Metodi di crittografia omomorfica per la protezione della privacy nell'analisi dei dati

Tesi di Laurea Magistrale in Ingegneria Informatica

Ignazio Daniele Di Blasi

Relatore: Prof. Giuseppe Lo Re

Correlatori: Ing. Pierluca Ferraro

UNIVERSITÀ DEGLI STUDI DI PALERMO

DIPARTIMENTO DI INGEGNERIA

LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

Metodi di crittografia omomorfica per la protezione della privacy nell'analisi dei dati

Tesi di Laurea di

Ignazio Daniele Di Blasi

Relatore:

Ch.mo Prof. Giuseppe Lo Re

Correlatore:

Prof. Pierluca Ferraro

Sommario

Il machine learning consente ai sistemi informatici di apprendere dai dati e di prendere decisioni in modo autonomo, con applicazioni in numerosi settori, dalla finanza alla medicina, dalla produzione industriale alla sicurezza informatica. Tuttavia, il suo utilizzo comporta significativi rischi per la sicurezza e la privacy dei dati. Per addestrare i modelli, infatti, è necessario accedere a grandi quantità di dati personali, esponendo potenzialmente informazioni sensibili a violazioni della privacy. Anche i modelli stessi possono essere vulnerabili ad attacchi avversari, in cui i dati di input vengono manipolati per ingannare il modello, compromettendo la sicurezza di sistemi cruciali. Inoltre, esiste il rischio che i modelli rivelino informazioni confidenziali attraverso le loro previsioni.

Le contromisure tradizionali, come l'anonimizzazione dei dati e la crittografia, sono strumenti essenziali per proteggere la privacy, ma non sempre risultano sufficienti. L'anonimizzazione può essere aggirata, specialmente se i dati anonimi vengono combinati con altre fonti di informazione, e la crittografia tradizionale richiede che i dati siano decifrati prima dell'elaborazione,

creando potenziali vulnerabilità durante il processo di calcolo. Per affrontare queste sfide, la crittografia omomorfica offre una soluzione avanzata, consentendo di eseguire calcoli su dati cifrati e garantendo una maggiore protezione della privacy.

In questa tesi verrà proposto un sistema di analisi dei dati che utilizza la crittografia omomorfica per garantire la riservatezza dei dati durante tutte le fasi di addestramento e predizione. Verranno analizzate le tecniche di crittografia omomorfica e le minacce specifiche legate al machine learning, inclusi i possibili attacchi e vulnerabilità. Inoltre, il sistema sarà verificato sperimentalmente utilizzando dataset rappresentativi e confrontato con le soluzioni presenti in letteratura, al fine di valutarne l'efficienza e la sicurezza.

Indice

Introduzione **4**

1 Homomorphic Encryption **7**

1.1 Partially Homomorphic Encryption 8

1.1.1 RSA 9

1.2 Somewhat Homomorphic Encryption 10

1.3 Fully Homomorphic Encryption 10

1.3.1 Ideal Lattice-Based 11

1.3.2 Over Integers 14

1.3.3 (R)LWE-based 16

1.3.4 NTRU-like 17

2 Privacy-Preserving Machine Learning **19**

2.1 Rischi per la sicurezza e la privacy nel Machine Learning 20

2.1.1 Memorizzazione dei dati 21

2.1.2 Attacchi di ricostruzione 21

2.1.3 Attacchi di inversione 22

2.1.4 Attacchi di inferenza sull'appartenenza 23

2.2 Tecniche di Privacy Preservation 23

2.2.1 Federated Learning 24

2.2.2 Secure Multi-Party Computation 25
2.2.3 Differential Privacy 25
2.2.4 Homomorphic Encryption 26

3 Analisi dei dati privacy-preserving con crittografia omomorfica 28
3.1 Schemi di riferimento 28
3.2 Librerie di riferimento 29
3.3 Reti Neurali 31
3.3.1 Neurone 31
3.3.2 Fully Connected Layer 33
3.3.3 Batch Normalization Layer 34
3.4 Soluzioni in letteratura 35
3.4.1 Nandakumar, Ratha, Pankanti e Halevi 35

4 HEDAS: Homomorphic Encrypted Data Analysis System 39
4.1 Architettura 39
4.2 Gestione delle chiavi 40
4.3 Codifica del testo 40
4.4 Implementazione del sistema 40
4.5 Rischi generali 40
4.5.1 Attacchi esterni 40
4.5.2 Attacchi interni 41
4.6 Caso studio: Credit Score 44
4.6.1 Rischi 45
4.7 Scenario di HEDAS 47

5 Valutazioni sperimentali 48
5.1 Parametro di sicurezza 48

5.2 Correttezza dell'implementazione 48
5.3 Prestazioni al variare del parametro di sicurezza 48
5.4 Confronto delle prestazioni della rete neurale proposta da Nandakumar 49
5.5 Caso studio: Credit Score 49

Conclusioni **50**

Elenco delle figure **52**

Bibliografia **53**

Introduzione

Il machine learning, o apprendimento automatico, è uno dei campi di ricerca più rivoluzionari del nostro tempo. Si tratta di un ramo dell'intelligenza artificiale che permette ai sistemi informatici di apprendere dai dati e migliorare le proprie prestazioni senza essere esplicitamente programmati per farlo. Questa capacità di "imparare" dai dati ha aperto la strada a innovazioni che stanno cambiando profondamente il nostro mondo.

In effetti, i progressi nel campo del machine learning hanno potenziato e trasformato anche i sistemi di analisi dei dati, rendendoli strumenti essenziali in numerosi settori, dalla finanza alla medicina, dalla produzione industriale alla sicurezza informatica. Ad esempio, nel campo della finanza, il machine learning è alla base di sistemi di trading automatizzato che possono analizzare enormi volumi di dati di mercato in tempo reale e prendere decisioni di investimento in una frazione di secondo.

I sistemi di analisi dei dati sono anche al centro delle tecnologie che si usano quotidianamente. Questi sistemi apprendono dalle preferenze e dai comportamenti degli utenti per offrire servizi sempre più personalizzati, come nel caso degli assistenti vocali come Siri o Alexa, dei suggerimenti di film su Netflix o di prodotti su Amazon, e dei filtri antispam nelle e-mail.

Ma le tecnologie legate all'intelligenza artificiale e all'apprendimento automatico non si limitano a migliorare i prodotti e i servizi esistenti; stanno anche aprendo nuove possibilità in aree che solo pochi anni fa sembravano fantascienza. Si pensi alle auto a guida autonoma, che utilizzano algoritmi di apprendimento automatico per riconoscere oggetti sulla strada, prevedere il comportamento degli altri veicoli e prendere decisioni di guida in tempo reale. Oppure si

consideri il campo della linguistica computazionale, dove il machine learning è alla base dei modelli di linguaggio come ChatGPT, che possono comprendere e generare testo in modo naturale e coerente.

Man mano che la tecnologia continua a evolversi, si vedranno applicazioni sempre più sofisticate e diffuse. Si potrebbe assistere alla nascita di sistemi capaci di apprendere non solo dai dati ma anche dall'esperienza diretta, simili all'apprendimento umano. Questo potrebbe portare a macchine ancora più intelligenti, in grado di affrontare problemi complessi in ambiti come la scienza, l'etica e l'arte.

Nonostante i numerosi benefici introdotti dalle tecniche di apprendimento automatico e dai sistemi di analisi dei dati, emergono preoccupazioni significative in merito alla privacy e alla sicurezza dei dati. Gli algoritmi di apprendimento automatico, per essere efficaci, richiedono l'accesso a grandi quantità di dati, spesso contenenti informazioni personali e sensibili. Questo solleva il rischio che tali dati possano essere utilizzati in modo non autorizzato o cadere vittima di abusi, mettendo in pericolo la privacy degli individui.

Inoltre, i sistemi di analisi dei dati non sono immuni da attacchi informatici. È possibile manipolare i dati di input o sfruttare vulnerabilità negli algoritmi per indurre errori nelle decisioni automatizzate, con potenziali conseguenze gravi in settori critici. Per mitigare questi rischi, è fondamentale adottare misure di protezione avanzate, garantire una gestione etica dei dati e sviluppare protocolli di sicurezza rigorosi che tutelino sia la privacy che l'integrità dei sistemi.

Un'area di ricerca promettente è l'integrazione dell'analisi dei dati con la crittografia omomorfica. Questa tecnologia consente di eseguire calcoli su dati cifrati senza doverli decifrare, garantendo così che le informazioni sensibili rimangano protette durante l'intero processo di elaborazione.

L'obiettivo della tesi è studiare l'efficacia e l'applicabilità del machine learning in un contesto reale utilizzando la crittografia omomorfica per preservare la riservatezza dei dati. Sviluppare tecniche di analisi dei dati che utilizzano la crittografia omomorfica rappresenta un passo avanti significativo per garantire che i benefici di questa tecnologia possano essere sfruttati senza com-

promettere la riservatezza delle informazioni, risolvendo uno dei principali ostacoli all'adozione su larga scala in ambiti sensibili.

A tal fine, verrà proposto un sistema chiamato HEDAS (Homomorphic Encrypted Data Analysis System), che utilizza la crittografia omomorfica per proteggere i dati durante tutte le fasi del processo di analisi, dall'addestramento alla predizione. Saranno inoltre esplorate le tecniche crittografiche più adatte e le possibili minacce specifiche legate al machine learning. Il sistema sarà poi testato sperimentalmente su dataset rappresentativi e confrontato con soluzioni esistenti per valutarne l'efficienza e la sicurezza.

La tesi è suddivisa nei seguenti capitoli.

Nel capitolo 1 si esamina l'evoluzione della crittografia omomorfica con particolare attenzione ai progressi teorici e pratici che hanno reso possibile l'implementazione moderna di questo potente strumento per la protezione dei dati.

Nel capitolo 2 vengono esplorati i rischi legati all'applicazione del machine learning, come gli attacchi avversari e le vulnerabilità della privacy, e analizzate le possibili contromisure.

Nel capitolo 3 si analizza lo stato dell'arte del Privacy-Preserving Machine Learning, esaminando come la crittografia omomorfica venga utilizzata per preservare la riservatezza dei dati durante le fasi di addestramento e di predizione, valutando le soluzioni più recenti e innovative nel campo.

Nel capitolo 4 si presenta un'analisi dell'architettura e dell'implementazione di HEDAS, evidenziando i dettagli tecnici del sistema, le sue funzionalità principali e le misure di sicurezza adottate per garantire la protezione dei dati durante l'elaborazione.

Nel capitolo 5 si verifica la correttezza di HEDAS e si analizzano le sue prestazioni computazionali, valutando l'efficienza delle operazioni crittografiche e le implicazioni in termini di tempi di calcolo e scalabilità per applicazioni reali.

Capitolo 1

Homomorphic Encryption

In greco antico, il termine "ὁμός" (homos) significa "uguale", mentre "μορφή" (morphe) significa "forma". Nel contesto della crittografia, il concetto di omomorfismo viene utilizzato come una forma di cifratura.

Homomorphic Encryption (HE) è uno schema crittografico che consente di eseguire determinate operazioni computazionali su dati cifrati, mantenendo inalterate le caratteristiche della funzione e il formato dei dati stessi. Ad esempio, con la cifratura tradizionale, per sommare due messaggi cifrati $E(m_1)$ e $E(m_2)$, è necessario decifrarli, eseguire l'operazione $m_1 + m_2$ e poi cifrare nuovamente il risultato. Al contrario, con uno schema di cifratura omomorfica additiva, è possibile ottenere direttamente $E(m_1 + m_2)$ senza dover decifrare i messaggi, preservando così la riservatezza del loro contenuto [1].

Uno schema di cifratura è omomorfico rispetto ad un'operazione "$\star$" se esiste un'operazione "$\diamond$" sui testi cifrati tale che:

$$E(m_1) \diamond E(m_2) = E(m_1 \star m_2), \quad \forall m_1, m_2 \in M,$$

dove E è l'algoritmo di cifratura e M è l'insieme di tutti i possibili messaggi.

Questo schema crittografico rivoluziona il cloud computing rendendolo adatto anche per

applicazioni che richiedono la totale riservatezza dei dati, poiché impedisce al fornitore del servizio e a eventuali terzi di accedere a qualsiasi informazione sensibile.

1.1 Partially Homomorphic Encryption

Gli schemi crittografici Partially Homomorphic Encryption (PHE) permettono di eseguire operazioni algebriche sui dati cifrati senza doverli prima decifrare, ma con la limitazione che supportano solo una specifica operazione, come l'addizione o la moltiplicazione. Questa limitazione rende gli schemi PHE molto efficienti, poiché richiedono poche risorse computazionali e sono semplici da implementare. La loro efficienza li rende pratici per applicazioni specifiche, come i sistemi di pagamento, dove è necessario eseguire operazioni semplici sui dati senza rivelare il contenuto originale.

RSA è uno dei più noti schemi crittografici e, pur essendo principalmente utilizzato per la cifratura standard, può essere considerato un esempio di schema PHE per operazioni moltiplicative. Altri schemi PHE sono riportati nella Tabella 1.1 con relativa operazione omomorfica supportata.

	Operazione omomorfica	
Schema	**+**	*****
RSA (1978)		✓
GM (1982)	✓	
El-Gamal (1985)		✓
Benaloh (1994)	✓	
Paillier (1999)	✓	

Tabella 1.1: Schemi PHE. Tabella tratta da [1].

