, 2017. doi: 10.1109/SP.2017.12.

Jungho Moon, Dongwoo Yoo, Xiaoqian Jiang, and Miran Kim. THOR: Secure transformer inference with homomorphic encryption. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 3765–3779, 2025. doi: 10.1145/3719027.3765150.

John X. Morris, Volodymyr Kuleshov, Vitaly Shmatikov, and Alexander M. Rush. Text embeddings reveal (almost) as much as text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 12448–12460, 2023. doi: 10.18653/v1/2023.emnlp-main.765.

Alexander Nemecek, Wenbiao Li, Xiaoqian Jiang, Jaideep Vaidya, and Erman Ayday. Quantifying memorization and privacy risks in genomic language models. *arXiv preprint arXiv:2603.08913*, 2026.

Eric Nguyen, Michael Poli, Marjan Faizi, Armin Thomas, Michael Wornow, Callum Birch-Sykes, Stefano Massaroli, Aman Patel, Clayton Rabideau, Yoshua Bengio, Stefano Ermon, Christopher Ré, and Stephen Baccus. HyenaDNA: Long-range genomic sequence modeling at single nucleotide resolution. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 43177–43201, 2023.

Amir Nuriyev and Gabriel Kulp. Expert selections in MoE models reveal (almost) as much as text. *arXiv preprint arXiv:2602.04105*, 2026.

Sofiane Ouaari, Jules Kreuer, and Nico Pfeifer. How private are DNA embeddings? inverting foundation model representations of genomic sequences. *arXiv preprint arXiv:2603.06950*, 2026.

Qi Pang, Jinhao Zhu, Helen Möllering, Wenting Zheng, and Thomas Schneider. BOLT: Privacy-preserving, accurate and efficient inference for transformers. In *IEEE Symposium on Security and Privacy (S&P)*, pp. 4753–4771, 2024. doi: 10.1109/SP54263.2024.00130.

Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. DeepSpeed-MoE: Advancing mixture-of-experts inference and training to power next-generation AI scale. In *International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 18332–18346, 2022.

Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM*, 56(6):1–40, 2009. doi: 10.1145/1568318.1568324.

Noam Shazeer. GLU variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*, 2017.

Bowen Shen, Yuyue Chen, Peng Yang, Bin Zhang, Xi Zhang, and Zoe L. Jiang. SecMoE: Communication-efficient secure MoE inference via select-then-compute. *arXiv preprint arXiv:2601.06790*, 2026.

Florian Tramèr and Dan Boneh. Slalom: Fast, verifiable and private execution of neural networks in trusted hardware. In *International Conference on Learning Representations (ICLR)*, 2019.

Itay Yona, Ilia Shumailov, Jamie Hayes, and Nicholas Carlini. Stealing user prompts from mixture of experts. *arXiv preprint arXiv:2410.22884*, 2024.

Jiawen Zhang, Xinpeng Yang, Lipeng He, Kejia Chen, Wen-jie Lu, Yinghao Wang, Xiaoyang Hou, Jian Liu, Kui Ren, and Xiaohu Yang. Secure transformer inference made non-interactive. In *Network and Distributed System Security Symposium (NDSS)*, 2025. doi: 10.14722/ndss.2025.230868.

Yifan Zhou, Tianshi Xu, Jue Hong, Ye Wu, and Meng Li. CryptoMoE: Privacy-preserving and scalable mixture of experts inference via balanced expert routing. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025a.

Zhihan Zhou, Yanrong Ji, Weijian Li, Pratik Dutta, Ramana V. Davuluri, and Han Liu. DNABERT-2: Efficient foundation model and benchmark for multi-species genomes. In *International Conference on Learning Representations (ICLR)*, 2024.

