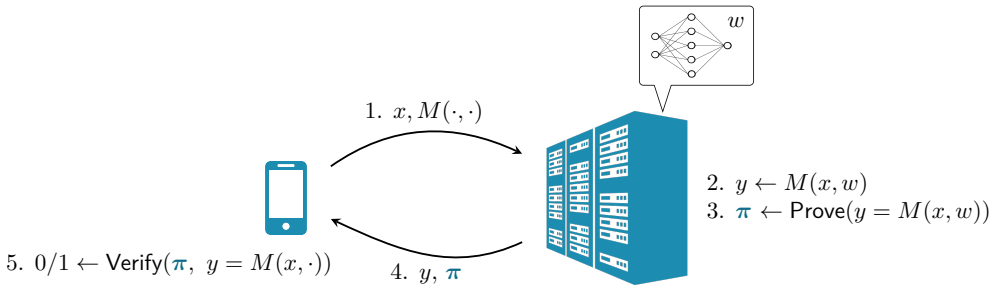
zkML: Provable ML inference through Zero-Knowledge Proofs



Machine learning (ML) models are increasingly used for a variety of applications, but without guarantees about fairness or model quality. Consider a client who wants to evaluate a model M on an input x, but lacks the computational resources, or knowledge of the model's weights, and therefore delegates the inference to a server. In an ordinary deployment, the client must trust that the server ran the expected model, used the provided input, and executed every operation faithfully, yielding $y = M(x, w)$. Recomputing the inference on the client's side would defeat the purpose of delegation.

Verifiable computation (VC) addresses this problem by requiring the server to produce, together with y, a cryptographic proof π that the claimed computation was performed correctly. The client must be able to verify π in a much less expensive way than recomputing the inference. Zero-knowledge proofs (ZKPs) can additionally guarantee that the client learns nothing about the server's private model weights w. The application of these techniques to ML is commonly called *zero-knowledge machine learning* (**zkML**). A typical goal is to prove the following statement with respect to a user input x, server output y, and public commitment c_w:

$$\exists\, w: \quad M(x,w) = y \quad \wedge \quad \mathsf{Commit}(w) = c_w$$

Constructing practical zkML systems is challenging. Neural-network inference contains large linear-algebra operations, non-linear functions, floating-point arithmetic, and memory-intensive execution. These operations must be represented through arithmetic constraints, lookup arguments, sumcheck protocols, or related proof-system machinery. Quantization is a necessary step for proving, but introduces questions about scaling, rounding, clipping, and overflow.

The objective of this thesis is to investigate one or more of the following directions:

1. analyse or improve a component of a zkML pipeline, such as non-linear operators, quantization, commitments, recursion, folding, or memory usage;
2. compare alternative proving strategies or arithmetizations for similar models, precision, and security assumptions;
3. implement a verifiable-inference prototype for a selected model or technique with reusable benchmarks to clearly state practical scalability;

Practicalities

Promotor:	Frederik Vercauteren		
Daily supervisor:	Valerio Ardizio	valerio.ardizio@esat.kuleuven.be	office B01.29
	Jannik Spiessens	jannik.spiessens@esat.kuleuven.be	office B01.15

Nature of the work:	40% literature, 20% theoretical work, 40% software
Number of students:	1