UQAC
Université du Québec
à Chicoutimi

SÉCURISATION DE DONNÉES SENSIBLES À L'AIDE D'AUTOENCODEUR CONVOLUTIONNEL PROFOND POUR IMAGES

PAR ABIB SY

MÉMOIRE PRÉSENTÉ À L'UNIVERSITÉ DU QUÉBEC À CHICOUTIMI COMME EXIGENCE PARTIELLE EN VUE DE L'OBTENTION DU GRADE DE MAÎTRE ÈS SCIENCES (M. SC.) EN INFORMATIQUE

QUÉBEC, CANADA

RÉSUMÉ

Plusieurs méthodes traditionnelles sont utilisées pour sécuriser les données sensibles, telles que les algorithmes de cryptographie comme AES-HMAC-SHA256, Twofish et Chacha20. Toutefois, des études récentes ont montré que ces algorithmes de cryptographie présentent des failles de sécurité. Dans ce mémoire, nous explorons l'utilisation d'un modèle de cryptographie basé sur un autoencodeur convolutionnel profond et nous comparons ses performances aux algorithmes traditionnels de cryptographie. Nous présentons les résultats d'une étude comparative basée sur plusieurs métriques. Ainsi, nous incorporons des métriques telles que la similarité en cosinus, l'entropie, le taux de Kendall et de Spearman, et l'erreur quadratique moyenne (MSE) pour une évaluation complète de la performance et de la sécurité de l'approche proposée, en plus des mesures de temps de cryptage et de décryptage.

Les résultats obtenus sont très prometteurs. Notre modèle est le plus performant pour deux métriques essentielles, l'entropie et l'erreur quadratique moyenne. Nous obtenons une entropie de fichier décrypté de 8,01, contre 7,99 pour les trois autres modèles standard, avec une MSE très faible de 0,003, contre 105,43 pour l'AES, qui reste l'algorithme traditionnel le plus efficace par rapport aux autres algorithmes.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ABRÉVIATIONS

AES	Advanced Encryption Standard
HMAC	Hash-based Message Authentication Code
SHA256	Secure Hash Algorithm 256-bit
MSE	Mean Squared Error
PBKDF2	Password-Based Key Derivation Function 2
RSA	Rivest-Shamir-Adleman (public-key cryptography algorithm)
CBC	Cipher Block Chaining
CAE	Convolutional Auto-Encoder
AE	Auto-Encoder
CNL	Convolutional Neural Layer
PNL	Perceptron Neural Layer
NIST	National Institute of Standards and Technology
EOS	End of Support (common in software lifecycle) or Equation of State (in physics)
TLS	Transport Layer Security
SSH	Secure Shell
IPSec	Internet Protocol Security
OpenSSL	Open Secure Sockets Layer
RAM	Random Access Memory
TPU	Tensor Processing Unit
JPEG	Joint Photographic Experts Group
JPEG-LS	JPEG Lossless Standard
PNG	Portable Network Graphics
L2	Layer 2 (in networking) or Euclidean norm
RGB	Red, Green, Blue
XOR	Exclusive OR
FPGA	Field-Programmable Gate Array
VHSIC	Very High-Speed Integrated Circuit
VHDL	VHSIC Hardware Description Language
PHT	Pseudo-Hadamard Transform
MDS	Maximum Distance Separable
QR	Quarter Round

DÉDICACE

À ma chère mère,

Ton amour inébranlable et ta foi en moi ont été mon refuge et ma force. Ta bravoure a pavé le chemin de mes rêves, même quand les autres doutaient. C'est avec un cœur rempli d'amour que je te dédie ce travail, car c'est pour toi que je m'efforce chaque jour d'être le meilleur de moi-même. Ton bonheur et ta fierté sont les étoiles qui guident mes pas. Longue et paisible vie à toi.

Je dédie également ce mémoire à la mémoire de mon père. Son absence laisse un vide incommensurable, mais je trouve du réconfort dans la prière qu'Allah, le Tout-Puissant, l'accueille en son vaste paradis. Amine.

Que cette réalisation soit le reflet de votre lumière dans ma vie.

REMERCIEMENTS

Je tiens à exprimer ma plus profonde gratitude envers mon directeur de recherche, Monsieur Fehmi Jaafar, pour son soutien indéfectible tout au long de la réalisation de ce mémoire de maîtrise. Ses encouragements incessants, son expertise et sa disponibilité ont été des piliers solides sur lesquels je suis appuyé. Je suis particulièrement reconnaissant pour les bourses d'études qu'il m'a accordées, qui ont été essentielles pour poursuivre mes recherches avec sérénité et détermination. Sa capacité à me motiver et à me stimuler intellectuellement m'a poussé à dépasser mes limites et je suis fier de l'avoir comme mentor.

Je tiens également à remercier chaleureusement mon codirecteur de recherche, Monsieur Kevin Bouchard, dont les conseils avisés et le soutien ont grandement contribué à enrichir ce travail.

Mes remerciements vont également à l'ensemble du personnel du Département de Mathématiques et Informatique (DIM) de l'UQAC pour leur environnement stimulant et accueillant. Une mention spéciale à Madame Andréanne Riverin, dont le professionnalisme exemplaire et l'empathie remarquable à gérer mon dossier ont grandement facilité mon parcours académique.

