Medical Privacy-Preservation Using Zero-Knowledge Machine Learning

Richard Huynh

Department of Engineering, Cal State University Long Beach

ABSTRACT

Privacy is a critical concern when handling sensitive data in machine learning, particularly in healthcare, where regulations such as the Health Insurance Portability and Accountability Act (HIPAA) and the General Data Protection Regulation (GDPR) govern data access and usage. Traditional machine learning approaches rely on large amounts of data for accurate predictions, increasing the risk of exposing both training and prediction data. Zero-Knowledge Machine Learning (ZKML) offers a promising solution to preserving medical data privacy through private inferences while maintaining compliance with existing regulations.

In this research, we explore the use of EZKL, a python library that aids in generating ZKP circuits, to transform a logistic regression model into a zero-knowledge succinct non-interactive argument of knowledge (z-SNARK) circuit. This model predicts disease risk based on patient data, ensuring that inferences can be verified without revealing sensitive information. Additionally, we integrate TenSEAL to encrypt test input data, showing that encrypted inputs yield the same inference results when decrypted, preserving accuracy while ensuring privacy. The results indicate that, while ZKML provides strong privacy guarantees on inferences, there are weaknesses from its training method to overall computational trade-offs. Our findings demonstrate the feasibility of ZKML for secure medical AI applications but also highlight key limitations in our approach that need to be addressed before it can be fully integrated into real-world applications.

1. INTRODUCTION

Medical data is among the most sensitive types of personal information as it contains details about a patient's health history, diagnoses, and treatments. Not only would exposure of such data would violate several regulations, but it would also lead to severe consequences such as insurance discrimination and the breach of patient confidentiality. To address these risks, governments and organizations have established strict regulations such as the Health Insurance Portability and Accountability Act (HIPAA) in the United States and GDPR in the European Union, which enforce strict data access controls and privacy safeguards. Security and Privacy of Health Records: Concerns and Challenges is a review that presents concerns about people's medical data shifting from paper records to electronic medical records [5]. When asked about their privacy on personal health, only 39% felt that it was safe and secure while a few neither worried nor had faith in the security of their data. Nonetheless, the shift to electronic medical records was met with several security and privacy features that ensured confidentiality, integrity, and accountability in handling sensitive medical data. By moving medical data into machine learning, new security and privacy features need to be developed to ensure the same benefits.

Machine learning in its own right has revolutionized several industries, transforming the way we interact with technology. Machine learning models analyze vast amounts of data, uncovering patterns that may be difficult for humans to detect. Acting as a second set of eyes, it could be leveraged in the digital healthcare industry to improve medical diagnosis and patient outcomes. However, traditional machine learning models would require access to large volumes of sensitive patient data, leading to concerns about privacy and compliance with regulations in which these models pose a risk to patient confidentiality. Without protective features, attackers can exploit vulnerabilities through membership inference or model inversion attacks, potentially reconstructing sensitive medical information from trained models. These challenges address the requirement for privacy-preserving machine learning (PPML) techniques that enable secure computations while protecting patient confidentiality. Similarly to the shift to electronic medical records [5], the use of machine learning models on medical data will require the implementation of security features that guarantee privacy. To create and maintain trust within health care, machine learning models must ensure that the data they use for training and inferences remain private.

Zero-Knowledge Machine Learning (ZKML) offers a promising solution by allowing model predictions to be verified without revealing the input data or the model's parameters. In this paper, we demonstrate the process of taking a machine learning model and converting it to a Zero-Knowledge circuit. With the ZK circuit, we can generate proofs given an input and verify both in house or on the chain using an Ethereum Virtual Machine (EVM) for smart contracts. Throughout this method, we show that through Zero-Knowledge Proofs (ZKP), the proofs leak no information about the input data or from the model.