Zhihan Zhou, Robert Riley, Satria Kautsar, Weimin Wu, Rob Egan, Steven Hofmeyr, Shira Goldhaber-Gordon, Mutian Yu, Harrison Ho, Fengchen Liu, Feng Chen, Rachael Morgan-Kiss, Lizhen Shi, Han Liu, and Zhong Wang. GenomeOcean: An efficient genome foundation model trained on large-scale metagenomic assemblies. *bioRxiv*, 2025b. doi: 10.1101/2025.01.30.635558. Preprint 2025.01.30.635558, version 1.

## A Proof of Proposition 1

Let $w_i$ be row $i$ of the integer weight $W$ and $m_t = \mathrm{round}(\Delta_t x_t)$ with $\Delta_t = 2^{Q-2}/(\beta\|x_t\|_2)$. Write $m_t = \Delta_t x_t + \rho_t$ with $\|\rho_t\|_\infty \le 1/2$. By Cauchy–Schwarz,

$$|w_i^\top m_t| \le \|w_i\|_2 \, \|m_t\|_2 \le \beta \left(\Delta_t \|x_t\|_2 + \|\rho_t\|_2\right) = 2^{Q-2} + \beta\|\rho_t\|_2.$$

Since $\|\rho_t\|_2 \le \sqrt{d}/2$ and, for every weight configuration we deploy, $\beta\sqrt{d}/2 \le 2^{Q-2}$ (the largest value, $\beta \approx 2^{19.8}$ for int16 weights with $d = 8192$, gives $\beta\sqrt{d}/2 \approx 2^{25.3} \ll 2^{46}$), we obtain $|w_i^\top m_t| < 2^{Q-1}$. The integer $w_i^\top m_t$ therefore lies in the centered representative range of $\mathbb{Z}_{2^Q}$, and the decrypted value $w_i^\top m_t + w_i^\top e_t$ recovers it up to the noise term, whose magnitude is at most $\beta\|e_t\|_2$, orders of magnitude below $2^{Q-1}$. The modulus switch to $2^{Q'}$ adds a rounding error of at most $1/2$ per coefficient, amplified by the ternary secret in the $A$ part, which fixes the admissible $Q'$. □

**Error decomposition.** A decrypted expert output differs from the exact fp64 expert by four terms: (i) per-row weight quantization, (ii) rounding of $m_t$, relative size $\sqrt{d}/(2\Delta_t\|x_t\|) = \beta\sqrt{d}/2^{Q-1}$, (iii) LWE noise $We$, and (iv) modulus-switch rounding. With int16 weights, $Q = 48$ and $Q' = 24$, the measured mean relative error of an expert output is $2.76 \times 10^{-4}$ (maximum $3.79 \times 10^{-4}$ over 13,824 expert calls), dominated by term (i).

## B Security

**Leakage function.** The orchestrator and every server observe, for each job, the tuple (layer, expert, stage, number of columns), the public mask seed, the ciphertext $B$, and the modulus-switched output. The client re-encrypts after every nonlinearity with fresh secrets, so no ciphertext is ever a function of another ciphertext's secret. Under decision module-LWE with rank $r = 8$ over $R = \mathbb{Z}[X]/(X^{256} + 1)$, the pair $(A, -As + e)$ is indistinguishable from uniform; with $k$ independent secrets sharing one $A$ this extends by a standard hybrid over the $k$ columns. Adding $M$ is a bijection on the uniform distribution, so the entire view is simulatable from the leakage function alone, for any coalition of servers and the orchestrator. The fixed public mask of hint mode is the multi-secret setting in which the same public $A$ is reused with fresh secrets and fresh noise, which is again a hybrid over independent-secret samples. Modulus switching, the split $A/B$ widths and uplink low-bit rounding are deterministic functions of the ciphertext and cannot increase an adversary's advantage.

