

apprentissage automatique préservant la
confidentialité basé sur le chiffrement
homomorphique

Serigne Saliou Gueye

April 22, 2021

REMERCIEMENTS

Acknowledgement goes here

RESUMÉ

INTRODUCTION GÉNÉRALE

ABSTRACT

CONTENTS

I	Contexte et introduction	2
1	Notion Préliminaires de la cryptographie	2
1.1	Introduction	2
1.2	Sécurité inconditionnelle	3
1.2.1	Notion de sécurité parfaite	3
1.2.2	principe de Kerckhoff	4
1.2.3	Principe de Shannon	4
1.2.4	Théorème du secret parfait	4
1.3	définition de sécurité pour la cryptographie à clé privée	7

1.3.1	sécurité EAV	8
1.3.2	Attaque par texte clair choisi (Chosen plaintext Attack CPA)	8
1.3.3	Attaque à chiffré choisi (Chosen Cipher text Attack)(CCA)	10
1.4	Cryptographie à clé publique	11
1.4.1	Fonction à sens unique	11
1.4.2	Fonction de hashage	12
1.4.3	Modélisation du chiffrement à clés publique	12
2	Le chiffrement homomorphe	13
2.1	Introduction	13
2.2	Concept du chiffrement homomorphe	13
2.2.1	Modélisation du chiffrement homomorphe	13
2.3	Sécurité des cryptosystèmes homomorphe	16
2.3.1	La Malléabilité	16
3	Notion Préliminaires de machine learning	18
3.1	Introduction	18
3.2	Classification binaire	18
3.3	Classification naïve bayésienne	21
3.4	Regression linéaire	22
3.5	Annalyse des composantes principales (ACP)	23
3.5.1	théorème (Eckart Young,1936)	23
II	Système de chiffrement homomorphe	25
4	Etude des cryptosystemes homomorphes	25
4.1	cryptosystème simplement homomorphe de RSA	25
4.2	Cryptosystème de Paillier	27
4.2.1	Le système de chiffrement	28
4.2.2	Réduction en RSA	31
4.2.3	Preuve de sécurité	33
4.3	Chiffrement entièrement homomorphe à partir de l'apprentissage avec erreurs (LWE)	35
4.3.1	Le problem LWE (learning with error)	36
4.3.2	Généralités sur le schéma :La méthode des vecteurs propres approximatifs	37
4.3.3	Outils pour le chiffrement homomorphe	37
4.3.4	Le schéma de chiffrement	38

4.4	Réduction des schémas basés sur le treillis pour l'apprentissage avec des erreurs	42
4.5	Encodage des nombres réels	45
III	application à la machine learning	47
5	Application du chiffrement homomorphe aux algorithmes de machine learning	47
5.1	Classification binaire	48
5.2	Regression lineaire	49

PART I

CONTEXTE ET INTRODUCTION

CHAPTER 1

NOTION PRÉLIMINAIRES DE LA CRYPTOGRAPHIE

Avant de passer à la définition du chiffrement homomorphe, nous allons revenir sur quelques notions cryptographiques à savoir les différents systèmes cryptographique (systèmes à clés secrètes et à clés privés) et aussi parler de la sécurité prouvée en cryptographie. Pour des traitements beaucoup plus approfondis de ces concepts nous renvoyons le lecteur sur [8]

1.1 Introduction

La cryptographie est une discipline des maths incluant des principes et méthodes permettant de garantir les services de sécurité de l'information .Sa principale mission est de garantir la sécurité des communications c'est-à-dire de permettre à des entités qui ne se font pas confiance en général de communiquer en toute sécurité en présence de potentiels adversaires (susceptibles entre autres d'accéder à des secrets en violant la confidentialité, d'intercepter et de modifier les informations échangées ou d'usurper des identités lors d'une communication).

La cryptographie ne prend en charge que les 4 premiers sur les 5 services de sécurité fondamentaux que sont la Confidentialité, l'intégrité, l'authentification, la Non Répudiation et la Disponibilité.Elle est composée des systèmes à clé secrète et des systèmes à clés publiques.

Dans un système à clés secrètes les entités se partagent la même clé pour le chiffrement et le déchiffrement (on parle de cryptographie symétrique)

Dans le cas du système à clé publique, on utilise deux clés dont l'une soit k' est difficile à déduire de l'autre soit k (on parle aussi de cryptographie asymétrique). La clé k est appelée clé publique et est utilisée pour le chiffrement ou la vérification de signature selon le système ; la clé k' est appelée clé privée et est utilisée pour le déchiffrement ou la signature selon le système.

Les algorithmes de chiffrements et les protocoles cryptographique sont impliqués dans une variété d'applications critiques: : les banques en ligne, le vote électronique, la vente aux enchères électroniques... En tant que tels, leurs sécurités doivent donc être formellement vérifiées et prouvées

1.2 Sécurité inconditionnelle

1.2.1 Notion de sécurité parfaite

Un cryptosystème est parfois définis par trois algorithmes :Algorithme de génération des clés ,de chiffrement et de déchiffrement ainsi qu'une spécification d'un espace de message M avec $\#M > 1$

L'algorithme de génération des clés que nous notons GEN est un algorithme probabiliste qui produit une clé k choisit selon une certaine distribution. Nous désignons par K l'espace des clés c'est à dire l'ensemble de toutes les clés possibles qui peuvent être sortis par GEN

L'algorithme de chiffrement que nous allons noter par ENC prend en entrée une clé $\kappa \in K$ et un message $m \in M$ produit un chiffré c .Nous considérons que l'algorithme de chiffrement est probabiliste donc $ENC(k,m)$ peut alors générer des chiffrés différents lorsque le même message est utilisé. On note $c \mapsto ENC_{\kappa}(m)$. On note C l'ensemble de tous les chiffrés possibles

L'algorithme de déchiffrement noté par DEC prend en entrée une clé κ et un chiffré c et produit le message initial m . Contrairement à ENC l'algorithme de déchiffrement est déterministe puisque $DEC_{\kappa}(c)$ donne la même sortie à chaque exécution on notera donc $m := DEC_{\kappa}(c)$ pour désigner le caractère déterministe

Il est évident que si l'attaquant détient la clé alors il pourra déchiffrer tous les messages échangés par les entités. Alors qu'en est-il de ENC ? N'est-il pas mieux de garder tous les deux (l'algorithme et la clé) secret?

Auguste Kerckhoffs a soutenu le contraire à la fin du 19e siècle lors de l'élucidation de plusieurs principes de conception de chiffrements militaires. L'un des plus important d'entre eux, désormais simplement connu sous le nom de principe de Kerckhoff.

1.2.2 principe de Kerckhoff

La méthode de chiffrement ne doit pas être tenue secrète, et elle doit pouvoir tomber entre les mains de l'attaquant sans inconvénient.

Pour pouvoir bien comprendre ce principe et voir l'importance de garder la clé secrète nous pouvons considérer l'exemple suivant

Soit m un message $\in \{0, 1\}^n$. On définit notre algorithme de chiffrement (ENC) en faisant la somme directe entre m et $\kappa \in K$ $c = m \oplus \kappa$. On peut voir dans cet algorithme de chiffrement que si κ est fixé alors ce cryptosystème ne sera pas sûr et l'attaquant obtiendra le message initial à partir du chiffré en faisant $m = c \oplus \kappa$.

Par contre si κ est choisi de façon uniformément aléatoire dans l'espace des clés $\{0, 1\}^n$ et gardé secret pour l'attaquant alors le résultat du chiffré sera uniformément distribué dans l'espace des chiffrés $\{0, 1\}^n$ et la sécurité sera atteinte

1.2.3 Principe de Shannon

Un chiffré doit apporter de la confusion et de la diffusion, c'est-à-dire :

Confusion

Il n'y a pas de relation algébrique simple entre le clair et le chiffré. En particulier, connaître un certain nombre de couples clair-chiffré ne permet pas d'interpoler la fonction de chiffrement pour les autres messages.

Diffusion

La modification d'une lettre du clair doit modifier l'ensemble du chiffré. On ne peut pas casser le chiffre morceau par morceau.

1.2.4 Théorème du secret parfait

Avant d'énoncer ce théorème, nous allons considérer les deux exemples ci-dessous

exemple 1

Considérons l'exemple simple du chiffrement de César suivant

Soit $\kappa \in \{0, \dots, 25\}$ avec $\Pr[K = \kappa]$ la probabilité que la clé soit κ (on considère que les clés sont équiprobables) donc $\Pr[K = \kappa] = \frac{1}{26}$. Supposons que nous avons la distribution suivante sur M : $\Pr[M = a] = 0,7$ et $\Pr[M = z] = 0,3$. ($\Pr[M = a]$ probabilité que le message soit a et $\Pr[M = z]$ probabilité que le message soit z). Ainsi on cherche alors la probabilité que le message chiffré soit B

On peut voir qu'il n'y a deux possibilités : soit $M=a$ et $K=1$, soit $M=z$ et $K=2$. Par indépendance de M et K nous avons $\Pr[M = a \wedge K = 1] = \Pr[M = a] * \Pr[K = 1]$, de

même $\Pr[M = z \wedge K = 2] = \Pr[M = z] \cdot \Pr[K = 2]$.

Par conséquent,

$$\Pr[C = B] = \Pr[M = a \wedge K = 1] + \Pr[M = z \wedge K = 2] = 0,7 \cdot \frac{1}{26} + 0,3 \cdot \frac{1}{26} = \frac{1}{26} \text{ donc } \Pr[C = B] = \frac{1}{26}$$

Nous pouvons également calculer la probabilité conditionnelle par exemple calculer la probabilité que le chiffré B soit issu de a, $\Pr[a | B]$. En utilisant le théorème de Bayes nous obtenons :

$$\Pr[M = a | C = B] = \frac{\Pr[C = B | M = a] \cdot \Pr[M = a]}{\Pr[C = B]}$$

Notons que $\Pr[C = B | M = a] = \frac{1}{26}$, puisque si $M = a$ alors la seule voie $C = B$ peut se produire si $\kappa = 1$ ce qui se produit avec une probabilité de $\frac{1}{26}$

$$\text{Donc } \Pr[M = a | C = B] = \frac{\Pr[C = B | M = a] \cdot \Pr[M = a]}{\Pr[C = B]} = \frac{\frac{1}{26} \cdot 0,7}{\frac{1}{26}}$$

$$\Pr[M = a | C = B] = 0,7$$

Conclusion

on a $\Pr[M = a] = \Pr[M = a | C = B] = 0,7$.

Ce résultat montre que la probabilité d'avoir une information claire sur le clair a ne varie pas même si on connaît son chiffré B

exemple 2

Considérez à nouveau le chiffre de décalage, mais avec la distribution suivante sur M:

$$\Pr[M = kim] = 0,5, \Pr[M = ann] = 0,2, \Pr[M = boo] = 0,3$$

Calculons alors la probabilité pour que le chiffré C soit égale à DQQ. La seule façon dont ce chiffré peut se produire est si $M = ann$ et $K = 3$, ou $M = boo$ et $K = 2$, ce qui arrive avec la probabilité $0,2 \cdot \frac{1}{26} + 0,3 \cdot \frac{1}{26} = \frac{1}{52}$.

Nous pouvons également calculer la probabilité que le chiffré DQQ soit issu de ann.? Un calcul comme ci-dessus en utilisant le théorème de Bayes donne $\Pr[M = ann | C = DQQ] = 0,4$

conclusion

On a $\Pr[M = ann] \neq \Pr[M = ann | C = DQQ]$ dans cet exemple nous avons vu avec cette distribution que la probabilité d'avoir une information claire ann varie si son chiffré DQQ est connu

Commentaire

Nous pouvons maintenant définir la notion de secret parfait. On imagine un adversaire qui connaît la distribution de probabilité de M c'est-à-dire que l'adversaire

connaît la probabilité que différents messages soient envoyés.

L'adversaire connaît également le schéma de chiffrement utilisé. La seule chose Inconnue de l'adversaire est la clé partagée par les parties. Un message est choisi par l'une des parties et chiffré, et le texte chiffré résultant est transmis à l'autre partie. L'adversaire peut espionner la communication des parties, et ainsi observer ce texte chiffré. (Autrement dit, c'est un Attaque de texte chiffré uniquement « ciphertext-only attack », où l'attaquant ne voit qu'un seul texte chiffré.)

Dans un schéma parfaitement secret, l'observation de ce texte chiffré ne devrait avoir aucun effet sur la connaissance de l'adversaire concernant le message réel qui a été envoyé; autrement dit, la probabilité a posteriori qu'un message $m \in M$ ait été envoyé, conditionné par le texte chiffré qui a été observé, ne devrait pas être différente de la probabilité a priori que m serait envoyé. Cela signifie que le texte chiffré ne révèle rien sur le message envoyé sous-jacent et que l'adversaire n'apprend absolument rien sur le texte en clair qui a été chiffré. Formellement

Definition 1

Un cryptosystème $(\text{GEN}, \text{ENC}, \text{DEC})$ avec M l'espace de messages est parfaitement secret si pour toute distribution de probabilité sur M et $\forall m \in M$ et $c \in C$ tel que $\Pr[C=c]>0$ on a ;

$$\Pr[M = m \mid C = c] = \Pr[M = m].$$

(L'exigence que $\Pr[C = c] > 0$ est une condition nécessaire pour éviter le Conditionnement sur un événement à probabilité nulle.)

Nous pouvons voir que le chiffrement de César n'est pas parfaitement secret

Théorème du secret parfait, Shannon

Soit $(M, K, C, \text{GEN}, \text{ENC}, \text{DEC})$ un cryptosystème. On suppose que $\#M = \#K = \#C < \infty$ et que $\Pr[M=m] > 0 \forall m \in M$. Alors ce cryptosystème est à secret parfait Si seulement si

- La distribution des clés suit une loi uniforme
- pour tout clair m appartient à M

$$\phi_m = \begin{cases} K \rightarrow C \\ \kappa \rightarrow \text{ENC}_\kappa(m) \end{cases}$$

est une bijection

Démonstration

On suppose avoir secret parfait. On montre d'abord la bijectivité, puis l'équiprobabilité.

