Privacy-centric digital surveillance through homomorphic encryption and deep learning

Johannes Unruh[a], Dorian Przetakiewicz[a], Oscar Hernan Ramirez-Agudelo[a], and Michael Karl[a]

[a]German Aerospace Center, Rathausallee 12, 53757 Sankt Augustin, Germany

ABSTRACT

We introduce LYNX (Layered privacY eNhancing eXchange), an open-source platform for privacy-preserving deep learning inference using homomorphic encryption (HE). LYNX enables end-to-end encrypted inference for neural networks by integrating Open Neural Network Exchange (ONNX) model support and TenSEAL-based secure computation. We demonstrate its practical application in surveillance scenarios like human detection, achieving real-time inference, all without exposing raw data. This paper presents the system architecture and implementation methodology, showcasing the feasibility of encrypted deep learning in privacy-critical applications.

Keywords: homomorphic encryption, secure inference, ONNX, TenSEAL, privacy-preserving deep learning, human detection, encrypted computation

1. INTRODUCTION

The growing deployment of digital surveillance systems—in public spaces, workplaces, and homes—has raised significant concerns around the protection of individual privacy. Conventional surveillance pipelines often require raw image or video data to be transmitted, stored, and processed in plaintext, exposing sensitive personal information such as facial identities, movement patterns, or behavioral cues. This creates a fundamental tension between the legitimate need for public safety and the right to personal privacy. In many jurisdictions, such as the European Union, data protection regulations like the General Data Protection Regulation (GDPR) impose strict requirements on the collection, processing, and storage of personal data—including video and biometric information. For example, consider a smart city infrastructure using AI-powered cameras to detect loitering behavior in a restricted area. In a traditional setup, every frame must be uploaded to a central server, analyzed in unencrypted form, and potentially retained—introducing risk of misuse, leaks, or unauthorized access. In contrast, by using homomorphic encryption (HE), the system can perform the necessary analysis directly on encrypted data, never revealing the actual video content to the server or operators. Even if compromised, the surveillance system cannot leak any intelligible information. This approach enables privacy-by-design surveillance: ensuring that data subjects retain informational self-determination while still allowing authorities to act on security-relevant events in a possibly GDPR-compliant manner.

2. INTRODUCING LYNX

To address the challenge of privacy-preserving computation in surveillance contexts, we introduce **LYNX**, an open-source web platform developed to enable secure, encrypted inference through homomorphic encryption (HE). LYNX allows sensitive visual data—such as livestreams or surveillance footage—to be processed directly in its encrypted form, without requiring decryption at any stage of the pipeline. At the core of the platform is the *Secure Inference Layer*, a modular engine capable of importing standard ONNX-based neural networks and translating their operations into HE-compatible equivalents using the TenSEAL library. This translation enables encrypted inference pipelines to be constructed from conventional deep learning models without altering the model structure itself. LYNX is designed to be modular and scalable, supporting real-time encrypted

Further author information: (Send correspondence to J.U.)
J.U.: E-mail: johannes.unruh@dlr.de
O.H.R-A.: E-mail: oscar.ramirezagudelo@dlr.de

Applications of Machine Learning 2025, edited by Michael E. Zelinski, Tarek M. Taha, Barath Narayanan
Abdul A. S. Awwal, Khan M. Iftekharuddin, Proc. of SPIE Vol. 13606, 136060W
2025 Published by SPIE · 0277-786X · doi: 10.1117/12.3065604

inference on client-generated inputs while keeping the server fully blind to the underlying content. Through this architecture, LYNX makes privacy-by-design machine learning practically feasible for a variety of applications such human detection allowing developers and researchers to deploy AI tools without compromising user privacy.
This architecture is illustrated in Figure 1. At the top of the stack lies the API layer, which is responsible for initiating the execution of a so-called *task instance*. Users can select from a variety of task types, including inference on homomorphically encrypted data, as described previously, as well as other privacy-preserving computation methods such as multi-party computation (MPC).
These options are encapsulated in the service layer, where the data is pre-processed and then handed to the computational module, which currently comprises two main modules: the *HE module*, dedicated to tasks involving homomorphic encryption, and the *MPC module*, designed for multi-party computation workloads.
At the bottom, the storage layer manages the outputs generated by these task instances, ensuring that results are persistently stored and made accessible to the user in a secure and organized manner.