**Concrete parameters.** We use $n = 2048$, ternary secrets and centered-binomial noise with $k = 21$ (standard deviation 3.24). The HE standard (Albrecht et al., 2021) lists $\log q \le 54$ as 128-bit secure at $n = 2048$ for ternary secrets. We additionally estimate the primal-uSVP and dual attacks (Albrecht et al., 2015) with the core-SVP cost $2^{0.292b}$ for BKZ block size $b$ and with the BKZ-sieve full-cost model of the HE standard tables (Figure 6, Table 2). All deployed moduli ($Q \le 48$) lie well inside the standard's 128-bit region. For the int4 deployment at $Q = 32$, the dimension can be lowered to $n = 1536$ while retaining 138.1 bits of core-SVP security and 169.0 bits under the BKZ-sieve model.

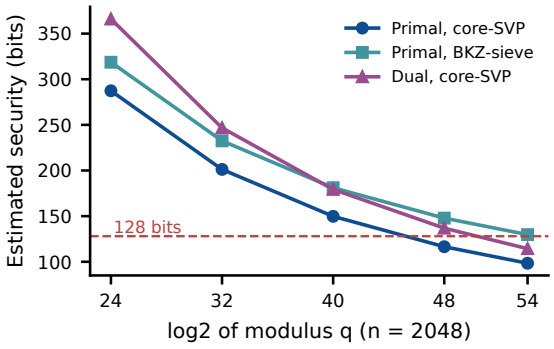



Figure 6: Estimated security of $n = 2048$ module-LWE with ternary secrets and $\sigma = 3.24$ as a function of the modulus. The deployed moduli are $Q = 48$ (int16 experts), $Q = 40$ (int8) and $Q = 32$ (int4).

Table 2: Concrete security estimates for $n = 2048$ (bits). The BKZ-sieve column follows the cost model of the HE standard tables.

| $\log_2 q$ | BKZ block size $b$ | Primal, BKZ-sieve | Primal, core-SVP (classical) | Dual, core-SVP |
|---|---|---|---|---|
| 24 | 984 | 318.6 | 287.3 | 366.0 |
| 32 | 689 | 232.5 | 201.2 | 246.7 |
| 40 | 513 | 181.1 | 149.8 | 179.3 |
| 48 | 399 | 147.9 | 116.5 | 136.8 |

## C EXPERIMENTAL SETUP

**Model.** GenomeOcean-MoE (Zhou et al., 2025b): 24 layers, 8 experts per layer, top-2 routing, hidden size 3072, expert width 8192, SwiGLU experts, grouped-query attention with 12 query and 4 key-value heads, vocabulary of 4097 BPE tokens. Of 15.1B parameters, 14.5B (95.8%) are expert weights, and experts account for 85.7% of the per-token linear FLOPs.

**Data.** Twelve complete bacterial reference genomes from NCBI RefSeq (Table 3) are used for evaluation. Four further genomes (*Francisella tularensis*, *Agrobacterium fabrum*, *Shewanella oneidensis* and *Bacteroides thetaiotaomicron*) are used only for quantization calibration and as additional classes in the 16-way routing-leakage task. For fidelity, each seed draws two non-overlapping 6,000-base tiles per genome, which are tokenized and truncated to 1024 tokens, giving 72 windows over three seeds. System measurements use 12 windows (four per seed) with two warm-up passes excluded.

Table 3: Evaluation genomes.

| Accession | Organism | Accession | Organism |
|---|---|---|---|
| NC_000913.3 | *Escherichia coli* K-12 MG1655 | NC_002516.2 | *Pseudomonas aeruginosa* PAO1 |
| NC_000915.1 | *Helicobacter pylori* 26695 | NC_003197.2 | *Salmonella enterica* Typhimurium LT2 |
| NC_000962.3 | *Mycobacterium tuberculosis* H37Rv | NC_003210.1 | *Listeria monocytogenes* EGD-e |
| NC_000964.3 | *Bacillus subtilis* 168 | NC_003888.3 | *Streptomyces coelicolor* A3(2) |
| NC_002505.1 | *Vibrio cholerae* N16961 chr. I | NC_007795.1 | *Staphylococcus aureus* NCTC 8325 |
| NC_009089.1 | *Clostridioides difficile* 630 | NC_016845.1 | *Klebsiella pneumoniae* HS11286 |