1.1 Related Work

The paper "A Secure Framework for Privacy-Preserving Analytics in Healthcare Records Using Zero-Knowledge Proofs and Blockchain in Multi-Tenant Cloud Environments" presents a novel approach to securing sensitive healthcare data in cloud-based multi-tenant environments [2]. As healthcare institutions increasingly rely on cloud computing for data storage and analytics, concerns over unauthorized access, data leakage, and regulatory compliance have become significant challenges. The study aims to address these issues by leveraging cryptographic techniques to enable secure and privacy-preserving analytics.

To achieve this, the authors propose a framework that integrates Zero-knowledge proofs (ZKPs) and blockchain technology. ZKPs allow verification of computations without exposing the underlying data, ensuring that healthcare analytics can be performed securely. Meanwhile, blockchain technology is incorporated to maintain data integrity, auditability, and access control through a decentralized and tamper-resistant ledger. By combining these techniques, the framework ensures that sensitive patient data remains protected, even in a shared cloud environment.

The paper discusses various use cases where this framework can be applied, including privacy-preserving predictive analytics for disease detection. By enabling secure machine learning on encrypted medical datasets, healthcare institutions can collaborate on research and data-driven decision-making without compromising patient privacy. The authors also evaluate the framework's performance in a simulated cloud environment, demonstrating that the approach incurs minimal computational and storage overhead while maintaining strong security guarantees.

This research is particularly relevant to our study, as we focus on zero-knowledge machine learning (ZKML) for medical data privacy. While the discussed paper primarily explores the use of multi-tenant cloud environments and blockchain auditing, our approach leverages zero-knowledge proofs and attempts at homomorphic encryption to achieve privacy-preserving inference and model verification on medical diagnostics. By integrating these cryptographic techniques with EZKL, our study builds upon existing frameworks but aims to achieve stronger privacy guarantees with minimal computational overhead.

2. BACKGROUND

This section covers background knowledge on terminology discussed throughout the paper. This includes information on machine learning, zero-knowledge proofs, and the combination of the two, zero-knowledge machine learning.

2.1 Machine Learning

Machine learning is a subset of artificial intelligence that aims to learn patterns from data and make decisions without being explicitly programmed. ML models are trained using mathematical techniques to generalize patterns and make accurate predictions on unseen data. A typical model consists of 3 types of layers:

- Input Layer - Receives raw data
- Hidden Layers - Process and extract meaningful features
- Output Layer - Produces final prediction

The complexity of a model depends on the number of hidden layers, with deeper models having more layers to capture intricate patterns. During training, data flows through these layers, where the nodes apply mathematical transformations to extract relevant features. These are typically activation functions that make a model nonlinear to help with learning complex relationships in the data. Each node is associated with weights, which adjust during the training to improve accuracy. To improve the model performance, a loss function evaluates how well the model's predictions are compared to the actual values. Through an optimization process like gradient descent, the model adjusts its weights iteratively to minimize the error. However, the goal is to not have the model memorize the data and underperform on unseen data.

Machine learning models can be designed differently depending on the type of problem it is trying to solve. Some examples are:

- Classification - Identifying and assigning data to predefined categories
- Regression - Predicting continuous values
- Pattern Recognition - Identifying trends and structures in data

In our research, we use logistic regression for binary classification of heart disease risk. Logistic regression is an extension of linear regression but applies a sigmoid activation function to transform predictions into probabilities. Linear regression is an ML model that establishes a linear relationship between independent variables in the data [8]. The goal of the model is to minimize the error between the predicted and actual value. The prediction for logistic regression is given by the equation $\hat{y} = \frac{1}{1+e^{-(\theta_0+\sum_{i=1}^{n}\theta_i x_i}}$ where $\hat{y}$ is the predicted value, θ_o is the intercept, and θ_i is the weight coefficient for feature x_i. Through each iteration of training, the coefficient is adjusted using binary cross-entropy loss to optimize classification performance. Each iteration updates the coefficients, refining the model's ability to differentiate between the two classes accurately. Figure 1 illustrates an idea of what the logistic regression model performs in each layer.