Bijectivité

S'il existe m tel que $\phi_m : k \mapsto \text{ENC}_k(m)$ est non surjective, il existe $c \in C$ tel que $\forall \kappa \in K, \text{ENC}_\kappa(m) \neq c$. En particulier $\Pr[C = c \mid M = m] = 0$, d'où l'on tire $\Pr[M = m \mid C = c] = 0 \neq \Pr[M = m] > 0$ impossible. Donc ϕ_m est surjective, et par égalité des cardinaux bijective.

Equiprobabilité

Soit $c \in C$ un chiffré fixé. Pour tout $m \in M$, on note $\kappa(m)$ l'unique clef telle que $\text{ENC}_{\kappa(m)}(m) = c$ (d'après la bijectivité). On a $\Pr[M = m \mid C = c] = \Pr[K = \kappa(m)]$

Puisque par hypothèse $\Pr[M = m \mid C = c] = \Pr[M = m]$, on obtient $\Pr[K = \kappa(m)] = \Pr[C = c]$. Or le chiffrement $m \mapsto \text{ENC}_k(m)$ est injectif à clef fixé, donc bijectif. Donc pour toute clef κ , il existe m tel que $\kappa = \kappa(m)$.

Ainsi, $\Pr[K = \kappa] = \Pr[K = \kappa(m)] = \Pr[C = c]$ est constante, et vaut $\frac{1}{\#K}$

Dans le sens réciproque, il suffit d'effectuer le calcul : par équiprobabilité des clefs

1.3 définition de sécurité pour la cryptographie à clé privée

Avant de parler de la sécurité, nous allons définir formellement ce que signifie être sûr pour un système cryptographique. La sécurité d'un cryptosystème est évaluée du fait qu'un adversaire efficace peut ou non différencier ou distinguer le chiffré de deux textes clairs dans une preuve par jeux donnée.

Fonction négligeable

Une fonction F de $\mathbf{N} \rightarrow \mathbf{R}$ est dite négligeable si pour tout polynôme P il existe $n_0 \in \mathbf{N}$ tel que pour tout $n > n_0$ on a :

$$F(n) < \frac{1}{P(n)}$$

Dans ce qui suit le terme on utilisera le terme « sécurité eav » pour désigner la sécurité du jeu en présence d'un espion comme noté dans la [8]

1.3.1 sécurité EAV

indistinguabilité

On considère une expérience noté PrivK^{eav} dans lequel un adversaire A émet deux messages m_0, m_1 et reçoit le chiffré d'un de ces messages. La définition de l'indistinguabilité stipule qu'un schéma E est sûr si aucun adversaire A ne peut déterminer lequel des deux messages a été chiffré on dit alors que le cryptosystème est indistinguable

Définition de l'expérience

L'expérience est définie pour un schéma de chiffrement à clé privée $E = (\text{GEN}, \text{ENC}, \text{DEC})$, un adversaire A , et une valeur n pour le paramètre de sécurité :

1) L'adversaire A émet deux messages $m_0, m_1 \in M$

2) Une clé κ est générée à l'aide de l'algorithme GEN , inconnue de l'adversaire de plus un bit aléatoire $b \in \{0, 1\}$ est choisi pour sélectionner m_0, m_1 . On calcule $c = \text{ENC}_\kappa(m_b)$ et on donne le chiffré c à A comme challenge

3) A émet alors un bit b' dans le but d'avoir $b' = b$

4) Le résultat de l'expérience est 1 si $b' = b$ et 0 sinon. Dans le cas où le résultat est 1 on dit que A a réussi et on note $\text{PrivK}^{eav}_{A,E} = 1$

Ainsi formellement nous pouvons définir la « sécurité EAV » comme suit

Définition 2

Un schéma de chiffrement à clé privée $E = (\text{GEN}, \text{ENC}, \text{DEC})$ est indistinguable sous une attaque eav (attaque en présence d'espion) si pour tout algorithme probabiliste, il existe une fonction négligeable negl telle que

$$\Pr[\text{PrivK}^{eav}_{A,E}(n) = 1] \leq \frac{1}{2} + \text{negl}(n)$$

Où la probabilité est prise aléatoirement et l'expérience également (c'est-à-dire le choix de la clé, du bit b ainsi que tout paramètre utilisé par ENC).

1.3.2 Attaque par texte clair choisi (Chosen plaintext Attack CPA)

Formellement, nous modélisons les attaques par texte choisi en donnant à l'adversaire A l'accès à un oracle de chiffrement, vu comme une « boîte noire » qui chiffre les

messages choisis par A à l'aide d'une clé k inconnue de lui. Autrement dit, nous imaginons que A a accès à un "oracle" $\text{ENC}_k(-)$; lorsque A interroge cet oracle en lui fournissant un message m comme entrée, l'oracle renvoie un texte chiffré $c \leftarrow \text{ENC}_k(m)$ comme réponse. (Si ENC est randomisé, l'oracle utilise un nouvel aléa chaque fois qu'il répond à une requête). L'adversaire peut interagir avec l'oracle de chiffrement de manière adaptative, autant de fois qu'il le souhaite.

Considérons l'expérience suivante définie pour tout schéma de chiffrement $E = (\text{GENC}, \text{ENC}, \text{DEC})$, l'adversaire A, et la valeur n pour le paramètre de sécurité :

- 1.) Une clé k est générée en utilisant GEN. L'adversaire A, qui ne connaît pas la clé, est autorisé à effectuer un nombre polynomial de requêtes à un oracle de chiffrement ENC_k .
- 2.) A émet deux messages $m_0, m_1 \in M$
- 3.) On choisit un bit aléatoire $b \in \{0, 1\}$. On calcule $c = \text{ENC}_k(m_b)$ et on donne le chiffré c à A comme le challenge
- 4.) A continue à avoir accès à l'oracle de chiffrement ENC_k .
- 5.) A sort un bit b' dans le but d'avoir $b' = b$
- 6.) La sortie de l'expérience est 1 si $b' = b$ et 0 sinon. Nous appelons le premier cas le succès de A et le désignons par $\text{PrivK}^{cpa}_{A,E} = 1$.

Définition 3

Un schéma de chiffrement à clé privée $E = (\text{GEN}, \text{ENC}, \text{DEC})$ est sûr en cas d'attaque par texte plat choisi (CPA) si, pour tous les adversaires probabilistes en temps polynomial A, il existe une fonction négligeable negl telle que

$$\Pr[\text{PrivK}^{cpa}_{A,E}(n) = 1] \leq \frac{1}{2} + \text{negl}(n)$$

proposition 1

Un schéma de chiffrement E dont l'algorithme de chiffrement ENC est déterministe ne peut pas être CPA-sûr

Démonstration

Considérons l'adversaire A suivant qui joue le jeu de sécurité CPA comme suit :

- 1.) Sélectionner deux messages aléatoires distincts $m_0, m_1 \in M$.
- 2.) Utiliser l'oracle de chiffrement pour obtenir les chiffrés $c_0 = ENC_{\kappa}(m_0)$ et $c_1 = ENC_{\kappa}(m_1)$.
- 3.) Sortir les deux messages $m_1, m_1 \in M$
- 4.) A la réception du texte chiffré c, si $c = c_0$, sortir le bit 0, sinon sortir le bit 1.

Comme ENC est déterministe, le texte chiffré du challenge c, étant un chiffrement de m_0 ou m_1 , sera égal à c_0 ou c_1 . Ainsi, A pourra réussir avec une probabilité non négligeable sur $\frac{1}{2}$ (en particulier, A réussira toujours). Ainsi, E ne peut pas être CPA-sûr

1.3.3 Attaque à chiffré choisi (Chosen Cipher text Attack)(CCA)

Qu'est-ce que cela signifie pour un schéma de chiffrement de résister contre les attaques à chiffre choisi ? Comme d'habitude pour définir une notion de sécurité appropriée, nous devons définir deux choses : la capacité supposé de l'attaquant et ce qui constitue une attaque réussie .Pour ce dernier nous suivrons l'approche adoptée dans les définitions précédentes .Nous allons donc donner à l'attaquant un texte chiffré c comme challenge issu des messages m_0 ou m_1 (chacun choisi avec une probabilité égale)et on considère que le schéma est cassé si l'attaquant peut déterminer lequel des deux messages a été chiffre avec une probabilité significative meilleur que $\frac{1}{2}$

L'attaque par chiffré choisi (CCA) permet à l'adversaire de disposer à la fois d'un oracle de chiffrement et de déchiffrement qu'il peut interroger à tout moment pendant le jeu.

D'autres textes distinguent les attaques par chiffré choisi non adaptative et les attaques par chiffres choisis adaptative .Les premières également appelé lunchtime permettent seulement à l'adversaire d'utiliser l'oracle de déchiffrement avant de recevoir le challenge (le chiffré) .La seconde un modèle d'attaque beaucoup plus puissant permet à l'adversaire de continuer d'utiliser l'oracle de déchiffrement après avoir reçu le challenge ,mais évidemment il n'est pas autorisé à interroger l'oracle de déchiffrement avec le challenge . Dans ce mémoire nous faisons allusion à la seconde cas lorsque nous parlerons d'attaque CCA Comme nous pouvons le voir la sécurité CCA est beaucoup plus forte que la sécurité CPA ou EAV et les implique tous les deux. Nous présenterons les définitions formelles ci-dessous

Considérons l'expérience suivante définie pour tout schéma de chiffrement $E = (GENC, ENC, DEC)$, l'adversaire A, et la valeur n pour le paramètre de sécurité :

1) Une clé est générée à l'aide de l'algorithme de génération des clés GEN. L'adversaire A qui ne connaît pas la clé, peut effectuer un nombre polynomial de requête à un oracle de chiffrement et de déchiffrement

2) A envoie deux messages m_0, m_1

3) On choisit un bit aléatoire $b \in \{0, 1\}$. On calcule $c = \text{ENC}_k(m_b)$ et on donne c à A comme challenge

4) A continue à avoir accès à l'oracle de chiffrement et de déchiffrement mais avec la restriction que A ne peut pas exécuter l'oracle de déchiffrement sur le challenge c

5) A produit un bit b' dans le but d'avoir $b' = b$

6) Le résultat de l'expérience est 1 si $b' = b$ et 0 sinon. Si $b = b'$ nous disons que A a réussi et nous le désignons par $\text{PrivK}^{cca}_{A,E} = 1$

.

Définition 3

Un schéma de chiffrement à clé privée $E = (\text{KeyGen}, \text{Enc}, \text{Dec})$ est (CCA) sûr si, pour tous les adversaires probabilistes en temps polynomial A, il existe une fonction négligeable negl telle que

$$\Pr[\text{PrivK}^{cca}_{A,E}(n) = 1] \leq \frac{1}{2} + \text{negl}(n)$$

1.4 Cryptographie à clé publique

1.4.1 Fonction à sens unique

Une fonction $f : A \rightarrow B$ est dite à sens unique s'il est facile à calculer $f(x)$ pour tout $x \in A$ (complexité polynomiale) et est difficile (complexité exponentielle) pour presque tout $y \in B$, de trouver x tel que $y = f(x)$.

Une fonction est à sens unique avec trappe si l'on connaît un secret permettant de l'inverser

1.4.2 Fonction de hashage

$\{0, 1\}^*$ ensemble des chaînes de longueur quelconque, $\{0, 1\}^l$ ensemble des chaînes longueur fixe avec $l \neq 0$.

définition 3

Une fonction $H : \{0, 1\}^* \rightarrow \{0, 1\}^l$ est dite fonction de hachage si :

1. Pour tout x , $H(x)$ est facile à calculer, $H(x)$ est appelé le hash ou l'empreinte de x
2. Étant donné $H(x)$, il est difficile de trouver y tel que $y = H(x)$ (fonction à sens unique) ;
3. Étant donné x , il est difficile de trouver x' tel que $x \neq x'$ et $H(x) = H(x')$, (collusions faibles) ;
4. Il est difficile de trouver x et x' (de son choix) $x \neq x'$ tel que $H(x) = H(x')$, (collusions fortes) ;

1.4.3 Modélisation du chiffrement à clés publique

Un schéma de chiffrement à clé publique de $E=(\text{GEN}, \text{ENC}, \text{DEC})$ avec un espace de clés $\mathcal{K} = \mathcal{K}_s \times \mathcal{K}_p$, un espace de message \mathcal{M} et un espace chiffré \mathcal{C} est un 3-tuple d'algorithme efficace dans lequel :

$\text{GEN} : \lambda \rightarrow \mathcal{K}$ algorithme de génération des clés

$\text{ENC}_\kappa : \mathcal{M} \rightarrow \mathcal{C}$ est une fonction à sens unique (avec trappe qui est un inverse à gauche) appelée fonction de chiffrement et qui dépend d'un paramètre κ appelé clé(publique)

$\text{DEC}_{\kappa'} : \mathcal{C} \rightarrow \mathcal{M}$ est la trappe et est appelée fonction de déchiffrement (dépendant de la clé κ') et on a $\text{DEC}_{\kappa'}(\text{ENC}_\kappa(m)) = m$

Il existe des définitions identiques pour la sécurité EAV, CPA et CCA pour les systèmes de chiffrements à clé publique. Nous renvoyons le lecteur à [16] pour les détails.

proposition 2

Un cryptosystème à clé publique est EAV-sûr si et seulement il est Cpa-sûr

Démonstration

Dans la version à clé publique du jeu de sécurité EAV, l'adversaire A a accès à la clé publique pk . Par conséquent, A peut calculer $\text{Enc}_{pk}(m)$ lui-même pour tout message m , ce qui est équivalent à donner à A l'accès à un oracle de chiffrement.

Par conséquent, pour les schémas de chiffrement à clé publique, la sécurité EAV est équivalente à la sécurité CPA.

CHAPTER 2

LE CHIFFREMENT HOMOMORPHE

2.1 Introduction

A présent que nous avons vu les notions préliminaires de la cryptographie, nous pouvons entamer la théorie du chiffrement homomorphe.

Dans ce chapitre nous introduisons le concept du chiffrement homomorphe de manière formelle mais aussi par analogie en utilisant l'exemple de la bijouterie présenté dans [6]. Ensuite nous allons parler de l'efficacité et de la sécurité du chiffrement homomorphe.