**Metrics.** Fidelity is the mean over tokens and windows of $\mathrm{KL}(p_{\mathrm{ref}} \,\|\, p_{\mathrm{arm}})$ between next-token distributions, where the reference is the fp32 model with original weights (Section 4.1) or the plaintext model with the same public quantized experts (Sections 4.3 and 4.5). Top-1 agreement is the fraction of positions with equal argmax. Confidence intervals resample the 12 genomes with replacement (10,000 resamples). Latency is the median over windows of wall-clock seconds per 1024-token window. Traffic counts every byte of every job message, both directions, divided by the number of tokens.

**Hardware and software.** All measurements use NVIDIA H100 80 GB GPUs, PyTorch with `torch._int_mm` for int8 GEMMs and Triton for the fused epilogue. The client, orchestrator and four server containers run as separate container instances that communicate over ZeroMQ TCP on one node; the client uses one GPU and the four server containers share a second GPU. The CKKS baseline uses TenSEAL (Benaissa et al., 2021) on the CPU of the same node.

## D KERNEL DETAILS AND BIT-EXACTNESS

The server computes $Y = W[A|B] \bmod 2^Q$ followed by the switch to $2^{Q'}$. Weights are stored as $L_a$ balanced int8 limbs ($L_a = 2$ for int16, 1 for int8 and int4) stacked row-wise, and the ciphertext columns as $L_c = \lceil Q/8 \rceil$ balanced base-256 digits laid out column-major. One `torch._int_mm` of the $(L_a m) \times d$ stacked weights with the $d \times (L_c \,\mathrm{cols})$ digits yields all $L_a L_c$ int32 partial products. A Triton kernel then forms $\sum_{a,b} 2^{8(a+b)} P_{ab} \bmod 2^Q$, skipping pairs with $8(a+b) \geq Q$, and applies the rounding modulus switch. Laying the digit matrix out column-major raised int8 throughput by

5.3 times over the row-major layout. Table 4 reports server job times for both projection shapes, three job sizes and four weight configurations. Every entry is bit-exact against an FP64 reference implementation on random operands, and a self-check at every $(Q, Q')$ used in Section 4.5 is run before each experiment.

Table 4: Server job time (ms, H100) including mask expansion, GEMM and modulus switch, for $S$ tokens per job. Units are int8 GEMM units after limb-pair pruning. All results are bit-exact.

| Shape (out × in) | $S$ | int16, $Q$=48 (11) | int8, $Q$=40 (5) | int4, $Q$=36 (5) | int4, $Q$=32 (4) |
|---|---|---|---|---|---|
| up/gate, $16384 \times 3072$ | 256 | 6.49 | 3.63 | 3.64 | 3.03 |
| | 1024 | 8.52 | 4.86 | 4.74 | 3.89 |
| | 4096 | 16.90 | 9.27 | 9.26 | 7.50 |
| down, $3072 \times 8192$ | 256 | 6.56 | 5.01 | 5.00 | 4.12 |
| | 1024 | 8.54 | 6.42 | 6.41 | 5.29 |
| | 4096 | 16.48 | 12.30 | 12.31 | 10.01 |

## E FULL FIDELITY RESULTS

Table 5 lists every arm of the fidelity experiment on the bf16 backbone and the fp32 backbone. Replaying the routing decisions of the fp32 model (gold routing) shows that almost all of the remaining KL in both the exact and the encrypted arm comes from bf16 routing flips rather than from the experts: with gold routing both arms fall to about $4 \times 10^{-4}$, and the two differ by only $2.2 \times 10^{-6}$.

