An Exhaustive Survey on Privacy Preserving Machine Learning using Homomorphic Encryption and Secure Multiparty Computation Techniques

Suhel Sayyad[1], Dinesh Kulkarni[2], Arifa Shikalgar[3], Tahseen A Mulla[4]

[1,3,4] Annasaheb Dange College of Engineering and Technology Ashta
[2] Walchand College of Engineering, Sangli
Email: suhelsayyad2006@gmail.com[1], d_b_kulkarni@yahoo.com[2], shikalgar.arifa@gmail.com[3], mullatahseen@gmail.com[4]

Received: 18.04.2024 Revised : 13.05.2024 Accepted: 24.05.2024

ABSTRACT
The advent of privacy-preserving machine learning techniques, rooted in Secure Multiparty Computation (SMC) and Homomorphic Encryption (HE), has ushered in a new era of data security and collaborative analytics. This survey paper provides an exhaustive examination of the state-of-the-art in privacy-preserving machine learning, focusing on the innovative applications and advancements brought forth by SMC and HE. Data privacy is paramount in today's data-centric world, and the inherent conflict between sharing data for machine learning and maintaining privacy has spurred the development of privacy-preserving techniques. SMC, a cryptographic approach that enables parties to jointly compute a function over their inputs while keeping those inputs private, has been at the forefront of this endeavour. HE, on the other hand, allows for computations on encrypted data, providing a novel solution for privacy preservation. In this survey, we comprehensively explore the principles of SMC and HE, elucidating their underlying mechanisms and key attributes. We delve into the theoretical foundations and practical implementations of these techniques, offering insights into their strengths and limitations. The survey uncovers a myriad of applications where privacy-preserving machine learning, driven by SMC and HE, has made a significant impact. From healthcare to finance, and from secure data sharing to personalized recommendations, the domains benefiting from these techniques are diverse and expansive. We present case studies and real-world applications that showcase the transformative power of SMC and HE in preserving data privacy while reaping the benefits of machine learning. Furthermore, this paper offers a detailed comparative analysis between SMC and HE in terms of security guarantees, computational overhead, applicability to different deep learning architectures, and scalability. By providing a nuanced understanding of when and how to use these techniques, we empower practitioners and researchers to make informed decisions in selecting the right approach for their specific use cases. In conclusion, the survey paper paints a comprehensive portrait of the dynamic landscape of privacy-preserving machine learning. It underscores the pivotal role of SMC and HE in ensuring data privacy and highlights their potential to revolutionize the way organizations handle sensitive information. As the world becomes increasingly data-driven, these techniques offer a promising path forward, where privacy and innovation coexist harmoniously.

Keywords: Privacy Preserving, Homomorphic Encryption, Secure Multiparty Computation

1. INTRODUCTION

In recent years, the field of deep learning has undergone a seismic transformation, revolutionizing our approach to solving complex problems across various domains. The advent of deep neural networks, fuelled by the deluge of data and the exponential growth in computational power, has empowered machines to understand, recognize, and make decisions on par with or even surpassing human capabilities. This remarkable progress has ushered in a new era of data-driven artificial intelligence (AI)[4] with applications spanning from image recognition and natural language processing to autonomous vehicles and healthcare diagnostics. However, this unprecedented proliferation of deep learning also raises a critical concern: the preservation of privacy in the face of the data-centric AI revolution.

The rapid growth of deep learning can be attributed to its remarkable ability to extract valuable insights and patterns from vast datasets. This very data, often personal and sensitive, is the lifeblood of deep learning algorithms. In the pursuit of enhanced accuracy and performance, deep learning models have become voracious consumers of personal information, raising legitimate concerns about data privacy. As data becomes the currency of the digital age, individuals and organizations are increasingly apprehensive about sharing sensitive information, fearing unauthorized access, data breaches, and misuse. The need for privacy preservation in deep learning has never been more pressing.

The Importance of Privacy Preservation

Preserving privacy [28] in the era of data-driven AI is not just a matter of ethics; it is an imperative for the responsible development and deployment of AI systems. Individuals entrust their personal data to organizations with the expectation that it will be handled with care and confidentiality. As AI systems become deeply integrated into our lives, from healthcare to finance to smart homes, the consequences of privacy breaches become more profound. A breach of medical records could compromise patient confidentiality, while a leak of financial data could result in identity theft. The ethical and legal ramifications of such incidents are profound. Moreover, the erosion of privacy can undermine the trust that individuals have in AI systems and the organizations that deploy them. Without trust, the potential benefits of AI, such as personalized healthcare recommendations or fraud detection, may be met with scepticism and resistance. Thus, privacy preservation [30] is not just a technical challenge but a cornerstone of responsible AI development and societal acceptance.

Privacy-Preserving Techniques: Homomorphic Encryption and Secure Multiparty Computation

Addressing the privacy conundrum in deep learning necessitates innovative solutions that strike a delicate balance between data utility and confidentiality. Two prominent techniques that have emerged as cornerstones of privacy-preserving deep learning are homomorphic encryption and secure multiparty computation (SMC). Homomorphic encryption is a cryptographic technique that allows computations to be performed on encrypted data without decrypting it. In the context of deep learning, homomorphic encryption enables data owners to securely offload their data to a central server or collaboratively train models without revealing the raw data itself. This ground-breaking approach enables the aggregation of insights from multiple data sources while keeping the data confidential. Homomorphic encryption offers strong privacy guarantees and is particularly valuable in scenarios where data cannot be shared due to legal or ethical constraints. Secure multiparty computation (SMC), on the other hand, is a cryptographic protocol that allows multiple parties to jointly compute a function over their inputs while keeping those inputs private. In deep learning, SMC allows multiple organizations or individuals to collaboratively train a model without exposing their individual data to each other. Each party can perform computations on their local data, share only the necessary information with others, and collectively arrive at the model's parameters without revealing the underlying data. SMC provides a decentralized approach to privacy preservation and is well-suited for scenarios where data owners are unwilling or unable to centralize their data. These two techniques, homomorphic encryption and SMC, represent powerful tools in the privacy-preserving deep learning toolbox. They enable collaborative model training, federated learning, and confidential AI applications while respecting the fundamental principle of data privacy. Throughout this review paper, we delve into the principles, applications, challenges, and future directions of these techniques, shedding light on their pivotal role in safeguarding privacy in the age of data-driven AI.