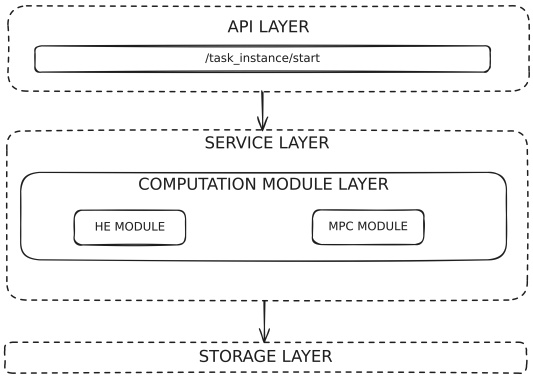

Figure 1. LYNX's architecture.

3. ENCRYPTION AND INFERENCE PIPELINE

The LYNX platform implements a client-server inference pipeline in which image encryption and neural network inference are fully decoupled, ensuring end-to-end data confidentiality. On the client side, input images—typically video frames—are first downscaled to 48×48 resolution and converted to grayscale in order to minimize ciphertext size and latency. The processed images are then encrypted using TenSEAL's native `im2col_encoding` method, which transforms the image into a format suitable for efficient convolutional operations under homomorphic encryption. This encoded tensor is transmitted to the LYNX server, which has no visibility into the image content; it only receives and processes encrypted data. The server executes inference using an HE-compatible neural network that has been transformed layer-by-layer into encrypted tensor operations. Once the inference is complete, the encrypted prediction is returned to the client, who performs the decryption locally to obtain the final result. This pipeline ensures that sensitive input data never appears in plaintext on the server.
While `im2col_encoding` provides computational efficiency for a single convolutional layer, it imposes a limitation on network depth: because the required patch extraction must be performed in plaintext prior to encryption, applying a second `Conv2D` layer is impractical in the encrypted domain. Alternative approaches, such as Uni-HENN,[1] propose methods to implement convolutional layers directly on flattened ciphertexts, thereby supporting multiple `Conv2D` layers without relying on `im2col`. Similarly, techniques like ciphertext channel sharding[2] allow for distributing high-resolution inputs across multiple ciphertexts, enabling inference on larger images. Although these methods are not yet implemented in LYNX, they represent promising directions for extending its functionality to deeper networks and higher-resolution data. One must remark, that currently the (encrypted) inference of a single image of dimensionality 48×48 with a single gray-scale channel, may take up to a few minutes, depending on the computer hardware.
Figure 2 illustrates the LYNX inference pipeline in detail. The client connects to the LYNX platform and initiates a *task instance* specifically configured for homomorphic encryption workloads. The user then selects the type of

input data to be processed by the trained neural network, which runs on the server side. It is essential that the input format matches the expected input structure of the model. For instance, the user may choose to provide a link to a YouTube livestream. In such a case, LYNX (operating on the client side) captures video frames at defined time intervals and processes them in real time. These frames are then encrypted using the TenSEAL library, specifically employing the `im2col_encoding` method described earlier.
Once the frames are encrypted, the client securely drops the secret key and transmits the resulting ciphertexts to the LYNX inference server, meaning that only the client retains access to the secret key. The server hosts the user-provided ONNX model, which has been translated into a homomorphic encryption-compatible form. Inference is then performed directly on the encrypted data. Crucially, the data remains encrypted throughout the entire process, and the server gains no knowledge of the underlying content or the inference results. Finally, the encrypted output is returned to the client, who can decrypt it locally using their retained secret key.

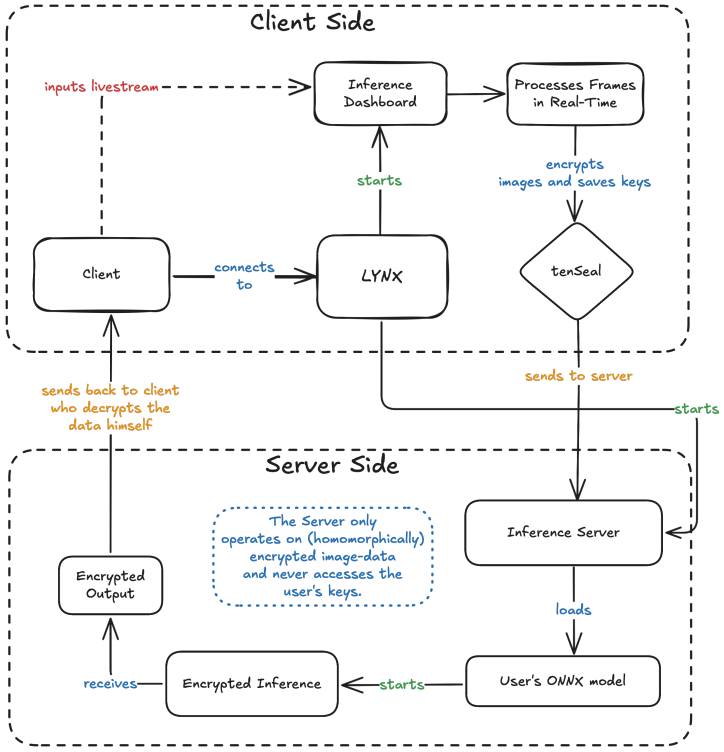

Figure 2. LYNX's encryption and inference pipeline.