Table 5: Fidelity on 72 windows. KL to the fp32 model and the KL added by encryption relative to exact fp64 experts, with 95% cluster-bootstrap CIs.

| Arm | KL to fp32 model | Added KL vs exact [95% CI] | Top-1 agreement | Non-inferior |
|---|---|---|---|---|
| bf16 local inference | 0.0033 | $3.63 \times 10^{-4}$ [$1.19 \times 10^{-4}$, $6.40 \times 10^{-4}$] | 94.4 | no |
| Exact fp64 experts | 0.0030 | $0.00 \times 10^{0}$ [$0.00 \times 10^{0}$, $0.00 \times 10^{0}$] | 94.5 | yes |
| Plaintext int16 experts | 0.0032 | $1.95 \times 10^{-4}$ [$3.48 \times 10^{-5}$, $3.80 \times 10^{-4}$] | 94.4 | yes |
| CipherGenome, $Q'$=24 (default) | 0.0032 | $2.54 \times 10^{-4}$ [$1.21 \times 10^{-4}$, $3.95 \times 10^{-4}$] | 94.3 | yes |
| CipherGenome, $Q'$=28 | 0.0030 | $6.48 \times 10^{-5}$ [$-3.22 \times 10^{-5}$, $1.56 \times 10^{-4}$] | 94.4 | yes |
| Exact experts, gold routing | 0.0004 | $-2.58 \times 10^{-3}$ [$-3.17 \times 10^{-3}$, $-2.02 \times 10^{-3}$] | 95.9 | yes |
| CipherGenome, gold routing | 0.0004 | $-2.58 \times 10^{-3}$ [$-3.17 \times 10^{-3}$, $-2.02 \times 10^{-3}$] | 95.8 | yes |
| CipherGenome, fp32 backbone | 0.0002 | $-2.82 \times 10^{-3}$ [$-3.42 \times 10^{-3}$, $-2.25 \times 10^{-3}$] | 99.7 | yes |
| Known-bad: int8, $Q'$=16 | 0.0309 | $2.79 \times 10^{-2}$ [$2.20 \times 10^{-2}$, $3.41 \times 10^{-2}$] | 84.7 | no |

## F INVERSION ATTACK DETAILS

The attacker hosts expert 0 or expert 3 of layer 0, 6, 12, 18 or 23 and receives every job for that expert. For each seed it trains two decoders, a two-layer MLP on single job columns and a sequence model over consecutive columns, to predict the input token from the features it observes, using windows from training genomes, and it is evaluated on windows from held-out genomes. The plaintext view gives the decoder the expert's input activations. The ciphertext view gives it the CipherGenome ciphertext columns. Two positive controls break the cryptography on purpose: reusing one secret for all jobs, and setting the secret to zero so that $B$ is a noisy plaintext. Table 6 lists the best decoder per cell, averaged over three seeds. A cell passes if the upper 95% Wilson bound of the ciphertext-view accuracy stays within one percentage point of the majority-class baseline.

## G ROUTING-COUNT LEAKAGE

We simulate an orchestrator that records, for each window, the routing counts it would observe under different policies, and train logistic regression and an MLP to predict the source genome. Views are exact token-level routes (V1), exact per-expert token counts (V2), counts rounded up to 256-slot blocks as transmitted in compact mode (V3), and the set of active experts per layer (V4). Figure 7 shows that leakage grows with the number of tokens per request: at 256 tokens the block counts carry no information (8.2% against 8.3% chance on 12 genomes), while at 1024 and 4096 tokens

Table 6: Token recovery top-1 (%) by a server hosting one expert, mean over seeds, best of two decoders. Positive controls at layer 0: reused secret 98.8%, zero secret 98.8%.