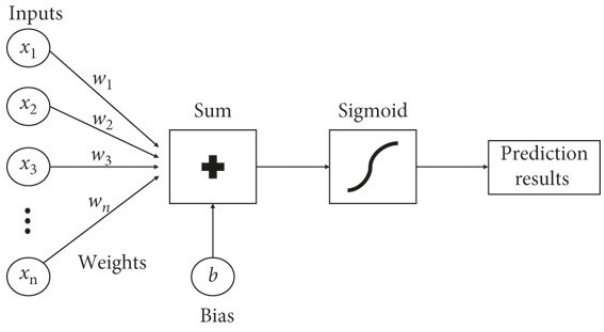

Figure 1: Logistic Regression Model

2.2 Zero-Knowledge Proofs

Zero-knowledge proofs (ZKPs) are a cryptographic protocol that allows one party, the prover, to convince another party, the verifier, that a certain statement is true without revealing any information beyond the truth of the statement. The goal of using ZKPs is to enable secure verification of claims without having to expose any sensitive data, making it essential for privacy-preserving computations.

There are 3 key properties of ZKPs [8]:

- Completeness: If the statement is true, an honest prover can convince an honest verifier.
- Soundness: If the statement is false, no cheating power can convince an honest verifier.
- Zero-knowledge: If the statement is true, the verifier learns nothing other than the fact that the statement is true.

The completeness property entails that there are no false negatives while soundness ensures that there are no false positives. These 2 properties build confidence for when the verifier approves of the proof. Zero-knowledge provides security and privacy, preventing any information from being exposed during the process.

The 2 key roles in ZKPs are the prover and the verifier. As seen in Figure 2, the prover possesses the secret information and needs to convince someone that they know the secret information. The verifier is the entity that needs to be convinced that the prover has the information. This process can be accomplished through interactive or non-interactive verifications. Interactive proofs are where the prover and verifier go through multiple rounds and proofs before the verifier can be convinced. Between rounds, the verifier asks questions based on the prover's answers [6]. Non-interactive proofs on the other hand only require a single message from the prover [9].

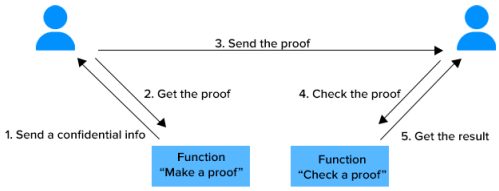

Figure 2: Proof and Verification

2.3 Zero-Knowledge Machine Learning

Zero-knowledge machine learning is an advanced application of ZKPs, integrating privacy-preserving techniques onto machine learning. The core idea of ZKML is to enable model inference and training while ensuring that sensitive information, such as input data and model parameters, remains confidential [7]. Unlike traditional machine learning approaches, where inference requires access to raw data, ZKML generates a cryptographic proof that verifies the correctness of an inference without exposing any underlying data. This is achieved by converting the machine learning model into a zero-knowledge circuit, which allows verifiable computation. Figure 3 below illustrates the design of a typical ZKML system.

Figure 3: ZKML Design

In ZKML, the machine learning model acts as the prover, generating predictions without revealing sensitive details about the data or model parameters. The verifier, typically an on-chain entity using an EVM smart contract, validates the proof to confirm the correctness of the computation. The ZKML process generally follows these steps [7]:

- Data Encryption: The input data is secured before processing. Homomorphic encryption is commonly used to enable computations directly on encrypted data.
- Model Training: The machine learning model is trained on encrypted or privacy-preserved data through methods such as federated learning to reduce computational overhead.
- Proof Generation: When making predictions, the model produced both an inference result and a zero-knowledge proof to verify correctness.
- Proof Verification: The verifier checks the proof without accessing any additional information, ensuring that the model's prediction is accurate and trustworthy.