Gli schemi PHE offrono un buon livello di sicurezza, basato sulla difficoltà di risolvere problemi matematici complessi senza la conoscenza della chiave privata, come la fattorizzazione di grandi numeri primi o il logaritmo discreto. Tuttavia, la loro limitata versatilità operativa li rende inadatti a scenari dove è richiesta una varietà di operazioni sui dati cifrati.

1.1.1 RSA

RSA rappresenta uno dei primi esempi di sistemi di crittografia omomorfica e fu introdotto da Rivest, Shamir e Adleman [2] poco dopo l'invenzione della crittografia a chiave pubblica da parte di Diffie e Hellman [3].

RSA è definito come segue:

- **Generazione delle chiavi**:
  - Generare due numeri primi p e q.
  - Calcolare $n = pq$.
  - Calcolare $\phi = (p-1)(q-1)$.
  - Scegliere un intero e tale che $\gcd(e, \phi) = 1$.
  - Calcolare $d = e^{-1} \bmod \phi$.
  - Distribuire la chiave pubblica (e, n).
  - Mantenere riservata la chiave privata (d, n).
- **Cifratura**:
  - Convertire il messaggio da cifrare in testo in chiaro m tale che $0 \leq m < n$.
  - Calcolare il testo cifrato $c = m^e \bmod n$.
- **Decifratura**:
  - Recuperare il testo in chiaro $m = c^d \bmod n$
- **Proprietà omomorfica**:
  - Per due testi in chiaro m_1 e m_2:

$$E(m_1) \cdot E(m_2) = (m_1^e \bmod n) \cdot (m_2^e \bmod n) = (m_1 \cdot m_2)^e \bmod n = E(m_1 \cdot m_2)$$

1.2 Somewhat Homomorphic Encryption

Gli schemi crittografici Somewhat Homomorphic Encryption (SWHE) consentono di eseguire operazioni additive e moltiplicative su dati cifrati, ma solo per un numero limitato di volte prima che sia necessario decifrare i dati e procedere a una nuova cifratura per eseguire ulteriori operazioni.

Polly Cracker [4] è uno dei primi schemi SWHE e consente di eseguire sia operazioni additive che moltiplicative su dati cifrati. Tuttavia, la dimensione del testo cifrato cresce esponenzialmente con l'esecuzione delle operazioni, rendendo lo schema meno pratico per calcoli complessi. In particolare, l'operazione moltiplicativa è particolarmente onerosa in termini di risorse computazionali. Sebbene siano state proposte varianti più efficienti nel corso degli anni [5], molte di esse si sono rivelate vulnerabili ad attacchi crittografici [6], limitando l'applicabilità pratica di questo approccio.

Lo schema crittografico BGN, introdotto da Boneh, Goh e Nissim [7], rappresenta un passo significativo verso gli schemi FHE, poiché consente di eseguire un numero arbitrario di operazioni additive e una sola operazione moltiplicativa su dati cifrati. Una caratteristica distintiva di BGN è che la dimensione del testo cifrato rimane costante, indipendentemente dal numero di operazioni additive eseguite. Questa proprietà lo rende particolarmente efficiente per applicazioni che richiedono molteplici operazioni sui dati cifrati, pur mantenendo il limite di una singola moltiplicazione. La capacità di eseguire entrambe le operazioni aritmetiche conferisce una maggiore flessibilità, ma comporta anche una maggiore complessità e una minore efficienza rispetto agli schemi PHE [8].

1.3 Fully Homomorphic Encryption

Gli schemi crittografici Fully Homomorphic Encryption (FHE) consentono di eseguire un numero illimitato di operazioni omomorfiche su dati cifrati. La prima proposta teorica praticabile di uno schema FHE è stata introdotta da Craig Gentry nel suo lavoro di dottorato nel 2009 [9].

Il lavoro di Gentry non solo ha fornito un schema FHE praticabile dal punto di vista teorico, ma ha anche offerto un quadro generale per la formulazione e lo sviluppo di schemi FHE futuri.

Sebbene lo schema FHE ideale basato su reticoli proposto da Gentry sia molto promettente, presenta anche molti ostacoli, come i costi computazionali in termini di applicabilità nella vita reale, e alcuni dei suoi concetti matematici avanzati lo rendono complesso e difficile da implementare. Pertanto, molti nuovi schemi e ottimizzazioni hanno seguito il suo lavoro per affrontare i suddetti ostacoli.

Di seguito sono illustrati i principali tipi di schemi FHE:

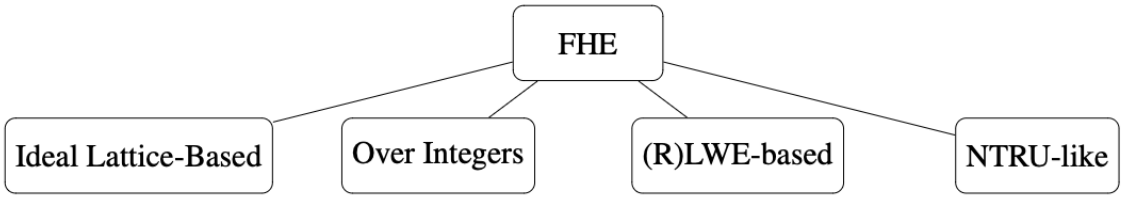


Figura 1.1: Evoluzioni degli schemi FHE. Immagine tratta da [1].

1.3.1 Ideal Lattice-Based

Un reticolo è la combinazione lineare di vettori indipendenti $b_1, b_2, \ldots, b_n$ (vettori base del reticolo).

$$L = \sum_{i=1}^{n} \vec{b_i} * v_i, \quad v_i \in \mathbb{Z}$$

Una base è definita "buona" o "ottimale" se è composta da vettori indipendenti corti e quasi ortogonali, altrimenti è definita "cattiva" o "non ottimale", come riportato in Figura 1.2.

Due problemi fondamentali nella teoria dei reticoli, e nella crittografia basata su di essi, sono Closest Vector Problem (CVP) e Shortest Vector Problem (SVP).

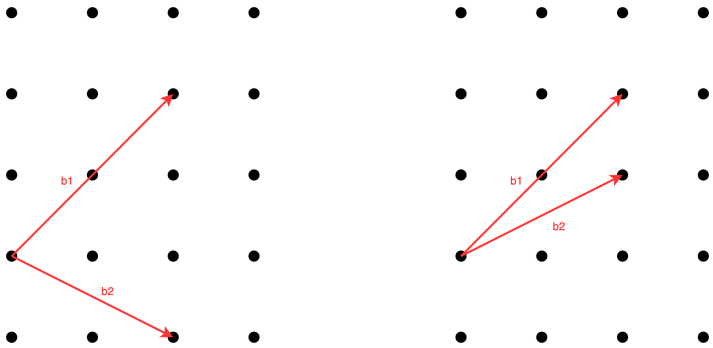


Figura 1.2: Esempio di una base ottimale e di una base non ottimale.

CVP

Dato un reticolo L e un vettore $\vec{t}$, il problema del vettore più vicino (CVP) riguarda la ricerca del vettore $\vec{v} \in L$ tale che $\|\vec{v} - \vec{t}\|$ sia minimo.

SVP

Dato un reticolo L, il problema del vettore più corto (SVP) riguarda la ricerca del vettore $\vec{v} \in L$ non nullo tale che $\|\vec{v}\|$ sia minimo.

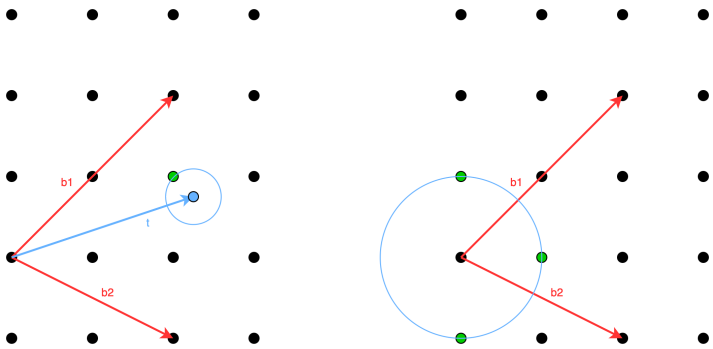


Figura 1.3: Esempio di problemi CVP e SVP.

Dato un reticolo L e una base $b_1, b_2, \ldots, b_n$, il problema di riduzione del reticolo consiste nel trovare una base ottimale che renda semplici i problemi computazionali sui reticoli, infatti la definizione di base "ottimale" e base "non ottimale" si basa sulla difficoltà di risolvere problemi

computazionali sui reticoli. Nel caso di una base ottimale, i problemi CVP e SVP possono essere risolti in tempo polinomiale, mentre nel caso di una base non ottimale, i migliori algoritmi conosciuti risolvono questi problemi solo in tempo esponenziale.

Il lavoro di Gentry inizia con uno schema SWHE basato su reticoli, dove la sicurezza del sistema si fonda sulla difficoltà del problema di riduzione del reticolo. Gentry introduce una quantità controllata di rumore nel testo cifrato, la cui gestione è fondamentale per mantenere la sicurezza e la funzionalità dello schema. Dopo un certo numero di operazioni omomorfiche, il rumore accumulato nel testo cifrato può raggiungere una soglia critica oltre la quale la funzione di decifratura non riesce più a recuperare correttamente il messaggio originale. Per affrontare questo problema, Gentry ha sviluppato due tecniche innovative, chiamate "squashing" e "bootstrapping", che sono fondamentali per trasformare uno schema SWHE in uno schema FHE.

Squashing

La tecnica di "squashing" è stata progettata per ridurre la complessità del processo di decifratura e la quantità di rumore nel testo cifrato. Attraverso questa tecnica, il circuito di decifratura viene semplificato, riducendo la profondità e quindi la crescita del rumore permettendo di gestire un numero maggiore di operazioni omomorfiche prima che il rumore diventi un problema.

Bootstrapping

Il bootstrapping è una componente fondamentale degli schemi FHE. È stato progettato per ridurre il rumore accumulato nel testo cifrato, permettendo l'esecuzione di un numero illimitato di operazioni omomorfiche senza compromettere l'integrità del risultato.

Durante l'esecuzione di operazioni omomorfiche, il rumore, che è una parte intrinseca della cifratura, aumenta progressivamente. Se il rumore supera una certa soglia, diventa impossibile ottenere risultati corretti dalle operazioni omomorfiche. Il bootstrapping risolve questo problema attraverso un processo di "ricifratura".

Il processo di bootstrap consiste nell'eseguire la funzione di decifratura omomorfica sul testo cifrato. Questa operazione non rivela il testo in chiaro, ma elimina una porzione significativa del rumore accumulato. Successivamente, viene eseguita una nuova cifratura omomorfica che produce un nuovo testo cifrato con una quantità di rumore significativamente ridotta. In questo modo, si può ripristinare la capacità del sistema di eseguire ulteriori operazioni omomorfiche. Il processo di bootstrap è illustrato in Figura 1.4.

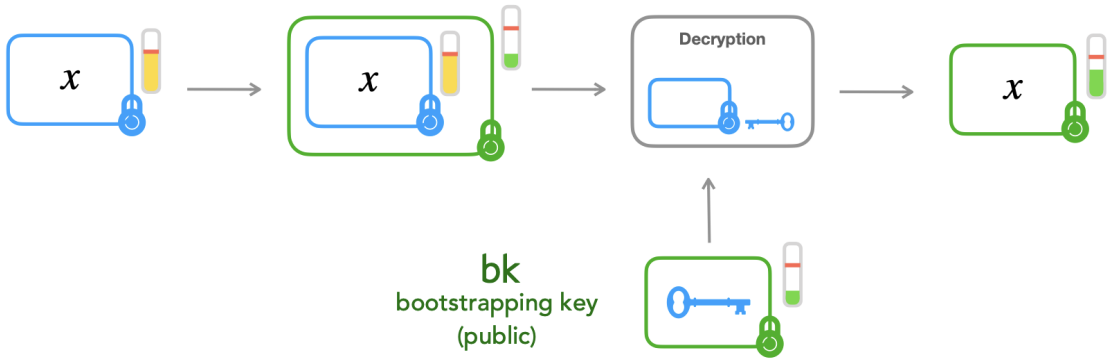


Figura 1.4: Processo di bootstrap.

1.3.2 Over Integers

Lo schema DGHV basato su interi è stato introdotto da Dijk, Gentry, Halevi e Vaikuntanathan [10] e la sua sicurezza si basa su Sparse Subset Sum Problem (SSSP) e su Approximate Greatest Common Divisor (AGCD).

Sparse Subset Sum Problem

Dato un insieme $A = \{a_1, a_2, \ldots, a_n \mid a_i \in \mathbb{Z}, i = 1, 2, \ldots, n\}$ e un obiettivo $t \in \mathbb{Z}$, il problema SSSP consiste nel trovare $\bar{A} = \{\bar{a_1}, \bar{a_2}, \ldots, \bar{a_k}\} \subseteq A$ tale che

$$\sum_{i=1}^{k} \bar{a_i} = t$$

Approximate Greatest Common Divisor

Dato un insieme $A = \{a_1, a_2, \ldots, a_n \mid a_i \in \mathbb{Z}, i = 1, 2, \ldots, n\}$ e un numero $p \in \mathbb{Z}$, il problema AGCD consiste nel trovare un valore $d \in \mathbb{Z}$ tale che

$$\left| x_i - d \cdot \left\lfloor \frac{x_i}{d} \right\rfloor \right| \leq \varepsilon$$

Lo schema DGHV è uno dei primi e più semplici schemi FHE. Tuttavia, questa semplicità si traduce in una maggiore complessità computazionale e in una minore efficienza operativa. Le operazioni di cifratura e decifratura richiedono numerosi calcoli aritmetici su numeri grandi, rendendo lo schema meno pratico per applicazioni reali rispetto a soluzioni FHE più moderne e ottimizzate.