| Layer | Expert | Plaintext job | CipherGenome job | Wilson upper bound | Majority baseline | Pass |
|---|---|---|---|---|---|---|
| 0 | 0 | 99.8 | 1.20 | 1.56 | 0.94 | yes |
| 0 | 3 | 96.6 | 0.42 | 0.66 | 0.64 | yes |
| 6 | 0 | 61.9 | 0.27 | 0.46 | 0.63 | yes |
| 6 | 3 | 69.9 | 0.22 | 0.40 | 0.30 | yes |
| 12 | 0 | 49.4 | 0.26 | 0.42 | 0.28 | yes |
| 12 | 3 | 41.8 | 0.19 | 0.35 | 0.18 | yes |
| 18 | 0 | 58.1 | 0.72 | 1.03 | 0.70 | yes |
| 18 | 3 | 68.5 | 0.28 | 0.48 | 0.57 | yes |
| 23 | 0 | 69.9 | 0.27 | 0.51 | 0.21 | yes |
| 23 | 3 | 73.6 | 0.38 | 0.61 | 0.36 | yes |

they reach 58.6% and 82.7%. Table 7 lists the padding policies on the 16-genome task; layermax at either granularity is the practical operating point.

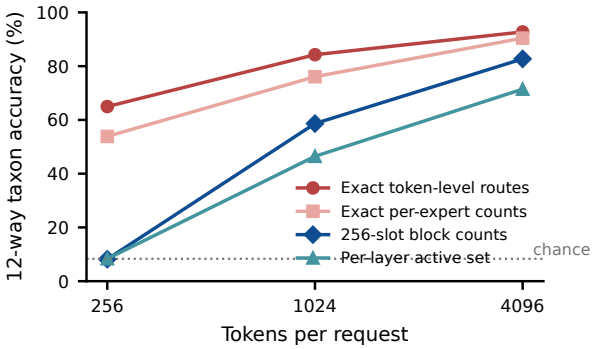



Figure 7: Accuracy of a 12-way genome classifier from routing information visible at different granularities, as a function of the request length; best of logistic regression and MLP, pooled over three seeds.

Table 7: 16-way genome identification from routing counts under padding policies (chance 6.25%), with slots sent relative to compact mode.

| Tokens | Policy | Accuracy (%) | 95% CI | Slots vs compact |
|---|---|---|---|---|
| 1024 | Compact (256-slot blocks) | 54.9 | [44.2, 65.5] | 1.00 |
| | Layermax, 256-slot blocks | 8.9 | [3.6, 15.2] | 1.38 |
| | Layermax, 16-token columns | 25.0 | [19.1, 31.5] | 0.95 |
| | Oblivious | 6.25 (chance) | by construction | 2.71 |
| 4096 | Compact (256-slot blocks) | 78.8 | [71.5, 85.9] | 1.00 |
| | Layermax, 256-slot blocks | 23.3 | [16.2, 32.8] | 1.22 |
| | Layermax, 16-token columns | 43.8 | [32.6, 55.2] | 1.12 |

## H System and bandwidth details

Table 8 gives the end-to-end latency and online traffic of every system configuration at batch sizes 1 and 4. The one-time hint download is 23.0 GB, fetched in about 20 s over the loopback link, and is input-independent and reusable across all requests. Table 9 lists all wire-compression configurations that are non-inferior to their plaintext model.

## I Quantized experts

HE-compatible quantization constrains the weight to the form $W \approx \mathrm{diag}(r)\, W_{\mathrm{int}}\, \mathrm{diag}(1/s)$: the per-row scale $r$ is applied by the client after decryption, and the AWQ input scale $s$ is folded into the plaintext activations before encryption, so the server holds only $W_{\mathrm{int}}$. Group-wise scales are not

Table 8: End-to-end latency (median seconds per 1024-token window) and online traffic, 12 windows per arm.

| Configuration | Latency, batch 1 | Latency, batch 4 | MB/token, batch 1 | MB/token, batch 4 |
|---|---|---|---|---|
| Plaintext expert parallelism (no privacy) | 1.33 | 1.53 | 2.95 | 2.95 |
| CipherGenome, compact | 24.28 | 9.57 | 31.32 | 12.41 |
| CipherGenome, hint | 6.85 | 5.37 | 8.94 | 6.81 |
| CipherGenome, hint + layermax | 8.23 | 6.43 | 12.09 | 7.87 |
| CipherGenome, hint, int8 AWQ+GPTQ, $Q$=32 | 5.03 | 4.56 | 7.30 | 5.60 |