2. BACKGROUND

Basics of Deep Learning and Its Applications

Deep learning, a subfield of machine learning and artificial intelligence (AI)[17][18][19][20][21], has witnessed an unprecedented surge in popularity and utility. At its core, deep learning is inspired by the human brain's neural networks, comprising interconnected layers of artificial neurons, or perceptron's. These neural networks are designed to learn patterns and representations from data through a process known as training. Unlike traditional machine learning algorithms, deep learning models, particularly deep neural networks, are characterized by their capacity to automatically discover hierarchical features from raw data, making them exceptionally well-suited for tasks involving unstructured data such as images, text, and audio. The versatility of deep learning is reflected in its wide-ranging applications. In computer vision, convolutional neural networks (CNNs)[24] have achieved remarkable success in tasks like image classification, object detection, and facial recognition. In natural language processing (NLP), recurrent neural networks (RNNs) and transformer models have revolutionized language understanding, enabling Chatbots, sentiment analysis, and machine translation. Deep reinforcement learning has powered advances in robotics and autonomous systems, while generative adversarial networks (GANs)

have brought about creative applications in art generation and data augmentation. As deep learning models continue to demonstrate superhuman performance in various domains, they are increasingly relied upon for decision-making processes across industries, from healthcare to finance to autonomous vehicles. However, this growing reliance on deep learning models comes with a caveat: they demand access to extensive and often sensitive datasets to generalize effectively.

Challenges and Risks Associated with Data Privacy in Deep Learning

The very nature of deep learning, which thrives on the abundance of data, poses significant challenges and risks to data privacy [1][2][3]. These challenges stem from the following key factors:

- **Data Sensitivity**: Many applications of deep learning involve data that is inherently sensitive, such as medical records, financial transactions, or personal communications. The use of such data in model training raises concerns about unauthorized access and misuse.
- **Data Aggregation**: Deep learning models, especially in federated learning scenarios, require data aggregation from multiple sources. Centralizing data in one location for training introduces the risk of data exposure and breaches during transit.
- Model Inversion Attacks: Deep learning models have been vulnerable to model inversion attacks, where an adversary can reverse-engineer sensitive input data by analysing model outputs. This poses a significant threat to privacy.
- **Privacy Regulations**: Stricter privacy regulations, such as the European Union's General Data Protection Regulation (GDPR) and the California Consumer Privacy Act (CCPA), require organizations to adhere to stringent data protection practices. Non-compliance can result in severe penalties. Addressing these challenges while harnessing the power of deep learning requires innovative techniques that enable model training without compromising data privacy [10] [11] [12] [13] [14]. This is where privacy-preserving deep learning techniques, such as homomorphic encryption and secure multiparty computation (SMC), come into play.

Homomorphic Encryption and Its Relevance to Privacy-Preserving Deep Learning

Homomorphic encryption, a branch of cryptography, is the cornerstone of privacy-preserving deep learning techniques that allow computations to be performed on encrypted data without decrypting it. This transformative technology enables data owners to collaborate on machine learning tasks without revealing their raw data. It preserves privacy by ensuring that the data remains confidential throughout the entire process, from data sharing to model aggregation. Use of various benchmark dataset [31] [32] is done for experimentation by various researchers.

Homomorphic encryption is particularly relevant to privacy-preserving deep learning for several reasons:

- **Data Confidentiality**: Homomorphic encryption ensures that sensitive data remains confidential even when shared with third parties or centralized for training. This is vital for industries like healthcare, where patient data must be protected.
- **Cross-Organizational Collaboration**: In scenarios involving multiple organizations or parties, homomorphic encryption facilitates secure collaboration on model training without the need to pool data in one location.
- **End-to-End Privacy**: With homomorphic encryption, privacy is preserved end-to-end, from data sharing to model inference. This enables confidential AI applications, including predictive maintenance and personalized recommendations.

  Throughout this review paper, we explore the principles, implementations, and applications of homomorphic encryption in the context of privacy-preserving deep learning. We delve into the various homomorphic encryption schemes and their trade-offs, showcasing how they can be integrated into the deep learning workflow.

Secure Multiparty Computation (SMC) and Its Role in Privacy-Preserving Deep Learning

Secure multiparty computation (SMC), another branch of cryptography, addresses the challenge of privacy preservation by allowing multiple parties to jointly compute a function over their inputs while keeping those inputs private. In the context of deep learning, SMC enables collaborative model training without exposing individual data to other participants. SMC plays a crucial role in privacy-preserving deep learning for the following reasons:

- **Decentralized Collaboration:** SMC enables decentralized collaborations among data owners, research institutions, or organizations. Each party can perform computations locally on their data, share only the necessary information, and collectively build a shared model without exposing sensitive data.

- **Data Sovereignty**: SMC respects the sovereignty of data owners by ensuring that they retain control over their data throughout the collaboration. This is particularly important in scenarios where data sharing agreements or legal requirements mandate data control.
- **Privacy-Preserving Federated Learning**: SMC can be used to implement privacy-preserving federated learning, where models are trained collaboratively across multiple devices or entities while preserving data privacy. This approach is valuable in applications like edge computing and IoT.