DGHV è definito come segue:

- **Generazione delle chiavi**:
  - Generare un numero dispari p intero (chiave privata).
  - Generare un insieme di valori $q_i, r_i,\ i = 1, 2, \ldots, n$.
  - Generare un insieme di valori $\{x_0, x_1, \ldots, x_n\}$ interi (chiave pubblica) dove $x_0 > x_i$ e $x_i = pq_i + r_i,\ i = 1, 2, \ldots, n$.
- **Cifratura**:
  - Il messaggio m deve essere contenuto in $\mathbb{F}_2$
  - Generare un numero $r \in \mathbb{Z}$.
  - Calcolare il testo cifrato c come segue:

    $$c = (m + 2r + 2\sum_{i \in S} x_i) \bmod x_0$$

    dove S è un sottoinsieme di $\{1, 2, \ldots, n\}$.

- **Decifratura**:
  - Recuperare il testo in chiaro $m = (c \bmod p) \bmod 2$

1.3.3 (R)LWE-based

Gli schemi FHE basati su Learning With Errors (LWE) e Ring Learning With Errors (RLWE) sono fondamentali per la crittografia omomorfica moderna. Entrambi si basano sulla difficoltà di risolvere problemi di algebra lineare con errori introdotti appositamente.

Uno dei primi schemi basati su LWE è stato proposto da Gentry, Sahai e Waters [11]. Lo schema GSW propone un approccio diverso per eseguire operazioni omomorfiche, utilizzando il metodo degli autovettori approssimati. Questa tecnica riduce la crescita dell'errore introdotto dalle moltiplicazioni omomorfiche a un piccolo fattore polinomiale. Tuttavia, uno dei principali svantaggi degli schemi basati su LWE è l'inefficienza pratica, poiché producono testi cifrati di grandi dimensioni e richiedono una significativa potenza computazionale per eseguire operazioni omomorfiche.

Il Ring Learning With Errors (RLWE) è una generalizzazione del problema LWE in cui le operazioni vengono eseguite all'interno di anelli polinomiali anziché in spazi vettoriali. Questa struttura consente una rappresentazione più compatta dei testi cifrati e operazioni aritmetiche più efficienti, riducendo così sia la dimensione del testo cifrato che la complessità computazionale.

Nel 2017, Cheon, Kim, Kim e Song hanno proposto un nuovo schema RLWE, noto come schema CKKS [12]. Una delle caratteristiche distintive dello schema CKKS è la sua capacità di eseguire operazioni su numeri reali approssimati, che vengono rappresentati attraverso numeri complessi. Questo approccio è essenziale per supportare le operazioni sui numeri in virgola mobile e, sebbene non offra una precisione assoluta, fornisce una rappresentazione approssimativa sufficiente per molte applicazioni pratiche.

Nel 2018, lo schema CKKS è stato notevolmente migliorato grazie all'introduzione di tecniche avanzate di ottimizzazione, come il packing basato sul Chinese Remainder Theorem (CRT).

Questa tecnica consente di raggruppare più valori numerici all'interno di un singolo testo cifrato. Il packing migliora l'efficienza permettendo di eseguire operazioni in parallelo su più dati cifrati e riducendo lo spazio necessario per la loro memorizzazione. Questo rende lo schema particolarmente efficiente per applicazioni come il machine learning, dove l'elaborazione di grandi volumi di dati è essenziale.

CKKS è stato soggetto a innumerevoli studi, e nonostante l'intensa analisi da parte di esperti di crittografia, è stata individuata una sola vulnerabilità. Nel 2020, Li e Micciancio hanno proposto un attacco basato su tecniche di algebra lineare e riduzione su reticoli, che potrebbe consentire a un avversario con accesso a numerose cifrature e alla funzione di decifratura di dedurre la chiave privata [13]. Tuttavia, questa vulnerabilità può essere facilmente mitigata attraverso strategie preventive, come evitare di condividere risultati decifrati o inserire piccoli errori casuali durante il processo di decifratura. Inoltre, questa vulnerabilità non si presenta in tutte le situazioni, ma solo in contesti molto specifici, il che conferma l'elevata resistenza dello schema.

1.3.4 NTRU-like

Lo schema NTRU (N-th degree Truncated Polynomial Ring Units) è stato introdotto da Hoffstein, Pipher e Silverman nel 1998 [14]. NTRU è uno schema crittografico basato su reticoli e, nel corso degli anni, ha subito diverse modifiche, tra cui una variante basata sull'assunzione RLWE, e ha portato all'introduzione dello schema MultiKey FHE (MKFHE) [15].

La sicurezza di questo schema si basa su tre componenti principali: la sicurezza circolare, il problema RLWE, e il problema del rapporto dei piccoli polinomi decisionali (DSPR). Il problema DSPR afferma che è difficile distinguere tra h (come definito nella costruzione dello schema) e polinomi casuali uniformi in R_q.

Tuttavia, vari studi [16] [17] hanno dimostrato che esiste un attacco a sottocampo di reticolo che rende insicuro qualsiasi schema tipo NTRU basato sul problema DSPR per alcune scelte particolari di parametri. Per garantire la sicurezza di uno schema NTRU, è quindi necessario

aumentare significativamente i parametri rispetto alle dimensioni proposte prima della pubblicazione di questi attacchi. Questo incremento nei parametri ha reso gli schemi basati su NTRU significativamente meno efficienti rispetto ad alternative basate su RLWE e, di conseguenza, tali schemi non sono più utilizzati né supportati da alcuna libreria di crittografia moderna.

Capitolo 2

Privacy-Preserving Machine Learning

Gli algoritmi di machine learning basati su reti neurali hanno rivoluzionato il campo dell'intelligenza artificiale, diventando il fulcro della ricerca contemporanea in questo ambito. Queste tecniche avanzate hanno mostrato una capacità straordinaria di elaborare e analizzare grandi quantità di dati, raggiungendo risultati senza precedenti in numerosi settori applicativi. Tra questi, si possono annoverare il rilevamento dello spam, l'analisi del traffico, il rilevamento delle intrusioni nei sistemi informatici, le previsioni in ambito medico, il riconoscimento facciale e le previsioni finanziarie. Grazie alla loro capacità di apprendere da grandi volumi di dati, le reti neurali stanno trasformando il modo in cui si affrontano problemi complessi che richiedono un'elevata capacità di generalizzazione.

L'addestramento di modelli di machine learning, tuttavia, presenta sfide significative [18], in particolare per quanto riguarda la privacy. Per ottenere prestazioni elevate, questi modelli richiedono l'accesso ai dati grezzi, i quali spesso contengono informazioni sensibili che, se non gestite correttamente, possono esporre gli individui o le organizzazioni a rischi di violazione della privacy. La protezione di questi dati durante le fasi di addestramento e di predizione è diventata quindi una preoccupazione cruciale nel contesto della sicurezza informatica e dell'etica dell'intelligenza artificiale.

Con la crescita esponenziale dei servizi cloud, i sistemi di analisi dei dati sono sempre più

spesso eseguiti su infrastrutture cloud. Questi ambienti offrono vantaggi significativi, tra cui la scalabilità e la possibilità di ridurre i costi operativi, poiché le risorse computazionali necessarie per l'addestramento e il deployment dei modelli sono gestite da fornitori di servizi cloud. In questo contesto, emerge il concetto di Machine Learning as a Service (MLaaS) [19], un paradigma che consente alle organizzazioni di sfruttare i potenti strumenti di machine learning senza dover investire in infrastrutture hardware costose o gestire complessi processi di manutenzione.

Il MLaaS offre numerosi servizi pronti all'uso che permettono agli utenti di addestrare modelli, eseguire analisi predittive e implementare soluzioni AI con facilità. Questi servizi si sono diffusi rapidamente grazie alla loro capacità di rendere accessibile l'intelligenza artificiale a una vasta gamma di utenti, dalle piccole imprese alle grandi multinazionali, senza richiedere competenze tecniche approfondite.

Gli algoritmi di machine learning generalmente si articolano in due fasi principali. La prima è la fase di addestramento, in cui l'algoritmo apprende un modello basato su un insieme di dati etichettati. Questo modello è il risultato di un processo iterativo che ottimizza la capacità del modello di generalizzare a partire dai dati di addestramento. La seconda fase è quella di classificazione, in cui il modello addestrato viene utilizzato per fare previsioni su nuovi dati. Il risultato di questa fase è una predizione che può essere una classe, una probabilità o un valore continuo, a seconda del tipo di problema affrontato.

2.1 Rischi per la sicurezza e la privacy nel Machine Learning

Un aspetto fondamentale che deve essere considerato, soprattutto in applicazioni che trattano dati sensibili, è la riservatezza delle informazioni coinvolte. È essenziale che i dati di addestramento e il modello rimangano protetti e accessibili solo alle parti autorizzate. Questa necessità ha stimolato la ricerca in tecniche di machine learning sicuro, come l'apprendimento federato e

la crittografia omomorfica, che permettono di addestrare modelli senza mai esporre direttamente i dati grezzi a potenziali minacce.

2.1.1 Memorizzazione dei dati

Nel caso in cui il proprietario dei dati sia distinto dal soggetto che esegue i calcoli, anche se i dati vengono trasferiti utilizzando un canale sicuro e in forma cifrata, è necessario che il server che esegue i calcoli disponga della chiave di decifratura per poter elaborare i dati. Questo implica che, una volta decifrati, i dati siano temporaneamente accessibili in chiaro sul server, esponendoli così al provider del servizio di machine learning e aumentando i rischi di esposizione a potenziali vulnerabilità di sicurezza. La decifratura dei dati sul server consente l'analisi e l'elaborazione necessari, ma introduce un periodo in cui i dati sono visibili in chiaro, il che potrebbe renderli vulnerabili a minacce come accessi non autorizzati o attacchi informatici.

2.1.2 Attacchi di ricostruzione

Anche quando solo le caratteristiche estratte dai dati grezzi vengono trasferite e memorizzate sui server del soggetto che esegue i calcoli, si può incorrere in una minaccia significativa rappresentata dagli attacchi di ricostruzione. Questi attacchi mirano a ricostruire i dati privati originali a partire dalla conoscenza dei vettori di caratteristiche utilizzati dal modello di machine learning. Il rischio è particolarmente elevato quando i vettori di caratteristiche utilizzati nella fase di addestramento non vengono cancellati dopo la creazione del modello finale. Algoritmi come le Support Vector Machines (SVM) [20] [21] o i k-Nearest Neighbors (kNN) [22], che memorizzano direttamente i vettori di caratteristiche, possono essere particolarmente vulnerabili a questi tipi di attacchi.

Un esempio di attacco di ricostruzione riuscito è la ricostruzione di impronte digitali, dove un'immagine dell'impronta può essere ricostruita a partire da un modello che conserva le minuzie, ovvero le caratteristiche distintive dell'impronta [23]. Questo tipo di attacco dimostra

come un modello di machine learning, se non adeguatamente protetto, possa compromettere seriamente la sicurezza e la privacy dei dati.

Per ridurre il rischio di attacchi di ricostruzione, è importante evitare l'uso di modelli di machine learning che conservano esplicitamente i vettori di caratteristiche. Se tali modelli devono essere utilizzati, è essenziale garantire che i dati di addestramento non siano accessibili al soggetto che riceve i risultati del modello. Garantire una protezione adeguata durante la fase di utilizzo e conservazione dei dati è cruciale per prevenire la ricostruzione non autorizzata di informazioni sensibili.

2.1.3 Attacchi di inversione

Alcuni algoritmi di machine learning producono modelli in cui i vettori di caratteristiche espliciti non sono memorizzati, come avviene nelle reti neurali. In questi casi, la conoscenza dell'avversario è limitata a due principali modalità di accesso: "white-box" o "black-box".

Nel contesto di accesso white-box, l'avversario ha accesso al modello di machine learning stesso ma non ai vettori di caratteristiche memorizzati. Questo tipo di accesso è limitato alla struttura e ai parametri del modello, senza possibilità di ottenere informazioni dirette sui vettori di caratteristiche utilizzati. Nel contesto di accesso black-box, l'avversario interagisce con il modello solo attraverso le risposte restituite dal sistema quando invia nuovi campioni di test [24]. In questo caso, la conoscenza dell'avversario è limitata alle risposte del modello, come le probabilità di classificazione o i valori di decisione restituiti.

L'obiettivo degli attacchi di inversione del modello è ricreare vettori di caratteristiche che somigliano a quelli utilizzati per addestrare il modello di machine learning, utilizzando solo le predizioni. Questi attacchi sfruttano le informazioni di confidenza, come le probabilità di appartenenza a una classe o i valori di decisione forniti dal modello per i campioni di test. La tecnica implica la generazione di una media che rappresenta una certa classe, e può costituire una minaccia significativa per la privacy, soprattutto quando una classe corrisponde a un singolo individuo, come nel caso del riconoscimento facciale.