Table 9: Wire compression, 72 windows. $g$: token granularity; $Q'_A/Q'$: downlink widths of the $A$ and $B$ parts; $t$: uplink bits rounded away; $b$: windows per batch. KL is to the plaintext model with the same public experts.

| Experts | $Q$ | $n$ | $g$ | $Q'_A/Q'$ | $t$ | $b$ | KL [95% CI] | Expert error | MB/tok compact | MB/tok hint |
|---|---|---|---|---|---|---|---|---|---|---|
| int16 | 48 | 2048 | 256 | 24/24 | 0 | 1 | $1.5 \times 10^{-4}$ [$1.1 \times 10^{-4}$, $2.0 \times 10^{-4}$] | $2.9 \times 10^{-4}$ | 31.34 | 8.92 |
| int16 | 48 | 2048 | 16 | 24/24 | 0 | 1 | $1.5 \times 10^{-4}$ [$9.8 \times 10^{-5}$, $2.1 \times 10^{-4}$] | $2.9 \times 10^{-4}$ | 28.64 | 6.22 |
| int16 | 48 | 2048 | 16 | 24/22 | 0 | 1 | $1.7 \times 10^{-4}$ [$1.1 \times 10^{-4}$, $2.5 \times 10^{-4}$] | $2.9 \times 10^{-4}$ | 28.40 | 5.98 |
| int16 | 48 | 2048 | 16 | 24/20 | 0 | 1 | $1.5 \times 10^{-4}$ [$1.1 \times 10^{-4}$, $2.1 \times 10^{-4}$] | $3.1 \times 10^{-4}$ | 28.16 | 5.74 |
| int16 | 48 | 2048 | 16 | 24/22 | 8 | 1 | $1.4 \times 10^{-4}$ [$1.1 \times 10^{-4}$, $1.9 \times 10^{-4}$] | $3.0 \times 10^{-4}$ | 27.84 | 5.43 |
| int16 | 48 | 2048 | 256 | 24/24 | 0 | 8 | $1.9 \times 10^{-4}$ [$1.1 \times 10^{-4}$, $2.7 \times 10^{-4}$] | $2.9 \times 10^{-4}$ | 9.22 | 6.41 |
| int16 | 48 | 2048 | 16 | 24/22 | 8 | 8 | $1.7 \times 10^{-4}$ [$1.2 \times 10^{-4}$, $2.3 \times 10^{-4}$] | $3.0 \times 10^{-4}$ | 8.09 | 5.29 |
| int4 | 32 | 2048 | 256 | 24/24 | 0 | 1 | $8.1 \times 10^{-5}$ [$6.0 \times 10^{-5}$, $1.0 \times 10^{-4}$] | $2.8 \times 10^{-4}$ | 29.73 | 7.32 |
| int4 | 32 | 1536 | 16 | 24/22 | 0 | 1 | $1.4 \times 10^{-4}$ [$7.8 \times 10^{-5}$, $2.1 \times 10^{-4}$] | $2.5 \times 10^{-4}$ | 21.68 | 4.87 |
| int4 | 32 | 1536 | 16 | 24/22 | 4 | 1 | $9.8 \times 10^{-5}$ [$5.5 \times 10^{-5}$, $1.5 \times 10^{-4}$] | $2.7 \times 10^{-4}$ | 21.40 | 4.59 |
| int4 | 32 | 1536 | 16 | 24/22 | 0 | 8 | $9.9 \times 10^{-5}$ [$5.3 \times 10^{-5}$, $1.5 \times 10^{-4}$] | $2.5 \times 10^{-4}$ | 6.85 | 4.75 |

used because they would split a single ciphertext product into several. Calibration uses 4 held-out genomes. Table 10 lists every quantized configuration we deploy.