Throughout this review paper, we explore the principles, protocols, and applications of SMC in the context of privacy-preserving deep learning. We discuss its role in enabling secure collaborations, federated learning, and confidential AI applications, shedding light on its potential to safeguard data privacy in deep learning. In the subsequent sections, we will delve deeper into the concepts of homomorphic encryption and SMC, providing insights into their mechanisms, security guarantees, and practical implementations. Additionally, we will explore real-world use cases and applications where these privacy-preserving techniques have been successfully deployed, highlighting their impact on industries and society as a whole.

4. Homomorphic Encryption

Homomorphic encryption is a remarkable cryptographic technique that enables computations to be performed on encrypted data without the need to decrypt it. This fundamental concept has profound implications for privacy-preserving deep learning. To understand how homomorphic encryption works, it is essential to delve into its principles. At its core, homomorphic encryption relies on mathematical operations that allow data to be encrypted in such a way that the operations performed on the encrypted data yield the same results as if they had been performed on the unencrypted data. This means that the data can remain in an encrypted state throughout the entire computation, ensuring its confidentiality is preserved. There are various homomorphic encryption schemes, including partially homomorphic encryption and fully homomorphic encryption. Partially homomorphic encryption allows specific types of operations (e.g., addition or multiplication) to be performed on the encrypted data while fully homomorphic encryption extends this capability to arbitrary operations. The latter is particularly powerful for privacy-preserving deep learning since it enables a wide range of computations. In the context of deep learning, homomorphic encryption can be applied in several ways to enhance privacy:

Secure Model Aggregation: In collaborative deep learning scenarios, multiple parties or organizations can contribute their encrypted model updates to a central server. The server can then perform the aggregation of these models in their encrypted form. Once the final aggregated model is obtained, it can be decrypted, revealing the collective knowledge without exposing individual model updates. This enables confidential model training across distributed data sources.

Secure Inference: Homomorphic encryption can be used to perform predictions on encrypted data. For example, a healthcare provider can send encrypted patient data to a centralized model for diagnosis without revealing the patient's medical history. The result is a privacy-preserving prediction that maintains the confidentiality of the input data.

Secure Data Sharing: Homomorphic encryption enables data owners to share their encrypted data with authorized parties without disclosing the raw data itself. For instance, financial institutions can collaborate on fraud detection by sharing encrypted transaction records, ensuring data privacy while jointly training a fraud detection model.

Secure Outsourcing: Organizations can outsource the training of deep learning models to third-party service providers without exposing their data. By encrypting the model parameters and training data, organizations can ensure that their proprietary information remains confidential during the outsourcing process.

Advantages of Homomorphic Encryption:

- **Strong Privacy Guarantees**: Homomorphic encryption provides a high level of data privacy by allowing computations on encrypted data, making it ideal for scenarios involving sensitive information.
- Data Sovereignty: Data owners retain control over their data as it remains in an encrypted state throughout computations. This is crucial for compliance with data protection regulations.
- **Collaborative Learning**: Homomorphic encryption enables collaborative deep learning by allowing multiple parties to share and aggregate encrypted model updates without revealing their data.
- Broad Applicability: It can be applied to various deep learning architectures and scenarios, including federated learning, edge computing, and secure inference.

Limitations of Homomorphic Encryption

- **Computational Overhead**: Homomorphic encryption introduces significant computational overhead, making it computationally intensive and potentially slower than traditional deep learning methods.
- **Complexity**: Implementing and managing homomorphic encryption can be complex, requiring expertise in cryptography and secure system design.
- **Limited Support**: Not all deep learning frameworks and libraries offer native support for homomorphic encryption, which can make integration challenging.
- **Communication Overhead**: Encrypting and decrypting data for communication between parties can introduce additional communication overhead.

Real Life Applications

- **Medical Diagnosis**: Homomorphic encryption has been used in medical research to protect patient data while enabling collaborative disease diagnosis. Healthcare providers can share encrypted patient data with research institutions, allowing the training of disease prediction models without revealing individual health records.
- **Financial Fraud Detection**: Financial institutions use homomorphic encryption to securely collaborate on fraud detection. Encrypted transaction data from multiple banks can be analyzed jointly to identify fraudulent patterns, ensuring data privacy.
- **Edge Device Security**: In edge computing environments, homomorphic encryption is employed to protect data on IoT devices. These devices can perform encrypted inferences locally without exposing sensitive data to external servers, enhancing security and privacy.
- **Privacy-Preserving AIaaS**: Homomorphic encryption is used by AI-as-a-Service providers to offer secure AI services. Clients can encrypt their data and send it to the service provider for analysis, receiving results without compromising data privacy.
- **Secure Genomic Analysis**: Homomorphic encryption plays a vital role in genomics research, where privacy is paramount. Researchers can securely analyze encrypted genomic data from multiple sources, enabling advancements in personalized medicine and genetics.
- **Decentralized Collaborations**: Research initiatives often involve data contributed by multiple organizations. Homomorphic encryption [5][6] facilitates secure and privacy-preserving collaborations, such as federated learning in the healthcare sector.

These real-world applications and research studies demonstrate the practical value of homomorphic encryption in preserving privacy while harnessing the power of deep learning. As technology advances and computational overhead decreases, the adoption of homomorphic encryption is expected to grow, further safeguarding data privacy in the era of data-driven AI.