Un attacco di inversione del modello è stato dimostrato da Fredrikson, Ristenpart, Tech, Jha e Thomas [25], che include anche la ricostruzione dei dati, in quanto le caratteristiche ricostruite corrispondevano direttamente alle immagini facciali originali.

Per difendersi da tali attacchi, è necessario limitare l'accesso solo ai risultati del modello e restringere le informazioni restituite. Tuttavia, nel caso in cui si utilizzi un servizio MLaaS per addestrare una rete neurale o per ospitare un servizio di predizioni, non è possibile limitare il provider del servizio alle sole predizioni; esso avrà sempre un accesso in modalità white-box.

2.1.4 Attacchi di inferenza sull'appartenenza

Gli attacchi di inferenza sull'appartenenza mirano a determinare, avendo a disposizione un modello di machine learning e un campione specifico, se quest'ultimo è stato incluso nel set di addestramento. Questo obiettivo differisce da quello degli attacchi di inversione del modello, che non cercano di stabilire se un campione fosse presente nel set di addestramento e non producono un campione reale basandosi sull'output del modello.

Gli attacchi di inferenza sull'appartenenza sfruttano le differenze tra le previsioni del modello sui campioni presenti nel set di addestramento e quelli non inclusi. Sono stati sviluppati modelli di attacco che utilizzano come input l'etichetta corretta di un campione e la previsione fornita dal modello di machine learning [26]. Questi modelli di attacco sono stati addestrati con modelli ombra, generati da dati tramite tre metodi: attacco di inversione del modello, sintesi basata su statistiche, o dati reali rumorosi.

2.2 Tecniche di Privacy Preservation

Esistono diverse tecniche di preservazione della privacy che possono essere impiegate nell'ambito del machine learning. Tuttavia, non tutte le tecniche sono ugualmente adatte a ogni scenario, poiché ciascuna presenta specifici vantaggi e limitazioni a seconda del contesto di ap-

plicazione. Le garanzie crittografiche offerte da queste tecniche possono variare notevolmente, influenzando il livello di sicurezza e privacy che possono fornire.

2.2.1 Federated Learning

Federated Learning [27] è una tecnica utilizzata per l'addestramento di modelli di machine learning su dati decentralizzati, distribuiti tra più parti. L'obiettivo principale è addestrare un modello globale senza richiedere la condivisione diretta dei dati tra le parti coinvolte, preservando così la privacy [28]. Ogni parte addestra localmente un modello sui propri dati e condivide solo gli aggiornamenti dei gradienti, che vengono poi combinati per aggiornare il modello globale, come riportato in Figura 2.1.

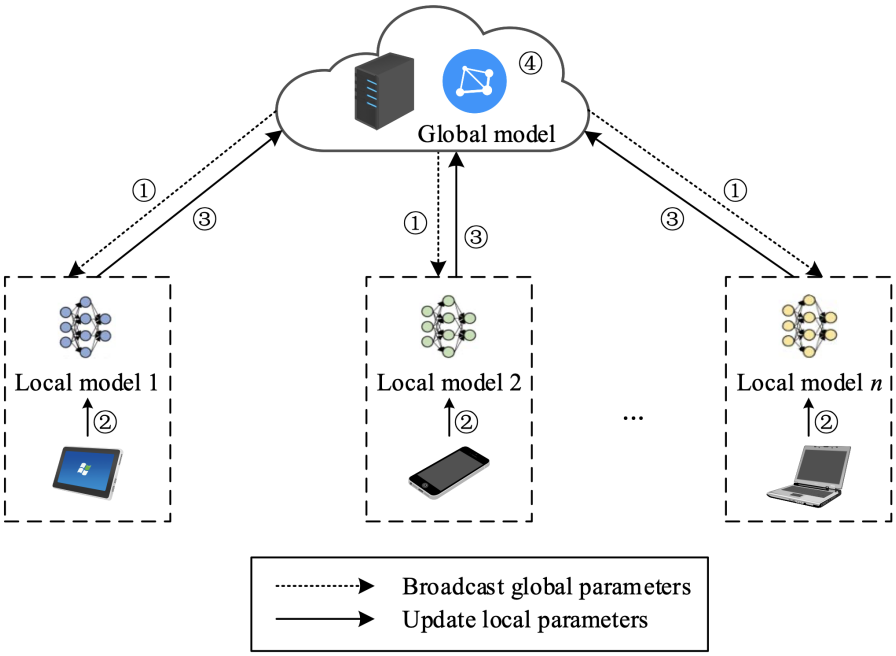


Figura 2.1: Processo di Federated Learning.

Per migliorare la sicurezza è possibile implementare tecniche aggiuntive come il Secure Multi-Party Computation (SMPC).

2.2.2 Secure Multi-Party Computation

Secure Multi-Party Computation (SMPC) [29] è un termine che descrive una serie di algoritmi e protocolli che permettono a due o più parti di valutare congiuntamente una funzione senza rivelare i propri input agli altri partecipanti. Utilizzando queste tecniche, sono stati sviluppati protocolli per valutare reti neurali preservando la privacy degli input [30] [31]. Tuttavia, questi protocolli spesso richiedono una significativa quantità di comunicazione durante il processo di calcolo, il che può causare problemi di latenza di rete. Per ridurre la quantità di comunicazione necessaria, è possibile eseguire parte della computazione onerosa in una fase offline preliminare, minimizzando così l'impatto sulla latenza durante la fase di calcolo attivo.

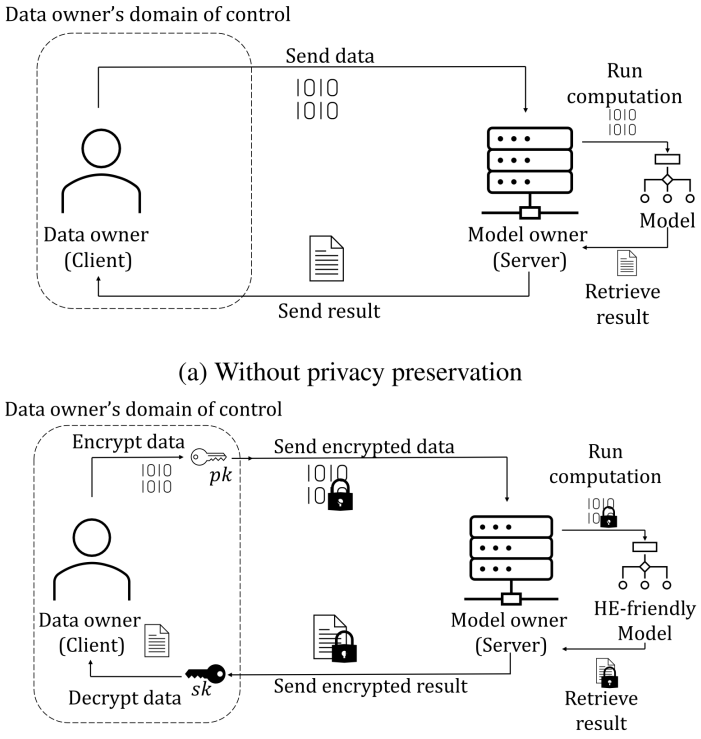
2.2.3 Differential Privacy

La Differential Privacy (DP) [32] aggiunge rumore casuale ai dati per offuscare le informazioni private contenute in essi. In base alla quantità di rumore aggiunto, è possibile calcolare un limite di probabilità sull'intrusione delle informazioni, che indica quanto è probabile che un avversario possa estrarre informazioni private dai dati. L'applicazione principale della DP è proteggere le informazioni personali nei dati di addestramento. L'obiettivo è prevenire che un attaccante estragga informazioni private dal modello addestrato controllando quanto ogni campione può influenzare i parametri durante l'addestramento. Inoltre, più parti possono utilizzarla per addestrare un modello condiviso su dati distribuiti senza rivelare i dati privati agli altri partecipanti.

Il principale punto di forza della DP è che opera su tipi di dati numerici standard, il che significa che la maggior parte delle librerie di machine learning e degli acceleratori hardware sono supportati, rendendo l'implementazione e l'overhead di esecuzione trascurabili. D'altra parte, l'utilizzo della DP può ridurre la qualità delle previsioni del modello.

2.2.4 Homomorphic Encryption

La cifratura omomorfica, come anticipato in precedenza, è una tecnica crittografica che consente di eseguire operazioni su dati cifrati senza doverli decifrare prima [33]. Questa proprietà è particolarmente utile nel contesto del machine learning, soprattutto quando si tratta di proteggere la privacy dei dati sensibili durante l'elaborazione su server remoti.



(a) Without privacy preservation

(b) With privacy preservation

Figura 2.2: Architettura di MLaaS, senza e con preservazione della privacy. Immagine tratta da [34].

Nello scenario classico (Figura 2.2 a), i dati necessari vengono inviati dal proprietario direttamente al server per l'elaborazione. Sebbene i dati vengano trasmessi in modo sicuro sulla rete, il server ha comunque pieno accesso ai dati grezzi, che utilizza per addestrare il modello di machine learning o per effettuare predizioni, i cui risultati vengono poi forniti in output. Tuttavia, questo approccio non preserva la privacy del proprietario dei dati, poiché il server ha accesso ai dati in chiaro. La crittografia omomorfica apre la strada a un'alternativa al tradizionale Machine

Learning as a Service. Nel secondo scenario (Figura 2.2 b), i dati necessari vengono prima cifrati utilizzando uno schema di cifratura omomorfica e poi inviati al server per l'elaborazione. Il server non ha accesso ai dati in chiaro, ma può comunque eseguire le operazioni necessarie (in base allo schema utilizzato) per addestrare il modello di machine learning o per effettuare predizioni. Infine, i risultati vengono restituiti al proprietario, che è l'unico in grado di decifrarli.

Capitolo 3

Analisi dei dati privacy-preserving con crittografia omomorfica

3.1 Schemi di riferimento

Gli schemi di crittografia omomorfica più studiati e utilizzati sono BGV [35], BFV [36], CKKS [12] e TFHE [37]. Ciascuno di questi schemi ha caratteristiche specifiche che li rendono adatti a diversi tipi di applicazioni.

BGV e BFV sono noti per supportare calcoli esatti su numeri interi, il che li rende ideali per applicazioni che richiedono precisione assoluta, come la crittografia per sistemi bancari o finanziari. Tuttavia, entrambi gli schemi devono affrontare il problema della crescita dei valori durante il calcolo, il che può rendere necessario l'uso di parametri più grandi per evitare errori di decifratura.

CKKS, d'altro canto, è progettato per eseguire calcoli approssimati su numeri reali, il che lo rende particolarmente utile in ambiti come l'analisi dei dati e il machine learning, dove una piccola approssimazione è accettabile. Questa caratteristica consente di eseguire calcoli più complessi e di supportare operazioni che richiedono l'uso di numeri in virgola mobile, pur introducendo un piccolo errore di approssimazione nei risultati.

Infine, TFHE è ottimizzato per operazioni a livello di bit e supporta una vasta gamma di porte logiche binarie, rendendolo ideale per costruire circuiti logici complessi. Una delle sue principali peculiarità è la velocità di bootstrapping, che lo rende uno degli schemi più efficienti per applicazioni che richiedono frequenti operazioni di riduzione del rumore nel testo cifrato.

Nella Tabella 3.1 sono riassunte le caratteristiche degli schemi FHE illustrati.

Schema	**Spazio dei messaggi**			**Operazioni supportate**				**SIMD**
	Intero	Reale	Binario	Somma	Prodotto	Divisione	Bitwise	
BFV	✓			✓	✓			✓
BGV	✓			✓	✓			✓
CKKS		✓		✓	✓	✓		✓
TFHE			✓				✓	

Tabella 3.1: Confronto tra schemi FHE riguardo allo spazio dei messaggi supportato, alle operazioni supportate sui dati cifrati (dove la divisione è supportata solo con divisori in chiaro) e al supporto per Single Instruction Multiple Data (SIMD). Tabella tratta da [34].

3.2 Librerie di riferimento

Esistono varie librerie open-source e strumenti sviluppati per implementare la crittografia omomorfica, che permettono di eseguire operazioni sui dati senza decifrarli. Ogni libreria ha caratteristiche diverse in termini di prestazioni, facilità d'uso e tipologia di cifratura supportata. Di seguito, una panoramica delle principali librerie disponibili:

- **SEAL** si distingue per la sua semplicità d'uso e le prestazioni ottimizzate. Sviluppata da Microsoft Research, SEAL supporta crittografia parzialmente e completamente omomorfica, offrendo implementazioni degli schemi BFV e CKKS. Grazie alla sua buona documentazione e API intuitive, SEAL è una scelta popolare sia per la ricerca che per l'implementazione in ambienti di produzione [38].

- **HElib**, sviluppata da IBM Research, è una delle prime librerie di crittografia omomorfica e ha avuto un ruolo pionieristico nello sviluppo del campo. HElib implementa princi-

palmente lo schema BGV e include ottimizzazioni avanzate per operazioni batch, risultando particolarmente utile in applicazioni che richiedono prestazioni elevate e gestione efficiente dei dati cifrati [39].