By incorporating ZKPs into machine learning, ZKML enhances privacy, trust, and security, making it a valuable technique for applications in healthcare, finance, and decentralized AI systems. It is important to note that not all ZKML approaches are the same and each proof system has trade-offs in terms of efficiency, proof size, and computational overhead, which will influence their adoption in ZKML applications.

3. IMPLEMENTATION AND EVALUATION

This section covers the methods used to implement ZKML and evaluates its outputs and privacy-preservation. We go over the specifics of the dataset used and how we trained a logistic regression model using the data. The trained model is then converted into a Zero-Knowledge circuit using EZKL, which automates the process and aids in implementing witnesses and verifiers for the inferences.

3.1 Dataset

To evaluate the effectiveness of our privacy-preserving machine learning approach, we use a medical dataset for

ID	Age	Gender	Diabetes	Hypertension	Obesity	Smoking	...	Health Insurance	Heart Attack Risk
1	42	Female	0	0	1	1	...	0	0
2	26	Male	0	0	0	0	...	0	0
3	78	Male	0	0	1	0	...	1	0
4	58	Male	1	0	1	0	...	0	0
5	22	Male	0	0	0	0	...	0	1

Table 1: Sample of Dataset

heart disease risk prediction. The dataset is based on Indian cardiovascular health statistics, medical research reports, and national surveys from various sources such as the Indian Council of Medical Research, National Health Statistics World Health Organization, and National Family Health Survey. It incorporates key medical and lifestyle risk facts such as diabetes, hypertension, obesity, smoking, exposure to air pollution, and access to healthcare care in India. The purpose of this data set is to provide a diverse representation across India's states and reflect urban-rural differences in healthcare and lifestyle to their risk of heart attacks [1].

The dataset was sourced from Kaggle and was made by Ankush Panday. It has 10,000 entries going as far back as 2019 and is still being updated annually. Each entry has 26 features, 23 of which will be used to train the logistic regression model. These features are aligned with the patient's health history, income, or emergency response time in their state. Table 1 below illustrates a sample of the dataset.

Using the dataset, we split it into training data and testing data where 80% of the data would be used for training and the other 20% would be used to test its precision. The reason for this split is so we can ensure that the model does not overgeneralize by memorizing the data on which it was trained on and instead would be able to perform well on unseen data. With this setup, we preprocess the input data using a ColumnTransformer, which handles different data types in our data. In our case, the gender and state get converted to one-hot encoding, which is a boolean list that indicates 1 in the index that classifies its value. The other features are set to a scalar value as they are already numerical values.

3.2 Model Implementation

Several machine learning models, including Random Forest, AdaBoost, and Linear Discriminant Analysis, were considered for this study. While all models exhibited similar precision, we selected logistic regression due to its simplicity and interpretability. The logistic regression model uses the binary cross-entropy loss function and a sigmoid activation function, making it well-suited for binary classification tasks like heart disease prediction.

Due to compatibility limitations with EZKL, the trained logistic regression model was converted into a PyTorch-based neural network. This equivalent neural network consists of:

- A single linear layer
- A sigmoid activation function

After conversion, the model maintained the same accuracy score of 71.75%, confirming that the model's performance remained equivalent to the transformation.

Following training, the PyTorch model was converted to Open Neural Network Exchange (ONNX) format to ensure compatibility with EZKL. ONNX is an open-source format where models are represented as computational graphs, simplifying conversion into circuits, and enabling interoperability across different frameworks and hardware accelerators. As shown in Figure 4, the ONNX format allows the model to be represented as a ZK circuit.

