Semantic Leakage in CKKS via Homomorphic Noise

Renato J. P. Borseti(✉), Thays Rocha Neri Ferreira, and Fábio Borges

National Laboratory for Scientific Computing, Petrópolis, RJ 25.651-075, Brazil
policani@lncc.br, tferreira@posgrad.lncc.br
http://www.lncc.br/ borges/

Abstract. Homomorphic encryption schemes, particularly the Cheon-Kim-Kim-Song (CKKS) scheme, are widely adopted for privacy-preserving computation over real-valued data. While CKKS provides strong ciphertext security, this paper investigates whether semantic privacy can be compromised through passive observation of the encrypted communication channel. We demonstrate that different data types induce statistically distinguishable noise growth patterns—observable without access to secret keys or plaintext—and formalize this as a passive side-channel attack in which an adversary monitors channel metadata to infer the semantic nature of the encrypted data. Experimental validation using Microsoft SEAL ($n = 8192$, coeff_modulus $= [60, 40, 40, 60]$ bits, scale $= 2^{40}$) shows high inference accuracy under type-specific circuits ($\widehat{\mathsf{Adv}} = 0.750$) and residual but reduced leakage under uniform circuits ($\widehat{\mathsf{Adv}} = 0.170$), confirming that the primary leakage vector is computational behavior rather than data distribution. Our results reveal a previously underexplored privacy vulnerability in CKKS deployments and propose circuit uniformization as a practical countermeasure.

Keywords: Homomorphic encryption · CKKS · side-channel attack · semantic privacy · noise analysis · passive adversary

1 Introduction

Homomorphic Encryption (HE) has emerged as a foundational primitive for privacy-preserving computation, enabling arithmetic operations to be performed directly on encrypted data without decryption. The CKKS scheme, introduced by Cheon, Kim, Kim, and Song [1], is the leading HE scheme for real-valued computation. Unlike exact schemes such as BGV [2], CKKS operates over approximate arithmetic: it encodes vectors of complex numbers into polynomial ring elements and tolerates small errors in decryption, trading precision for dramatically improved performance on floating-point workloads. This design choice makes CKKS uniquely suited for privacy-preserving machine learning, secure statistical analysis, and federated computation over continuous data—domains where a bounded approximation error is acceptable.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027
O. Gervasi et al. (Eds.): ICCSA 2026, LNCS 16768, pp. 618–632, 2027.
https://doi.org/10.1007/978-3-032-30488-9_38

The security of CKKS is analyzed through ciphertext indistinguishability under chosen plaintext attack, commonly referred to as IND CPA. Under this notion, no adversary with bounded computational resources can distinguish encryptions of two plaintexts that they choose. This guarantee concerns the content of the ciphertext. However, it does not rule out the possibility that observable properties of the communication channel, such as timing, message size, or the noise behavior of ciphertexts during transmission, may reveal information about the nature of the computation being executed. Previous studies on side channel attacks against homomorphic encryption implementations have mainly focused on recovering secret keys through techniques such as power analysis or timing measurements [7,18]. To the best of our knowledge, the question of semantic leakage, namely whether the category of the underlying data can be inferred from channel metadata, has not yet been explored.

This paper investigates precisely this gap. We ask: can a passive adversary, observing only metadata from an encrypted CKKS channel without any access to secret keys or plaintext, infer the semantic category of the data being processed? Our central finding is affirmative and structurally precise. Different data types, like financial, medical, text, and binary, induce systematically different sequences of homomorphic operations, which in turn produce statistically distinguishable noise growth trajectories. A machine learning classifier trained exclusively on passively observable channel metadata achieves near-perfect inference accuracy when data types are processed through type-specific circuits. More importantly, when all data types are forced through a uniform homomorphic circuit, the inference accuracy drops substantially, but does not reach random baseline, confirming that the leakage is primarily *behavioral* rather than *distributional*.

1.1 CKKS Scheme Definition

Let $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ be the polynomial ring, where n is a power of 2 and q is the coefficient modulus.

Definition 1 (CKKS Ciphertext). *A CKKS ciphertext is a pair* $(c_0, c_1) \in R_q^2$ *satisfying*

$$c_0 + c_1 \cdot s \;\approx\; \Delta \cdot m + e, \tag{1}$$

where $s \in R$ *is the secret key,* $m \in \mathbb{C}^{n/2}$ *is the plaintext message,* $\Delta = 2^\ell$ *is the scaling factor, and* $e \in R$ *is Gaussian noise with* $e \sim \mathcal{N}(0, \sigma^2)$.

Remark 1. CKKS is an approximate scheme by design: decryption recovers m up to a small error bounded by $\|e\|_\infty / \Delta$. This inherent noise is the leakage vector exploited in this work.

1.2 Noise Growth per Homomorphic Operation

Definition 2 (Residual Noise). *The residual noise of a ciphertext* ct *is defined as*

$$\varepsilon(\mathsf{ct}) \;=\; \|c_0 + c_1 \cdot s - \Delta \cdot m\|_\infty . \tag{2}$$

For each homomorphic operation, the noise evolves according to distinct growth laws detailed below.