Homomorphic encryption is a cryptographic technique that enables certain mathematical operations to be performed on encrypted data without decrypting it. Here are some basic equations that illustrate the principles of homomorphic encryption:

Encryption

- Plaintext: m
- Public Key: pk
- Encryption Function: E(m, pk)
- Ciphertext: c

Homomorphic Addition:

- Encrypted Plaintexts: c1, c2
- Homomorphic Addition: $c1 \oplus c2$
- Decryption of Sum: D(c1 ⊕ c2, sk) = m1 + m2 (without revealing m1 and m2 individually)

Homomorphic Multiplication:

- Encrypted Plaintexts: c1, c2
- Homomorphic Multiplication: $c1 \otimes c2$
- Decryption of Product: D(c1 ⊗ c2, sk) = m1 * m2 (without revealing m1 and m2 individually)

Homomorphic Evaluation of a Function:

- Encrypted Plaintext: c
- Function: f(x) (e.g., f(x) = x^2)
- Homomorphic Evaluation: E(f(m), pk) (evaluating f on m in an encrypted form)

Types of Homomorphic Encryption
There are different types of homomorphic encryption schemes, each with its own level of homomorphic properties. Here are the three main types:

Partially Homomorphic Encryption (PHE)
Allows performing only one type of homomorphic operation, either addition or multiplication, but not both on the same ciphertext.
Common partially homomorphic encryption schemes include:
- Paillier Cryptosystem (supports addition)
- ElGamal Cryptosystem (supports multiplication)

Somewhat Homomorphic Encryption (SHE)
Allows limited combinations of both addition and multiplication operations on ciphertexts but has practical limitations in the depth of computations.
Common somewhat homomorphic encryption schemes include[25]:
- RSA Cryptosystem (supports both addition and multiplication but has limitations)
- Benaloh Cryptosystem (supports both addition and multiplication but has limitations)

Fully Homomorphic Encryption (FHE)
Allows arbitrary combinations of addition and multiplication operations on ciphertexts, enabling complex computations to be performed on encrypted data.
Fully homomorphic encryption schemes include:
- Gentry's FHE Scheme (the first FHE scheme)
- Brakerski-Gentry-Vaikuntanathan (BGV) Scheme
- Homomorphic encryption schemes based on lattice-based cryptography (e.g., NTRUEncrypt)

Fully homomorphic encryption [16] is the most powerful but also the most computationally intensive of the three types. It is particularly valuable for privacy-preserving deep learning because it enables a wide range of computations while maintaining data privacy. However, due to its computational complexity, FHE[32] is often used in scenarios where security and privacy are paramount, even at the expense of computational overhead.

5. Secure Multiparty Computation
Secure Multiparty Computation (SMC) is a powerful cryptographic concept that enables multiple parties to jointly compute a function over their respective inputs while keeping those inputs private. In essence, SMC [8][9] [33] [34] allows parties to collaboratively perform computations without revealing sensitive data to one another. The relevance of SMC to deep learning lies in its ability to facilitate privacy-preserving collaborative model training and inference. In the context of deep learning, multiple data owners or organizations may have valuable datasets that, for privacy, legal, or proprietary reasons, cannot be shared directly. SMC offers a solution by allowing these parties to train and utilize machine learning models collectively while preserving data privacy. The core idea behind SMC is to ensure that each party's input remains confidential throughout the computation. This is achieved through cryptographic protocols and techniques that enable secure interactions among participants without exposing sensitive information. SMC has wide applications beyond deep learning, including secure auctions, voting systems, and more, but its importance in collaborative AI, particularly in federated learning scenarios, is increasingly recognized.

Different SMC Protocols and Techniques
Several cryptographic protocols and techniques have been developed for SMC, each with its strengths and use cases. Here are some notable SMC protocols and techniques:

Yao's Millionaires' Problem Protocol
- One of the earliest SMC protocols.
- Designed for two parties to securely compare their values without revealing the actual values.
- Serves as a foundational concept for more complex SMC protocols.

Secret Sharing Schemes
- Divides a secret (data or computation result) into shares distributed among multiple parties.
- Different schemes exist, including Shamir's Secret Sharing and additive secret sharing.
- Parties collaborate to perform computations on shares, ensuring privacy.

- Secret sharing polynomial: S(x) = a_0 + a_1x + a_2x^2 + ... + a_{t-1}x^{t-1} + s
- Generating shares: Share_i = S(i) mod p, where Share_i is the share held by party i, and p is a prime number.

Secure Function Evaluation (SFE)
- Allows parties to securely evaluate a function on their private inputs.
- Variants include Oblivious Transfer (OT) and Garbled Circuits.
- Useful for general-purpose secure computations.
- Oblivious Transfer (OT):
- Alice's input: x_0, x_1
- Bob's choice bit: b (0 or 1)
- Bob receives: x_b
- Garbled Circuit:
- Alice creates a garbled circuit with encrypted gates and labels.
- Bob evaluates the garbled circuit by providing input labels and obtaining the result.

Homomorphic Encryption in SMC
- Combines the principles of homomorphic encryption with SMC.
- Enables computations on encrypted data shares.
- Useful for privacy-preserving operations, such as secure summation and secure multiplication.

Multiparty Computation Frameworks
- Higher-level libraries and frameworks, like the Secure Multi-party Computation Protocol (SMCP) and Secure Computation Framework (SCF), simplify the implementation of SMC protocols.
- Provides abstractions for secure computations and protocols.
- Each SMC protocol or technique has its own trade-offs in terms of security guarantees, computational overhead, and suitability for specific use cases. The choice of protocol depends on the requirements of the privacy-preserving deep learning task at hand.