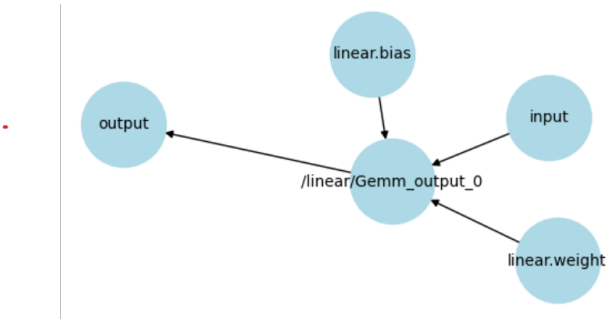

Figure 4: ONNX Representation of EZKL Circuit

As illustrated, the ONNX representation of the EZKL circuit:

- Takes in input data
- Utilizes the model's trained weights and biases
- Outputs the model's inference along with a cryptographic proof

By using an open format like ONNX, the model becomes compatible with a wide range of hardware optimizations and ONNX-compatible libraries, including EZKL, simplifying integration into Zero-Knowledge Proof (ZKP) systems.

EZKL is a library designed to automate Zero-Knowledge Proof (ZKP) generation for computational models [4], significantly simplifying the process of constructing ZK circuits. Given a model formatted in ONNX, EZKL transforms it into a ZKP-compatible circuit, enabling privacy-preserving inference.

To generate a proof, the ZKP circuit requires input data and calibration. In our implementation, we provide both plaintext and encrypted input data to compare inference performance under different privacy settings. Encryption is handled using TenSEAL, a library that enables homomorphic encryption (HE) on tensors, allowing computations to be performed without decrypting the data. Ideally, homomorphic encryption should also be applied during training, but due to computational constraints, this was not implemented—this limitation is further discussed in the Limitations section.

In practice, plaintext data should only be used locally on a private model to prevent exposure during inference and stop data from exiting the local environment. However, regardless of whether plaintext or encrypted inputs are used, data privacy is preserved throughout the entire computation and proof verification process.

We then move on to generating cryptographic keys for proving and verifying. These keys will be used by the witness, who will generate the proof, and the verifier, who confirms that the proof is correct. When generating the proof, the witness requires the input data and model parameters. These will result in a proof with the model's prediction and a proof hash for verification. For proof verification, EZKL provides 2 methods:

- On-Chain Verification (EVM Smart Contract)
- Off-Chain Verification (WebAssembly - WASM)

Going on-chain requires the use of Solidity-based smart contracts which are deployed on Ethereum-compatible platforms such as Remix Ethereum. This method provides a trustless verification mechanism but also requires higher computational costs. Using WASM allows verification directly in a web application, resulting in higher efficiency and reduced execution costs. While on-chain verification provides stronger trust guarantees, it is computationally expensive. In contrast, off-chain verification via WASM offers a faster and resource-efficient alternative but requires additional trust mechanisms to ensure the integrity of the verification process.

4. EVALUATION

4.1 Result

The logistic regression model initially achieved an accuracy of 71.75%. To assess the fidelity of the zero-knowledge circuit, EZKL provides a numerical fidelity report, summarized in Table 2. This report quantifies precision loss due to quantization, ensuring that the zero-knowledge proof (ZKP) maintains accuracy while optimizing for computational efficiency.

The mean absolute error (MAE) is approximately 1%, indicating a small average deviation between the original model's predictions and the EZKL circuit's outputs. Similarly, the mean squared error (MSE) is approximately 0.01%, which magnifies deviations but remains minimal. These low error rates confirm that the EZKL circuit accurately preserves the original model's behavior and accuracy.

Regarding the proof computation time, plaintext inference using EZKL took 13.75 seconds, while encrypted inference using homomorphic encryption (HE) required 13.81 seconds. The marginal increase in time suggests that HE introduces minimal additional computational overhead. However, compared to the original model—which performed inference in less than 1 second, ZKML incurs a significant increase in computation time due to the cost of proof generation.

Proof generation took 13.75 seconds per inference, while verification via WebAssembly (WASM) was completed in 0.06 seconds. Despite the added computational cost, this trade-off is acceptable for applications requiring strong privacy guarantees, where protecting sensitive medical data outweighs the need for real-time inference. In addition, the short verification time reflects the goal of reducing computational overhead on-chain such as Ethereum.