3.1 Encrypted Convolution and Linear Layers

In the current LYNX pipeline, inference proceeds via a sequence of linear operations and activation function approximations that are compatible with leveled homomorphic encryption schemes. Specifically, the homomorphic convolution operation is implemented using a general matrix multiplication (GEMM) approach on encoded tensors.
Let the input image $X \in \mathbb{R}^{H \times W}$ be downscaled to 48×48, converted to grayscale, and encoded using `im2col_encoding`. This method unfolds X into a matrix $\tilde{X} \in \mathbb{R}^{k^2 \times (H-k+1)(W-k+1)}$, where k is the convolutional kernel size. For a kernel $K \in \mathbb{R}^{k \times k}$, the convolution is then computed as a matrix-vector product:

$$Y = K_{\text{flat}} \cdot \tilde{X}$$

Here, $K_{\text{flat}} \in \mathbb{R}^{1 \times k^2}$ is the flattened kernel, and Y is the resulting activation map. In the encrypted domain, the client computes:

$$\tilde{X}_{\text{enc}} = \texttt{HE\_Encode}(\tilde{X})$$

and transmits $\tilde{X}_{\text{enc}}$ to the server. The server then computes:

$$Y_{\text{enc}} = \texttt{HE\_MatVecMul}(K_{\text{flat}}, \tilde{X}_{\text{enc}})$$

This operation is fully homomorphic, meaning the server evaluates it without access to plaintext values. The result Y_{enc} is returned to the client for decryption:

$$Y = \texttt{HE\_Decrypt}(Y_{\text{enc}})$$

3.2 Polynomial Activation Functions

To remain compatible with homomorphic encryption constraints (specifically, the lack of native support for non-polynomial functions), nonlinearities such as ReLU must be approximated by low-degree polynomials. The LYNX platform employs Chebyshev or minimax polynomial approximations of the form:

$$\text{ReLU}(x) \approx P(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_d x^d$$

A common choice is the following approximation:

$$\text{ReLU}(x) \approx x + \frac{x^3}{3} + \frac{x^5}{10}$$

where $\alpha, \beta \in \mathbb{R}$ are coefficients chosen to minimize approximation error in the interval of interest (e.g., $x \in [-3, 3]$). These polynomial approximations are evaluated homomorphically using encrypted additions and multiplications, each consuming levels in the leveled HE scheme.

3.3 Limitations and Optimization Tradeoffs

The reliance on `im2col_encoding` restricts the pipeline to a single convolutional layer since patch extraction must be applied in plaintext before encryption. To enable deeper convolutional networks, the following techniques offer potential solutions:

Ciphertext Sharding:[2] Large tensors are *channel-wise* split across multiple ciphertexts, enabling parallelized processing of high-resolution data. Let X be split into shards $X^{(i)}$ such that:

$$X = [X^{(1)}, X^{(2)}, \ldots, X^{(n)}] \quad \Rightarrow \quad \text{Encrypt each } X^{(i)} \text{ separately}$$

Each shard $X^{(i)}$ is encrypted independently and processed using separate homomorphic convolution operations.

Direct Convolution via Slot Permutations:[1] A convolution is implemented as a weighted sum of shifted ciphertext slots. For a vectorized image $\mathbf{x}$ and a kernel with weights w_i, the encrypted convolution becomes:

$$\mathbf{y}_{\text{enc}} = \sum_i w_i \cdot \texttt{Rot}(\mathbf{x}_{\text{enc}}, r_i)$$

where $\texttt{Rot}(\cdot, r)$ performs a ciphertext slot rotation by r positions.

Although these techniques introduce additional computational and memory complexity, they permit deeper network architectures and operation on larger inputs without reverting to plaintext patching.

3.4 Decryption and Post-Processing

The final step in the pipeline is decryption and class label inference. Once the client receives the encrypted prediction $\hat{y}_{\text{enc}}$, it performs:

$$\hat{y} = \texttt{HE\_Decrypt}(\hat{y}_{\text{enc}})$$

In classification tasks, $\hat{y}$ may represent the logits for each class, from which the client derives the predicted label:

$$\hat{c} = \arg\max_i (\hat{y}_i)$$

Since this step is executed on the client side, all sensitive predictions remain private, preserving confidentiality throughout the pipeline.