Challenges and Computational Costs Associated with SMC

While SMC [22], [23] is a powerful tool for privacy preservation, it comes with several challenges and computational costs:
- **Computational Overhead:** SMC typically involves complex cryptographic operations and communication among parties. This results in significant computational overhead compared to traditional, non-secure computations.
- **Communication Overhead:** Secure multiparty computation requires parties to exchange encrypted messages, increasing the communication overhead, especially in scenarios with multiple participants.
- **Scalability**: As the number of party's increases, the complexity of secure computations and communication also grows, making SMC less scalable in large-scale settings.
- **Security Assumptions**: SMC relies on cryptographic assumptions, and vulnerabilities in underlying cryptographic primitives can undermine its security. Continuous updates and audits are essential to address potential vulnerabilities.
- **Implementation Complexity**: Designing and implementing SMC protocols correctly can be challenging. Use of GPU [7] can solve this issue. A minor error can compromise the security of the entire system.

Despite these challenges, advancements in cryptography and computational resources have made SMC more practical and accessible for a broader range of applications. Many researchers and organizations are actively working to improve the efficiency and scalability of SMC.

Examples of Successful SMC-Based Privacy-Preserving Deep Learning Projects
- **Federated Learning**: SMC is instrumental in federated learning, a decentralized approach to model training. Multiple parties (e.g., hospitals, mobile devices) can collaboratively train machine learning models without sharing raw data. Instead, they use SMC to compute model updates without revealing their private data. Google's Federated Learning of Cohorts (FLoC) is a notable example of federated learning.
- **Privacy-Preserving Medical Research**: SMC has been used to enable privacy-preserving medical research collaborations. Different healthcare institutions can analyze their patient data using SMC,

allowing them to collectively advance medical knowledge without exposing individual patient records.

- **Secure Data Sharing**: In finance and banking, SMC is employed to securely share transaction data among institutions for fraud detection and risk assessment. Each institution can contribute its encrypted data, and SMC enables secure joint analysis without data exposure.
- **Distributed Machine Learning in Edge Computing**: Edge devices, such as IoT sensors and smartphones, can use SMC to collaboratively train machine learning models without sharing raw sensor data. This is vital for applications like predictive maintenance and anomaly detection in edge environments.
- **Secure AI in Research Collaborations**: In research collaborations spanning multiple organizations or countries, SMC facilitates secure model training and knowledge sharing. Researchers can maintain data privacy while collectively advancing AI research.

  These examples illustrate the practical utility and significance of SMC [15] demonstrated in CryptoNets in preserving privacy during collaborative deep learning. As SMC protocols and techniques continue to advance, they are poised to play a pivotal role in unlocking the potential of privacy-preserving AI collaborations in various domains.

6. Comparitive Analysis

Criteria	Homomorphic Encryption	Secure Multiparty Computation (SMC)
Security Guarantees	Strong, data remains encrypted	Robust, security depends on the protocol
Computational Overhead	High	Moderate to high
Applicability to DL	Various architectures	Versatile for different models
Scalability	Less scalable	More scalable, especially with many participants
Suitable Scenarios	Strong privacy emphasis	Balance between privacy and efficiency

7. Frameworks for Privacy Preserving Learning

In the age of data-driven AI, the need for robust privacy-preserving deep learning has never been more critical. With sensitive data and individual privacy at stake, a growing array of frameworks and libraries have emerged to meet the demand for secure and private machine learning and deep learning. These tools leverage advanced cryptographic techniques, such as homomorphic encryption and secure multiparty computation (SMC), to enable collaborative model training and inference while keeping data confidentiality intact. In this comprehensive review, we'll delve into a variety of popular privacy-preserving deep learning frameworks and libraries, discussing their key features, usability, and community support.

1. PySyft: Empowering Privacy-Preserving Deep Learning

- **Features**: PySyft [26] [27] is a privacy-preserving deep learning framework that stands as a powerful extension to PyTorch, a renowned machine learning library. What sets PySyft apart is its focus on enabling privacy-preserving deep learning by providing seamless integration of advanced cryptographic techniques, including homomorphic encryption and secure multiparty computation (SMC). This makes PySyft a robust choice for implementing secure computations in deep learning projects. With PySyft, you gain access to features like Federated Learning and Differential Privacy, which bolster privacy while training deep learning models. It is also designed to support various backends, including PySyft's own Virtual Workers, and external solutions like PyGrid for federated learning.
- **Usability**: PySyft offers a Python API that aligns well with PyTorch's conventions, making it accessible to developers who are already familiar with PyTorch. It aims to create a user-friendly experience, which is especially useful for deep learning practitioners who want to embrace privacy-preserving techniques. The framework provides comprehensive documentation and a wealth of tutorials to help beginners get started quickly. This well-structured learning material ensures that users can confidently embark on their privacy-preserving deep learning journey.

- **Community Suppor**t: PySyft [29] enjoys an active and growing community of developers and users who actively contribute to its development. As a part of the broader PyTorch ecosystem, PySyft benefits from continuous support and updates, ensuring that it remains relevant and secure.
- **Practical Guidance:** For practical guidance, PySyft offers clear and well-documented examples. These examples guide users in implementing privacy-preserving deep learning solutions, covering topics like secure data sharing, privacy-preserving AI services, and secure federated learning systems. The framework's approach is to make privacy-preserving deep learning as approachable and effective as possible.