- **TFHE**, sviluppata da INRIA, è una libreria specializzata in crittografia completamente omomorfica su cifrati torici. TFHE è particolarmente apprezzata per le sue elevate prestazioni nelle operazioni bitwise, rendendola ideale per calcoli logici su dati cifrati. La sua architettura modulare consente inoltre di personalizzare e ottimizzare ulteriormente le configurazioni per specifiche esigenze [40].

- **OpenFHE** è una libreria emergente, frutto della collaborazione tra i team dietro altre librerie, tra cui HElib. OpenFHE è progettata per essere una piattaforma robusta e flessibile per lo sviluppo e la ricerca in crittografia omomorfica, con supporto per schemi come BFV, CKKS, BGV e TFHE. Le sue ottimizzazioni per la performance e la scalabilità, insieme al supporto per Multy-Party Computation e Federated Learning, la rendono una scelta all'avanguardia per chi cerca soluzioni HE avanzate [41].

Libreria	**Schemi supportati**				**Bootstrapping**	**GPU**
	BGV	BFV	CKKS	TFHE		
SEAL	✓	✓	✓			
HElib	✓		✓		✓ (solo BGV)	
TFHE				✓	✓	
OpenFHE	✓	✓	✓	✓	✓	

Tabella 3.2: Confronto tra librerie di crittografia omomorfica. Tabella tratta da [34].

Uno schema riassuntivo degli schemi crittografici supportati e delle principali funzionalità offerte dalle librerie di crittografia omomorfica di riferimento è riportato nella Tabella 3.2. OpenFHE si distingue per il suo supporto completo di schemi crittografici e per la presenza di un'implementazione avanzata di bootstrapping, rendendola una delle soluzioni più complete nel panorama delle librerie di crittografia omomorfica. Tuttavia, nessuna delle librerie elencate offre

un'implementazione per GPU, limitando così le prestazioni nelle applicazioni che richiedono un'elaborazione intensiva.

3.3 Reti Neurali

Le reti neurali sono strutture computazionali ispirate al funzionamento del cervello umano, rappresentate come grafi diretti pesati. In questa rappresentazione, i nodi, detti anche neuroni, sono organizzati in livelli, e ogni nodo di un livello è collegato ai nodi del livello successivo tramite archi pesati.

L'obiettivo dell'addestramento di una rete neurale è quello di ottimizzare i pesi degli archi in modo tale che la rete possa eseguire una determinata attività, come la classificazione o la regressione. Per farlo, si utilizza un algoritmo di ottimizzazione iterativo come Stochastic Gradient Descent (SGD). Questo algoritmo aggiorna i pesi in direzione opposta al gradiente della funzione di loss, che misura quanto la previsione della rete si discosta dal valore desiderato.

Il processo di addestramento si ripete su un set di dati di esempio, con l'obiettivo di minimizzare la funzione di loss attraverso continui aggiornamenti dei pesi, fino a che la rete non raggiunge un livello di accuratezza soddisfacente.

Spesso, le soluzioni per la preservazione della privacy nelle reti neurali coprono solo la fase predittiva. D'altra parte, l'addestramento è più costoso in termini computazionali poiché richiede un passaggio in avanti e uno indietro attraverso il modello. L'aumento delle operazioni porta a un accumulo significativo di rumore, come riportato in Figura 3.1, rendendo l'addestramento un processo con una profondità moltiplicativa molto maggiore rispetto alla predizione.

3.3.1 Neurone

Il neurone è la parte fondamentale delle reti neurali. Il funzionamento del neurone può essere descritto come un processo sequenziale in cui i dati in ingresso vengono elaborati per produrre un risultato, come mostrato in Figura 3.2.

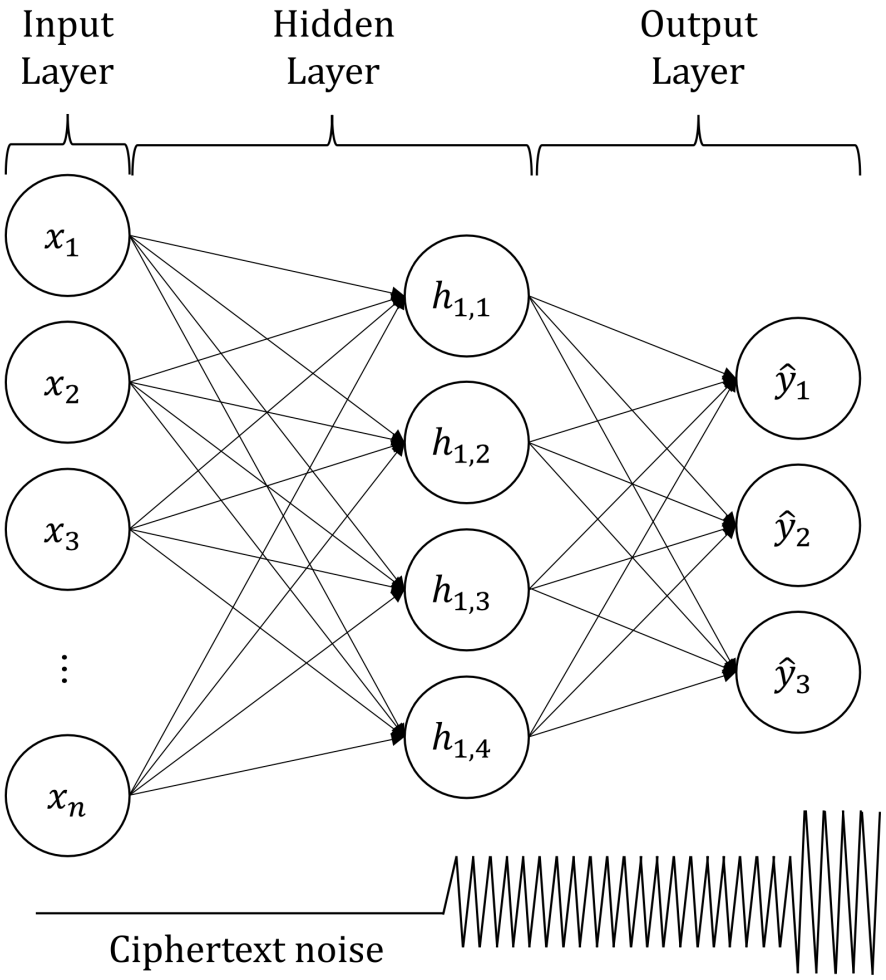


Figura 3.1: Accumulo di errore nella fase di predizione. Immagine tratta da [34].

Ogni neurone del livello corrente prende come input i valori provenienti dai neuroni del livello precedente, moltiplicati per un peso associato a ciascun collegamento. I risultati di queste moltiplicazioni vengono sommati per ottenere un valore aggregato. Questa somma pesata viene poi trasformata tramite una funzione di attivazione non lineare. Questa funzione introduce non linearità nel sistema, permettendo alla rete di apprendere e modellare comportamenti complessi. Esempi di funzioni di attivazione sono la funzione Sigmoid e la funzione ReLU (Rectified Linear Unit).

Sebbene la maggior parte dei calcoli, come somme e prodotti, possano essere eseguiti su dati

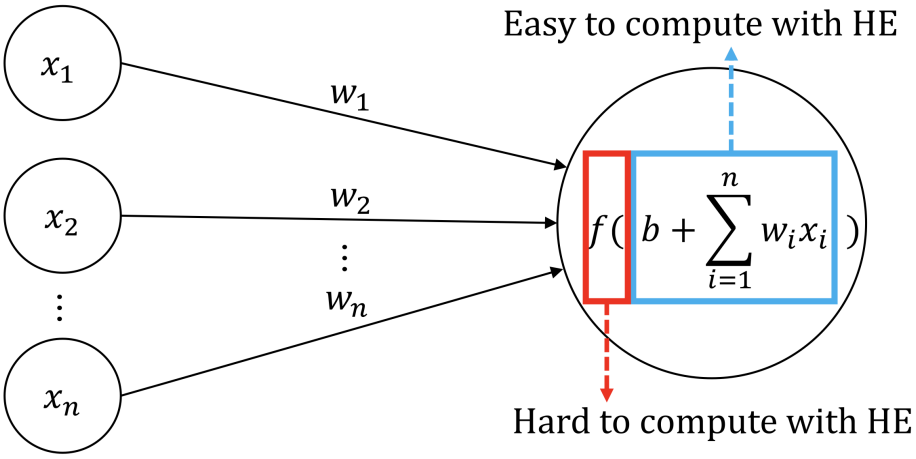


Figura 3.2: Calcolo di un neurone. Immagine tratta da [34].

cifrati, le funzioni di attivazione non lineari rappresentano una sfida significativa. Funzioni di attivazione comuni come ReLU (Rectified Linear Unit), Tanh (Tangente Iperbolica), Softmax e Sigmoid non possono essere valutate in modo efficiente sui dati cifrati con le tecnologie attuali di crittografia omomorfica. Senza queste funzioni di attivazione non lineari, una rete neurale sarebbe in grado di apprendere solo funzioni lineari.

3.3.2 Fully Connected Layer

I livelli completamente connessi, detti anche livelli densi, rappresentano la struttura più semplice e fondamentale nelle reti neurali. In questi livelli, ogni neurone di uno strato è connesso a tutti i neuroni dello strato precedente, e ciascuna connessione ha un peso associato. Invece di calcolare individualmente l'output di ciascun neurone, l'output di un intero strato di neuroni può essere calcolato in modo molto più efficiente utilizzando operazioni di algebra lineare. Nello specifico, l'output dell'intero strato può essere determinato tramite il prodotto matrice-vettore, in cui la matrice W contiene i pesi delle connessioni tra neuroni, il vettore degli input $\vec{x}$ rappresenta i valori di input provenienti dallo strato precedente, e il vettore $\vec{b}$ rappresenta il bias associato a ciascun neurone. Il risultato di questa operazione è il vettore degli output $\vec{y}$.

Questo processo può essere descritto dalla seguente equazione:

$$\vec{y} = W\vec{x} + \vec{b}$$

Questa formulazione consente di eseguire calcoli efficienti per l'intero strato in un solo passaggio, riducendo notevolmente la complessità computazionale rispetto a un calcolo neurone per neurone, rendendo così i livelli completamente connessi una componente essenziale e altamente versatile in molte architetture di reti neurali.

3.3.3 Batch Normalization Layer

La batch normalization è una tecnica introdotta per affrontare il problema dello spostamento interno della covarianza, un fenomeno che si verifica durante l'addestramento delle reti neurali quando la distribuzione degli input a ogni strato cambia continuamente, rallentando il processo di apprendimento e rendendo più difficile la convergenza. Forzando gli input a seguire una distribuzione normale con media zero e varianza unitaria, la batch normalization non solo accelera l'addestramento, ma può anche ridurre la necessità di selezionare accuratamente il tasso di apprendimento.

L'uso della batch normalization è altrettanto efficace quando viene combinata con funzioni di attivazione come Sigmoid e ReLU. La funzione Sigmoid, che comprime gli input in un intervallo tra 0 e 1, è soggetta al problema del gradiente evanescente per valori estremi degli input, rendendo difficile l'aggiornamento dei pesi durante l'addestramento. La ReLU, d'altro canto, è meno incline a questo problema ma può soffrire di un fenomeno noto come "dying ReLU", dove i neuroni possono rimanere bloccati a zero se ricevono input negativi. La batch normalization mitiga questi problemi normalizzando gli input prima che vengano passati alla funzione di attivazione, mantenendo i valori in un intervallo che consente alla Sigmoid e alla ReLU di operare in modo più efficiente. Questo non solo stabilizza il processo di addestramento, ma migliora anche la performance complessiva del modello.

Siano i valori di ingresso $x_1, x_2, \ldots, x_n$, il processo di batch normalization è definito come segue:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2$$

$$\hat{x_i} = \frac{x_i - \mu}{\sqrt{\sigma^2 + \varepsilon}}, \quad \varepsilon \approx 10^{-5}$$

L'output della batch viene infine scalato e shiftato con due parametri γ e β.

$$y_i = \gamma \hat{x_i} + \beta$$

Come per le funzioni di attivazione non lineari, anche l'inverso e la radice quadrata rappresentano una sfida significativa.

3.4 Soluzioni in letteratura

Sono state avanzate pochissime proposte per l'apprendimento delle reti neurali in forma cifrata con schemi FHE, principalmente a causa degli elevati costi computazionali associati a questa tecnologia. La maggior parte degli studi e delle proposte si concentra invece sull'uso della FHE per la fase di predizione, piuttosto che per l'addestramento dei modelli [42] [43]. Questo perché l'addestramento di reti neurali richiede un'enorme quantità di operazioni matematiche complesse, che diventano ancora più onerose se eseguite in un contesto cifrato.

3.4.1 Nandakumar, Ratha, Pankanti e Halevi

Il lavoro di Nandakumar, Ratha, Pankanti e Halevi nel 2019 [44] ha segnato una svolta significativa, estendendosi anche alla fase di addestramento. Questi ricercatori sono stati tra i primi a introdurre questo approccio innovativo e hanno successivamente brevettato la loro

soluzione [45], dimostrando il loro ruolo pionieristico nel campo della crittografia omomorfica applicata all'apprendimento automatico.

La loro implementazione si basa sull'uso della libreria open-source HElib e sull'impiego dello schema BGV, che permette l'esecuzione di operazioni cifrate su dati sensibili.