Through ZK verification, we confirmed that the model's predictions were accurately verified without revealing any sensitive data. The only outputs generated were:

- The model's inference, represented in hexadecimal format.
- A cryptographic proof, provided as a hash.

No additional information—such as model weights, input values, or intermediate computations—was exposed. This confirms that the proof successfully verifies the model's integrity without compromising privacy.

4.2 Performance

Among the libraries used for ZKML, EZKL stands out as one of the fastest and most efficient solutions available. A benchmark study by EZKL compared its performance against Orion and RISC Zero, two other prominent ZKML frameworks, and found significant advantages in proving time, memory efficiency, and ease of setup [3]. Their findings concluded that:

- Setup: EZKL provides a more streamlined and automated setup process compared to Orion and RISC Zero, which require more manual configuration and optimization steps. This makes EZKL easier to integrate, especially for developers new to zero-knowledge proofs.
- Proving Time: Across different machine learning models, EZKL demonstrated an average proving time 65.88× faster than RISC Zero and 2.92× faster than Orion.
- Memory Usage: EZKL reduced memory consumption by 63.95% compared to Orion and 98.13% compared to RISC Zero, making it significantly more resource-efficient.

In their tests, they used four models: Linear Regression, Random Forest Classification, Support Vector Machine (SVM) Classification, and Tree Ensemble Regression. For each model, the study measured the time taken to generate a proof and the corresponding memory usage. Figures 5 and 6 illustrate the benchmark results, showing EZKL's superior performance across both metrics.

As shown in Figure 5, EZKL outperforms Orion and RISC Zero in proving time across all models tested. Similarly, Figure 6 illustrates the substantial reduction in memory usage, reinforcing EZKL's efficiency. The key factor driving these performance differences lies in the underlying proof systems that each library uses. RIC Zero relies on a zkSTARK proof system using the FRI protocol, DEEP-ALI, and an HMAC-SHA-256-based PRF. Its zkVM-based approach executes each computational step individually, which increases proving overhead and memory usage. Orion also uses a zkSTARK proof system, but its execution is based on Cairo, a virtual machine that processes computations sequentially, contributing to higher resource demands. EZKL, in contrast, adopts a zkSNARK-based approach using Halo2. It optimizes performance further by replacing Halo2's default lookup mechanism with logUP lookup, which reduces proving costs by simplifying complex computations.

Mean Error	Median Error	Mean Abs Error	MAE	MSE	Mean Percent Error	Mean Abs Percent Error
0.010059655	0.010059655	0.010059655	0.010059655	0.00010119665	0.029023187	0.029023187