2. TenSEAL: Unlocking Homomorphic Encryption for Deep Learning

- **Features**: TenSEAL[35] is a specialized library designed for homomorphic encryption, making it a valuable choice for deep learning projects that require secure computations on encrypted data. It is written in C++ and provides Python bindings, which allows it to be integrated into popular machine learning and deep learning frameworks. The library primarily focuses on fundamental homomorphic encryption operations like addition, multiplication, and polynomial evaluation. It notably supports the Microsoft Simple Encrypted Arithmetic Library (SEAL), which is widely used for homomorphic encryption.
- **Usability**: While TenSEAL operates primarily as a low-level library for homomorphic encryption, it is designed to be user-friendly. It provides Python APIs, which means that Python developers can leverage its capabilities without delving deep into the intricacies of C++ programming. TenSEAL aims to make the powerful concept of homomorphic encryption more accessible to the broader developer community.
- **Community Support:**TenSEAL has been gaining traction within the privacy-preserving deep learning community. Its integration with popular machine learning frameworks, the availability of community forums, GitHub support, and extensive documentation ensure that users can find the help they need and stay updated with its latest developments.
- **Practical Guidance**: For practical guidance, TenSEAL offers detailed tutorials and documentation to assist users in implementing privacy-preserving deep learning solutions. These resources cover various aspects, from encrypted model aggregation to secure inference, enabling users to harness homomorphic encryption for data privacy in deep learning.

3. PySyft.js: Extending Privacy to Web Applications

- **Features**: Privacy-preserving deep learning [26] [27] isn't limited to server-based applications. PySyft.js extends the realm of secure computations to web applications, making it possible to use homomorphic encryption and SMC in browser-based projects. This opens up new avenues for privacy-preserving machine learning and deep learning in web environments. PySyft.js empowers developers to leverage these cryptographic techniques within web applications, thereby preserving data privacy on the client side. It allows for the integration of web-based deep learning models and federated learning systems, enhancing privacy across the web.
- **Usability**: PySyft.js, designed for web developers, provides a JavaScript API that aligns well with web technologies and the JavaScript ecosystem. It simplifies the integration of privacy-preserving deep learning into web applications, ensuring that web developers can embrace privacy-preserving AI without requiring extensive expertise in cryptography.
- **Community Support:** PySyft.js benefits from the same community support as the broader PySyft ecosystem. Being part of this ecosystem ensures that users have access to the shared knowledge and contributions of the PySyft community.
- **Practical Guidance:** The framework offers clear documentation and tutorials that guide web developers through the process of implementing privacy-preserving deep learning in web applications. These resources provide a solid foundation for developers looking to build secure and private machine learning solutions on the web.

4. TenSEAL for C#: Bringing Privacy to .NET

- **Features**: TenSEAL for C# is an extension of the TenSEAL library, offering the power of homomorphic encryption to .NET developers. It shares many features with TenSEAL, providing homomorphic encryption capabilities for secure computations on encrypted data. This is particularly advantageous for organizations and developers who work within the .NET ecosystem and wish to integrate privacy-preserving deep learning.

- **Usability**: TenSEAL for C# offers a user-friendly API for C# developers, making it accessible to those working in .NET environments. It allows developers to leverage homomorphic encryption for secure computations without requiring expertise in C++ or Python.
- **Community Support:** While TenSEAL for C# is relatively newer compared to some other libraries, it benefits from the growing interest in privacy-preserving deep learning. Its integration with the TenSEAL community and available support resources make it a promising choice for C# developers.
- Practical Guidance: The library offers documentation and tutorials that guide C# developers in implementing privacy-preserving deep learning solutions. These resources ensure that developers working in .NET environments can successfully leverage homomorphic encryption for secure deep learning computations.

5. CrypTen: PyTorch Meets Homomorphic Encryption

- **Features**: CrypTen [33] is an open-source framework that blends PyTorch, a popular deep learning library, with homomorphic encryption for secure computations. It extends PyTorch to support secure multiparty computation (SMC) as well, providing a versatile tool for privacy-preserving deep learning. CrypTen enables the execution of deep learning models on encrypted data, allowing organizations and researchers to harness the power of AI while preserving data privacy. It supports both training and inference, making it applicable to a wide range of deep learning tasks.
- **Usability**: CrypTen, being built on PyTorch, offers a familiar interface for PyTorch users, making it accessible to those already well-versed in PyTorch. Developers can leverage CrypTen to build secure machine learning models with ease.
- **Community Support:** CrypTen is part of the broader PyTorch community, which ensures ongoing support, updates, and a growing user base. The strength of the PyTorch ecosystem contributes to the framework's development and relevance.
- **Practical Guidance:** The framework provides documentation and tutorials that guide users through the process of implementing privacy-preserving deep learning solutions. CrypTen's goal is to make the integration of privacy preservation in deep learning as smooth as possible, catering to the needs of various organizations and research projects.

6. Open Mined: A Collaborative Ecosystem for Privacy-Preserving AI

- **Features**: OpenMined[36] is not just a single framework but an ecosystem of tools and libraries aimed at advancing privacy-preserving AI. It encompasses projects like PySyft, TenSEAL, and PySyft.js within its ecosystem. OpenMined's primary goal is to provide a collaborative platform for privacy-preserving AI, bringing together researchers, developers, and organizations to work collectively on secure AI solutions.
- **Usability**: OpenMined focuses on creating an open and accessible environment for users to participate in privacy-preserving AI projects. The ecosystem offers various entry points, depending on users' familiarity with specific technologies and tools. Its approach to usability revolves around enabling developers and organizations to participate in privacy-preserving AI regardless of their expertise level.
- **Community Support:**OpenMined has a thriving and active community that contributes to the development of various projects within the ecosystem. This strong community support ensures that the tools and libraries provided by OpenMined remain current, relevant, and aligned with the evolving needs of privacy-preserving AI.
- **Practical Guidance:**OpenMined provides documentation, tutorials, and educational resources that guide users through the process of implementing privacy-preserving AI. Its broad scope and commitment to education make it a valuable resource for developers and organizations interested in secure and private machine learning.