Un aspetto complesso da affrontare riguarda il calcolo della funzione di attivazione, della funzione di loss e delle rispettive derivate. Per eseguire questi calcoli in modo omomorfico, si possono adottare approcci che includono l'approssimazione delle funzioni tramite polinomi di basso grado o la pre-computazione e la ricerca in tabelle. Per limitare l'uso delle ricerche nella tabella, vengono scelte funzioni di attivazione e di loss con derivate più semplici, come la funzione Sigmoid e la funzione di loss quadratica.

Una delle operazioni più complesse e dispendiose nel contesto cifrato, specialmente con mini-batch SGD, è la moltiplicazione matriciale che viene affrontata con costo $O(n^2)$.

Gli esperimenti sono stati condotti utilizzando il dataset MNIST [46], ampiamente impiegato per il riconoscimento di cifre scritte a mano. Questo dataset comprende 60.000 immagini per l'addestramento e 10.000 immagini per il test, con ogni immagine in scala di grigi di dimensioni 28×28 pixel e le cifre posizionate al centro, mostrato in Figura 3.3.

La rete neurale impiegata in questo lavoro è una rete completamente connessa a 3 strati con funzione di attivazione Sigmoid. I dati sono stati normalizzati sottraendo la media e dividendo per la deviazione standard dei campioni di addestramento, trasformando ogni immagine in un vettore, che sarà fornito in input alla rete neurale. Poiché l'obiettivo è classificare ciascun input in una delle 10 classi numeriche (da "0" a "9"), lo strato di output contiene 10 neuroni. La funzione di loss utilizzata è la Mean Squared Error (MSE).

Sono stati considerati due set di parametri per la rete neurale a 3 strati. Il primo, denominato NN1, utilizza l'input completo di 784 dimensioni (28×28) e include due strati nascosti con 128 e 32 neuroni rispettivamente. Questo porta a un totale di 104.938 parametri ($128 * 785 + 32 * 129 + 10 * 33$). È stata considerata anche una rete molto più piccola, denominata NN2. Questa rete ha un input di dimensione 64 (ottenuta riducendo e ritagliando l'immagine a 8×8 pixel,

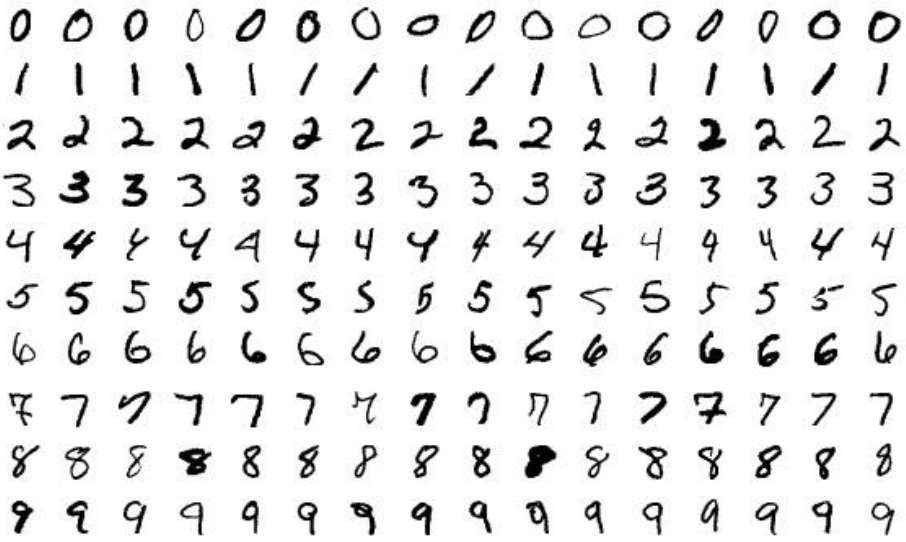


Figura 3.3: Esempi del dataset MNIST.

come mostrato in Figura 3.4) e include due strati nascosti con 32 e 16 neuroni. Questo porta a un totale di 2.778 parametri ($32*65+16*33+10*17$). Entrambe le reti sono state addestrate con dati in chiaro utilizzando SGD con mini-batch da 60 campioni per 50 epoche con accuratezza finale del 97.8% per NN1, del 96.4% per NN2 e del 96% per NN2 utilizzando operazioni su valori a virgola fissa, come mostrato in Figura 3.5.

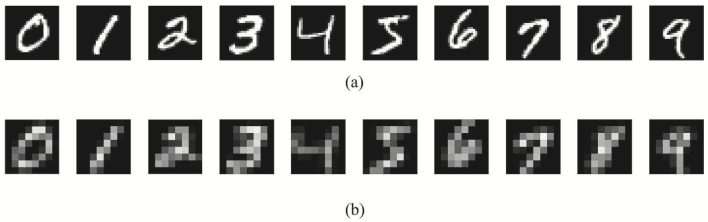


Figura 3.4: Esempi di ridimensionamento del dataset MNIST. Immagine tratta da [44].

Utilizzando parametri di sicurezza non ottimali per semplificare la computazione con testo cifrato e sfruttando la parallelizzazione su 30 core, viene stimato che la computazione di una singola mini-batch (forward e backward) da 60 elementi sulla rete NN2 richieda circa 40 minuti.

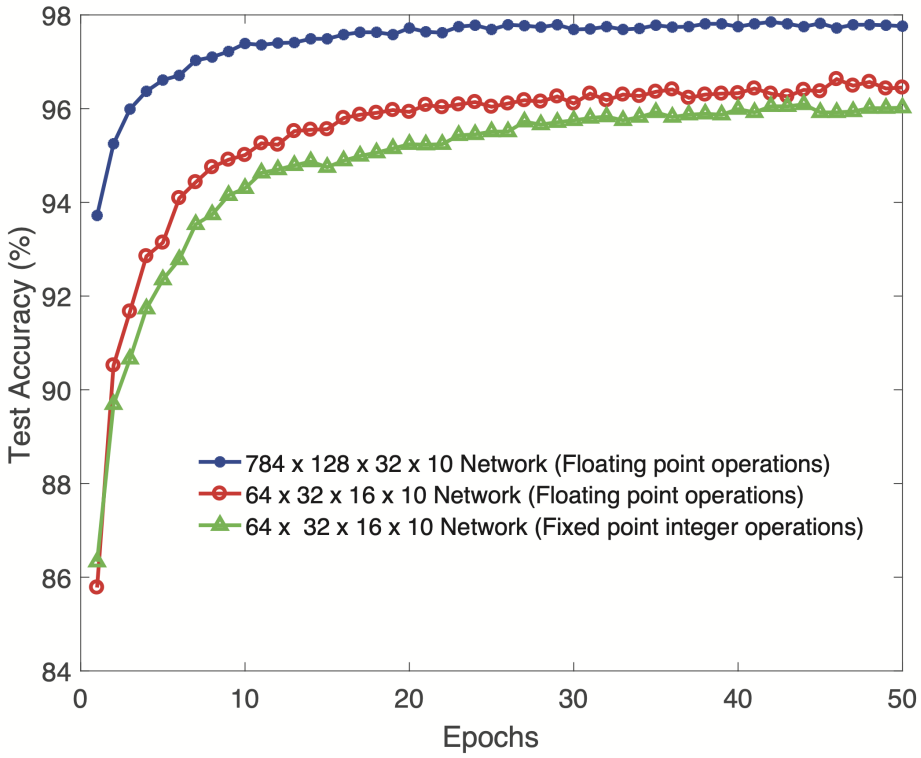


Figura 3.5: Accuratezza delle reti proposte da Nandakumar. Immagine tratta da [44].

Utilizzando gli stessi pesi iniziali della rete e lo stesso set di campioni, è stato osservato che i gradienti e l'aggiornamento dei pesi sono identici sia nel dominio in chiaro (con operazioni su valori a virgola fissa) sia in quello cifrato. Pertanto, è stato affermato che l'accuratezza di classificazione del modello addestrato utilizzando dati cifrati sarà la stessa della versione di NN2 con operazioni su valori a virgola fissa, cioè il 96%.

Capitolo 4

HEDAS: Homomorphic Encrypted Data Analysis System

Questa tesi propone HEDAS, un sistema di analisi dei dati privacy-preserving che sfrutta la crittografia omomorfica. HEDAS è progettato per consentire sia l'addestramento che la predizione di modelli di machine learning direttamente su dati cifrati, garantendo che i dati degli utenti rimangano sempre protetti. In questo modo, è possibile offrire un servizio di Machine Learning as a Service che preserva la privacy, senza mai esporre alcuna informazione in chiaro. L'architettura del sistema è progettata per essere flessibile ed estensibile, e per l'implementazione si è scelto di utilizzare C++ per la sua efficienza computazionale e la libreria OpenFHE per la sua completezza.

4.1 Architettura

OMISSIS

4.2 Gestione delle chiavi

OMISSIS

4.3 Codifica del testo

OMISSIS

4.4 Implementazione del sistema

OMISSIS

4.5 Rischi generali

Il sistema proposto ha richiesto uno studio approfondito dei potenziali rischi associati a tutte le fasi di utilizzo del servizio. Questo studio ha considerato vari tipi di minacce, tra cui attacchi esterni e interni.

4.5.1 Attacchi esterni

Durante tutte le fasi di utilizzo del sistema, le comunicazioni tra il client e il server sono protette dal protocollo TLS (Transport Layer Security). Questo protocollo è essenziale per garantire la sicurezza e la privacy dei dati trasmessi.

TLS utilizza algoritmi di cifratura avanzati per cifrare i dati mentre vengono trasmessi tra client e server. Questo significa che, anche se un attaccante riesce a intercettare i dati, non sarà in grado di leggerli o modificarli senza la chiave di cifratura. Inoltre, TLS fornisce meccanismi di autenticazione per verificare l'identità del server e, facoltativamente, del client. Attraverso l'uso di certificati digitali, TLS assicura che le parti coinvolte nella comunicazione siano quelle

legittime, riducendo il rischio di attacchi di tipo man-in-the-middle, dove un attaccante potrebbe fingersi un'entità legittima per intercettare o alterare i dati.

TLS garantisce anche l'integrità dei dati utilizzando funzioni di hash e codici di autenticazione per assicurare che i dati non siano stati alterati durante il transito. Se i dati vengono modificati in modo malevolo, tali modifiche vengono rilevate, impedendo che dati compromessi vengano accettati come validi. Inoltre, TLS protegge contro gli attacchi di replay, utilizzando numeri casuali e timestamp per assicurare che ogni comunicazione sia unica e non possa essere riutilizzata in modo fraudolento.

Nonostante le sue robuste misure di sicurezza, TLS non è immune da rischi. Una delle principali preoccupazioni è la vulnerabilità dei certificati digitali. Se un certificato viene compromesso o emesso erroneamente, potrebbe consentire a un attaccante di impersonare un server legittimo. È cruciale gestire e verificare regolarmente i certificati e utilizzare quelli emessi da autorità di certificazione affidabili per mitigare questo rischio.

Qualora un attaccante riuscisse a impersonificare il server e a ricevere i dati, è importante sottolineare che questi dati sono cifrati utilizzando la crittografia omomorfica. Pertanto, anche se un attaccante riuscisse a ottenere i dati cifrati, non sarebbe in grado di decifrarli senza la chiave privata, che è in possesso esclusivo del legittimo proprietario dei dati.

4.5.2 Attacchi interni

Il sistema proposto coinvolge due principali attori, ovvero il proprietario dei dati e il server che fornisce il servizio di MLaaS. Nell'analisi del sistema, è stato esaminato il comportamento del server sotto diverse condizioni, non limitandosi al solo scenario di comportamento totalmente onesto.

Honest But Curious

Nel contesto di un server "honest but curious", questo rispetta le regole del protocollo ma può tentare di ottenere informazioni aggiuntive dai dati o dalle comunicazioni, al di là di quanto

strettamente necessario per eseguire il servizio. Anche se non manipola i dati in modo malevolo, il suo desiderio di esplorare o raccogliere informazioni potrebbe rappresentare un rischio per la privacy.

Di seguito sono riportati alcuni dei rischi principali:

- **Memorizzazione e analisi prolungata dei dati cifrati**: un rischio intrinseco a qualsiasi sistema crittografico è la possibilità che il server memorizzi i dati cifrati per un periodo prolungato, sperando di sfruttare eventuali vulnerabilità future o algoritmi di crittoanalisi più avanzati. Anche se i dati sono attualmente protetti, l'evoluzione della tecnologia potrebbe in futuro rendere possibili attacchi che oggi sembrano impraticabili. Tuttavia, il sistema proposto utilizza CKKS, uno degli schemi FHE più studiati e affermati. CKKS è stato oggetto di numerosi studi e ricerche, dimostrando una robustezza significativa contro attacchi crittografici noti. Sebbene non esista una protezione assoluta contro tutte le possibili vulnerabilità future, l'uso di CKKS rappresenta una scelta consolidata e avanzata nel campo della crittografia omomorfica.

- **Inferenza tramite funzioni**: un altro rischio è che il server potrebbe tentare di ottenere informazioni sui dati cifrati se le funzioni che il proprietario dei dati può eseguire non sono sufficientemente restrittive. Se le funzioni consentite sono troppo generiche, potrebbero rivelare pattern o dettagli sui dati originali. Per mitigare questo rischio, il sistema limita le funzioni valutabili dal proprietario dei dati a quelle essenziali, come la funzione logistica o l'inverso della radice quadrata, in modo tale da non rivelare informazioni significative sui dati cifrati, mantenendo la sicurezza e la privacy del sistema.