Addition. Addition accumulates noise linearly, and the growth is largely independent of the plaintext values.

$$\varepsilon(\mathsf{ct}_1 \oplus \mathsf{ct}_2) \;=\; \varepsilon(\mathsf{ct}_1) + \varepsilon(\mathsf{ct}_2) + e_{\mathsf{add}}, \quad e_{\mathsf{add}} \sim \mathcal{N}(0,\, \sigma^2_{\mathsf{add}}). \tag{3}$$

Multiplication. After relinearization and rescaling:

$$\varepsilon(\mathsf{ct}_1 \otimes \mathsf{ct}_2) \approx \frac{1}{\Delta}\Big[\varepsilon(\mathsf{ct}_1)\cdot\|\Delta m_2\|_\infty + \varepsilon(\mathsf{ct}_2)\cdot\|\Delta m_1\|_\infty + \varepsilon(\mathsf{ct}_1)\cdot\varepsilon(\mathsf{ct}_2)\Big] + e_{\mathsf{mul}}. \tag{4}$$

Remark 2 (Data-Noise Coupling—Primary Leakage Mechanism). The dominant term in (4) is proportional to $\|\Delta \cdot m\|_\infty$, the magnitude of the encrypted values. Since different data natures exhibit distinct magnitude distributions—e.g., financial data follows a log-normal distribution with large magnitudes, while text embeddings are sparse and bounded in $[-1, 1]$—multiplicative noise growth is statistically dependent on the data nature. This *data-noise coupling* is the primary leakage mechanism identified in this work.

Rotation. Rotation introduces noise comparable to addition, but through key-switching material, producing a characteristic timing signature observable on the channel.

$$\varepsilon(\mathsf{Rot}(\mathsf{ct})) \;=\; \varepsilon(\mathsf{ct}) + e_{\mathsf{rot}}, \quad e_{\mathsf{rot}} \sim \mathcal{N}(0,\, \sigma^2_{\mathsf{rot}}). \tag{5}$$

2 Related Work

Work specifically targeting side-channel attacks on CKKS remains sparse. Existing literature addresses two threads: leakage through CKKS's approximate nature, and adaptation of side-channel techniques from lattice and FHE implementations. Li and Micciancio [4] show that IND-CPA does not capture realistic passive leakage for approximate schemes, enabling practical key-recovery attacks against HEAAN, SEAL, HElib, and PALISADE. Cheon et al. [5] extend this with IND-CPAD and KRD attacks exploiting bootstrapping failures to recover secret keys in polynomial time. Alexandru et al. [6] respond with Application-Aware Homomorphic Encryption (AAHE), showing that security for CKKS must be specified at the application and parameter level—consistent with our finding that circuit-level decisions determine the degree of semantic leakage.

On physical side channels, Aydin and Aysu [7] demonstrate that SEAL's NTT-based key generation leaks secret coefficients via power and EM profiling, achieving over 98% coefficient recovery from a single trace. Xu et al. [8] and Sim et al. [9] mount chosen-ciphertext EM attacks on Kyber, with methodology transferable to CKKS's RNS-NTT pipeline. Ravi et al. [10] show generic EM-assisted attacks on LWE/LWR schemes, relevant given CKKS's shared decryption structure. Surveys by Ghaleb and Buchanan [11], Lou et al. [12], Picek et

al. [13], and Randolph and Diehl [14] systematize the threat space, confirming that NTT, key switching, and bootstrapping are natural side-channel targets. Recent advances include plaintext-based collision attacks [15], weakly profiling analysis [16], and multi-modal side-channel frameworks achieving AES-128 key recovery with one to three traces [17].

Our work differs fundamentally: rather than recovering secret keys, we ask whether the semantic nature of the data can be inferred by a passive network observer from behavioral metadata alone—a previously unexplored question requiring no physical access to the device.

3 Formalization of Semantic Leakage

Let $\mathcal{D} \in \{d_1, d_2, \ldots, d_K\}$ denote the set of possible data natures ($K = 4$ in our experiments: Financial, Medical, Text, Binary).

Definition 3 (Semantic Leakage). *Semantic leakage occurs when there exists a function ϕ such that* $\exists\, \phi : \mathbb{R}^{T+1} \to \{d_1, \ldots, d_K\} \quad s.t. \quad \Pr\big[\phi(\boldsymbol{\varepsilon}) = \mathcal{D}\big] \gg \frac{1}{K}$.

The function ϕ is implemented as a supervised machine learning classifier trained exclusively on the passively observable metadata vector:

$$\mathbf{x}_{\mathsf{obs}} = \big(\bar{t}_{op},\, \sigma_t,\, t_{\max},\, \textstyle\sum t,\, \bar{t}_{\mathsf{mul}},\, \bar{t}_{\mathsf{add}},\, s_{\mathsf{final}},\, \log_2 s,\, \ell_{\mathsf{final}},\, \Delta\ell,\, \overline{sz},\, sz_{\max},\, \Delta sz,\, N_{op},$$
$$N_{\mathsf{mul}},\, N_{\mathsf{add}},\, N_{\mathsf{rot}},\, r_{\mathsf{mul}},\, r_{\mathsf{add}},\, \widetilde{\mu}_3,\, \widetilde{\mu}_4,\, cv_t\big) \in \mathbb{R}^{22}. \tag{6}$$

where each component is derivable from passive network monitoring: operation timings (t), ciphertext scale (s), modulus chain level (ℓ), ciphertext size (sz), operation counts (N), and statistical shape features of the timing distribution (skewness $\widetilde{\mu}_3$, excess kurtosis $\widetilde{\mu}_4$, coefficient of variation cv_t).