2.2 Concept du chiffrement homomorphe

2.2.1 Modélisation du chiffrement homomorphe

Définition

Un cryptosystème homomorphe $\varepsilon = (\text{GEN}, \text{ENC}, \text{DEC}, \text{Evaluate}, \mathcal{F})$ avec \mathcal{K} espace des clés, \mathcal{M} espace des textes clairs, \mathcal{C} espace des textes chiffrés et λ le paramètre de sécurité est un 5-uplet dont:

$\text{GEN} : \lambda \rightarrow \mathcal{K}$ est l'algorithme de génération des clés

$\text{ENC} : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{C}$ est l'algorithme de chiffrement

$\text{DEC} : \mathcal{K} \times \mathcal{C} \rightarrow \mathcal{M}$ l'algorithme de déchiffrement

$\text{Evaluate} : \mathcal{F} \times \mathcal{C}^* \rightarrow \mathcal{C}$ est la fonction d'évaluation qui prend en entrée une fonction autorisée $f \in \mathcal{F}$ et l'évalue sur un ensemble de texte chiffré (c_1, c_2, \dots, c_n) pour produire un autre chiffré c .

La définition ci-dessus stipule que si f représente une fonction de traitement alors on peut faire du traitement (évaluer) sur les données sans en avoir accès. Cela peut sembler paradoxal ou voir impossible, mais pour ce faire une idée de la solution, on peut considérer l'exemple suivant présenté dans l'étude de Gentry de 2010 [6].

Dans cet exemple Alice possède une bijouterie. Elle prend des matières premières dans son magasin (or, diamant...) et les transforme en coliers et bagues. Alice doit s'occuper de la commande de ses clients et elle est donc obligée à laisser à ses employés faire la transformation. Cependant Alice ne fait pas confiance à ses employés et elle craint qu'ils ne volent certaines matières premières à son insu. La solution d'Alice à cette énigme est la suivante. Elle crée une boîte en laquelle elle seule a accès grâce à une clé. La boîte est accompagnée de gants qui permettent de manipuler les objets qui s'y trouvent mais ne permettent pas de sortir quoi que soit de la boîte. Ensuite Alice met les matières premières dans la boîte et la ferme avec sa clé pour que son contenu ne puisse pas être retiré et demande à ses employés d'assembler les bijoux en utilisant les gants fournis par la boîte. Ainsi Alice utilise sa clé pour ouvrir la boîte et récupérer les bijoux assemblés.

Comme le note Gentry, cette analogie présente en effet des insuffisances, même si les employés ne peuvent pas sortir les matériaux, ils peuvent voir s'ils sont présents ou non et mieux ce que la boîte contient, ce qui ne correspond pas aux exigences de sécurité de la cryptographie. Cependant l'exemple démontre suffisamment l'idée principale du chiffrement homomorphe.

Une autre analogie, plus réaliste, est la suivante : supposons que la scolarité du département de LACGAA de l'UCAD conserve des dossiers détaillés sur tous ses étudiants et leurs résultats. Ainsi la scolarité décide de stocker toutes ces données chiffrées évidemment sur un service cloud. Ainsi les administrateurs de la scolarité veulent calculer à partir des données stockées sur le cloud la moyenne de chaque étudiant. Cependant ils ne veulent pas compromettre les données personnelles des étudiants en donnant au service tier (le cloud) la clé secrète.

Dans ce cas si les administrateurs utilisent le chiffrement homomorphe où la moyenne sera une fonction autorisée ($\mu \in F$), le service tier (cloud) pourra évaluer alors la moyenne des étudiants sur les données chiffrées et le renvoyer à son tour aux administrateurs de la scolarité qui pourront alors utiliser leur clé secrète pour déchiffrer la moyenne chiffrée des étudiants.

Notez que nous supposons que le service en cloud suit un modèle honnête mais curieux, ce qui signifie qu'il exécutera les instructions correctement (c'est-à-dire qu'il calculera la fonction de chiffrement souhaitée au lieu d'une fonction malveillante de son choix), mais s'il en a l'occasion, il essaiera d'obtenir des informations sur les données sous-jacentes chiffrées qu'il stocke.

Cependant, il existe un cryptosystème homomorphe trivial qui satisfait à notre déf-

inition du chiffrement homomorphe, où la fonction Evaluate ajoute simplement la description de la fonction aux chiffrés correspondants, et l'algorithme DEC se charge de l'évaluation de la fonction on a alors:

$$\text{Evaluate}(f, c_1, \dots, c_n) = (f, c_1, \dots, c_n) = c$$

L'algorithme DEC fait l'évaluation $\text{DEC}(c) = f(\text{DEC}(c_1), \dots, \text{DEC}(c_n))$

Definition 2

Un schéma de chiffrement est dit compact si le temps d'exécution du déchiffrement du chiffré évalué c (sortie de Evaluate) est égale au temps d'exécution du déchiffrement des chiffrés par lesquels il est créé c_i (entrée de Evaluate). La taille c ne peut pas être plus grande que la taille des c_i et le temps de déchiffrement de c doit être indépendant de la complexité de f , la fonction permise choisie.

définition 3

Un schéma de chiffrement homomorphe est dit à i -saut (i -hop homomorphe) s'il est possible d'appliquer le résultat de l'algorithme d'évaluation i fois de suite.

Une propriété intéressante que nous donne la compacité est que nous pouvons transformer un schéma de chiffrement homomorphe à clé privée en un schéma homomorphe à clé publique en particulier nous avons ce théorème suivant tiré de [15]

Théorème

Tout schéma de chiffrement à clé privée sécurisé à messages multiples qui est compact et homomorphe par rapport à l'addition modulo 2 peut être transformé en un schéma à clé publique sémantiquement sûr.

En particulier, si le schéma à clé privée permet $i + 1$ évaluations de la fonction d'évaluation homomorphique ($(i+1)$ -hop homomorphe) alors le schéma à clé publique correspondant permet i évaluations de la fonction d'évaluation homomorphique (i -hop homomorphe)

Démonstration

Voir [15] pour deux constructions explicites qui transforment de tels schémas de chiffrement à clé privée en schémas à clé publique.

2.3 Sécurité des cryptosystèmes homomorphes

Par rapport à la sécurité des schémas de chiffrement homomorphes, il est important de réfléchir sur le modèle d'attaque du serveur tiers (l'évaluateur) et voir ce qu'il est capable de faire. Dans l'exemple précédent nous avons supposé que le modèle est honnête mais curieux ce qui signifie que la partie tierce (serveur) suivra et exécutera les instructions correctement et essaiera d'en apprendre le plus possible sur les données chiffrées.

Notez qu'il y a d'autres modèles beaucoup plus adverses que nous pouvons considérer pour le serveur (par exemple, il ne calcule pas toujours la fonction appropriée), mais ces obstacles présentés peuvent être résolus en utilisant la délégation de calcul. Cette dernière permet d'établir un protocole pour qu'un tiers calcule une fonction et prouve efficacement qu'il a correctement calculé la fonction. Nous allons pas dans ce mémoire traiter la délégation des chiffrements homomorphes nous renvoyons le lecteur aux sources comme [18]

Une caractéristique des cryptosystèmes homomorphes qui a des implications importantes pour la sécurité est la malléabilité.

2.3.1 La Malléabilité

Définition 3

Un schéma de chiffrement E est malléable si, étant donné le texte chiffré d'un message, $c = ENC_K(m)$, il existe une fonction non constante f qui n'est pas la fonction d'identité, de sorte qu'un adversaire efficace A peut calculer le texte chiffré d'un certain message lié c' tel que $DEC_K(c') = f(m)$ avec une probabilité non négligeable.

Les cryptosystèmes homomorphes sont malléables par définition puisqu'ils permettent l'évaluation de diverses fonctions sur des chiffrés. Les cryptosystèmes malléables sont intrinsèquement limités dans la sécurité qu'ils offrent. Les chercheurs ont montré que plus que le système est malléable, plus qu'il est facile à casser. Comme les cryptosystèmes homomorphes sont malléables, nous avons donc la limitation suivante sur leur sécurité

proposition 1

Tout schéma de chiffrement malléable ne peut pas être sécurisé contre les attaques adaptatives par chiffré choisi (CCA).

Démonstration

Soit $E=(GEN,ENC,DEC,F,Evaluate)$ un cryptosystème homorphe malléable où pour un certain message m_0 , et pour tout $c \in$

$C_m = \{c \in C \mid \Pr[ENC_\kappa(m_0) = c] \text{ avec } \kappa \text{ et } ENC \text{ aléatoire}\}$ Comme E est malléable alors le message $f(m)$ a un chiffré de la forme $g(c)$ pour tout $c \in C_m$ où f, g peuvent être calculés en temps polynomial. Comme f est non constante, il existe un message m_1 tel que $f(m_1) \neq f(m_0)$. Alors, on considère l'adversaire A suivant :

- 1.) Soumet m_0 à l'oracle de chiffrement, et note $c_0 = ENC_\kappa(m_0)$
- 2.) Sort les messages $f(m_0)$ et $f(m_1)$
- 3.) A la réception du texte chiffré de défi c , A calcule $g(c_0)$. Si $c = g(c_0)$, alors la sortie $b = 0$. Sinon, tirer à pile ou face et sortir le résultat. Nous pouvons voir qu'un tel adversaire obtiendrait un avantage non négligeable, et donc notre schéma de chiffrement malléable E n'est pas sûr contre les attaques adaptatives par texte choisi.

À partir de là, nous nous rendons compte que, bien qu'il existe des schémas de chiffrement, même à clé publique, qui sont sûrs contre les attaques par texte chiffré choisi adaptatif [4], il est impossible de construire un schéma de chiffrement homomorphe qui satisfasse à cette exigence de sécurité. Cela dit, bien que les schémas de chiffrement homomorphes ne puissent pas formellement répondre à l'exigence de la CCA, nous pouvons les augmenter avec des méthodes de délégation de calcul qui peuvent aider à garantir que le serveur ne modifie pas de manière malveillante les chiffrés qu'il renvoie, ce qui est le type d'attaque qui motive le désir d'une sécurité CCA [18]. Il existe également de nombreuses d'autres normes de sécurité que les schémas de chiffrement homomorphes sont capables d'atteindre, comme la sécurité EAV et la sécurité CPA [8].

Les définitions de sécurité d'un chiffrement homomorphe sont les mêmes que celles d'un schéma de chiffrement simple. Pour un schéma de chiffrement régulier, imposant des conditions sur les opérations GEN, ENC, DEC , mais étant indépendantes de l'opération $Evaluate$.

CHAPTER 3

NOTION PRÉLIMINAIRES DE MACHINE LEARNING

3.1 Introduction

Dans cette section, nous allons voir quelques algorithmes de machine learning qui seront combinés ultérieurement avec les schémas de chiffrements homomorphes. Nous allons étudier les algorithmes d'apprentissage supervisé et non supervisé.

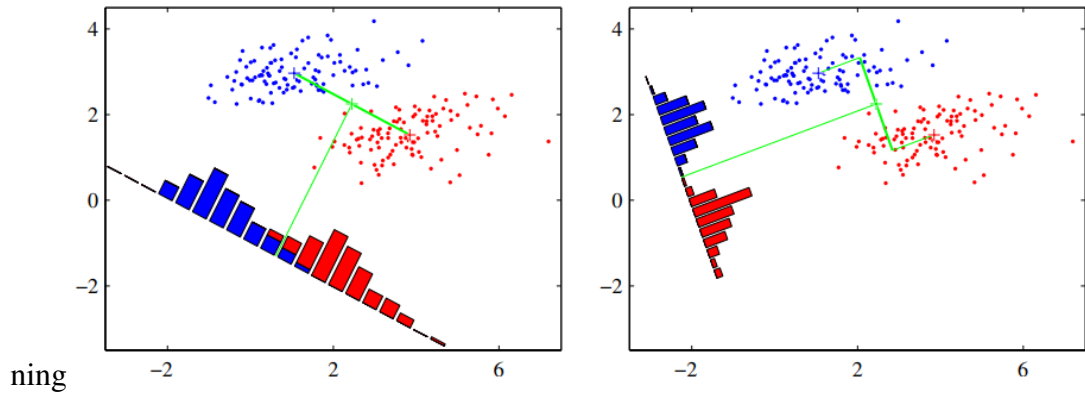
L'apprentissage supervisé consiste à déduire une fonction (modèle) à partir d'un ensemble de données étiquetées. Les deux types d'apprentissage supervisé sont la classification et la régression, le premier faisant référence au cas où les étiquettes forment un ensemble discret d'étiquettes de classe, et le second lorsque les étiquettes prennent des valeurs continues. Pour la classification, nous examinerons l'analyse discriminante linéaire, la classification naïve bayésienne et les arbres de décision.

D'autre part, l'apprentissage non supervisé consiste à déduire la structure cachée des données non étiquetées. Pour le cas de l'apprentissage non supervisé, nous allons voir l'analyse en composantes principales, qui est également un outil important pour la réduction de la dimension.

3.2 Classification binaire

La classification binaire est un problème consistant à prendre un vecteur de caractéristiques \mathbf{x} et à décider de laquelle des deux étiquettes (ou classes) il appartient. Par exemple on peut prendre un vecteur de caractéristiques contenant la taille et le poids d'une personne et essayer de prédire le sexe de la personne. Il existe de nombreux algorithmes qui traitent ce problème, notamment la régression logistique, les machines à vecteurs de support, etc. Dans cette section, nous examinerons la discrimination linéaire de Fisher, qui est une technique permettant d'effectuer une classification binaire lorsque les caractéristiques sont des quantités continues.

L'analyse discriminante linéaire de Fisher aborde le problème de la classification en trouvant un vecteur \mathbf{w} sur lequel nous pouvons projeter les données sur un vecteur tel qu'il est plus facile de définir un point qui sépare les deux classes avec une grande marge. Pour ce faire, nous choisissons un vecteur \mathbf{w} qui maximise le rapport entre les variances interclasse et intraclasse. Intuitivement, cela a du sens car nous voulons maximiser la séparation entre les différentes classes, tout en minimisant la séparation au sein de chaque classe. Une fois que nous avons ce vecteur \mathbf{w} , nous pouvons classer un point de données \mathbf{x} en calculant $\mathbf{w}^T \mathbf{x}$ et en le comparant au seuil c correspondant à notre point de séparation.