- **Attacchi di ricerca sul testo cifrato**: un rischio meno pratico ma teoricamente possibile è rappresentato dagli attacchi di ricerca sul testo cifrato. In tali attacchi, il server potrebbe cercare di confrontare il testo cifrato con diversi testi in chiaro per trovare una corrispondenza. Tuttavia, questi attacchi sono altamente complicati e computazionalmente onerosi, specialmente quando si utilizza una cifratura robusta. Inoltre, CKKS utilizza la tecnica

SIMD che, su anelli ciclotomici di dimensioni adeguate, rende gli attacchi di ricerca sul testo cifrato computazionalmente impossibili da eseguire.

Malicious

Nel contesto di un server malevolo, si aggiungono ulteriori rischi intrinseci che compromettono la correttezza dei dati, del modello e delle operazioni svolte dal server. Un server malevolo non solo cerca di estrarre informazioni dai dati cifrati, ma può anche compromettere attivamente l'integrità e la sicurezza del sistema.

Oltre ai rischi già menzionati, quali la memorizzazione prolungata dei dati cifrati e l'inferenza tramite funzioni, emergono rischi specifici associati al comportamento malevolo del server:

- **Alterazione del modello**: un server malevolo potrebbe addestrare il modello utilizzando dati errati, alterando i parametri della rete senza l'autorizzazione dell'utente. Questo potrebbe compromettere la qualità e l'affidabilità del modello, rendendo le predizioni e le decisioni basate su di esso meno accurate o completamente fuorvianti.

- **Alterazione dei risultati**: il server potrebbe manipolare i risultati finali, modificando le predizioni o le classificazioni fornite agli utenti. Ad esempio, potrebbe alterare le decisioni di classificazione per favorire un risultato desiderato o per sabotare il processo decisionale basato sui risultati del modello. Tali manipolazioni potrebbero avere conseguenze gravi, soprattutto in contesti critici dove le decisioni basate sui risultati del modello possono influenzare direttamente le azioni o i risultati.

Anche se il sistema adotta misure per mitigare i rischi associati a server "honest but curious", è importante sottolineare che non è possibile affidarsi a un server malevolo. La natura stessa dei server malevoli implica che potrebbero non solo tentare di accedere ai dati cifrati, ma anche addestrare o manipolare il modello in modi che non possono essere completamente mitigati. Di conseguenza, è fondamentale che i dati e il modello siano gestiti in modo sicuro e che si consideri

l'uso di server di fiducia o meccanismi di verifica per garantire l'integrità e l'affidabilità del sistema.

4.6 Caso studio: Credit Score

Il processo di credit scoring è una pratica fondamentale nel settore finanziario, utilizzata per valutare l'affidabilità creditizia di individui o aziende. Il punteggio di credito, o credit score, è un numero che rappresenta la probabilità che un soggetto sia in grado di rimborsare un prestito o un debito, basandosi sulla sua storia creditizia e su altri dati rilevanti. Questa valutazione è cruciale per banche, istituzioni finanziarie e altri creditori, poiché consente loro di prendere decisioni informate sul rischio associato a concedere crediti, riducendo il rischio di insolvenza.

Il credit score serve principalmente per determinare l'idoneità di un soggetto a ottenere prestiti, mutui, carte di credito e altre forme di credito. I creditori utilizzano il credit score per stabilire non solo se concedere il credito, ma anche a quali condizioni, come il tasso di interesse applicato, l'importo del prestito e la durata del finanziamento. Un punteggio elevato suggerisce che il soggetto è finanziariamente affidabile e ha una buona storia di rimborsi tempestivi, mentre un punteggio basso può indicare un rischio maggiore, portando a condizioni più severe o al rifiuto del credito.

Il processo di credit scoring richiede la raccolta e l'elaborazione di un'ampia gamma di dati personali e finanziari. Questi dati possono includere informazioni su reddito, debiti esistenti, cronologia dei pagamenti, informazioni su conti bancari, dettagli sull'occupazione e anche eventuali precedenti bancari negativi, come insolvenze o fallimenti. Dato il carattere estremamente sensibile di questi dati, è essenziale garantire un elevato livello di privacy. La protezione della privacy è necessaria per evitare che informazioni personali possano essere utilizzate impropriamente, esposte o vendute a terzi senza il consenso del soggetto interessato. Le normative come il GDPR in Europa impongono requisiti stringenti per la gestione di questi dati, richiedendo che siano trattati solo per scopi specifici, in modo trasparente e con il consenso esplicito dell'inte-

ressato. Queste normative prevedono anche il diritto all'oblio e la portabilità dei dati, elementi cruciali per la gestione responsabile delle informazioni personali.

4.6.1 Rischi

Quando si utilizza un servizio di MLaaS per il credit scoring, una delle minacce più preoccupanti non è solo legata agli attacchi esterni, ma al comportamento potenzialmente curioso o malevolo del server stesso, ovvero del fornitore del servizio. In questi scenari, non è un attaccante esterno che cerca di compromettere i dati, ma il server o chi lo gestisce che potrebbe abusare delle informazioni o dei modelli che ospita.

Durante il processo di addestramento o predizione, il server ha accesso a tutti i dati sensibili relativi al credit scoring, inclusi dati personali, come il reddito, i debiti, e la cronologia dei pagamenti di migliaia di clienti. Se il server decide di memorizzare e analizzare questi dati oltre i limiti concordati o stabiliti dalle normative, potrebbe accumulare una vasta mole di informazioni private, senza che l'ente finanziario o i clienti ne siano a conoscenza. Questa memorizzazione non autorizzata crea un enorme rischio di violazione della privacy, poiché il server potrebbe conservare i dati per più tempo del necessario o utilizzarli per scopi non dichiarati, come l'analisi di mercato, la vendita a terzi o il miglioramento dei propri modelli senza il consenso degli utenti.

Un server malevolo potrebbe inoltre applicare attacchi di ricostruzione in modo intenzionale. Poiché il server ospita il modello e ha accesso ai dati di input e alle risposte generate dal modello stesso, potrebbe sfruttare queste informazioni per ricostruire dati personali specifici. Per esempio, il server potrebbe esaminare le risposte del modello per identificare pattern o dettagli nascosti sui dati degli utenti. Nel caso del credit scoring, il server potrebbe riuscire a dedurre quali clienti hanno un reddito elevato o una storia di debiti particolarmente sensibile, effettuando ripetute interrogazioni e osservando le variazioni nei risultati. In questo modo, il server stesso potrebbe "scavare" nei dati e ricostruire profili dettagliati degli utenti senza che nessuno se ne accorga, compromettendo gravemente la riservatezza delle informazioni finanziarie.

Il server potrebbe anche mettere in atto attacchi di inversione, sfruttando la conoscenza approfondita del modello e dei risultati generati per dedurre i dati di input utilizzati per calcolare i punteggi di credito. Ad esempio, analizzando un punteggio di credito finale, un server malevolo potrebbe risalire alla cronologia finanziaria o ai dettagli personali di un individuo, come la quantità di debiti contratti o la puntualità nei pagamenti. Questa capacità di dedurre informazioni sensibili è particolarmente pericolosa poiché il server ha accesso sia al modello che ai dati in modo completo, permettendogli di scoprire dettagli che i clienti non avevano intenzione di condividere.

Un altro comportamento pericoloso che un server malevolo potrebbe adottare riguarda gli attacchi di inferenza sull'appartenenza. Con accesso completo al modello e ai dati, il server potrebbe analizzare i risultati per determinare se i dati di un cliente specifico siano stati utilizzati durante l'addestramento del modello. In questo modo, il server potrebbe rivelare informazioni sensibili, come il fatto che un individuo abbia richiesto un prestito o sia stato coinvolto in una transazione finanziaria ad alto rischio. Questo tipo di attacco potrebbe non solo violare la privacy degli utenti, ma anche esporli a potenziali discriminazioni o conseguenze legali, qualora queste informazioni venissero condivise o vendute a terze parti senza consenso.

Questi rischi rendono evidente quanto sia cruciale, nel credit scoring e in altri ambiti dove la privacy è essenziale, non solo considerare le minacce esterne ma anche monitorare attentamente il comportamento del fornitore del servizio. La fiducia nel server deve essere accompagnata da misure di sicurezza rigorose, come audit regolari, trasparenza nelle operazioni, l'uso di tecniche avanzate e la conformità a normative stringenti come il GDPR. Solo con queste protezioni si può garantire che il fornitore del servizio non utilizzi i dati o i modelli in modi non autorizzati o dannosi.

4.7 Scenario di HEDAS

Per il sistema proposto, è essenziale che il server aderisca rigorosamente al protocollo stabilito e non comprometta i dati o il modello attraverso operazioni omomorfiche non autorizzate. Il protocollo definisce chiaramente le operazioni specifiche e le restrizioni che il server deve seguire durante l'elaborazione dei dati cifrati e l'addestramento del modello.

La crittografia omomorfica consente al server di eseguire operazioni sui dati cifrati senza decifrarli, ma è cruciale che tali operazioni siano limitate esclusivamente alle funzioni autorizzate e predefinite nel protocollo di addestramento o predizione. Se il server esegue operazioni omomorfiche non autorizzate o manipola i dati in modi non previsti, potrebbe compromettere l'integrità del modello e la sicurezza dei dati.

Il protocollo prevede che il proprietario dei dati generi le chiavi (privata, pubblica, di bootstrap e di rotazione) e che la chiave privata, fondamentale per decifrare i dati, non venga mai comunicata al server per garantire la protezione dei dati stessi.

Le altre chiavi saranno invece comunicate al server, sempre utilizzando TLS per garantire la sicurezza delle comunicazioni. Queste chiavi, insieme ai dati cifrati e ai parametri del modello, permettono al server di eseguire operazioni omomorfiche sul modello e sui dati, che possono includere somme, prodotti e rotazioni. Tali operazioni, però, non consentono di recuperare i dati in chiaro, garantendo che le informazioni rimangano protette durante tutto il processo.

Le operazioni non lineari, come la funzione logistica, l'inverso o la radice quadrata, saranno eseguite dal proprietario dei dati, se incluse nella lista delle funzioni eseguibili. In questo caso, il client riceve la richiesta con i dati cifrati, provvede a decifrare i dati, applicare la funzione necessaria, cifrare nuovamente i dati e inviare il testo cifrato al server.

Al termine dei calcoli, il server fornisce i parametri finali o le predizioni al client, che sarà l'unico a poter accedere ai valori in chiaro utilizzando la chiave privata. Questo processo assicura che solo il proprietario dei dati possa decifrare e accedere alle informazioni sensibili, mantenendo la sicurezza e l'integrità del sistema.

Capitolo 5

Valutazioni sperimentali

HEDAS è stato accuratamente valutato e testato su una macchina dotata di un processore AMD Ryzen 5 3600, con 6 core e 12 thread, e 48 GB di RAM DDR4. I test sono stati eseguiti utilizzando la versione 1.2.0 di OpenFHE, rilasciata il 15 giugno 2024, che include miglioramenti significativi in termini di efficienza e sicurezza rispetto alle versioni precedenti.

5.1 Parametro di sicurezza

OMISSIS

5.2 Correttezza dell'implementazione

OMISSIS

5.3 Prestazioni al variare del parametro di sicurezza

OMISSIS

5.4 Confronto delle prestazioni della rete neurale proposta da Nandakumar

OMISSIS

5.5 Caso studio: Credit Score

OMISSIS

Conclusioni

Applicare la crittografia omomorfica all'analisi dei dati in modo privacy-preserving è tecnicamente fattibile, ma non senza notevoli difficoltà. Questa tecnologia consente di eseguire operazioni sui dati cifrati senza necessità di decifrarli, preservando così la riservatezza delle informazioni. Tuttavia, gli algoritmi di machine learning, specialmente quelli basati su reti neurali profonde, fanno ampio uso di funzioni non lineari, come le funzioni di attivazione (Sigmoid, ReLU, Tanh) che complicano l'applicazione della crittografia omomorfica. Queste funzioni sono difficili da trattare con gli schemi di crittografia omomorfica attuali, che sono più adatti per operazioni lineari.

Per affrontare queste sfide, una possibile soluzione è implementare un modello di collaborazione tra il proprietario dei dati e il server di calcolo. In questo scenario, il proprietario dei dati e il server lavorano insieme in modo tale che i dati in chiaro non siano mai esposti al server, riducendo il rischio di violazioni della privacy. Questo approccio consente di mantenere la riservatezza dei dati mentre si sfruttano le capacità computazionali del server, anche se introduce complessità nella gestione della collaborazione e delle operazioni computazionali.

Nonostante l'adozione del sistema proposto possa consentire l'analisi dei dati privacy-preserving, è fondamentale considerare che i tempi computazionali rimangono una limitazione significativa. La crittografia omomorfica comporta operazioni matematiche molto complesse, il che può rallentare notevolmente i tempi di elaborazione rispetto ai metodi tradizionali. Questo rallentamento è particolarmente accentuato quando si utilizzano parametri di sicurezza elevati per garantire una protezione adeguata dei dati, il che rende i tempi di calcolo poco praticabili

per applicazioni reali.