Table 2: Numerical Fidelity Report

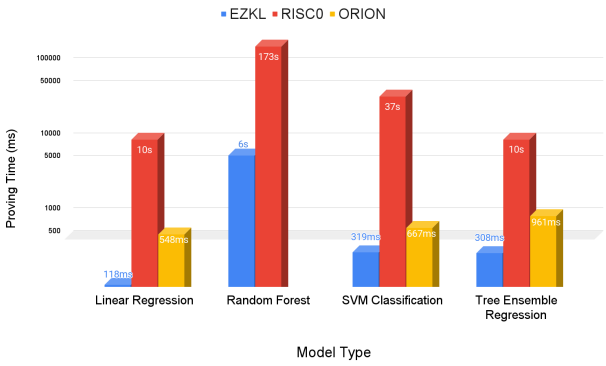

Figure 5: Proving Time to Model Type Among Libraries

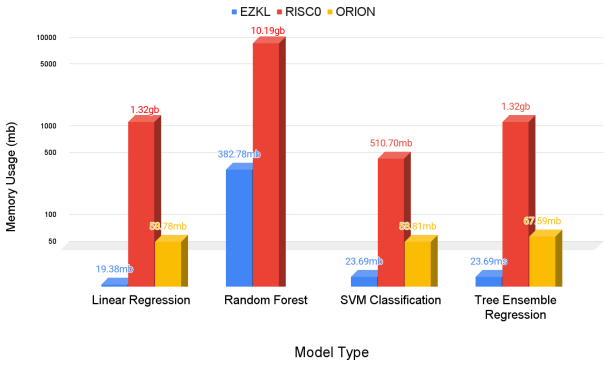

Figure 6: Memory Usage to Model Type Among Libraries

4.3 Limitations

One of the main limitations of ZKML is its high computational demand and the associated overhead. This overhead arises from the incompatibility between machine learning models and ZK circuits. Machine learning models typically use floating-point weights, which allow for precision adjustments during training, whereas ZK circuits rely on fixed-point arithmetic (integers) that lack the flexibility required to model complex behaviors with high precision [8]. As a result, ZK circuits must be adapted to approximate the operations of the machine learning model while balancing accuracy and privacy. This introduces a performance bottleneck, especially when trying to apply ZKML in privacy-sensitive domains such as healthcare.

Moreover, additional computational overhead is introduced when encrypted training techniques such as homomorphic encryption (HE) are introduced. HE enables computations on encrypted data, but this process is significantly more computationally expensive than traditional operations on plaintext data. This added complexity can drastically reduce the efficiency of ZKML systems, making them difficult to scale and deploy for real-world applications, particularly in domains like medical data analysis, where real-time processing is crucial. While ZKML is an area that is being studied, it remains in its early stages of research and development [3] and will be a while before production-ready projects appear.

Another key limitation is that EZKL, the library used to generate ZK circuits for machine learning models, is still in its early stages of development and is not yet ready for live production. Consequently, several compatibility issues prevent EZKL from offering full functionality and security guarantees. Currently, EZKL is optimized for models trained with PyTorch [4], and as a result, it cannot construct ZK circuits for models trained on encrypted data. This creates a significant challenge for applications that require end-to-end encryption during training, as ZKML's effectiveness relies on maintaining privacy from the start of the model's training process. While EZKL ensures privacy for inference stages, this limitation restricts its ability to safeguard training data, which is critical in privacy-preserving applications, particularly in healthcare settings where the training data is often the most sensitive.

5. CHALLENGES

ZKML presents several key challenges that must be addressed before it can be widely adopted in real-world applications, particularly in privacy-sensitive domains such as healthcare.

One challenge is generalizability, which refers to a model's ability to perform well on both seen and unseen data. Machine learning models typically rely on floating-point operations for precision, whereas zero-knowledge proofs (ZKPs) operate using integer-based arithmetic. This fundamental difference introduces accuracy loss when mapping floating-point computations to fixed-point representations [8]. Additionally, ZKPs rely on addition and multiplication, making it difficult to efficiently implement nonlinear activation functions, which are crucial for extracting complex features from data. As a result, alternative equations that approximate the nonlinear function must be used to resemble the original model.

Another challenge lies with efficient proof generation. While proof generation for small models is relatively fast, scaling to larger models results in exponential increases in computation time and memory requirements. In our study, the model was relatively small which reflected a rather fast proof generation. However, using a larger model such as the neural network VGG16, which has 138 million parameters, would require an estimated 10 years and 1 petabyte of storage to generate [8]. This makes ZKML impractical for deep learning models without significant optimization. Ongoing research suggests that efficiency could be improved through:

- Circuit optimization to reduce the number of constraints.
- More efficient verification techniques to speed up proof checking.

Outside of ZKPs, there exist challenges to the security of ZKMLs. While ZKML ensures correctness without revealing the model's inputs or parameters, it does not inherently encrypt the data. To achieve stronger privacy preservation, it must be combined with encryption techniques such as homomorphic encryption (HE), which further increases computational overhead. Aside from the model, ZKML remains vulnerable to various adversarial attacks, including:

- Membership Inference Attacks (MIAs): Attempting to infer whether specific data points were used during training.
- Data Poisoning: Manipulating training data to corrupt model outputs.
- Gaussian Noise Injection: Introducing noise to degrade model performance.