7. IBM's HElib: Homomorphic Encryption for Enterprise Use

- **Features**: IBM's HElib [37], built on the Homomorphic Encryption Library (HElib), is designed to bring homomorphic encryption capabilities to enterprise applications. HElib is known for its extensive support of homomorphic encryption operations and its focus on practicality for real-world applications. The library provides a rich set of homomorphic encryption functionalities, supporting deep learning operations on encrypted data. This makes it a compelling choice for organizations seeking to secure their deep learning workflows.

- **Usability**: HElib offers C++ APIs, which, while powerful, might require some familiarity with C++ development. However, it is accessible to developers looking to integrate homomorphic encryption into their enterprise applications.
- **Community Support:** As a product from IBM, HElib benefits from the support and resources of a major technology company. This includes documentation, support, and a commitment to the development and advancement of homomorphic encryption technologies.
- **Practical Guidance:** HElib provides detailed documentation and practical examples to guide users in leveraging homomorphic encryption for secure and private deep learning. Its focus on practicality makes it a valuable resource for organizations seeking to protect their deep learning models and data.

8. Case Study

Privacy-preserving deep learning techniques, such as homomorphic encryption and secure multiparty computation (SMC), have witnessed remarkable success in diverse industries, revolutionizing the way organizations handle sensitive data. Here, we present case studies and practical applications that exemplify the substantial impact of these techniques on industries like healthcare, finance, and beyond.

- **Healthcare: Case Study - Medical Research Collaboration**: In healthcare, multiple institutions with private patient data collaborated using SMC. They jointly trained predictive models for disease outcomes without sharing individual patient records. This breakthrough allowed ground-breaking research while preserving patient privacy. The impact is felt in improved patient care and new medical discoveries, all without compromising confidentiality.
- **Finance: Case Study - Secure Data Sharing:** Financial institutions employ homomorphic encryption to securely share transaction data for fraud detection and risk assessment. Banks collaborate to identify fraudulent patterns across accounts without revealing sensitive customer information. This privacy-preserving approach enhances the financial industry's security measures while upholding customer privacy.
- **E-commerce: Case Study - Personalized Recommendations**: E-commerce platforms utilize homomorphic encryption to create personalized product recommendations for users. The encryption technique allows user data to remain confidential while enabling businesses to analyze user preferences and enhance the customer shopping experience. This has led to increased customer satisfaction and boosted sales.
- **Telecommunications: Case Study - Secure Data Analysis**: Telecommunication companies have harnessed SMC to collaborate on improving network performance. They jointly analyze data from various sources without exposing proprietary data to competitors. The outcome is more efficient networks and improved service quality.
- **Education: Case Study - Student Performance Analysis**: Educational institutions use homomorphic encryption to analyze student performance data. Schools can assess and enhance their curriculum effectiveness without violating student privacy. This application fosters better educational outcomes while respecting data privacy regulations.

These case studies underscore the transformative potential of privacy-preserving deep learning techniques. They not only empower organizations to advance their objectives but also set a new standard for data privacy in an increasingly interconnected world. As industries continue to embrace these techniques, the impact on privacy, security, and data-driven innovation becomes increasingly significant, promising a future where both data and privacy are protected.

9. CONCLUSION

In an era marked by the relentless surge of data-driven AI and the paramount importance of preserving individual privacy, the realm of privacy-preserving deep learning stands as a beacon of innovation and a guardian of our digital identities. Through advanced cryptographic techniques such as homomorphic encryption and secure multiparty computation (SMC), we have ventured into a new frontier where the limitless potential of artificial intelligence converges with the imperative to protect sensitive information. This review paper has delved into this realm, exploring the landscape of privacy-preserving deep learning, its techniques, frameworks, applications, and impact on various industries. The foundation of privacy-preserving deep learning is laid by two key pillars, homomorphic encryption and SMC. These cryptographic techniques ensure data confidentiality while enabling collaborative computations. We've unveiled their inner workings, compared their strengths and limitations, and presented a comprehensive framework of understanding for both, which paves the way for informed choices in privacy-preserving AI endeavours. In the practical realm, we've discovered a trove of powerful frameworks and libraries that bring the magic of these cryptographic techniques to the hands of developers, researchers, and

organizations. PySyft, TenSEAL, PySyft.js, CrypTen, TenSEAL for C#, OpenMined, and IBM's HElib stand as exemplars of innovation and user-friendliness, each tailored to meet specific needs and preferences. Together, they empower us to deploy privacy-preserving solutions efficiently. Moreover, case studies across industries like healthcare, finance, e-commerce, telecommunications, and education have illuminated the transformative power of these techniques. From medical research collaborations to secure data sharing in finance, these cryptographic tools have ushered in a new era of data protection and informed decision-making, all without violating individual privacy. As we peer into the future, privacy-preserving deep learning remains at the forefront of technological progress. Its impact is profound, redefining the way organizations manage and analyze data, from hospitals enhancing patient care to financial institutions combating fraud. The journey towards preserving privacy while reaping the benefits of AI is ongoing, but the path is well-lit and brimming with potential.

In conclusion, privacy-preserving deep learning is not merely a concept; it's a tangible reality with the power to transform industries and protect the individual in our data-rich world. It is a testament to human ingenuity, where privacy and progress coexist harmoniously. As we embrace these techniques and frameworks, we embark on a collective mission to safeguard data and advance the frontiers of AI. The journey is just beginning, and the possibilities are limitless. The future is one where data is secure, AI is transformative, and privacy is preserved.

REFERENCES

[1] S. Chow, Y. He, and et al. Spice- simple privacy-preserving identity management for cloud environment. In ACNS 2012, volume 7341 of Lecture Notes in Computer Science. Springer, 2012.