Table 10: CipherGenome with quantized experts against the plaintext model with the same experts, fp32 backbone, 72 windows.

| Experts | Method | $Q/Q'$ | KL [95% CI] | Top-1 (%) | Non-inferior |
|---|---|---|---|---|---|
| int8 | RTN | 40/24 | $1.5 \times 10^{-4}$ [$9.1 \times 10^{-5}$, $2.1 \times 10^{-4}$] | 99.72 | yes |
| int8 | AWQ | 40/24 | $1.0 \times 10^{-4}$ [$6.2 \times 10^{-5}$, $1.5 \times 10^{-4}$] | 99.86 | yes |
| int8 | GPTQ | 40/24 | $1.8 \times 10^{-4}$ [$1.4 \times 10^{-4}$, $2.2 \times 10^{-4}$] | 99.64 | yes |
| int8 | AWQ+GPTQ | 40/24 | $8.0 \times 10^{-5}$ [$5.2 \times 10^{-5}$, $1.1 \times 10^{-4}$] | 99.85 | yes |
| int8 | AWQ | 32/24 | $3.0 \times 10^{-4}$ [$1.8 \times 10^{-4}$, $4.4 \times 10^{-4}$] | 99.55 | yes |
| int8 | AWQ+GPTQ | 32/24 | $1.9 \times 10^{-4}$ [$1.4 \times 10^{-4}$, $2.5 \times 10^{-4}$] | 99.58 | yes |
| int4 | RTN | 32/24 | $2.8 \times 10^{-4}$ [$1.7 \times 10^{-4}$, $4.0 \times 10^{-4}$] | 99.62 | yes |
| int4 | AWQ | 32/24 | $1.8 \times 10^{-4}$ [$7.1 \times 10^{-5}$, $3.1 \times 10^{-4}$] | 99.69 | yes |
| int4 | GPTQ | 32/24 | $2.2 \times 10^{-4}$ [$1.6 \times 10^{-4}$, $2.9 \times 10^{-4}$] | 99.59 | yes |
| int4 | AWQ+GPTQ | 32/24 | $1.1 \times 10^{-4}$ [$7.0 \times 10^{-5}$, $1.5 \times 10^{-4}$] | 99.81 | yes |
| int4, known-bad | AWQ+GPTQ | 32/14 | $2.0 \times 10^{-1}$ [$1.6 \times 10^{-1}$, $2.5 \times 10^{-1}$] | 67.25 | no |

## J PROTOCOL SUMMARY

Every experiment was run under a protocol written and frozen before any data were collected, with a single primary endpoint, three seeds (44, 142, 256) where applicable, and pilot runs excluded from all reported results. Table 11 summarizes the experiments reported in this paper.

Table 11: Experiments, primary endpoints and controls.

| Experiment | Primary endpoint | Control |
|---|---|---|
| Correctness gate | Eleven unit checks: ring products, encryption round trip, GEMM bit-exactness, modulus switch, byte accounting | All must pass |
| Fidelity (Section 4.1) | KL added by encryption vs exact experts; non-inferior if CI upper bound $< 5 \times 10^{-4}$ | Known-bad $Q'$=16 must fail |
| Inversion (Section 4.1) | Ciphertext-view top-1 per (layer, expert) cell | Reused and zero secrets must succeed |
| Routing leakage (Section 4.4) | Genome accuracy from block counts, $T = 1024$ | Chance level |
| System (Section 4.2) | Median latency ratio compact over hint, paired by window | Same GPUs for all arms |
| Quantized experts (Section 4.5) | KL to same quantized plaintext model | Known-bad $Q'$=14 must fail |
| Wire compression (Section 4.3) | KL to plaintext model and exact bytes per token | Three known-bad settings must fail |
| Kernels (Appendix D) | Server job time, bit-exactness | FP64 reference |