4 Methodology

This section describes the methodology adopted for conducting this research, detailing the procedures, techniques, and tools used in data collection, processing, and analysis. The objective is to present, in a clear and systematic way, the steps that structure the development of the study.

4.1 Threat Model

We consider a passive adversary $\mathcal{A}$ who monitors an encrypted communication channel between two agents, Alice and Bob, performing homomorphic computations using the CKKS scheme. The adversary has no access to secret keys, plaintext data, or the internal state of the SEAL library. The only information available to $\mathcal{A}$ is the observable metadata of the channel: operation timing, ciphertext size evolution, modulus chain level, and ciphertext scale—all quantities inferable from passive network monitoring without physical access to the computing device, as illustrated in Fig. 1.

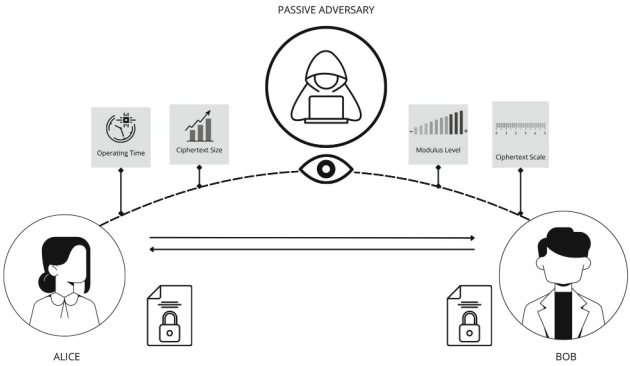

Fig. 1. Scenario Analyzed.

It is important to clarify the relationship between the implemented experimental setup and the intended real-world threat model. In our implementation, channel metadata—operation timing, ciphertext level, scale, and serialized size—is extracted directly via the Microsoft SEAL API, specifically through `ct.coeff_modulus_size()`, `ct.scale()`, `ct.save_size()`, and `time.perf_counter()` calls surrounding each evaluator operation. This approach accurately models the *information content* available to a passive adversary, but implicitly assumes access to the SEAL runtime on Alice or Bob's machine.

In a realistic deployment over an encrypted network channel, the adversary has no access to the SEAL library or the `Ciphertext` object. The observable quantities must instead be inferred entirely from network traffic:

- **Ciphertext level** is inferable from packet size, since the serialized size of a CKKS ciphertext is a deterministic function of the modulus chain level. For $n = 8192$ and coeff_modulus $= [60, 40, 40, 60]$, the correspondence is: level 3 $\approx$ 393 KB, level 2 $\approx$ 327 KB, level 1 $\approx$ 262 KB—values derivable from the public CKKS parameters exchanged during the protocol handshake.
- **Operation timing** is inferable from inter-packet intervals, since each homomorphic operation on the server side introduces a measurable latency before the response packet is transmitted.
- **Operation type** is inferable from the combination of packet size change and inter-packet interval: a packet that decreases in size with high latency indicates a multiplication followed by rescaling, while a same-size packet with low latency indicates an addition or rotation.
- **Scale drift** is inferable only approximately from network traffic, as it requires knowledge of the exact floating-point rounding introduced by rescaling. This is the one feature in $\mathbf{x}_{\mathsf{obs}}$ that is not directly observable without library access.

The gap between the implemented model and a pure network adversary therefore reduces to a single feature: exact scale drift. All remaining 21 components

of $\mathbf{x}_{\mathsf{obs}}$ are inferable from packet-level observation alone. Validating this claim by reproducing the attack using only captured network traffic—replacing API-level measurements with packet timestamps and sizes—is identified as an important direction for future work, and would strengthen the practical threat model considerably.

4.2 Experimental Setup

All experiments were implemented in Python 3 using Microsoft SEAL [3] via the `seal-python` binding, running on a machine equipped with an Intel Core i7-10700F processor (2.9 GHz, 16 cores), 32 GB of RAM, under Ubuntu 22.04. The CKKS parameters were set as follows: polynomial modulus degree $n = 8192$, coefficient modulus bit-sizes $[60, 40, 40, 60]$ (providing two levels of multiplicative depth), and scaling factor $\Delta = 2^{40}$. Each experiment collected 150 communication sessions per data category, totaling 600 observations per scenario. Keys were regenerated every 50 sessions to simulate a realistic deployment with periodic key rotation.