Pertanto, è cruciale continuare a investire nella ricerca e nello sviluppo della crittografia omomorfica. I progressi in questo campo sono essenziali per migliorare la sua efficienza e praticabilità. Con ulteriori avanzamenti, la crittografia omomorfica potrebbe diventare una tecnologia più accessibile e utile, aprendo nuove possibilità per garantire la riservatezza dei dati non solo nel machine learning, ma anche in numerosi altri ambiti, rendendo possibile l'implementazione di soluzioni più sicure e scalabili per la protezione delle informazioni sensibili.

Elenco delle figure

1.1 Evoluzioni degli schemi FHE. 11
1.2 Esempio di una base ottimale e di una base non ottimale. 12
1.3 Esempio di problemi CVP e SVP. 12
1.4 Processo di bootstrap. 14

2.1 Processo di Federated Learning. 24
2.2 Architettura di MLaaS, senza e con preservazione della privacy. 26

3.1 Accumulo di errore nella fase di predizione. 32
3.2 Calcolo di un neurone. 33
3.3 Esempi del dataset MNIST. 37
3.4 Esempi di ridimensionamento del dataset MNIST. 37
3.5 Accuratezza delle reti proposte da Nandakumar. 38

Bibliografia

[1] A. Acar, H. Aksu, A. S. Uluagac e M. Conti. «A survey on homomorphic encryption schemes: Theory and implementation». In: *ACM Computing Surveys (Csur)* 51.4 (2018), pp. 1–35.

[2] R. L. Rivest, L. Adleman, M. L. Dertouzos et al. «On data banks and privacy homomorphisms». In: *Foundations of secure computation* 4.11 (1978), pp. 169–180.

[3] W. Diffie e M. Hellman. «New directions in cryptography». In: *IEEE Transactions on Information Theory* 22.6 (1976), pp. 644–654.

[4] M. Fellows e N. Koblitz. «Combinatorial cryptosystems galore!» In: *Contemporary Mathematics* 168 (1994), pp. 51–51.

[5] M. R. Albrecht, P. Farshim, J.-C. Faugere e L. Perret. «Polly cracker, revisited». In: *International conference on the theory and application of cryptology and information security*. Springer. 2011, pp. 179–196.

[6] R. Steinwandt e W. Geiselmann. «Cryptanalysis of Polly Cracker». In: *IEEE Transactions on Information Theory* 48.11 (2002), pp. 2990–2991.

[7] D. Boneh, E.-J. Goh e K. Nissim. «Evaluating 2-DNF formulas on ciphertexts». In: *Theory of Cryptography: Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005. Proceedings 2*. Springer. 2005, pp. 325–341.

[8] V. Agate, P. Ferraro, G. Lo Re e S. K. Das. «BLIND: A privacy preserving truth discovery system for mobile crowdsensing». In: *Journal of Network and Computer Applications* 223 (2024).

[9] C. Gentry. «Fully homomorphic encryption using ideal lattices». In: *Proceedings of the forty-first annual ACM symposium on Theory of computing*. 2009, pp. 169–178.

[10] M. Van Dijk, C. Gentry, S. Halevi e V. Vaikuntanathan. «Fully homomorphic encryption over the integers». In: *Advances in Cryptology–EUROCRYPT 2010: 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30–June 3, 2010. Proceedings 29*. Springer. 2010, pp. 24–43.

[11] C. Gentry, A. Sahai e B. Waters. «Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based». In: *Advances in Cryptology–CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*. Springer. 2013, pp. 75–92.

[12] J. H. Cheon, A. Kim, M. Kim e Y. Song. «Homomorphic encryption for arithmetic of approximate numbers». In: *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23*. Springer. 2017, pp. 409–437.

[13] B. Li e D. Micciancio. «On the security of homomorphic encryption on approximate numbers». In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2021, pp. 648–677.

[14] J. Hoffstein, J. Pipher e J. H. Silverman. «NTRU: A Ring-Based Public Key Cryptosystem». In: *Proceedings of the Third International Symposium on Algorithmic Number Theory*. ANTS-III. Berlin, Heidelberg: Springer-Verlag, 1998, pp. 267–288.

[15] M. Clear e C. McGoldrick. «Multi-identity and multi-key leveled FHE from learning with errors». In: *Advances in Cryptology–CRYPTO 2015: 35th Annual Cryptology Conference,*

Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II 35. Springer. 2015, pp. 630–656.

[16] M. Albrecht, S. Bai e L. Ducas. «A subfield lattice attack on overstretched NTRU assumptions: Cryptanalysis of some FHE and graded encoding schemes». In: *Annual International Cryptology Conference*. Springer. 2016, pp. 153–178.

[17] J. H. Cheon, J. Jeong e C. Lee. «An algorithm for NTRU problems and cryptanalysis of the GGH multilinear map without a low-level encoding of zero». In: *LMS Journal of Computation and Mathematics* 19.A (2016), pp. 255–266.

[18] V. Agate, A. De Paola, S. Drago, P. Ferraro e G. Lo Re. «Enhancing IoT Network Security with Concept Drift-Aware Unsupervised Threat Detection». In: *2024 IEEE Symposium on Computers and Communications (ISCC)*. IEEE. 2024.

[19] M. Ribeiro, K. Grolinger e M. A. Capretz. «Mlaas: Machine learning as a service». In: *2015 IEEE 14th international conference on machine learning and applications (ICMLA)*. IEEE. 2015, pp. 896–902.

[20] V. N. Vapnik e A. Y. Chervonenkis. «A note on one class of perceptrons». In: *Automation and Remote Control* 24.6 (1963), pp. 1084–1093.

[21] C. Cortes e V. Vapnik. «Support-vector networks». In: *Machine Learning* 20.3 (1995), pp. 273–297.

[22] T. Cover e P. Hart. «Nearest neighbor pattern classification». In: *IEEE transactions on information theory* 13.1 (1967), pp. 21–27.

[23] J. Feng e A. K. Jain. «Fingerprint reconstruction: from minutiae to phase». In: *IEEE transactions on pattern analysis and machine intelligence* 33.2 (2010), pp. 209–223.

[24] F. Batool, F. Canino, F. Concone, G. Lo Re e Morana. «A Black-box Adversarial Attack on Fake News Detection Systems». In: *CEUR Workshop Proceedings*. Vol. 3731. IT. 2024.

[25] M. Fredrikson, S. Jha e T. Ristenpart. «Model inversion attacks that exploit confidence information and basic countermeasures». In: *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. 2015, pp. 1322–1333.

[26] R. Shokri, M. Stronati, C. Song e V. Shmatikov. «Membership inference attacks against machine learning models». In: *2017 IEEE symposium on security and privacy (SP)*. IEEE. 2017, pp. 3–18.

[27] B. McMahan, E. Moore, D. Ramage, S. Hampson e B. A. y Arcas. «Communication-efficient learning of deep networks from decentralized data». In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.

[28] A. Augello, G. Falzone e G. Lo Re. «DCFL: Dynamic Clustered Federated Learning under Differential Privacy Settings». In: *2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. 2023, pp. 614–619.

[29] A. C. Yao. «Protocols for secure computations». In: *23rd annual symposium on foundations of computer science (sfcs 1982)*. IEEE. 1982, pp. 160–164.

[30] S. Li, K. Xue, B. Zhu, C. Ding, X. Gao, D. Wei e T. Wan. «Falcon: A fourier transform based approach for fast and secure convolutional neural network predictions». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 8705–8714.

[31] J. Liu, M. Juuti, Y. Lu e N. Asokan. «Oblivious neural network predictions via minionn transformations». In: *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*. 2017, pp. 619–631.

[32] C. Dwork, F. McSherry, K. Nissim e A. Smith. «Calibrating noise to sensitivity in private data analysis». In: *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*. Springer. 2006, pp. 265–284.

[33] V. Agate, P. Ferraro e G. Lo Re. «A Privacy-Preserving System for Enhancing the QoI of Collected Data in a Smart Connected Community». In: *2024 IEEE Symposium on Computers and Communications (ISCC)*. IEEE. 2024, pp. 1–6.

[34] R. Podschwadt, D. Takabi, P. Hu, M. H. Rafiei e Z. Cai. «A survey of deep learning architectures for privacy-preserving machine learning with fully homomorphic encryption». In: *IEEE Access* 10 (2022), pp. 117477–117500.

[35] Z. Brakerski, C. Gentry e V. Vaikuntanathan. «Fully Homomorphic Encryption without Bootstrapping». In: *Proceedings of the 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*. 2011, pp. 311–330.

[36] Z. Brakerski, J. Fan e V. Vaikuntanathan. «Packing and Bootstrapping for Improved Efficiency of Fully Homomorphic Encryption». In: *Proceedings of the 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*. 2014, pp. 307–328.

[37] I. Chillotti, N. Gama e M. Georgieva. «Faster Fully Homomorphic Encryption: Bootstrapping in Less Than 0.1 Seconds». In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2016, pp. 1207–1218.

[38] Microsoft. *Microsoft SEAL: Simple Encrypted Arithmetic Library*. `https://github.com/microsoft/SEAL`.

[39] IBM Research. *HElib: High-performance Homomorphic Encryption Library*. `https://github.com/homenc/HElib`.

[40] TFHE developers. *TFHE: Fast Fully Homomorphic Encryption Library over the Torus*. `https://github.com/tfhe/tfhe`.

[41] OpenFHE developers. *OpenFHE: Open-source Fully Homomorphic Encryption Library*. `https://github.com/openfheorg/openfhe-development`.

[42] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig e J. Wernsing. «Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy». In: *International conference on machine learning*. PMLR. 2016, pp. 201–210.

[43] P. Mohassel e Y. Zhang. «Secureml: A system for scalable privacy-preserving machine learning». In: *2017 IEEE symposium on security and privacy (SP)*. IEEE. 2017, pp. 19–38.

[44] K. Nandakumar, N. Ratha, S. Pankanti e S. Halevi. «Towards deep neural network training on encrypted data». In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*. 2019.

[45] K. Nandakumar, N. K. Ratha, S. Halevi e S. Pankanti. *Learning and inferring insights from encrypted data*. US Patent 11,087,223. Ago. 2021.

[46] Y. LeCun, C. Cortes e C. J. Burges. *The MNIST Database of Handwritten Digits*. 1998. URL: http://yann.lecun.com/exdb/mnist/.

[47] V. Agate, A. De Paola, P. Ferraro, G. Lo Re e M. Morana. «SecureBallot: A secure open source e-Voting system». In: *Journal of Network and Computer Applications* 191 (2021), p. 103165. URL: https://www.sciencedirect.com/science/article/pii/S1084804521001776.

[48] X. Jiang, M. Kim, K. Lauter e Y. Song. «Secure outsourced matrix computation and application to neural networks». In: *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*. 2018, pp. 1209–1222.

[49] M. Kim, Y. Song, S. Wang, X. Yuhou e X. Jiang. «Secure Logistic Regression Based on Homomorphic Encryption: Design and Evaluation». In: *JMIR Medical Informatics* 6 (ago. 2017).

[50] H. Robbins e S. Monro. «A stochastic approximation method». In: *The annals of mathematical statistics* (1951), pp. 400–407.

[51] J. Kiefer e J. Wolfowitz. «Stochastic estimation of the maximum of a regression function». In: *The Annals of Mathematical Statistics* (1952), pp. 462–466.

[52] V. Agate, F. Concone, A. De Paola, P. Ferraro, G. Lo Re e M. Morana. «Bayesian Modeling for Differential Cryptanalysis of Block Ciphers: A DES Instance». In: *IEEE Access* 11 (2023), pp. 4809–4820.

[53] D. P. Kingma e J. Ba. «Adam: A method for stochastic optimization». In: *arXiv preprint arXiv:1412.6980* (2014).

[54] J. Duchi, E. Hazan e Y. Singer. «Adaptive subgradient methods for online learning and stochastic optimization.» In: *Journal of machine learning research* 12.7 (2011).

[55] T. Tieleman e G. Hinton. «Rmsprop: Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning». In: *COURSERA Neural Networks Mach. Learn* 17 (2012).

[56] C.-P. Schnorr. «A hierarchy of polynomial time lattice basis reduction algorithms». In: *Theoretical computer science* 53.2-3 (1987), pp. 201–224.

[57] C. P. Schnorr. «Lattice reduction by random sampling and birthday methods». In: *STACS 2003: 20th Annual Symposium on Theoretical Aspects of Computer Science Berlin, Germany, February 27–March 1, 2003 Proceedings 20*. Springer. 2003, pp. 145–156.

[58] O. Regev. «On lattices, learning with errors, random linear codes, and cryptography». In: *Journal of the ACM (JACM)* 56.6 (2009), pp. 1–40.

[59] X. Wang, M. Liu, Q. Liao e Y. Wang. «Quantum algorithms for solving the shortest vector problem and the closest vector problem». In: *International Conference on Information and Communications Security*. Springer. 2011, pp. 461–473.

[60] V. Agate, M. Curaba, P. Ferraro, G. Lo Re e M. Morana. «Secure e-Voting in Smart Communities». In: *CEUR Workshop Proceedings - ITASEC 2020*. 2020, pp. 1–11.

[61] Z. Yanling, D. Bimin e W. Zhanrong. «Analysis and study of perceptron to solve XOR problem». In: *The 2nd International Workshop on Autonomous Decentralized System, 2002*. 2002, pp. 168–173.

[62] R. Quinlan. *Statlog (Australian Credit Approval)*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C59012. 1987.