[2] Privacy Preserving Back-Propagation Neural Network Learning Made Practical with Cloud Computing. IEEE Transactions on Parallel and Distributed Systems, Vol. 25, No. 1, January 2014.

[3] N. Schlitter, A Protocol for Privacy Preserving Neural Network Learning on Horizontal Partitioned Data, Proc. Privacy Statistics in Databases (PSD 08), Sept. 2008

[4] Erich Schikuta and Erwin Mann, N2Sky- Neural Networks as Services in the Clouds. arXiv:1401.2468v1 [cs.NE] 10 Jan 2014.

[5] T. Chen and S. Zhong,Privacy-Preserving Backpropagation Neural Network Learning, IEEE Trans. Neural Network, vol. 20, no. 10, pp. 1554-1564, Oct. 2009.

[6] Mohammad Ali Kadampur, Somayajulu D.V.L.N. A Noise Addition Scheme in Decision Tree for Privacy Preserving Data Mining, Journal of Computing, Volumen 2, Issue 1, January 2010, ISSN 2151-9617

[7] Yong Liu, Yeming Xiao, Li Wang, Jielin Pan, Yonghong Yan. Parallel Implementation of Neural Networks Training on Graphic Processing Unit, 2012 5th International Conference on BioMedical Engineering and Informatics (BMEI 2012)

[8] PelinAngin, Bharat Bhargava, Rohit Ranchal, Noopur Singh. An Entitycentric Approach for Privacy and Identity Management in Cloud Computing, 2010 29th IEEE International Symposium on Reliable Distributed Systems.

[9] Scretan J, Georgiopoulos, M. A privacy preserving probabilistic neural network for horizontally partitioned databases. International Joint Conference on Neural Networks. Aug 2007. 70

[10] Barni M, Failla P, Sadeghi A. Privacy Preserving ECG Classification with branching programs and neural networks.IEEE Transaction. Information Forensics and Security. Volume 6, Issue 2, June 2011.

[11] Samet S. Privacy Preserving protocols for perceptron learning algorithm in neural networks. IEEE Conference on Intelligent Systems, Sept 2008.

[12] Mahmoud Barhamgi, Arosha K. Bandara, and Yijun Yu, Protecting Privacy in the Cloud: Current Practices, Future Directions, Computer IEEE Society February 2016.

[13] Majid Bashir Malik, A model for Privacy Preserving in Data Mining using Soft Computing Techniques. March 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom).

[14] Reza Shokri, Privacy-Preserving Deep Learning,, 2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton) Oct 2015.

[15] Nathan Dowlin, Ran Gilad-Bachrach, Kim Laine, Kristin Lauter, Michael Naehrig and John Wernsing, CryptoNets: Applying Neural Networks to Encrypted Data with High Throughput and Accuracy 29 December 2015

[16] Ryan Hayward, Chia-Chu Chiang, Parallelizing fully homomorphic encryption for a cloud environment. Journal of Applied Research and Technology 13 (2015) 245-252

[17] Bengio. Learning deep architectures for AI. Foundations and trends in machine learning, 2(1):1–127, 2009.

[18] L. Deng. A tutorial survey of architectures, algorithms, and applications for deep learning. APSIPA Trans. Signal and Information Processing, 3, 2014.
[19] A. Graves, A.-R. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In ICASSP, 2013.
[20] Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, et al. Deepspeech: Scaling up end-to-end speech recognition. arXiv:1412.5567 , 2014.
[21] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, et al. Deep neural networks for acoustic 71modeling in speech recognition: The shared views of four research groups. Signal Processing Magazine, 29(6):82–97, 2012.
[22] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, 2012.
[23] P. Simard, D. Steinkraus, and J. Platt. Best practices for convolutional neural networks applied to visual document analysis. In Document Analysis and Recognition, 2013.
[24] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In CVPR, 2014.
[25] Angel Yu, Wai Lok Lai, James Pay or Efficient Integer Vector Homomorphic Encryption, May 2015.
[26] PyTorch [Online]. Available: https://pytorch.org/
[27] PySyft. [Online]. Available: https://github.com/OpenMined/PySyft
[28] Karthiban, K., and S. Smys. Privacy preserving approaches in cloud computing; In 2018 2nd International Conference on Inventive Systems and Control (ICISC), pp. 462-467. IEEE, 2018.
[29] A generic framework for privacy preserving deep learning, Theo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert, Jonathan Passerat-Palmbach, 13 Nov 2018.
[30] Mohammad Al-Rubaie, Privacy Preserving Machine Learning: Threats and Solutions at IEEE Security and Privacy Magazine, 2018.
[31] UCI Machine Learning Heart https://archive.ics.uci.edu/ml/datasets/heart+disease Disease
[32] HElib [Online]. Available: https://github.com/homenc/HElib Dataset:
[33] Brian Knott, Shobha Venkataraman, AwniHannun, Shubho Sengupta, Mark Ibrahim, Laurens van der Maaten, CRYPTEN: Secure Multi-Party Computation Meets Machine Learning 35th Conference on Neural Information Processing Systems (NeurIPS 2021) 72
[34] Xu, K., Zhu, W. and Darve, E. Distributed machine learning for computational engineering using MPI. Preprint at arXiv https://arxiv.org/abs/2011.01349 (2020)
[35] A. Benaissa, B. Retiat, B. Cebere, A.E. Belfedhal, "TenSEAL: A Library for Encrypted Tensor Operations Using Homomorphic Encryption", ICLR 2021 Workshop on Distributed and Private Machine Learning.
[36] OpenMined Url: https://github.com/OpenMined/
[37] IBM HeLIB: https://github.com/homenc/HElib