De manière plus formelle, considérons un problème à deux classes dans lequel il y a N_1 points de la classe C_1 et N_2 points de la classe C_2 , de sorte que les vecteurs moyens des deux classes sont donnés par:

$$u_1 = \frac{1}{N_1} \sum_{i=1}^n X_i, \text{ et } u_2 = \frac{1}{N_2} \sum_{i=1}^n X_i \quad (3.1)$$

La mesure la plus simple de la séparation des classes, lorsqu'elle est projetée sur \mathbf{w} , est la séparation des moyennes des classes projetées. Ceci suggère que nous pourrions choisir \mathbf{w} de manière à maximiser

$$u_2 - u_1 = \mathbf{w}^T (u_2 - u_1) \quad (3.2)$$

$$u_k = \mathbf{w}^\top(u_k) \quad (3.3)$$

La variance intra-classe des données transformées de la classe C_k est donc donnée par

$$\sigma_k^2 = \sum_{n \in C_k} (Y_n - u_k)^2 \quad (3.4)$$

où

$$Y_n = \mathbf{w}^\top x_n \quad (3.5)$$

Nous pouvons définir la variance intra-classe totale pour l'ensemble des données comme étant simplement $\sigma_1^2 + \sigma_2^2$. Le critère de Fisher est défini comme étant le rapport entre la variance interclasse et la variance interne à la classe et est donné par:

$$\begin{aligned} S &= \frac{\sigma^2(intra)}{\sigma^2(extra)} \\ &= \frac{(u_2 - u_1)^2}{\sigma_1^2 + \sigma_2^2} \\ &= \frac{(\mathbf{w}^\top(u_2 - u_1))^2}{(\sum_{i,c_i=1} (\mathbf{w}^\top(x_i - u_1)))^2 + (\sum_{i,c_i=2} (\mathbf{w}^\top(x_i - u_2)))^2} \\ &= \frac{\mathbf{w}^\top \mathbf{S}_B \mathbf{w}}{\mathbf{w}^\top \mathbf{S}_W \mathbf{w}} \end{aligned} \quad (3.6)$$

où \mathbf{S}_B est la matrice de covariance entre les deux classes C_1 et C_2 avec

$$\mathbf{S}_B = (u_2 - u_1)(u_2 - u_1)^\top \quad (3.7)$$

et \mathbf{S}_W la matrice de covariance intra-classe avec

$$\mathbf{S}_W = \sum_{i,c_i=1} (x_i - u_1)(x_i - u_1)^\top + \sum_{i,c_i=2} (x_i - u_2)(x_i - u_2)^\top \quad (3.8)$$

En suivant la présentation de [10], nous pouvons dériver par rapport à w pour obtenir

$$\frac{\partial S}{\partial w} = 0 \Rightarrow (w^T S_B w) S_W w = (w^T S_W w) S_B w \quad (3.9)$$

$S_B w$ est dans la même direction que $u_2 - u_1$ et que $(w^T S_B w)$ et $(w^T S_W w)$ sont des constantes. Puisque nous ne nous intéressons qu'à la direction et non au facteur de w , nous pouvons éliminer les constantes et multiplier les deux côtés par S_W^{-1} pour obtenir une solution optimale de w

$$w^* \propto S_W^{-1}(u_2 - u_1)$$

Une fois que le w optimale calculé, nous devons trouver un seuil de classification c . D'habitude c est pris comme étant le point médian entre les deux moyennes dans la direction donnée par w^* .

$$c = \frac{w^{*T}(u_2 - u_1)}{2} \quad (3.10)$$

Ainsi, étant donné un point x , on calcule $w^{*T}x$ et on classe x selon que cette valeur est supérieure ou inférieure à c (seuil).

3.3 Classification naïve bayésienne

Un classificateur Naïve Bayes est une méthode pour classer des vecteurs caractéristiques à valeur discrète. En particulier, étant donné un vecteur de données $x \in \{1, \dots, K\}^n$, où K est le nombre de valeurs possibles pour chaque variable caractéristique et D est le nombre de variable caractéristiques. Cela nous permet d'écrire la densité conditionnelle de classe comme suit :

$$P(x | C = c) = \prod_{j=1}^D P(x_j | C = c) \quad (3.11)$$

Les classificateurs de Naïve Bayes sont qualifiés de "naïves" car ils supposent l'indépendance des variables, ce qui est rarement le cas. Malgré cela, ils donnent parfois de bon résultat [5]

Comme son nom l'indique, les classificateurs Naïve Bayes constituent une autre façon d'aborder le problème de la classification. Si nous avons K classes possibles, alors le modèle se compose de classe de probabilité à priori $\{Pr(C = ci)\}_{i=1}^K$ (c'est-à-dire la probabilité à priori de la classe c_i) et des densités conditionnelles de classe $\{P(X = x | C = ci)\}_{i=1}^K$, c'est-à-dire la probabilité qu'un objet de la classe c_i possède un ensemble de variables caractéristiques x .

À titre d'exemple, supposons que nous ayons une liste de maladies $\{m_1, \dots, m_n\}$ et

un vecteur de symptômes (s_1, \dots, s_m) où $s_i=1$ si la personne présente le symptôme i et 0 si elle ne le présente pas. Le modèle de Naïve Bayes spécifie $P(s_i = 1|m_j) \forall i, j$ (c'est-à-dire la probabilité qu'une personne présente le symptôme i étant donné qu'elle est atteinte de la maladie j) et $P(m_j) \forall j$ (c'est-à-dire la probabilité qu'une personne soit atteinte de la maladie j). Ensuite, pour calculer la probabilité qu'une personne atteinte d'une maladie m_k présente un ensemble particulier de symptômes s'_1, \dots, s'_m , on calcule

$$P((s'_1, \dots, s'_m) | m_k) = \prod_{j=1}^m P(s_j = s'_j | m_k)$$

Avec ce modèle, la classification fonctionne alors en utilisant une règle de décision à posteriori maximale. Étant donné un point de données x , nous calculons la probabilité postérieure de son appartenance à chaque classe c_i et classons x dans la classe ayant la probabilité postérieure la plus élevée. Plus précisément, nous calculons.

$$\begin{aligned} c^* &= \arg \max_{i \in 1, \dots, k} P(C = c_i | X = x) = \arg \max_{i \in 1, \dots, k} \frac{P(X=x|C=c_i)P(C=c_i)}{P(X=x)} \\ &= \arg \max_{i \in 1, \dots, k} P(X = x | C = c_i)P(C = c_i) \end{aligned} \quad (3.12)$$

Notez que dans la dernière étape, puisque $P(X = x)$ est une constante pour toutes les classes le problème se ramène alors à chercher des numérateurs. Comme nous le verrons plus tard, le fait de ne pas avoir cette étape de division rend un classificateur de Naïve Bayes plus facile à mettre en œuvre sur un schéma de chiffrement partiellement homomorphe.

3.4 Régression linéaire

La régression linéaire est l'un des algorithmes de machine learning les plus fondamentaux. Nous modélisons une variable cible ou de sortie comme une fonction linéaire de variables d'entrée (qui ont vraisemblablement une certaine relation avec la variable cible), plus un terme d'erreur normalement distribué, formellement

$$y = \langle \beta, \mathbf{x} \rangle + \epsilon \quad (3.13)$$

Par exemple, nous pouvons modéliser l'espérance de vie d'une personne comme une fonction linéaire de son revenu et de son état de santé, donc si nous avons un vecteur $x = (\text{revenu}, \text{état de santé})$ qui contient le revenu d'une personne et son état de santé, nous pouvons alors calculer l'estimation de l'espérance de vie de cette personne en utilisant

$$esprancedevie = \langle \beta, \mathbf{x} \rangle + \epsilon \quad (3.14)$$

Maintenant la question est de savoir quelle β choisir? Pour résoudre β , on calcule le β qui minimise l'erreur des moindres carrés, c'est-à-dire

$$\beta = \arg \min_{\beta} \left(\frac{1}{2} \sum (y_i - \langle \beta, \mathbf{x}_i \rangle)^2 \right)$$

Le $\frac{1}{2}$ est pris par commodité lorsque nous prenons la dérivée. Si nous avons n caractéristiques et d points de données, alors X désigne la matrice de conception $d \times n$ dont la i -ème ligne et la j -ème colonne contiennent la valeur de la j -ème caractéristique du i -ème point de données, la solution du problème d'optimisation ci-dessus est la suivante

$$\beta^* = (X^T X)^{-1} X^T y \quad (3.16)$$

3.5 Analyse des composantes principales (ACP)

Un problème courant dans l'apprentissage automatique est la malédiction de la dimensionnalité. Il s'agit du phénomène selon lequel le volume d'un espace caractéristique augmente de manière exponentielle avec le nombre de dimensions (ex $\{0, 1\}^d$ a 2^d éléments). Par conséquent, de nombreux algorithmes de machine learning ne sont pas adaptés aux grandes dimensions. Une façon de résoudre ce problème consiste à réduire la dimensionnalité des données tout en préservant leur structure essentielle. Pour être plus précis, si nous voulons réduire la dimension à la taille L , nous voulons un ensemble de L vecteurs W de telle sorte que nous minimisions l'erreur de reconstruction, définie comme suit

$$J(W, Z) = \frac{1}{N} \sum_{i=1}^N \|x_i - \tilde{x}_i\|^2 \quad (3.17)$$

où $\tilde{x}_i = V z_i$ avec la contrainte que V soit orthonormé. Intuitivement V est une matrice de taille $D \times L$ qui renvoie z_i , représentation de faible dimension à la représentation de dimension D , z_i est la représentation de faible dimension de x_i calculée par $Z = V^T X$.

Le théorème suivant nous donne la solution optimale qui minimise l'erreur de reconstruction entre x et \tilde{x}_i

3.5.1 théorème (Eckart Young, 1936)

L'unique solution de $\min_{J,v}(Z,v)$ avec $J(Z,v) = \|X - vZ\|^2$, $\|v\| = 1$ est donné par $Z^* = v^{*T} X$ où v^* est le vecteur propre normalisé associé à la plus grande valeur propre λ de XX^T . de plus on a $\|Z^*\|^2 = \sqrt{\lambda}$

preuve

Voir [12] pour la démonstration

Ainsi, nous voyons que le problème de la réduction de la dimensionnalité via l'ACP peut être résolu en obtenant les vecteurs propres de la matrice de covariance empirique normalisée ayant les plus grandes valeurs propres absolues. En particulier, notre représentation à faible (L) dimension d'un vecteur de données x est donnée par $y = V_L x$ (où y a une dimension L). Le vecteur propre ayant la i -ème plus grande valeur propre absolue est appelé la i -ème composante principale.

PART II

SYSTÈME DE CHIFFREMENT HOMOMORPHE

CHAPTER 4

ETUDE DES CRYPTOSYSTEMES HOMOMORPHES

Dans ce chapitre, nous étudions la construction de plusieurs cryptosystèmes homomorphes. Pour démontrer que les schémas de chiffrement homomorphes ne sont pas aussi insaisissables qu'il n'y paraît, nous commençons par examiner comment l'algorithme RSA est simplement ou partiellement homomorphe. Ensuite, dans la section suivante, nous examinons le cryptosystème Paillier, qui est utilisé dans certaines des applications de machine learning que nous explorerons plus tard. Ensuite, nous présentons un schéma de chiffrement entièrement homomorphe [3] basé sur l'hypothèse d'apprentissage avec erreurs (learning with errors). Enfin, bien que nous ne présentions pas un schéma de chiffement homomorphe strictement basé sur des problèmes de treillis (lattice problems), nous montrons une réduction du problème du plus court vecteur de treillis (shortest vector lattice problems) à l'apprentissage avec erreurs avant de conclure par une brève section sur le codage des nombres réels.

4.1 cryptosystème simplement homomorphe de RSA

L'algorithme RSA conçu par Rivest, Shamir et Adleman en 1978 est largement utilisé aujourd'hui pour transmettre en toute sécurité de petites clés secrètes entre

les deux parties, qui peuvent ensuite être utilisées pour communiquer en toute sécurité avec des messages beaucoup plus importants grâce à des systèmes de chiffrement à clé privée efficaces.(chiffrement hybride)

Nous allons explorer les propriétés homomorphes que le RSA possède, même s'il n'a pas été spécifiquement conçu dans ce but. Tout d'abord, nous définissons formellement le RSA ci-dessous :

Algorithm 1 Algorithme de RSA [8]

procedure GEN((1^λ))

On Choisit 2 nombres premiers aléatoires p, q , et on calcule $N = pq$

On calcule $\phi(N) = (p - 1)(q - 1)$

On choisit $e > 1$ tel que $\text{pgcd}(\phi(N), e) = 1$ et on calcule $d = e^{-1} \bmod \phi(N)$

return (N, e, d)

procedure ENC((m, N, e))

return $c = m^e \bmod N$

end procedure

procedure DEC((c, N, d))

return $m = c^d \bmod N$

end procedure

Nous pouvons facilement voir que les schémas de chiffrement et de déchiffrement de RSA sont corrects puisque on a :

$$\begin{aligned} DEC_{(N,d)}(ENC_{(N,e)}(m)) &= DEC_{(N,d)}(m^e \bmod N) \\ &= m^{de} \bmod N \\ &= m \bmod N \end{aligned}$$

De plus, notez que l'algorithme RSA prend pour acquis que nous pouvons générer des nombres premiers de λ -bit pour tout $\lambda \in \mathbb{Z}^+$. Ce n'est certainement pas une hypothèse triviale, mais de nombreux schémas simples qui génèrent des nombres de λ -bit aléatoires et utilisent des tests de primalité comme le test de Miller-Rabin fonctionnent bien en pratique [8]. Une discussion plus approfondie des méthodes permettant de générer des nombres premiers de λ -bit dépasse le cadre de ce mémoire, nous supposons donc que nous pouvons générer efficacement de tels nombres premiers.