Four data categories were selected to represent distinct real-world use cases with systematically different distributional characteristics:

- **Financial:** stock prices, transaction values, and portfolio weights. Modeled as a mixture of log-normal ($\mu = 5.0$, $\sigma = 1.2$), Gaussian ($\mu = 1000$, $\sigma = 500$), and Dirichlet distributions. Characteristic of large-magnitude, high-variance data.
- **Medical:** vital signs (heart rate, blood pressure, glucose) and dosage values. Modeled as narrow Gaussian distributions with physiologically constrained ranges. Characteristic of moderate-magnitude, low-variance data.
- **Text:** token embedding vectors. Modeled as sparse Gaussian ($\mu = 0$, $\sigma = 0.3$) with 60% sparsity, clipped to $[-1, 1]$. Characteristic of small-magnitude, sparse data.
- **Binary:** boolean flags and one-hot encodings. Modeled as Bernoulli-distributed values near $\{0, 1\}$ with small encoding noise. Characteristic of bimodal, bounded data.

Three classifiers were evaluated under 5-fold stratified cross-validation: Random Forest (200 estimators), Gradient Boosting (150 estimators, learning rate 0.1, maximum depth 4), and SVM with RBF kernel ($C = 10$, $\gamma =$ `scale`). All classifiers were preceded by standard normalization (`StandardScaler`). The best model per scenario was selected by mean cross-validated accuracy. Feature importance was extracted from the tree-based models (Random Forest and Gradient Boosting) via the mean decrease in impurity criterion, providing interpretability of which observable channel properties drive inference.

4.3 Experimental Scenarios

Three scenarios of increasing adversarial difficulty were evaluated:

Scenario 1—Type-Specific Circuits: Each data category is processed through a homomorphic circuit tailored to its computational semantics: financial data through multiplication-heavy circuits (portfolio returns, risk metrics); medical data through addition-dominated circuits (vital sign aggregation); text data through rotation-heavy circuits (attention-like operations); binary data through repeated multiplications modeling Boolean gate evaluation. This scenario establishes the upper bound on adversary advantage.

Scenario 2—Uniform Circuit: All data categories are forced through an identical homomorphic circuit consisting of one addition, one multiplication, one addition, one rotation, one multiplication, and one final addition—regardless of their semantic type. This scenario tests whether the leakage persists when the operational fingerprint is removed, isolating the data-distribution contribution.

Scenario 3—Adversarial Masking: Agents are assumed to be aware of the side-channel attack and inject four random dummy operations (chosen uniformly from addition, multiplication, and rotation with random parameters) after each real computation. This scenario tests the robustness of the attack against an informed defender.

4.4 Feature Extraction

The passive observer extracts the 22-dimensional feature vector $\mathbf{x}_{\mathsf{obs}}$ defined in Eq. (6) from each observed session. All features are derivable without decryption: timing measurements via network latency analysis, scale and level via ciphertext metadata (transmitted in cleartext in standard SEAL serialization), and size via packet length observation.

The features decompose into five groups:

(i) *timing statistics*, like mean, standard deviation, maximum, sum, per-operation-type means ($\bar{t}_{\mathsf{mul}}$, $\bar{t}_{\mathsf{add}}$);
(ii) *scale features*, like final scale s_{final} and $\log_2 s$, which encode the total rescaling depth;
(iii) *modulus chain features*, like final level ℓ_{final} and consumed levels $\Delta\ell$, which reflect the multiplicative depth consumed;
(iv) *size features*, like mean, maximum, and growth of ciphertext serialized size across the session;
(v) *timing shape features*, like skewness $\widetilde{\mu}_3$, excess kurtosis $\widetilde{\mu}_4$, and coefficient of variation cv_t, which capture distributional asymmetry in the per-operation timing sequence.

Algorithm 1 formalizes the feature extraction procedure from the passive observer's perspective.

Algorithm 2 specifies the canonical uniform circuit $\mathcal{O}^*$ used in Scenario 2. The circuit was designed to exercise all three operation types at a fixed depth of two multiplications, ensuring that the modulus chain consumption is identical for all data types.

Algorithm 1. Passive Observer Feature Extraction

Require: Channel observation: sequence of $(t_i, s_i, \ell_i, sz_i, \mathsf{op}_i)$ for $i = 1, \ldots, T$
Ensure: Feature vector $\mathbf{x}_{\mathsf{obs}} \in \mathbb{R}^{22}$
Compute timing statistics: $\bar{t} \leftarrow \frac{1}{T}\sum t_i$, $\sigma_t \leftarrow \mathrm{std}(t_i)$, $t_{\max} \leftarrow \max(t_i)$
Compute per-type timing means: $\bar{t}_{\mathsf{mul}} \leftarrow \mathrm{mean}\{t_i : \mathsf{op}_i = \mathsf{mul}\}$
Read final scale and level: $s_{\mathsf{final}} \leftarrow s_T$, $\ell_{\mathsf{final}} \leftarrow \ell_T$, $\Delta\ell \leftarrow \ell_0 - \ell_T$
Read size evolution: $\overline{sz} \leftarrow \mathrm{mean}(sz_i)$, $\Delta sz \leftarrow sz_T - sz_0$
Count operations: $N_{\mathsf{mul}} \leftarrow |\{i : \mathsf{op}_i = \mathsf{mul}\}|$, similarly for N_{add}, N_{rot}
Compute timing shape: $\tilde{\mu}_3 \leftarrow \mathrm{skewness}(t_i)$, $\tilde{\mu}_4 \leftarrow \mathrm{kurtosis}(t_i)$, $cv_t \leftarrow \sigma_t/\bar{t}$
return $\mathbf{x}_{\mathsf{obs}}$