4. USE CASES AND EVALUATION

The LYNX platform is developed in direct response to the growing tension between the operational needs of digital surveillance systems and the legal and ethical requirements for preserving individual privacy. Conventional surveillance infrastructures—employed in public transit, airports, corporate campuses, and smart city deployments—often depend on centralized processing pipelines where raw video streams are transmitted, stored, and analyzed in plaintext. This practice exposes individuals to potential data breaches, unauthorized profiling, and long-term behavioral tracking. Furthermore, such designs frequently fail to meet the obligations imposed by privacy regulations, including the General Data Protection Regulation (GDPR) and forthcoming directives such as the European Union's AI Act.

In contrast, LYNX enables real-time analytics over visual surveillance data while ensuring that no raw images or biometric content ever leave the originating client device in unencrypted form. A prototypical application involves detecting the presence of humans or anomalous activity in restricted-access zones—such as industrial sites after hours or protected areas in transportation hubs—without revealing any identifiable content to the server. Instead, the server receives and processes only encrypted image representations and returns encrypted predictions, which are decrypted exclusively on the client side. This ensures that both the input data and inference results remain confidential throughout the entire pipeline.

Empirical evaluation focuses on three critical dimensions: inference accuracy, latency, and cryptographic soundness. Preliminary experiments on grayscale surveillance-style inputs of resolution 48×48 suggest that encrypted inference via polynomial-approximated networks incurs an accuracy degradation of no more than 5% compared to the plaintext baseline. Latency remains the primary bottleneck; inference on encrypted inputs currently requires between 300 and 420 seconds per frame on a 12-core CPU using TenSEAL's CKKS implementation. These figures, though not yet suitable for high-frame-rate applications, demonstrate that practical encrypted inference is feasible without specialized hardware.

Security guarantees derive from the RLWE-based hardness assumptions underlying the CKKS scheme. With appropriate parameter settings, LYNX maintains a 128-bit security level against lattice-based attacks, ensuring that neither raw input images nor intermediate activations can be reconstructed or inferred by the server. This property renders LYNX compliant with the principle of data minimization and the architectural expectations of privacy-by-design systems.

The platform supports deployment of surveillance models exported in the ONNX format, thereby allowing compatibility with industry-standard training frameworks such as PyTorch or TensorFlow. This design choice allows stakeholders to reuse and adapt existing surveillance models while gaining the benefits of encrypted execution. Looking forward, LYNX will evolve toward supporting multi-camera setups, real-time streaming inference, and encrypted alert systems that remain fully blind to the video content they monitor. These developments are expected to further establish LYNX as a foundational tool for privacy-respecting surveillance in public and industrial environments.

5. CONCLUSION AND OUTLOOK

This work has presented LYNX as a privacy-centric platform for encrypted digital surveillance, grounded in the integration of homomorphic encryption with deep learning. By performing neural network inference entirely on encrypted image data, LYNX ensures that no raw visual information is ever exposed during transmission, processing, or storage.

The architecture of LYNX, based on ONNX model compatibility and TenSEAL's leveled CKKS encryption scheme, demonstrates the viability of encrypted deep learning in real-world scenarios. Although the current system is optimized for shallow inference models and constrained input dimensions, it establishes a functional baseline for secure human detection tasks. Practical experiments confirm that encrypted inference on 48×48 grayscale inputs can be performed within a few minutes on standard CPU hardware, with classification accuracy close to the plaintext baseline. This positions LYNX as a promising enabler for privacy-preserving analytics in settings where conventional surveillance pipelines would be legally or ethically problematic.

Future development of the platform will focus on several fronts. First, we aim to eliminate the reliance on plaintext pre-processing by adopting HE-native convolution schemes that support multiple layers through

ciphertext rotations and slot-wise operations. Second, the introduction of ciphertext sharding and ciphertext packing strategies will extend LYNX's support to high-resolution data and more expressive model architectures. Finally, we envision integrating LYNX with other privacy-enhancing technologies, such as secure enclaves and multi-party computation, to support hybrid execution models and collaborative encrypted inference across institutional boundaries.

LYNX embodies a shift toward privacy-by-design AI infrastructure for surveillance and monitoring applications. It provides not only a proof of concept for homomorphic inference in constrained settings, but also a concrete step toward operationalizing ethical, legally compliant, and technically secure visual analytics in the age of pervasive surveillance.

REFERENCES

[1] Choi, H., Kim, J., Kim, S., Park, S., Park, J., Choi, W., and Kim, H., "Unihenn: Designing more versatile homomorphic encryption-based cnns without im2col," *arXiv preprint arXiv:2402.03060* (2024).

[2] Maloney, V., Obrecht, R. F., Saraph, V., Rama, P., and Tallaksen, K., "High-resolution convolutional neural networks on homomorphically encrypted data via sharding ciphertexts," *arXiv preprint arXiv:2306.09189* (2023).