Dans ce qui , nous montrerons que RSA est un schéma de chiffrement homomorphe valide qui supporte la multiplication modulo N.

Proposition1

RSA est un schema de chiffrement valide et partiellement homomorphe. En particulier, la multiplication des messages chiffrés modulo N correspond à la multiplication des messages clairs modulo N.

Démonstration

Soit $ENC_{(N,e)}$ l'algorithme de chiffrement et m_1 et $m_2 \in \mathbb{Z}_N$ deux messages,alors on a

$$\begin{aligned} ENC_{(N,e)}(m_1)ENC_{(N,e)}(m_2) &= m_1^e . m_2^e \text{ mod } N \\ &= (m_1 m_2)^e \text{ mod } N \\ &= ENC_{(N,e)}(m_1 m_2) \end{aligned}$$

Ainsi, on peut voir que multiplier les chiffrés modulo N correspond à multiplier les messages en clair modulo N.

Malheureusement, même si nous prenons l'hypothèse RSA comme acquise, il existe un certain nombre d'attaques sur le RSA qui le rendent peu sûr [8]. En fait, comme le RSA est déterministe, il ne peut pas être CPA-sûr. Cependant, nous pouvons au moins voir que l'idée d'un chiffrement homomorphe est réalisable car RSA est capable d'obtenir l'homomorphisme sans que ce soit son objectif.

4.2 Cryptosystème de Paillier

Dans cette section, nous explorons un schéma de chiffrement homomorphe additif proposé par Paillier en 1999 [13].Ce schéma utilise le problème de la classe de résidu composite, qui est étroitement lié au système RSA.Bien qu'il ne soit pas totalement homomorphe, le schéma de chiffrement de Paillier est utilisé dans de nombreuses applications de machine learning que nous explorons dans ce mémoire. Une caractéristique intéressante du cryptosystème de Paillier est qu'il est homomorphe sur un grand nombre de messages , c'est-à-dire sur \mathbb{Z}_N . Un exemple courant de cas d'utilisation du cryptosystème de Paillier est le calcul du nombre de votes chiffrés, puisque les sommes résultantes peuvent atteindre des totaux élevés [8].Depuis sa création, le cryptosystème de Paillier a également subi des améliorations pour le rendre plus efficace .Nous décrivons l'algorithme ci-dessous (Algorithme 2), nous illustrons la réduction à RSA et nous donnons une preuve de sécurité, en suivant les exposés de [13], [8]. En particulier, nous

présentons de nombreux détails de [13] mais pour la clarté de la présentation, nous utilisons la notation et la terminologie de [8].

4.2.1 Le système de chiffrement

Pour le cryptosystème de Paillier, l'espace de message $M = \mathbb{Z}_N$, l'espace de clés $\mathcal{K} = \mathcal{K}_{pk} \times \mathcal{K}_{sk}$ où $\mathcal{K}_{pk} = \mathbb{Z}_N$ et l'espace des chiffrés $\mathcal{C} = \mathbb{Z}_{N^2}$

Algorithm 2 Algorithme de Paillier [8], [13]

procedure GEN((1^λ))

On Choisit 2 nombres premiers aléatoires p, q , de λ -bit

On calcule $N=pq$ et $\phi(N)=(p-1)(q-1)$

return $(N, \phi(N))$

end procedure

procedure ENC((m, N))

$r \leftarrow \mathbb{Z}_{N^*}$

$c = (1 + N)^m \cdot r^n \pmod{N^2}$

return c

end procedure

procedure DEC($(c, N, \phi(N))$)

Calculer $\tilde{c} = c^{\phi(N)} \pmod{N^2}$

Calculer $\tilde{m} = \frac{\tilde{c}-1}{N}$

Calculer $m = \tilde{m} \cdot \phi(N^{-1}) \pmod{N^2}$

return m

end procedure

Pour démontrer l'exactitude du cryptosystème de Paillier, nous examinons l'espace du texte chiffré : le groupe $\mathbb{Z}_{N^2}^*$ sous multiplication modulo N^2 et introduisons la notion de N^{ieme} résidus, qui sont importants pour le cryptosystème de Paillier.

Un entier $z \in \mathbb{Z}_{N^2}^*$ est un N^{ieme} résidus modulo N^2 , s'il existe $y \in \mathbb{Z}_{N^2}^*$ tel que

$$z \equiv y^N \pmod{N^2}$$

(4.1)

Pour revenir à la démonstration de l'exactitude du cryptosystème, nous devons prouver que le schéma de chiffrement est univoque et que le système de déchiffrement est correct. La première est importante car si deux plaintexts sont mis en correspondance avec le même ciphertext, alors l'algorithme de déchiffrement ne pourra pas déchiffrer correctement. Donc, fixons un N^{ieme} résidu $g \in \mathbb{Z}_{N^2}^*$ et Notons ϵ_g la fonction

$$\epsilon_g : \mathbb{Z}_N \times \mathbb{Z}_N^* \longrightarrow \mathbb{Z}_{N^2}^* \quad (4.2)$$

$$(x, y) \longrightarrow g^x \cdot y^N \pmod{N^2} \quad (4.3)$$

lemme 1

Si l'ordre de $g \in \mathbb{Z}_{N^2}^*$ est un multiple non nul de N , alors ϵ_g est bijective

preuve

Nous allons d'abord montrer que $\mathbb{Z}_N \times \mathbb{Z}_N^*$ et $\mathbb{Z}_{N^2}^*$ ont le même nombre d'éléments, $\mathbb{Z}_N \times \mathbb{Z}_N^*$ a $N \phi(N)$ éléments avec ϕ la fonction d'Euler.

Soit $N = \prod_i p_i^{e_i}$ la décomposition en produit de facteur premier de N . notez que

$$\begin{aligned} \phi(N) &= \prod_i \phi(p_i^{e_i}) \text{ (est multiplicative)} \\ &= \prod_i p_i^{e_i-1} (p_i - 1) \end{aligned}$$

Ainsi on a,

$$\begin{aligned} \phi(N^2) &= \prod_i \phi(p_i^{2e_i}) \\ &= \prod_i p_i^{2e_i-1} (p_i - 1) \\ &= \left(\prod_i p_i^{e_i} \right) \left(\prod_i p_i^{e_i-1} (p_i - 1) \right) \\ &= N \phi(N) \end{aligned}$$

Ainsi $\mathbb{Z}_N \times \mathbb{Z}_N^*$ et $\mathbb{Z}_{N^2}^*$ ont le même nombre d'éléments, il suffit alors de montrer que ϵ_g est injective.

Supposons que $g^{x_1} \cdot y_1^N = g^{x_2} \cdot y_2^N \pmod{N^2} \implies g^{x_2-x_1} \left(\frac{y_2}{y_1} \right)^N \equiv 1 \pmod{N^2}$.

soit λ le plus grand entier pour lequel il existe un élément de $\mathbb{Z}_{N^2}^*$ d'ordre λN . Ainsi $(\frac{y_2}{y_1})^{\lambda N} \equiv 1 \pmod{N^2} \implies g^{\lambda(x_2-x_1)} \equiv 1 \pmod{N^2}$. Ceci implique que $\lambda(x_2 - x_1)$ est un multiple de l'ordre de g et donc un multiple de N . Ainsi $\text{Pgcd}(\lambda, N) = 1 \implies x_2 - x_1 \equiv 0 \pmod{N}$ donc $(\frac{y_2}{y_1})^N \equiv 1 \pmod{N^2}$ qui admet une solution unique $\frac{y_2}{y_1} = 1$ sur $\mathbb{Z}_{N^2}^*$. d'où $x_1 - x_2 = 0$ et $y_1 - y_2 = 0$ et ϵ_g est bijective

Il découle alors de la bijectivité que l'algorithme de chiffrement ne fait pas correspondre deux messages distincts sur le même texte chiffré. Quant à l'exactitude de l'algorithme de déchiffrement du cryptosystème de Paillier, nous avons la proposition suivante

Proposition 2

L'algorithme de déchiffrement du cryptosystème Paillier est correct. Plus précisément, pour un certain $m \in \mathbb{Z}_N$, alors $\text{DEC}_{\phi(N)}(\text{ENC}_N(m)) = m$

démonstration

Observons que

$$\begin{aligned} \tilde{c} &= c^{\phi(N)} \pmod{N^2} \\ &= (1 + N)^{m\phi(N)} r^{n\phi(N)} \pmod{N^2} \\ &\Leftrightarrow (m\phi(N) \pmod{N}, r^{\phi(N)} \pmod{N}) \text{ (bijection)} \\ &= (m\phi(N) \pmod{N}, 1) \text{ (} r^{\phi(N)} = 1 \pmod{N}, \text{ si } r \text{ et } N \text{ sont premiers entre eux)} \\ &\Leftrightarrow (1 + N)^{m\phi(N)} \pmod{N^2} \text{ (bijection)} \\ &= 1 + m\phi(N)N \pmod{N^2} \end{aligned}$$

Ainsi, on obtient

$$\left(\frac{\tilde{c} - 1}{N}\right) \cdot \phi(N)^{-1} = m$$

Ainsi le schéma de déchiffrement est donc correcte

Maintenant que nous avons vérifié l'exactitude des algorithmes de chiffrement et de déchiffrement, nous pouvons passer à l'examen de la sécurité du cryptosystème de Paillier. Le problème difficile sur lequel le cryptosystème de Paillier est basé est le problème de résidu composite décisionnel, qui consiste à distinguer entre les éléments uniformes de $\mathbb{Z}_{N^2}^*$ et les n^{ieme} résidus de $\mathbb{Z}_{N^2}^*$. Plus formellement, nous définissons ce que signifie le fait que le problème de résidu composite décisionnel soit difficile, en utilisant la définition de [8].

Définition 1

Le problème de la résiduosit  composite d cisionnelle est difficile si, pour tous les algorithmes probabilistes en temps polynomial D, il existe une fonction n gligeable $negl$ telle que pour r choisi al atoirement parmi $\mathbb{Z}_{N^2}^*$

$$|Pr[D(N, r^{N \bmod N^2}) = 1] - Pr[D(N, r) = 1]| \leq negl(n)$$

Dans la pr sentation originale de ce sch ma par Paillier, il explore le probl me de la n^{ieme} classe de r sidu et d montre que sa r solution nous donne un distingueur pour le probl me de r sidu composite d cisionnel ci-dessus. La n^{ieme} classe de r sidu est d finie comme suit :

D finition 2

Soit $g \in \mathbb{Z}_{N^2}^*$ ayant un ordre non nul multiple de N . Nous appelons la N^{ieme} classe de r sidu de w par rapport   g , l'unique entier $[[w]]_g \in \mathbb{Z}_n$ pour lequel il existe y tel que

$$\epsilon_g([w]_g, y) = ([w]_g)^g \cdot y^N = w$$

Notez que ce probl me est bien d fini car $[[w]]_g$ est unique, ce qui vient du fait que ϵ_g est bijective (par le Lemme).

4.2.2 R duction en RSA

D finition 3

Le N^{ieme} probl me de classe de r sidu de la base g , $Classe[N, g]$ est le probl me suivant.  tant donn  $w \in \mathbb{Z}_{N^2}^*$ et une base g , calculer $[[w]]_g$.

Le r sultat suivant nous montre comment le probl me de la N^{ieme} Classe de R siduosit  nous donne un distingueur D pour le probl me de la r siduosit  composite d cisionnelle.

lemme 2

$[[w]]_g = 0$ si et seulement si w est un N^{ieme} r sidu modulo N^2

Démonstration

Voir [13] pour la démonstration.

Ainsi, si nous pouvions résoudre le problème de la N^{ieme} classe de résidu, un simple distingueur D pourrait prendre r et calculer $[[r]]_g$ pour un certain g approprié. Si le résultat est 0, alors il sort 1 (en devinant que c'est un N^{ieme} résidu). Sinon, il produit 0. Il est facile de voir que cela présente un avantage non négligeable. En effet, par le lemme 1 et le lemme précédent, une proportion non triviale d'éléments de $\mathbb{Z}_{N^2}^*$ ne sont pas des N^{ieme} résidus, c'est-à-dire exactement une proportion $\frac{N-1}{N}$ d'entre eux.

Jusque ici, une instanciation particulière du problème de la N^{ieme} classe de résidu dépend de plusieurs paramètres, à savoir N , g et w . D'après les deux lemmes suivants, il s'avère que tous ces paramètres n'influencent pas la difficulté du problème. Tout d'abord, notons que dans la construction de ce distingueur, nous n'avons pas précisé quelle valeur de g choisir. Le lemme suivant nous montre que cela n'a pas d'importance.

lemme 3

La classe $[N, g]$ est aléatoirement auto-réductible sur tous les g' dont l'ordre est un multiple non nul de N .

démonstration

Voir [13] pour la démonstration

Une autre propriété intéressante de la classe $[N, g]$ est le fait qu'elle est aléatoirement auto-réductible. Plus précisément, nous pouvons réduire une instance du problème pour une valeur $w \in \mathbb{Z}_{N^2}^*$ en une instance aléatoire $w' \in \mathbb{Z}_{N^2}^*$ tiré d'une distribution uniforme.

lemme 4

La classe $[N, g]$ est aléatoirement auto-réductible sur $w \in \mathbb{Z}_{N^2}^*$

démonstration

En effet, on peut facilement transformer tout $w \in \mathbb{Z}_{N^2}^*$ en une instance aléatoire $w' \in \mathbb{Z}_{N^2}^*$ avec une distribution uniforme, en posant $w' = wg^ab \pmod{N^2}$ où a et b sont pris uniformément au hasard sur \mathbb{Z}_N (l'événement $b \notin \mathbb{Z}_N$ se produit avec une probabilité négligeable). Après avoir calculé $[[w']]_g$, il suffit de retourner

$$[[w]]_g = [[w']]_g - a \bmod N.$$

Ainsi, nous voyons que nos choix de $w \in \mathbb{Z}_{N^2}^*$ et g un N^{ieme} résidu modulo N n'affectent pas la difficulté du problème de la N^{ieme} classe de résidu. Par conséquent, il nous suffit de choisir le paramètre N , et nous pouvons donc désigner une instantiation de ce problème par $Classe[N]$.