Je ne saurais oublier de remercier tous les professeurs que j'ai eu le privilège de rencontrer pendant mes études. Leurs enseignements précieux m'ont permis de construire une base solide de connaissances et de compétences.

Enfin, je tiens à exprimer toute ma reconnaissance à ma famille. Leur soutien inconditionnel, leur amour et leur foi en mes capacités ont été les fondements de ma persévérance et de ma réussite. C'est à eux que je dédie ce succès.

AVANT-PROPOS

C'est avec une profonde humilité et une grande reconnaissance que je me penche sur ces pages, fruit d'un parcours aussi exigeant qu'enrichissant. Ce mémoire de maîtrise, qui s'aventure au cœur de la cryptographie et de la sécurité des données, est bien plus qu'un simple document académique : c'est le récit d'un voyage intellectuel, le témoignage d'une passion pour la protection de notre intimité numérique à l'ère de l'information.

Lorsque j'ai rencontré mon directeur de recherche, notre échange sur les défis contemporains de la cryptographie a été l'étincelle initiale qui a allumé la flamme de ce projet. Nous nous sommes rapidement accordés sur un sujet qui, nous en étions convaincus, ne se contentait pas d'effleurer la surface d'un domaine en constante évolution, mais plongeait plutôt dans les profondeurs de ses enjeux les plus complexes.

Le chemin parcouru depuis ce jour a été pavé de défis, d'obstacles, mais aussi de découvertes et de satisfactions inestimables. Chaque page de cette mémoire porte l'empreinte des heures de recherche, de discussion et d'analyse, toutes animées par une quête incessante de compréhension et de maîtrise.

Que ce mémoire sert non seulement à valider mon parcours académique, mais aussi à ouvrir de nouvelles voies de réflexion pour sécuriser encore davantage notre avenir numérique. C'est avec un sentiment de contribution, aussi modeste soit-elle, à ce domaine vital, que je vous invite à explorer ce travail.

CHAPITRE I

INTRODUCTION

1.1 CONTEXTE DE LA RECHERCHE

1.1.1 ÉVOLUTION DE LA SÉCURITÉ DES DONNÉES

L'avènement de l'intelligence artificielle et des approches d'apprentissage profond a révolutionné la sécurité des données, offrant des perspectives inédites pour la protection des informations numériques LeCun *et al.* (2016). Ainsi, l'ère numérique a transformé la gestion de l'information, accentuant la nécessité de méthodes de sécurité robustes pour préserver l'intégrité et la confidentialité des données. Dans ce contexte en constante évolution, les autoencodeurs émergent comme une solution prometteuse, exploitant leur capacité à apprendre des représentations codées pour renforcer la sécurité.

Historiquement, la protection des informations a évolué des techniques simples telles que le chiffrement de César aux systèmes complexes de cryptographie numérique comme le RSA et l'AES. L'essor du cloud computing et de l'informatique quantique a introduit de nouveaux défis, nécessitant des approches innovantes pour sécuriser les données contre des menaces de plus en plus sophistiquées. C'est dans ce contexte que les autoencodeurs représentent une avancée dans ce domaine, offrant des encodages uniques et complexes pour une sécurité accrue.

1.1.2 LIMITATIONS DES MÉTHODES TRADITIONNELLES

Les méthodes traditionnelles de cryptographie, bien que largement éprouvées, révèlent leurs limites face à l'évolution des attaques et à la croissance exponentielle des données

Example & Case (2019). Les algorithmes standards peinent à s'adapter aux formats de données complexes, comme les images, nécessitant des approches plus flexibles et adaptées.

La rigidité des systèmes classiques de chiffrement symétrique et asymétrique les rend vulnérables face à des attaques sophistiquées et limite leur efficacité dans la gestion des volumes croissants de données numériques. Ces limitations soulignent la nécessité d'innover en matière de sécurité pour répondre aux enjeux actuels de protection des informations.

1.1.3 POTENTIEL DES AUTOENCODEURS DANS LA CRYPTOGRAPHIE

L'exploration des autoencodeurs dans le domaine de la cryptographie ouvre des horizons prometteurs, particulièrement pour la sécurisation des images numériques Doe & Smith (2020a). Ces modèles d'apprentissage profond, capables de générer des représentations codées difficiles à inverser, offrent une nouvelle approche pour renforcer la confidentialité des données.

En outre, cette capacité des autoencodeurs à produire des encodages compacts et résistants au déchiffrement sans clé appropriée représente une avancée majeure pour la sécurité des données en transit et au repos. Toutefois, leur mise en œuvre soulève des défis, notamment en termes de ressources informatiques et de volume de données nécessaires à l'entraînement, qui doivent être prises en compte pour assurer leur efficacité dans des contextes réels.

1.1.4 IMPORTANCE DE LA SÉCURITÉ DES IMAGES NUMÉRIQUES

Dans de nombreux secteurs, les images numériques véhiculent des informations critiques, nécessitant une protection rigoureuse pour prévenir les fuites et les abus Tech (2021). Que ce soit dans le domaine médical, financier ou de la sécurité nationale, la compromission des données visuelles peut avoir des conséquences désastreuses.