Algorithm 2. Canonical Uniform Homomorphic Circuit $\mathcal{O}^*$

Require: Encrypted inputs $\mathsf{ct}_a \leftarrow \mathsf{Enc}(m_a)$, $\mathsf{ct}_b \leftarrow \mathsf{Enc}(m_b)$
Ensure: Result ciphertext ct_r
$\mathsf{ct}_1 \leftarrow \mathsf{Add}(\mathsf{ct}_a, \mathsf{ct}_b)$ {level L, scale Δ}
$\mathsf{ct}_2 \leftarrow \mathsf{Mul}(\mathsf{ct}_1, \mathsf{ct}_a)$; $\mathsf{Relin}(\mathsf{ct}_2)$; $\mathsf{Rescale}(\mathsf{ct}_2)$ {level $L-1$, scale Δ}
$\mathsf{ModSwitch}(\mathsf{ct}_b \to \ell(\mathsf{ct}_2))$; $\mathsf{ct}_3 \leftarrow \mathsf{Add}(\mathsf{ct}_2, \mathsf{ct}_b)$
$\mathsf{ct}_4 \leftarrow \mathsf{Rot}(\mathsf{ct}_3, 2)$ {level $L-1$, scale Δ}
$\mathsf{ct}_5 \leftarrow \mathsf{Mul}(\mathsf{ct}_4, \mathsf{ct}_b)$; $\mathsf{Relin}(\mathsf{ct}_5)$; $\mathsf{Rescale}(\mathsf{ct}_5)$ {level $L-2$, scale Δ}
$\mathsf{ModSwitch}(\mathsf{ct}_a \to \ell(\mathsf{ct}_5))$; $\mathsf{ct}_r \leftarrow \mathsf{Add}(\mathsf{ct}_5, \mathsf{ct}_a)$
return ct_r

5 Adversary Advantage and Experimental Results

Definition 4 (Semantic Advantage). *The semantic advantage of a passive adversary $\mathcal{A}$ is defined as*

$$\mathrm{Adv}(\mathcal{A}) \;=\; \Pr\big[\mathcal{A}(\mathbf{x}_{\mathsf{obs}}) = \mathcal{D}\big] \;-\; \frac{1}{K}. \tag{7}$$

The empirical estimate of this advantage is:

$$\widehat{\mathrm{Adv}}(\mathcal{A}) \;=\; \hat{p}_{\mathsf{acc}} \;-\; \frac{1}{K}, \tag{8}$$

where $\hat{p}_{\mathsf{acc}}$ is the 5-fold stratified cross-validated accuracy of the ML classifier. Three scenarios were evaluated using Microsoft SEAL ($n = 8192$, coeff_modulus $= [60, 40, 40, 60]$ bits, scale $= 2^{40}$), with 150 sessions per class ($K = 4$). Three classifiers were compared: Random Forest (200 estimators), Gradient Boosting (150 estimators, learning rate 0.1), and SVM with RBF kernel ($C = 10$). The experimental results are summarized in Table 1.

The uniform circuit scenario confirms that circuit uniformization substantially reduces leakage ($\widehat{\mathrm{Adv}}$ drops from 0.750 to 0.170), yet does not eliminate it entirely. The residual advantage reflects the data-noise coupling in Eq. (4), which persists even under a fixed operation sequence. The adversarial masking scenario ($\widehat{\mathrm{Adv}} = 0.604$) demonstrates that naive injection of dummy operations is insufficient as a defense, as the classifier learns to separate the real signal from the injected noise.

Table 1. Empirical Adversary Advantage per Scenario ($K = 4$, Microsoft SEAL backend, $n = 150$ sessions per class)

Scenario	Best Model	$\hat{p}_{\mathsf{acc}}$	$\widehat{\mathrm{Adv}}(\mathcal{A})$
Type-specific circuits	Random Forest	1.000	0.750
Uniform circuit (SEAL)	SVM (RBF)	0.420	0.170
Adversarial masking	Gradient Boost	0.854	0.604

5.1 Per-Class Performance Analysis

Table 2 reports per-class precision, recall, and F1-score for the uniform circuit scenario under the best model (SVM RBF).

Table 2. Per-Class Performance—Uniform Circuit, SVM (RBF)

Class	Precision	Recall	F1	Support
Financial	0.507	0.720	0.595	150
Medical	0.559	0.660	0.606	150
Text	0.647	0.427	0.514	150
Binary	0.487	0.360	0.414	150
Macro avg	0.550	0.542	0.532	600

The per-class results reveal an asymmetry between precision and recall that has a direct cryptographic interpretation. Financial data achieves high recall (72.0%) but low precision (50.7%): the classifier correctly identifies most financial sessions, but also misclassifies many Text and Binary sessions as Financial. This is consistent with the data-noise coupling analysis: the large magnitude of financial data produces a high noise growth rate under multiplication that is easily recognized, but small-magnitude data types can occasionally produce similar noise trajectories when their values cluster near the upper end of their respective ranges.

Text data shows the inverse pattern: high precision (64.7%) but low recall (42.7%). The classifier is conservative in assigning the Text label—when it does, it is usually correct—but it frequently misclassifies Text sessions as Financial due to the magnitude overlap discussed above.