Théorème (Réduction en RSA)

La classe $[N]$ est réductible en RSA

Démonstration

Nous savons que les éléments de la classe $[N, g]$ avec un ordre non nul multiple de N sont équivalentes à g . En utilisant $g = 1 + N$, nous avons seulement besoin de montrer que

$$RSA \Rightarrow Class[N, 1 + N]$$

Supposons maintenant que nous ayons un oracle S pour RSA .

On a $c = (1+N)^m y^N \bmod N^2$ avec $m \in \mathbb{Z}_N$ et $y \in \mathbb{Z}_N^*$. En développant $(1+N)^m$ (avec la loi de binome) et en appliquant la réduction modulo N on obtient $c = y^N \bmod N$ et on peut calculer y en utilisant l'oracle S . On a alors $y = S(c \bmod N)$ ainsi nous avons

$$\begin{aligned} c &= (1 + N)^m y^N \bmod N^2 \\ \frac{c}{y^N} &= (1 + N)^m \bmod N^2 \\ &= 1 + mN \bmod N^2 \end{aligned}$$

Ainsi on pourra calculer $m = [[c]]_{1+N}$, comme souhaité, ce qui résout la classe $[N]$

Ainsi, nous avons montré que le problème décisionnel du cryptosystème de Paillier et RSA sont des schémas de chiffrement étroitement liés en ce sens et que le premier est au plus aussi difficile que la factorisation du RSA. Cependant, cela ne le rend pas automatiquement sûr. Ci-dessous, nous montrons que le cryptosystème de Paillier est CPA-sécurisé en supposant que le problème de résidu composite décisionnel est difficile, en utilisant une preuve de [8].

4.2.3 Preuve de sécurité

Avant de présenter la preuve de la sécurité, nous montrons un résultat standard de la théorie des groupes.

proposition 3

Si G est un groupe fini et que $g \in G$ soit choisi arbitrairement. Alors, si on choisit une distribution uniforme $k \in G$, alors la distribution de $g \circ k$ sur G est également uniforme.

Démonstration

On a pour tout $g' \in G$, il existe un $k \in G$ (c-à-d $k = g^{-1} \circ g'$) tel que
 $g \circ k = g \circ g^{-1} \circ g' = g'$

Maintenant, nous pouvons prouver la sécurité CPA conditionnelle du cryptosystème de Paillier.

Théorème

Si le problème de résidu composite décisionnel est difficile, alors le cryptosystème de Paillier est CPA-sécurisé.

démonstration

Soit $\epsilon = (GEN, ENC, DEC)$ le cryptosystème de Paillier. Comme le système de Paillier est un schéma de chiffrement à clé publique, il suffit de montrer que E est sécurisé par EAV. Alors, par la proposition 2 du chapitre 1, cela implique qu'il est sécurisé par CPA. Nous allons construire une réduction du problème de résidu composite décisionnel pour casser la sécurité EAV pour le cryptosystème de Paillier. Ensuite, comme nous supposons que le problème de résidu composite décisionnel est difficile, nous ne devons pas être en mesure de casser la sécurité EAV pour le cryptosystème de Paillier. Soit \mathbb{A} un adversaire probabiliste en temps polynomial. Considérons l'algorithme D suivant qui tente de résoudre le problème de résidu composite décisionnel en se déroulant comme suit et prenant en entrée N , et y

- 1.) Définir $pk = N$ et exécuter $\mathbb{A}(pk)$ pour obtenir deux messages $m_0, m_1 \in M$
- 2.) Choisir un bit uniforme $b \in \{0, 1\}$ et calculer le chiffré $c = (1 + N)^{m_b} \cdot y \bmod N^2$
- 3.) On donne le chiffré c à \mathbb{A} comme challenge et \mathbb{A} produit un bit b' . Si $b=b'$ on sort 1 sinon 0

Premièrement, si l'entrée de D avait un y qui a été tiré de en prenant un aléa $r \in \mathbb{Z}_{N^2}^*$ et en l'élevant à la puissance N , on aura alors $c = (1 + N)^{m_b} \cdot r^N \bmod N^2$. Dans ce cas, nous notons que si \mathbb{A} a été capable de distinguer correctement les chiffrés de m_0 et m_1 (c'est-à-dire si $b = b'$), cela est équivalent à ce que D réussisse à déterminer si y a été tiré uniformément de $\mathbb{Z}_{N^2}^*$ ou de $Res(N^2)$. Par conséquent,

nous avons que.

$$Pr[D(N, r^N \bmod N^2) = 1] = Pr[PubK^{eav}_{A,\epsilon}(n) = 1]$$

Maintenant, on observe que si notre entrée à D avait un y qui était tiré au hasard de $\mathbb{Z}_{N^2}^*$, alors par la Proposition 3 précédent, le texte chiffré c est distribué uniformément dans $\mathbb{Z}_{N^2}^*$, indépendamment de m. Ainsi, la probabilité que $b' = b$ est exactement $\frac{1}{2}$ et en particulier

$$Pr[D(N, r) = 1] = \frac{1}{2}$$

Maintenant, en utilisant l'hypothèse que le problème de résidu composite décisionnel est dur, nous savons alors qu'il existe une fonction négligeable $negl$ telle que

$$|Pr[D(N, r^N \bmod N^2) = 1] - Pr[D(N, r) = 1]| \leq negl(n)$$

en remplaçant on a

$$|Pr[PubK^{eav}_{A,\epsilon}(n) = 1] - \frac{1}{2}| \leq negl(n)$$

et donc

$$|Pr[PubK^{eav}_{A,\epsilon}(n) = 1]| \leq \frac{1}{2} + negl(n)$$

Cela montre donc que le cryptosystème de Paillier est sécurisé par EAV, et donc sécurisé par CPA.

4.3 Chiffrement entièrement homomorphe à partir de l'apprentissage avec erreurs (LWE)

Jusqu'à présent, les systèmes que nous avons examinés (c'est-à-dire RSA et le cryptosystème Paillier) étaient partiellement homomorphes. Les cryptosystèmes partiellement homomorphes sont limités dans la mesure où ils ne supportent pas l'évaluation de toutes les fonctions f possibles sur les textes chiffrés. Cette restriction limite la mesure dans laquelle ils peuvent être utilisés, comme dans les cas où nous ne savons pas quelles opérations spécifiques seront nécessaires. En effet, nous aimerions pouvoir exécuter des algorithmes de machine learning de pointe sur des données chiffrées sans avoir à modifier le schéma de chiffrement chaque fois que nous devons prendre en charge une nouvelle opération, ou restreindre nos

algorithmes à un certain ensemble d'opérations.

D'autre part, un schéma de chiffrement totalement homomorphe nous permettrait d'évaluer des fonctions arbitraires sur les données. Le premier schéma de chiffrement totalement homomorphe a été proposé par Gentry [6], en utilisant une variante plus forte de l'hypothèse d'apprentissage avec erreurs pour les treillis idéaux et une hypothèse supplémentaire sur les sommes de sous-ensembles épars. Nous ne couvrons pas la construction de Gentry dans ce mémoire, mais notons qu'elle a servi de base à de nombreux schémas de chiffrement totalement homomorphes construits par la suite.

Bien que le fait que nous puissions construire des schémas de chiffrement totalement homomorphes soit un énorme succès, cette flexibilité n'est pas sans inconvénients. En effet, comme indiqué dans [9], même si des solutions pour le chiffrement totalement homomorphe ont été constamment proposées et améliorées, les schémas de chiffrement totalement homomorphes semblent être loin d'être suffisamment efficaces pour être utilisés dans la pratique. Cependant, les schémas de chiffrement partiellement homomorphes ont tendance à être beaucoup plus rapides, ce qui justifie leur utilisation dans de nombreuses applications de machine learning. Quoi qu'il en soit, nous poursuivons dans ce qui suit une exploration du schéma de chiffrement totalement homomorphe de [3], basé sur les hypothèses $\text{LWE}(\text{learning with error})$.

4.3.1 Le problem LWE (learning with error)

LWE est le problème difficile sur lequel repose le schéma de chiffrement entièrement homomorphe de [3]. De manière informelle, cette hypothèse stipule que, étant donné \mathbf{u} et $\langle \mathbf{u}, \mathbf{v} \rangle$ il est difficile de résoudre \mathbf{v} si un petit terme de bruit est ajouté à $\langle \mathbf{u}, \mathbf{v} \rangle$. Nous donnons une définition formelle ci-dessous.

Définition 1

Étant donné un paramètre de sécurité λ , soit $n = n(\lambda)$ une dimension entière, $q = q(\lambda) \geq 2$ un entier et $\chi = \chi(\lambda)$ une distribution sur \mathbb{Z} . Le problème de l'apprentissage avec erreurs (LWE) est de distinguer les deux distributions suivantes :

- 1) Dans la première distribution, on échantillonne (\mathbf{a}_i, b_i) uniformément de \mathbb{Z}^{n+1}_q
- 2) Dans la deuxième distribution, on tire d'abord $s \leftarrow \mathbb{Z}^n_q$ uniformément, puis on échantillonne $(\mathbf{a}_i, b_i) \in \mathbb{Z}^{n+1}_q$ en échantillonnant $\mathbf{a}_i \leftarrow \mathbb{Z}^n_q$ uniformément, $e_i \leftarrow \chi$, et en fixant $b_i = \langle \mathbf{a}_i, s \rangle + e_i$.

L'hypothèse d'apprentissage avec des erreurs (LWE) est alors que le problème ci-dessus est irréalisable.

4.3.2 Généralités sur le schéma : La méthode des vecteurs propres approximatifs

L'idée générale de ce schéma de chiffrement est de réaliser un chiffrement homomorphe basé sur LWE où les opérations d'addition et de multiplication correspondent directement à l'addition et à la multiplication de matrices. L'idée générale est qu'étant donné un message m et une clé secrète v avec au moins un coefficient v_i grand, le texte chiffré est alors une matrice C telle que

$$C.v = m.v + e$$

avec e un vecteur d'erreur faible

Par conséquent, la clé secrète v est un vecteur propre approximatif du texte chiffré C avec une valeur propre égale au message m . Pour déchiffrer, nous allons extraire la i ème ligne de C (correspondant à la plus grande entrée dans v , v_i), et calculons

$$x = \langle C_i, v_i \rangle = m.v_i + e_i \implies m = \left\lfloor \frac{x}{v_i} \right\rfloor$$

Comme l'erreur e_i est faible par rapport au coefficient v_i , elle n'affectera pas l'arrondi de $\frac{x}{v_i}$ et ne faussera pas le déchiffrement de m . Par conséquent, le schéma de chiffrement sera correct.

4.3.3 Outils pour le chiffrement homomorphe

D'après la Généralité précédente, nous pouvons voir qu'il est important de garder le terme d'erreur borné. Par conséquent, nous définissons ce que signifie garder un texte chiffré fortement limité et nous introduisons certaines méthodes dans l'algorithme 3 qui nous permettront de maintenir cette propriété. Pour la notation, supposons que nous ayons un vecteur \mathbf{a} , alors soit $a_{i,j}$ le j ème bit (où $j = 0$ est le bit le moins significatif) de l'élément i du vecteur.

Définition 2

Un texte chiffré C est B -fortement-limité si son message associé m , et les coefficients de C ont une norme d'au plus 1, tandis que les coefficients de son terme d'erreur e ont une norme d'au plus B

Les méthodes ci-dessous seront utilisées dans les algorithmes de chiffrement, de déchiffrement et d'évaluation homomorphes pour garder nos chiffrés fortement

limités. *BitDecomp* renvoie simplement les coefficients binaires pour chaque entrée d'un vecteur \mathbf{a} . BitDecomp^{-1} ramène ces coefficients au vecteur \mathbf{a} , mais notez que les entrées de *BitDecomp*-1 ne doivent pas nécessairement être dans $\{0, 1\}$. *Flatten* utilise ces deux opérations pour accepter des coefficients binaires mal formés (c'est-à-dire qui ne sont pas dans $\{0, 1\}$) et les représenter avec des coefficients binaires valides sans changer les entrées du vecteur sous-jacent. Enfin, *PowersOf2* crée simplement des copies de chaque entrée multipliée par des puissances de 2

Algorithm 3 Algorithme de *BitDecomp*, *Flatten*, *Powersof2*

procedure BITDECOMP((\mathbf{a}))

return $(a_{1,0}, \dots, a_{1,l-1}, \dots, a_{k,0}, a_{k,l-1})$

end procedure

procedure BITDECOMP $^{-1}((a_{1,0}, \dots, a_{1,l-1}, \dots, a_{k,0}, a_{k,l-1}))$

RETURN $(\sum_j a_{1,j}, \dots, \sum_j a_{k,j})$

END PROCEDURE

PROCEDURE FLATTEN($(a_{1,0}, \dots, a_{1,l-1}, \dots, a_{k,0}, a_{k,l-1})$)

RETURN $\text{BITDECOMP}(\text{BITDECOMP}^{-1}(a_{1,0}, \dots, a_{1,l-1}, \dots, a_{k,0}, a_{k,l-1}))$

END PROCEDURE

PROCEDURE POWERSOF2($(\mathbf{2})$)

RETURN $(b_1, 2b_1, \dots, 2^{l-1}b_1, \dots, b_k, 2b_k, \dots, 2^{l-1}b_k)$

END PROCEDURE

Quelques propriétés simples à noter à propos de ces méthodes

1) $\langle \text{BitDecomp}(\mathbf{a}), \text{Powersof2}(\mathbf{b}) \rangle = \langle \mathbf{a}, \mathbf{b} \rangle$

2) Pour \mathbf{a} de dimension N , Si $\mathbf{a}' = \text{BitDecomp}(\mathbf{a})$, $\langle \mathbf{a}', \text{Powersof2}(\mathbf{b}) \rangle = \langle \text{BitDecomp}^{-1}(\mathbf{a}'), \mathbf{b} \rangle = \langle \text{Flatten}(\mathbf{a}'), \text{Powersof2}(\mathbf{b}) \rangle$

Comme *Flatten* convertit les décompositions de bits en représentations binaires valides, c'est-à-dire avec des chiffres dans $\{0, 1\}$, il est utile donc de s'assurer que les coefficients de nos vecteurs/matrices sont petits. Rappelons que ceci est important pour garantir que les termes d'erreur restent limités tout au long de l'évaluation homomorphe.