These are attack vectors that adversaries can use to corrupt the model's outputs. Attacks like MIAs are especially dangerous when the aim of using ZKML is to keep the model and its training data private. A potential mitigation strategy is adversarial training, where the model is trained with adversarial examples to improve robustness against such attacks [4].

Addressing these challenges requires a combination of hardware acceleration, cryptographic optimizations, and advancements in proof systems. Research into efficient ZK-friendly activation functions, as well as hybrid approaches combining HE and ZKML, could significantly improve the practicality of privacy-preserving machine learning.

6. CONCLUSION

In this study, we explored the potential of Zero-Knowledge Machine Learning (ZKML) for privacy-preserving medical inference, demonstrating how EZKL can convert a logistic regression model into a zero-knowledge circuit while maintaining accuracy. Our results confirm that ZKML ensures data privacy during inference by generating cryptographic proofs without exposing sensitive patient information. The fidelity analysis shows a minimal precision loss while benchmarking results highlight the efficiency of EZKL compared to other ZKML frameworks.

Despite these promising results, ZKML remains computationally intensive, with significant overhead from proof generation and verification. Adding additional measures such as encrypted training to ZKML further increases the overhead, making it difficult to implement in real-world applications. The reliance on fixed-point arithmetic and the incompatibility of current ZKML tools with encrypted training further limit its real-world applicability. Additionally, EZKL is still in its early stages of development, requiring further optimizations before it can be deployed in production medical environments.

Looking ahead, advancements in proof efficiency, hardware acceleration, and better integration with homomorphic encryption could significantly improve the practicality of ZKML for healthcare and other privacy-sensitive domains. As research in this area progresses, ZKML has the potential to become a cornerstone of privacy-preserving AI, enabling secure medical diagnostics, confidential model sharing, and verifiable machine learning in real-world applications.

REFERENCES

[1] Ankit, "Heart attack risk amp; prediction dataset in india," 2025. [Online]. Available: https://www.kaggle.com/dsv/10853291

[2] S. Bharath Babu and K. R. Jothi, "A secure framework for privacy-preserving analytics in healthcare records using zero-knowledge proofs and blockchain in multi-tenant cloud environments," *IEEE Access*, vol. 13, pp. 8439–8455, 2025.

[3] EZKL, "Benchmarking zkml frameworks." [Online]. Available: https://blog.ezkl.xyz/post/benchmarks/

[4] EZKL, "The ezkl system." [Online]. Available: https://docs.ezkl.xyz/

[5] I. Keshta and A. Odeh, "Security and privacy of electronic health records: Concerns and challenges," *Egyptian Informatics Journal*, vol. 22, no. 2, pp. 177–183, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1110866520301365

[6] T. Liu, X. Xie, and Y. Zhang, "zkcnn: Zero knowledge proofs for convolutional neural network predictions and accuracy," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 2968–2985. [Online]. Available: https://doi.org/10.1145/3460120.3485379

[7] Q. Team, "Zero-knowledge machine learning: A beginner's guide." [Online]. Available: https://www.quillaudits.com/blog/ai-agents/zero-knowledge-machine-learning-zkml

[8] Z. Xing, Z. Zhang, Z. Zhang, Z. Li, M. Li, J. Liu, Z. Zhang, Y. Zhao, Q. Sun, L. Zhu, and G. Russello, "Zero-knowledge proof-based verifiable decentralized machine learning in communication network: A comprehensive survey," 2023.

[9] S. Zapechnikov, "Privacy-preserving machine learning as a tool for secure personalized information services," *Procedia Computer Science*, vol. 169, pp. 393–399, 2020, postproceedings of the 10th Annual International Conference on Biologically Inspired Cognitive Architectures, BICA 2019 (Tenth Annual Meeting of the BICA Society), held August 15-19, 2019 in Seattle, Washington, USA. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050920303598