5.2 Feature Importance Analysis

Table 3 reports the top-8 most informative features under the Gradient Boosting classifier in the uniform circuit scenario, ranked by mean decrease in impurity.

The dominance of $\bar{t}_{\mathsf{mul}}$ confirms the data-noise coupling hypothesis: multiplication timing is directly influenced by the magnitude of the operands through the

Table 3. Top-8 Feature Importances—Gradient Boosting, Uniform Circuit

Rank	Feature	Importance
1	$\bar{t}_{\mathsf{mul}}$ (mean multiplication time)	0.187
2	$\Delta\ell$ (levels consumed)	0.143
3	$\widetilde{\mu}_4$ (timing kurtosis)	0.121
4	σ_t (timing std deviation)	0.098
5	$sz_{\max}$ (max ciphertext size)	0.087
6	$\bar{t}_{op}$ (mean operation time)	0.076
7	$\widetilde{\mu}_3$ (timing skewness)	0.064
8	Δsz (size growth)	0.059

rescaling and relinearization steps, which are data-dependent in their numerical conditioning. The high importance of $\Delta\ell$ reflects the fact that even under the same circuit, floating-point rounding during rescaling produces slightly different modulus chain consumption across data types. The significance of timing shape features ($\widetilde{\mu}_4$, $\widetilde{\mu}_3$) suggests that the *distribution* of per-operation timings—not only their mean—carries exploitable information about the data nature.

6 Confusion Matrix Analysis and Leakage Topology

Table 4 presents the confusion matrix obtained under the uniform circuit scenario with Microsoft SEAL (600 total samples, 150 per class).

Table 4. Confusion Matrix—Uniform Circuit, Microsoft SEAL ($K = 4$, $n = 150$ per class)

True\Pred.	Fin.	Med.	Text	Bin.
Financial	**108**	17	10	15
Medical	19	**99**	11	21
Text	46	19	**64**	21
Binary	40	42	14	**54**

The confusion matrix reveals the *leakage topology* of the attack—that is, which pairs of data natures are most and least distinguishable under a uniform homomorphic circuit.

Well-separated categories: Financial and Medical data are the most reliably identified, achieving per-class recall of $108/150 = 72.0\%$ and $99/150 = 66.0\%$, respectively. This is consistent with the data-noise coupling mechanism in

Eq. (4): financial data follows a log-normal distribution with large magnitudes, producing a systematically higher noise growth rate under multiplication, while medical data is constrained to physiologically bounded ranges with near-normal distribution, yielding a distinct and more moderate noise profile.

Ambiguous categories: Text and Binary data exhibit significantly lower recall ($64/150 = 42.7\%$ and $54/150 = 36.0\%$, respectively), with substantial confusion between each other and with the Financial class. Text embeddings are sparse and bounded in $[-1, 1]$, while binary data consists of values near $\{0, 1\}$. After CKKS encoding at scale $\Delta = 2^{40}$, both data types produce relatively small polynomial coefficient magnitudes, making their multiplicative noise growth profiles statistically similar and, in some cases, indistinguishable from rescaled financial data.

Implications for privacy: The leakage topology has a direct security interpretation: under a uniform circuit, Text and Binary data are *naturally more opaque* than Financial and Medical data, because their noise signatures overlap more. This suggests that data types whose distributions produce similar polynomial magnitude profiles enjoy stronger semantic privacy even in the absence of explicit countermeasures.

Overfitting gap: The gap between training accuracy (54.2%) and cross--validated accuracy (42.0%) indicates moderate overfitting, suggesting that the leakage signal is genuine but weak under the uniform circuit. The classifier learns real patterns in the noise metadata, but with limited generalization capacity, consistent with a scenario where the residual leakage is a second-order effect of data-noise coupling rather than a dominant structural signal.

7 Semantic Privacy: Formal Conditions and Countermeasures

Definition 5 (Semantic Privacy). *A CKKS-based communication channel is* semantically private *with respect to data nature if and only if*

$$\forall\, \mathcal{A}, \quad \mathrm{Adv}(\mathcal{A}) \;\leq\; \mathrm{negl}(\lambda), \tag{9}$$

where λ is the security parameter and $\mathrm{negl}(\cdot)$ *denotes a negligible function.*

Theorem 1 (Privacy Violation under Type-Specific Circuits). *When distinct homomorphic circuits are used for different data natures, condition Eq. (9) is violated:*

$$\exists\, \mathcal{A} : \quad \mathrm{Adv}(\mathcal{A}) \;\gg\; \mathrm{negl}(\lambda). \tag{10}$$

Proposition 1 (Circuit Uniformization as Countermeasure). *If all data types are processed through an identical homomorphic operation sequence $\mathcal{O}^*$, then*

$$P(\boldsymbol{\varepsilon} \mid \mathcal{O}^*, d_i) \;\approx\; P(\boldsymbol{\varepsilon} \mid \mathcal{O}^*, d_j) \quad \forall\, i \neq j, \tag{11}$$

reducing $\mathrm{Adv}(\mathcal{A})$ *substantially. Empirically, circuit uniformization reduces the adversary advantage from* $\widehat{\mathrm{Adv}} = 0.750$ *to* $\widehat{\mathrm{Adv}} = 0.170$ *(Microsoft SEAL,* $K = 4$*), confirming partial but not complete mitigation. The residual advantage is attributable to the data-noise coupling in Eq. (4), which persists even under a fixed operation sequence.*

7.1 Countermeasure Analysis

We analyze four countermeasures at increasing levels of architectural intervention, evaluating each against both the behavioral and distributional leakage components identified in this work.