4.3.4 Le schéma de chiffrement

Nous allons maintenant explorer les opérations de chiffrement, de déchiffrement et d'évaluation homomorphes du schéma de chiffrement totalement homomorphe

de [3].

Rappelons que l'espace du texte chiffré est constitué de matrices $\mathbf{N} \times \mathbf{N}$ sur \mathbf{Z}_q pour un paramètre de dimension \mathbf{N} et un module q . De manière plus détaillée, supposons que C_1 et C_2 sont des chiffrés de m_1 et m_2 , respectivement

$$C_1.v = m_1.v + e_1 \quad (4.4)$$

$$C_2.v = m_2.v + e_2 \quad (4.5)$$

avec e_i très petit. Pour vérifier si ce schéma est homomorphe, nous pouvons alors voir que la somme et les produits des chiffrés sont donnés par:

$$(C_1 + C_2).v = (m_1 + m_2).v + (e_1 + e_2) \quad (4.6)$$

$$\begin{aligned} (C_1 \circ C_2)v &= C_1(m_2v + e_2) \\ &= m_2(m_1v + e_1) + C_1e_2 \\ &= m_1m_2v + (m_2e_1 + C_1e_2) \end{aligned}$$

où les termes $(e_1 + e_2)$ et $(m_2e_1 + C_1e_2)$ demeurent petits. Formellement, le schéma est décrit ci-dessous. Suivant la présentation de [3], nous décomposons l'algorithme GEN en trois parties : Setup, SecretKeyGen, et PublicKeyGen, comme ci-dessous dans l'Algorithme 4 :

Algorithm 4 Algorithme de Setup, SecretKeyGen, and PublicKeyGen

procedure SETUP($(1^\lambda, 1^L)$)

Choisir un module de $\kappa = \kappa(\lambda, L)$

Choisir la dimension latice (trellis) $n = n(\lambda, L)$

Choisir la distribution d'erreur $\chi = \chi(\lambda, L)$ de manière appropriée pour LWE
qui atteint au moins 2^λ sécurité contre les attaques connues

Choisir le paramètre $m = m(\lambda, L) = \Theta(n \log(q))$

return (κ, n, χ, m)

end procedure

procedure SECRETKEYGEN((κ, n, χ, m))

Echantillon $t \leftarrow \mathbf{Z}_q^n$

Définir $s_k = (1, -t_1, \dots, -t_n) \in \mathbf{Z}_q^{n+1}$

return s_k

end procedure

procedure PUBLICKEYGEN($(\kappa, n, \chi, m, s_k)$)

Echantillonner une matrice $B \leftarrow \mathbf{Z}_q^{m \times n}$

Echantillonner un vecteur $e \leftarrow \chi^m$

Définir $b = B.t + e$ où t peut être récupéré à partir de s_k

Calculer $p_k = A$ qui est une $(n + 1)$ matrice de colonnes où la 1ère colonne est b et les n colonnes suivantes sont B

return p_k (notez que $p_k.s_k = e$ où p_k est une matrice et s_k et e sont des vecteurs).

end procedure

L'algorithme 5 ci-dessous présente les principales opérations du schéma de chiffrement. Notez qu'il existe deux algorithmes de déchiffrement : algorithme où le message provient d'un petit espace d'échantillonnage (par exemple, $\{0, 1\}$), et un qui fonctionne pour tout message dans \mathbf{Z}_q , bien que nous ne présentons que le cas où q est une puissance de 2. Ce dernier algorithme de déchiffrement MPDecryption provient de [11] et un traitement du cas général q peut être trouvé là aussi. Nous précisons également la manière dont les opérations homomorphes seront effectuées dans l'algorithme 6 ci-dessous :

Algorithm 5 Le schéma de chiffrement complet

procedure ENC($(\kappa, n, \chi, m, p_k, \mu \in \mathbb{Z}_q)$)

Echantillonnez une matrice uniforme $R \in \{0, 1\}^{N \times m}$

Calculer $C = Flatten(\mu.I_N + BitDecomp(R.A)) \in \mathbb{Z}^{N \times N}_q$

return C

end procedure

procedure DEC($(\kappa, n, \chi, m, s_k, C)$)

Calculer $v = Powersof2(s_k)$

Notez que les l premiers coefficients de v sont $1, 2, \dots, 2^{l-1}$

Choisir i tel que $v_i = 2^i \in (\frac{q}{4}, \frac{q}{2})$

Calculer $x_i = \langle C_i, \mathbf{v} \rangle$ où C_i est le i eme ligne de C

Calculer $\mu = \left\lfloor \frac{x_i}{v_i} \right\rfloor$

return μ

end procedure

procedure MPDECRYPTION $((\kappa, n, \chi, m, s_k, C))$

voir [11]

end procedure

Algorithm 6 Opérations homomorphes

```
procedure ADD( $(C_1, C_2)$ )  
    Calculer  $C_1 + C_2$   
    return  $Flatten(C_1 + C_2)$   
end procedure  
procedure MULTCONST( $(C, \alpha)$ )  
    Calculer  $M_\alpha = Flatten(\alpha \cdot I_N)$   
    return  $Flatten(M_\alpha \cdot C)$   
end procedure  
procedure MULT( $(C_1, C_2)$ )  
    return  $Flatten(C_1 \times C_2)$   
end procedure  
procedure NAND( $(C_1, C_2)$ )  
    return  $Flatten(I_N - C_1 \cdot C_2)$   
end procedure
```

Cette construction particulière utilise également les innovations de [22] pour rendre facultative l'étape de bootstrapping commune à de nombreux schémas basés sur l'apprentissage avec les erreurs, ce qui le rendant plus efficace.

4.4 Réduction des schémas basés sur le treillis pour l'apprentissage avec des erreurs

Outre l'apprentissage avec des erreurs (LWE), les problèmes basés sur des treillis constituent une autre primitive cryptographique importante. Ils sont importants car les hypothèses de dureté les plus défavorables pour ces problèmes basés sur des treillis résistent même contre les ordinateurs quantiques. En outre, plusieurs schémas de chiffrement entièrement homomorphes ont été construits à l'aide de ces problèmes de treillis [19]. Bien que nous n'explorions aucun schéma de chiffrement strictement basé sur des problèmes de treillis, nous introduirons brièvement la théorie de la cryptographie basée sur les treillis, puis nous présenterons une réduction du problème de treillis GapSVP souvent utilisé à l'apprentissage avec erreurs.

Définition 4

Un treillis (de rang complet) à n dimensions est un sous-groupe discret et additif de \mathbf{R}^n . De manière équivalente, si nous avons une base de n vecteurs linéairement indépendants $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\} \subset \mathbf{R}^n$ alors le treillis Λ généré par la base \mathbf{B} est l'ensemble

$$\Lambda = \mathcal{L}(\mathbf{B}) = \left\{ \sum_i^n c_i \mathbf{b}_i \mid c_i \in \mathbb{Z} \right\} \quad (4.7)$$

Définition 5

Le treillis dual d'un treillis Λ , noté Λ^* est $\Lambda^* = \left\{ \mathbf{x} \in \mathbf{R}^n \mid \forall \mathbf{v} \in \Lambda, \langle \mathbf{x}, \mathbf{v} \rangle \in \mathbb{Z} \right\}$.

Définition 6

La distance minimale $\lambda_l(\Lambda)$ d'un treillis Λ (dans la norme l_2 par défaut) est la distance minimale entre deux points distincts quelconques du treillis dans Λ . Cela équivaut à la longueur du vecteur non nul le plus court du treillis, Soit

$$\lambda_l(\Lambda) = \min_{x_1 \neq x_2, x_1, x_2 \in \Lambda} \|x_1 - x_2\| = \min_{c \neq 0; x \in \Lambda} \|x\| \quad (4.8)$$

Définition 7

Étant donné une fonction $\gamma(n) \geq 1$, le problème du vecteur le plus court $GapSVP_\gamma(\mathbf{B}, d)$, où \mathbf{B} est une base d'un treillis à n dimensions $\Lambda = \mathcal{L}(\mathbf{B})$ et $d \in \mathbb{R}^+$ est le problème de déterminer si $\lambda_l(\Lambda) \leq d$. En particulier, il requiert une sortie de 1 si $\lambda_l(\Lambda) \leq d$ et une sortie de 0 si $\lambda_l(\Lambda) > \gamma(n).d$

En 2009, Peikert a prouvé une réduction d'une variante particulière du problème du plus court vecteur $GapSVP_{\zeta, \gamma}$ défini ci-dessous à des versions correspondantes du problème de l'apprentissage avec erreurs [14]. Pour être complet, nous présentons l'énoncé et la preuve du théorème dans [14]. Tout d'abord, nous définissons la variante $GapSVP_{\zeta, \gamma}$ du problème du plus court vecteur

Définition 8

Étant donné les fonctions $\zeta(n) \geq \gamma(n) \geq 1$, le problème du plus court vecteur ζ -to- γ $GapSVP_{\zeta, \gamma}(\mathbf{B}, b)$, où

- 1.) \mathbf{B} est une base d'un treillis à n dimensions $\Lambda = \mathcal{L}(\mathbf{B})$ où $\lambda_l(\Lambda) \leq \zeta(n)$.
- 2.) $\min_i \|b_i\| \geq 1$

$$3.) 1 \leq d \leq \frac{\zeta(n)}{\gamma(n)}$$

est le problème consistant à déterminer si $\lambda_l(\wedge) \leq d$. En particulier, il requiert une sortie de 1 si $\lambda_l(\wedge) \leq d$ et une sortie de 0 si $\lambda_l(\wedge) > \gamma(n).d$.

Notons que pour tout $\zeta(n) \geq 2^{\frac{n}{2}}$, $\text{GapSVP}_{\zeta, \gamma}$ est équivalent au problème standard du plus court vecteur GapSVP_{γ} . Ceci est dû au fait que, par l'algorithme de Lenstra-Lenstra-Lovász [17], une base arbitraire B pour un treillis \wedge peut être réduite en temps polynomial à une autre base B' qui couvre le même treillis, et telle que

$$\lambda_l(\wedge) \leq \|\mathbf{b}'_1\| \leq 2^{\frac{n}{2}} \cdot \min_i \|b_i\| \quad (4.9)$$

Puisque nous pouvons toujours mettre à l'échelle la base du treillis d'origine B de telle sorte que $\min_i \|b_i\| = 1$, nous pouvons réduire toute instance du problème du plus court vecteur à une instance où la distance minimale est garantie comme étant inférieure à $2^{\frac{n}{2}}$. Cela rend la limite imposée par la condition (1) trivialement vraie. Nous n'avons pas à nous soucier des conditions (2) et (3) car nous pouvons toujours réduire les bases de sorte que (2) soit vraie, et la solution est triviale si (3) n'est pas vraie (et si (1) et (2) sont toutes deux vraies).

Maintenant nous allons passer à la réduction, nous énonçons quelques lemmes qui seront utilisés. Pour un traitement plus complet de l'objet de ces lemmes, voir [14]. Pour la notation, on désigne par $B_n \subset \mathbf{R}^n$ la boule unitaire ouverte (dans la norme l_2), c'est-à-dire l'ensemble $B_n = \{x \in \mathbf{R}^n \mid \|x\| \leq 1\}$.

lemme 5

Soit $\Delta(X, Y)$ la différence statistique entre les distributions X et Y , et $U(S)$ la distribution uniforme sur un ensemble S . Pour toute constante c , $d > 0$ et tout $z \in \mathbf{R}^n$ avec $\|z\| < d$, et $d' = d \cdot \sqrt{\frac{n}{c \log(n)}}$, on obtient que

$$\Delta(U(d' \cdot B_n), U(\mathbf{z} + d' \cdot B_n)) \leq 1 - \frac{1}{\text{poly}(n)} \quad (4.10)$$

Démonstration

Voir [7] pour la démonstration

Nous utilisons également le concept de paramètre de lissage de [1]. Intuitivement, le paramètre de lissage est le plus petit r qui lisse la structure discrète de \wedge , jusqu'à une certaine erreur statistique.

Nous allons énoncer le théorème de la réduction, il y'a d'autres lemmes intéressants qui pourront aider à la compréhension et à la démonstration du théorème, nous allons renvoyer la lecture à [14] pour un traitement complet

Theorème

Soit $\alpha = \alpha(n) \in (0, 1)$ et $\gamma = \gamma(n) \geq \frac{n}{\alpha \sqrt{\log(n)}}$. Soit $\zeta = \zeta(n) \geq \gamma(n)$ et $q = q(n) \geq \zeta(n) \cdot \sqrt{\frac{\log(n)}{n}}$. Alors, il existe une réduction probabiliste en temps polynomial de la résolution de $GapSVP_{\zeta, \gamma}$ dans le pire des cas (avec une probabilité écrasante) à la résolution de $LWE_{q, \alpha}$ en utilisant des échantillons $\text{poly}(n)$.

Démonstration

voir [14] pour la démonstration

Cependant, [14] ne fonctionne que pour les problèmes LWE avec de grands modules $q \geq q^{\frac{n}{2}}$. Plus tard, il est prouvé une réduction similaire qui fonctionne pour des modules polynomiaux. Ces résultats montrent que le GapSVP peut être réduit au LWE.

4.5 Encodage des nombres réels

Un détail technique important qui doit être discuté lors de l'examen des applications des schémas de chiffrement partiellement/totalement homomorphes aux algorithmes d'apprentissage automatique est le codage des nombres réels. La plupart des schémas de chiffrement homomorphes utilisent \mathbf{Z}_q pour un certain q comme espace de message. Par conséquent, ils sont naïvement incapables de chiffrer des valeurs réelles. Dans cette section, nous explorons brièvement une méthode de codage de nombres réels employée dans les applications de machine learning utilisées dans [20], [17].

Dans [20], les nombres réels sont comme des vecteurs de bits signés(+ ou -) de taille $L + l + 1$ bits, et un nombre réel $a \in \mathbf{R}$ est représenté par .

$$a = \sum_{-L}^l a_k 2^k \quad (4.11)$$

où $a_k \in \{0, 1\}$ si $a \geq 0$ et $a_k \in \{0, -1\}$ si $a < 0$

Pour les opérations sur les nombres réels avec ce codage, on a

$$a + b = \sum_{-L}^l (a_k + b_k) 2^k = \left\langle [2^{-L}, \dots, 2^0, \dots, 2^l], (\text{Bits}(a) + \text{Bits}(b)) \right\rangle$$

$$\begin{aligned}
ab &= \left\langle [2^{-L}, \dots, 2^0, \dots, 2^l], (Bits(a)) \right\rangle \cdot \left\langle Bits(b), [2^{-L}, \dots, 2^0, \dots, 2^l] \right\rangle \\
&= [2^{-L}, \dots, 2^0, \dots, 2^l] \cdot \left\langle Bits(a), Bits(b) \right\rangle \cdot [2^{-L}, \dots, 2^0, \dots, 2^l]
\end{aligned}$$

Notez que ce schéma de codage est flexible, car nous pouvons choisir L à l'avance pour obtenir la précision décimale souhaitée et l pour définir notre valeur entière maximale. On pourrait craindre que cette perte de précision n'ait peut avoir un impact négatif sur les performances prédictives de nos algorithmes d'apprentissage automatique. Cependant, des chercheurs ont étudié l'impact de la perte de précision sur des algorithmes tels que les classificateurs de réseau bayésien et n'ont constaté aucune perte de performance significative, à condition que les probabilités ne soient pas trop proches de 0 ou de 1 et que les probabilités postérieures sur les classes soient suffisamment différentes. sur les classes sont suffisamment différentes [22] . Cela suggère qu'il existe des algorithmes d'apprentissage algorithmes d'apprentissage automatique capables de tolérer un certain niveau de perte de précision.

PART III

APPLICATION À LA MACHINE LEARNING

CHAPTER 5

APPLICATION DU CHIFFREMENT HOMOMORPHE AUX ALGORITHMES DE MACHINE LEARNING

Dans ce chapitre, nous allons explorer les implémentations d'algorithmes de machine learning populaires à l'aide de schémas de chiffrement homomorphes. La possibilité d'implémenter des algorithmes d'apprentissage automatique sur des schémas de chiffrement homomorphes est utile car elle nous permet d'exécuter de puissantes techniques d'analyse sur des données chiffrées. Par exemple, dans notre exemple précédent avec la scolarité du département de LACGAA de l'UCAD, nous ne voulons peut-être pas seulement calculer la moyenne des étudiants, mais aussi trouver des prédictions de bonnes notes pour les étudiants, ou classer les étudiants dans diverses catégories, tout en maintenant la confidentialité. Ainsi, le but du chiffrement homomorphe est de permettre le calcul ou le traitement sur des données chiffrées sans révéler aucun détail sur les données.

Bien qu'il existe des schémas de chiffrement totalement homomorphes, les schémas de chiffrement partiellement homomorphes sont parfois suffisants pour mettre en œuvre toutes les opérations nécessaires à certains algorithmes de machine learning. Dans d'autres cas, nous pouvons utiliser des schémas de chiffrement totale-

ment homomorphes ou apporter de légères modifications à l'algorithme pour qu'il fonctionne avec des schémas partiellement homomorphes [17]. Pour certaines sections, nous présentons également les preuves de sécurité correspondantes pour les algorithmes de machine learning respectifs.

5.1 Classification binaire

Discriminante linéaire de Fisher

Rappelons que le but de la discriminante linéaire de Fisher est de trouver un vecteur \mathbf{w} sur lequel nous pouvons projeter les données et qui maximise le rapport entre les variances interclasses et les variances intraclasse. En suivant la présentation de [10], nous avons trouvé que le choix optimal de \mathbf{w} était :

$$\mathbf{w}^* \propto \mathbf{S}_W^{-1}(\mathbf{u}_2 - \mathbf{u}_1)$$

et en particulier, si nous voulons classifier, nous devons d'abord calculer $c = \mathbf{w}^{*\top}(\frac{\mathbf{u}_1 + \mathbf{u}_2}{2})$ où $\frac{\mathbf{u}_1 + \mathbf{u}_2}{2}$ est le point médian entre les deux moyennes dans la direction donnée par \mathbf{w} . Ensuite, nous pouvons classifier le point x_k en comparant $\mathbf{w}^{*\top}x_k$ à c .

Bien que [17] ne spécifie pas un schéma de chiffrement particulier, il mentionne qu'il a l'intention de faire fonctionner cet algorithme sur un schéma de chiffrement partiellement homomorphe qui supporte les additions et les multiplications limitées. Par conséquent, ils définissent la notion d'un algorithme D-polynomial, ce qui signifie que l'algorithme peut être exprimé comme un polynôme de degré borné.

Cependant, comme nous pouvons le voir, la détermination de ce \mathbf{w} optimal nécessite l'inversion de la matrice \mathbf{S}^{-1}_W , qui à son tour nécessite une division et n'est donc pas D-polynomiale [17]. Dans [17], ils présentent une approche par les moindres carrés pour calculer une approximation du vecteur optimal \mathbf{w}^* . L'idée est que nous pouvons définir une fonction d'erreur et la minimiser en utilisant des algorithmes comme la descente de gradient.

De manière plus détaillée, soit $\mathbf{d} = \mathbf{u}_1 - \mathbf{u}_2$ être la différence entre les moyennes conditionnelles de classe. Ensuite, supposons que nous essayons de minimiser une fonction de coût raisonnable (notez que le \mathbf{w} optimale est $\mathbf{w} = \mathbf{S}^{-1}_W \cdot \mathbf{d} = \mathbf{S}^{-1}_W(\mathbf{u}_2 - \mathbf{u}_1)$).

$$E(\mathbf{w}) = \frac{1}{2} \|\mathbf{S}_W \mathbf{w} - \mathbf{d}\| \quad (5.1)$$

à l'exception qu'au lieu de la norme euclidienne standard, comme dans [17], nous utilisons $\|\mathbf{v}\|^2 = \mathbf{v}^\top \mathbf{S}_W \mathbf{v}$ pour un meilleur conditionnement. Pour minimiser cette

fonction de coût, nous prenons d'abord le gradient, ce qui donne

$$\Delta_w(E(\mathbf{w})) = \mathbf{S}_W \mathbf{w} - \mathbf{d} \quad (5.2)$$

Ainsi, si on applique la descente de gradient avec un taux d'apprentissage η , on constate qu'à chaque étape, on fait

$$\begin{aligned} \mathbf{w}_{i+1} &= \mathbf{w}_i - \eta(\mathbf{S}_W \mathbf{w}_i + \mathbf{d}) \\ &= (\mathbf{I} - \eta \mathbf{S}_W) \mathbf{w}_i - \eta \mathbf{d} \end{aligned}$$

Ensuite, si on commence avec $\mathbf{w}_0 = 0$, on peut analyser la récurrence pour voir que la k ème approximation pour \mathbf{w} . est

$$\mathbf{w}_k = \eta \sum_{i=0}^{k-1} (\mathbf{I} - \eta \mathbf{S}_W)^i \mathbf{d}$$

Cet algorithme convergera si le rayon spectral (ou la norme de la valeur propre correspondant à son vecteur propre dominant) de $\mathbf{I} - \mathbf{S}_W$ est inférieur à 1. Cependant, nous pouvons toujours y parvenir en choisissant η suffisamment petit. Il est facile de voir que cet algorithme est D -polynomial avec $D = 2(k + 1) + 1$

5.2 Regression lineaire

Dans cette section, nous étudions comment mettre en œuvre la régression linéaire, un algorithme de machine learning très fondamental, en utilisant un schéma de chiffrement homomorphe. Notez que la construction précédente montre on appliquer un algorithme de descente de gradient pour le discriminant linéaire de Fisher et qui peut également être appliqué à la régression linéaire, mais nous n'irons pas plus loin dans ce domaine. La construction que nous explorons ici est due à [21] et nous permet d'effectuer la régression sur les données chiffrés. Ils mettent en œuvre un schéma de chiffrement entièrement homomorphe similaire à celui discuté dans [2]. Ainsi, leurs limitations ne sont pas tant sur les opérations qu'ils peuvent effectuer, mais plutôt sur la surcharge de performance liée à l'utilisation d'un schéma de entièrement homomorphe.

Rappelons brièvement qu'en régression linéaire, nous modélisons la variable de sortie comme une fonction linéaire des variables d'entrée plus un terme d'erreur normalement distribué, c'est à dire.

$$y = \langle \beta, \mathbf{x} \rangle + \epsilon \quad (5.3)$$

Pour résoudre le β , nous calculons le β qui minimise l'erreur des moindres carrés. Rappelons que si X désigne la matrice de conception $d \times n$ la solution en forme du problème d'optimisation ci-dessus est la suivante

$$\beta^* = (X^T X)^{-1} X^T y \quad (5.4)$$

Notez que les multiplications matricielles et les multiplications matrice-vecteur peuvent être facilement traitées par les sytemes partiellement homomorphe tels que le cryptosystème de Paillier, car ils supportent l'addition et la multiplication. Cependant, l'inversion de matrice est un peu plus problématique. Heureusement, [21] mentionne que nous pouvons utiliser la règle de Cramer pour calculer ces inverses. Tout d'abord, nous énonçons la règle de Cramer

Theorème 1(la règle de Cramer)

BIBLIOGRAPHY

- [1] Daniele Micciancio † Oded Regev ‡. “Worst-case to Average-case Reductions based on Gaussian Measures”. In: (). URL: <https://cims.nyu.edu/~regev/papers/average.pdf>.
- [2] Zvika Brakerski. “Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP”. In: (). URL: <https://eprint.iacr.org/2012/078.pdf>.
- [3] Amit Sahai Craig Gentry and Brent Waters. Homomorphic encryption from learning with errors. URL: <https://www.iacr.org/archive/crypto2013/80420297/80420297.pdf>.
- [4] RONALD CRAMER† and VICTOR SHOUP‡. “DESIGN AND ANALYSIS OF PRACTICAL PUBLIC-KEY ENCRYPTION SCHEMES SECURE AGAINST ADAPTIVE CHOSEN CIPHERTEXT ATTACK”. In: (). URL: <https://epubs.siam.org/doi/pdf/10.1137/S0097539702403773>.
- [5] Pedro Domingos and Michael Pazzani. “On the Optimality of the Simple Bayesian Classifier under Zero-One Loss”. In: (). URL: http://engr.case.edu/ray_soumya/mlrg/optimalilty_of_nb.pdf.
- [6] Craig Gentry. “Computing Arbitrary Functions of Encrypted Data”. In: (March 2010,). URL: <https://cacm.acm.org/magazines/2010/3/76272-computing-arbitrary-functions-of-encrypted-data/fulltext?mobile=false>.

- [7] Oded Goldreich and Shafi Goldwasser. “On the limits of non-approximability of lattice problems”. In: (). URL: <https://dl.acm.org/doi/10.1145/276698.276704>.
- [8] Yehuda Lindell Jonathan Katz. Introduction to Modern Cryptography Principles and Protocols. Chapman and Hall CRC, 2007.
- [9] Michael Naehrig Kristin Lauter and Vinod Vaikuntanathan. “Can homomorphic encryption be practical?” In: (). URL: <https://eprint.iacr.org/2011/405.pdf>.
- [10] CHRISTOPHER M.BISHOP. “PATTERN RECOGNITION AND MACHINE LEARNING”. In: (). URL: <https://epubs.siam.org/doi/pdf/10.1137/S0097539702403773>.
- [11] Daniele Micciancio¹ and Chris Peikert. “Trapdoors for Lattices:Simpler, Tighter, Faster, Smaller”. In: (). URL: <https://www.iacr.org/archive/eurocrypt2012/72370695/72370695.pdf>.
- [12] Kevin P. Murphy. Machine Learning: A Probabilistic Perspective. URL: http://noiselab.ucsd.edu/ECE228/Murphy_Machine_Learning.pdf.
- [13] Pascal Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. URL: <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.112.4035&rep=rep1&type=pdf>.
- [14] Chris Peikert. “Public-Key Cryptosystems from the Worst-Case Shortest Vector Problem”. In: (). URL: <https://web.eecs.umich.edu/~cpeikert/pubs/svpcrypto.pdf>.
- [15] Ron Rothblum. “Homomorphic Encryption:from Private-Key to Public-Key”. In: (). URL: <https://www.iacr.org/archive/tcc2011/65970216/65970216.pdf>.
- [16] Djiby SOW. “Introduction à la Sécurité Prouvée en Cryptographie à Clés publique. 2016. URL: https://nitaj.users.lmno.cnrs.fr/cimpamaure/cours_securite-prouvee-crypto_Djiby_SOW-CIMPA.pdf.
- [17] Kristin Lauter Thore Graepel and Michael Naehrig. “ML Confidential:Machine Learning on Encrypted Data”. In: (). URL: <https://eprint.iacr.org/2012/323.pdf>.
- [18] Kai-Min Chung -Yael Kalai - Salil Vadhan[‡]. “Improved Delegation of Computation using Fully Homomorphic Encryption”. In: (April 28, 2010). URL: <https://people.seas.harvard.edu/~salil/research/delegation-eprint-apr10.pdf>.
- [19] Zvika Brakerski Vinod Vaikuntanathan[†]. “Lattice-Based FHE as Secure as PKE”. In: (). URL: <https://eprint.iacr.org/2013/541.pdf>.

- [20] Yoshinori Aono Takuya Hayashi Le Trieu Phong Lihua Wang. “Fast and Secure Linear Regression and Biometric Authentication with Security Update”. In: (). URL: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.732.3597&rep=rep1&type=pdf>.
- [21] David J. Wu and Jacob Haven. “Using Homomorphic Encryption for Large Scale Statistical Analysis”. In: (). URL: https://www.cs.virginia.edu/dwu4/papers/FHE-SI_Report.pdf.
- [22] Craig Gentry Zvika Brakerski. “Fully Homomorphic Encryption without Bootstrapping”. In: (). URL: <https://eprint.iacr.org/2011/277.pdf>.