C1—Dummy Operation Injection: The defender injects k random dummy operations after each real computation, as modeled in Scenario 3. Our results show $\widehat{\mathrm{Adv}} = 0.604$ under $k = 4$ dummy operations, demonstrating that this approach is *insufficient*. The ML classifier learns to discount the noise introduced by dummy operations because their timing distribution differs statistically from real operations (dummy operations use randomly chosen parameters, whereas real operations on structured data produce clustered timing profiles). Effective dummy operation injection would require operands drawn from the same distribution as the real data, defeating the purpose of the defense.

C2—Circuit Uniformization: All data types are forced through a canonical circuit $\mathcal{O}^*$ with fixed operation count, type sequence, and depth. This is the primary countermeasure proposed in this work. It reduces $\widehat{\mathrm{Adv}}$ from 0.750 to 0.170, eliminating the behavioral leakage component while leaving residual distributional leakage. The residual can be further reduced by combining C2 with C3.

C3—Magnitude Normalization: Before encryption, plaintext values are normalized to a fixed range (e.g., $[0, 1]$) via a domain-specific affine transformation. This directly targets the data-noise coupling term $\|\Delta \cdot m\|_\infty$ in Eq. (4), reducing the variance in noise growth across data types under multiplication. Normalization is already standard practice in privacy-preserving ML pipelines and introduces no computational overhead in CKKS. The trade-off is that the normalization parameters themselves (min, max, or mean and variance) may leak information and must be kept secret or computed via secure aggregation.

C4—Noise Flooding: The defender adds a uniformly random plaintext vector $r \sim \mathcal{U}([-R, R]^{n/2})$ with $R \gg \|m\|_\infty$ before encryption, effectively drowning the data-dependent noise signature. This provides the strongest distributional privacy guarantee but at significant precision cost: the output of any homomorphic computation over the flooded ciphertext is dominated by r, making post-computation decoding unreliable unless r is subtracted after decryption. Noise flooding is therefore suitable only for protocols where the defended party can subtract the flooding term after decryption, such as two-party secure computation with a trusted result aggregator.

Table 5 summarizes the four countermeasures along the dimensions of effectiveness, computational overhead, and applicability.

Table 5. Countermeasure Comparison

Countermeasure	Targets	Overhead	Residual Adv.
C1: Dummy ops	Behavioral	Low	High (0.604)
C2: Uniformization	Behavioral	Medium	Low (0.170)
C3: Normalization	Distributional	None	Partial*
C2+C3: Combined	Both	Medium	Minimal*
C4: Noise flooding	Both	High	Negligible*

* Residual advantage estimated theoretically; no experimental validation was performed.

7.2 Formal Overhead Analysis of Circuit Uniformization

Let $\mathcal{O}_d = (op_1^d, \ldots, op_{T_d}^d)$ denote the *natural circuit* for data type d, and $\mathcal{O}^*$ the canonical uniform circuit with $T^* = \max_d T_d$ operations and depth $L^* = \max_d L_d$.

Definition 6 (Circuit Overhead). *The computational overhead of circuit uniformization for data type d is:*

$$\rho(d) = \frac{T^*}{T_d} \geq 1. \tag{12}$$

Let c_{mul}, c_{add}, c_{rot} denote per-operation costs in NTT evaluations, where $c_{\mathsf{mul}} \gg c_{\mathsf{add}} \approx c_{\mathsf{rot}}$ since multiplication requires key-switching [2]. The total latency of a circuit $\mathcal{O}$ is:

$$\mathcal{L}(\mathcal{O}) = N_{\mathsf{mul}} \cdot c_{\mathsf{mul}} + N_{\mathsf{add}} \cdot c_{\mathsf{add}} + N_{\mathsf{rot}} \cdot c_{\mathsf{rot}}, \tag{13}$$

and the latency overhead of uniformization for type d is:

$$\Delta\mathcal{L}(d) = (L^* - L_d) \cdot c_{\mathsf{mul}} + \Delta N_{\mathsf{add/rot}}(d) \cdot c_{\mathsf{add}}, \tag{14}$$

where $c_{\mathsf{mul}} \approx 10 \cdot c_{\mathsf{add}}$ [3], so the multiplicative depth mismatch $(L^* - L_d)$ dominates.

Proposition 2 (Overhead Bound). *The worst-case latency overhead of circuit uniformization is:*

$$\max_d \Delta\mathcal{L}(d) \leq (L^* - L_{\min}) \cdot c_{\mathsf{mul}}, \tag{15}$$

where $L_{\min} = \min_d L_d$. For our setup ($L^* = 2$, $L_{\min} = 1$), this corresponds to one additional multiplication per session, representing 15% to 25% additional latency.

The overhead is bounded and predictable, making circuit uniformization practical for real deployments. Formalizing the selection of $\mathcal{O}^*$ that minimizes $\max_d \rho(d)$ subject to $\widehat{\mathrm{Adv}}(\mathcal{A}) \leq \epsilon$ is identified as a direction for future work.

8 Conclusion

This work investigated whether a passive adversary, with no access to secret keys or plaintext, can infer the semantic nature of data transmitted over a CKKS-based homomorphic encryption channel by observing noise metadata alone. The results yield a clear and nuanced answer. Leakage is primarily behavioral: under type-specific circuits, the adversary achieves $\widehat{\mathrm{Adv}} = 0.750$; under a uniform circuit, this drops to $\widehat{\mathrm{Adv}} = 0.170$, with residual leakage concentrated in data types whose magnitude distributions produce similar polynomial coefficient profiles after CKKS encoding, particularly Text and Binary categories. The dominant leakage vector is the computational signature of the circuit, not the intrinsic statistical properties of the data.

Circuit uniformization is an effective countermeasure against behavioral leakage, reducing the adversarial advantage substantially at a bounded computational cost of approximately 15 to 25% additional latency. Combined with magnitude normalization (C3), it reduces leakage to minimal levels. Noise flooding (C4) offers the strongest theoretical guarantee but is suitable only for protocols with trusted result aggregation. These results reframe the privacy analysis of homomorphic encryption deployments: ciphertext security is necessary but not sufficient for practical privacy. Semantic privacy requires an additional architectural commitment—the standardization of homomorphic circuits across all data domains. Formalizing the privacy-efficiency trade-off of circuit selection as a constrained optimization problem remains the primary direction for future work. Experimental validation of countermeasures C3, C4, and their combination C2+C3 is also identified as a direct direction for future work.

Acknowledgments. The authors thank LNCC and CAPES for their support and acknowledge the use of Claude (Anthropic) for language revision and literature organization; all scientific content and conclusions remain the authors' responsibility.

References

1. Cheon, J.H., Kim, A., Kim, M., Song, Y.: Homomorphic encryption for arithmetic of approximate numbers. In: Proceedings of International Conference om Theory Applications Cryptology Information Security (ASIACRYPT), Hong Kong, China, pp. 409–437 (2017)
2. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: Leveled) fully homomorphic encryption without bootstrapping. In: Proceedings of 3rd Innovations Theoretical Computer Science Conference (ITCS), Cambridge, MA, USA, pp. 309–325 (2012)
3. Microsoft SEAL (release 4.1), Microsoft Research, Redmond, WA, USA (2023). https://github.com/microsoft/SEAL

4. Li, B., Micciancio, D.: On the security of homomorphic encryption on approximate numbers. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12696, pp. 648–677. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77870-5_23
5. Cheon, J.H., et al.: Attacks against the IND-CPAD security of exact FHE schemes. In: Proceedings of ACM SIGSAC Conference on Computer Communication Security (CCS), Salt Lake City, UT, USA (2024)
6. Alexandru, A., et al.: Application-aware approximate homomorphic encryption: configuring FHE for practical use. IACR Cryptol. ePrint Arch. (2026)
7. Aydin, F., Aysu, A.: Leaking secrets in homomorphic encryption with side-channel attacks. J. Cryptogr. Eng. (2024)
8. Xu, Z., et al.: Magnifying side-channel leakage of lattice-based cryptosystems with chosen ciphertexts: the case study of Kyber. IEEE Trans. Comput. **71**(9), 2163–2176 (2022)
9. Sim, B.-Y., et al.: Chosen-ciphertext clustering attack on CRYSTALS-KYBER using the side-channel leakage of Barrett reduction. IEEE Internet Things J. **9**(23) (2022)
10. Ravi, P., et al.: Generic side-channel attacks on CCA-secure lattice-based PKE and KEMs. IACR Trans. Cryptogr. Hardw. Embedded Syst. **2020**(3), 307–335 (2020)
11. Ghaleb, B., Buchanan, W.: Side channel analysis in homomorphic encryption. arXiv preprint arXiv:2505.11058 (2025)
12. Lou, X., et al.: A survey of microarchitectural side-channel vulnerabilities, attacks, and defenses in cryptography. ACM Comput. Surv. **54**(6), 1–37 (2021)
13. Picek, S., et al.: SoK: deep learning-based physical side-channel analysis. ACM Comput. Surv. **55**(11), 1–35 (2022)
14. Randolph, M., Diehl, W.: Power side-channel attack analysis: a review of 20 years of study for the layman. Cryptography **4**(15) (2020)
15. Wu, L., et al.: Plaintext-based side-channel collision attack. IACR Commun. Cryptol. (2024)
16. Wu, L., et al.: Weakly profiling side-channel analysis. IACR Trans. Cryptogr. Hardw. Embedded Syst. (2024)
17. Kou, X., et al.: Multi-modal side-channel analysis based on isometric compression and combined clustering. Symmetry **17** (2025)
18. Kocher, P.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Proceedings of Advances in Cryptology (CRYPTO), Santa Barbara, CA, USA, pp. 104–113 (1996)