La cryptographie des images doit relever des défis spécifiques, tels que la taille des fichiers, la préservation des métadonnées et la qualité visuelle. Face à ces enjeux, les autoencodeurs représentent une solution prometteuse, offrant une sécurité renforcée tout en maintenant la fidélité des images Deng *et al.* (2009).

Dans le contexte des habitats intelligents, la sécurité des images numériques est cruciale. Ces environnements, dotés de capteurs et de caméras, recueillent des données visuelles pour diverses applications, allant de la gestion énergétique à la sécurité domestique. Par exemple, au Laboratoire d'Intelligence Ambiante pour la Reconnaissance d'Activités (LIARA) de l'Université du Québec à Chicoutimi, les recherches se concentrent sur le développement d'habitats intelligents équipés de capteurs ambiants. Ces technologies permettent au personnel médical ou aux proches de reconnaître, suivre et prédire les activités des personnes âgées, en perte d'autonomie ou ayant un handicap. Bien que ces capteurs fournissent des informations précieuses pour le bien-être des résidents, ils soulèvent également des préoccupations en matière de sécurité. En effet, la collecte de données augmente les risques pour la vie privée, rendant indispensable la mise en œuvre de méthodes de cryptographie robustes pour protéger les informations sensibles tout en préservant la fonctionnalité des systèmes d'habitat intelligent.

1.1.5 DÉFIS ET OPPORTUNITÉS

L'intégration des autoencodeurs dans les stratégies de cryptographie présente des défis, mais également des opportunités pour améliorer la sécurité des données sensibles Innovate & Solutions (2022). La complexité informatique et les exigences en matière de données pour l'entraînement sont des obstacles à surmonter, mais les bénéfices potentiels en termes de protection des informations justifient les efforts de recherche et de développement dans ce domaine.

Par ailleurs, les autoencodeurs offrent une nouvelle dimension à la sécurité des données, en proposant des systèmes dynamiques et résistants face aux menaces émergentes. Leur intégration dans les infrastructures existantes pourrait transformer la manière dont nous concevons et appliquons la cryptographie.

1.2 PROBLÉMATIQUE DE LA RECHERCHE

Bien que les algorithmes standards tels que AES, HMAC-SHA256, Twofish, et Chacha20 soient largement utilisés, leur efficacité face aux techniques modernes d'attaque reste une question ouverte Innovate & Solutions (2022). De plus, la gestion efficace de la sécurité des données dans des formats complexes comme les images nécessite des solutions innovantes. La question centrale de cette recherche est donc : comment les autoencodeurs convolutionnels profonds peuvent-ils renforcer la sécurité des données sensibles, notamment des images, en se comparant avec les méthodes traditionnelles de cryptographie ?

1.3 CONTRIBUTION DE LA RECHERCHE

Cette recherche présente un modèle cryptographique basé sur un autoencodeur convolutionnel profond pour les images, offrant une nouvelle approche dans le domaine de la cryptographie. En comparant avec les méthodes de cryptographie traditionnelles, ce travail met en évidence l'efficacité supérieure des autoencodeurs en termes d'entropie et d'erreur quadratique moyenne (MSE), indiquant une sécurité et une fidélité de reconstruction améliorées.

Notre approche diffère des autres méthodes qui utilisent des autoencodeurs, notamment en ce qui concerne les paramètres choisis pour la comparaison. Nous incorporons des métriques telles que la similarité en cosinus, l'entropie, le taux de Kendall et de Spearman et l'erreur

quadratique moyenne (MSE) pour une évaluation complète de la performance et de la sécurité du modèle, en plus des mesures de temps de cryptage et de décryptage.

1.4 MÉTHODOLOGIE DE LA RECHERCHE

Cette étude s'appuie sur une méthodologie rigoureuse pour évaluer le potentiel de l'autoencodeur convolutionnel en profondeur dans le domaine de la cryptographie des images. La démarche comprend plusieurs étapes clés décrites par la figure 1.1, visant à concevoir, entraîner et évaluer un modèle d'autoencodeur adapté à la sécurisation des données visuelles.

1.4.1 CONCEPTION DU MODÈLE

Le cœur de notre recherche repose sur la conception d'un autoencodeur convolutionnel profond, spécifiquement adapté pour traiter des images complexes. Ce modèle est structuré pour apprendre efficacement des représentations codées des données d'entrée, offrant ainsi une base solide pour le chiffrement des images.

1.4.2 ENTRAÎNEMENT DU MODÈLE

Pour l'entraînement, nous avons sélectionné un ensemble de données d'images variées, couvrant différents types de contenu visuel. Ensuite, le modèle a été entraîné en utilisant la bibliothèque Keras de TensorFlow, une plateforme reconnue pour sa flexibilité et son efficacité dans le domaine de l'apprentissage automatique. Les paramètres d'entraînement, tels que le nombre d'époques et la taille du lot, ont été soigneusement ajustés pour optimiser les performances du modèle. De même, la progression de l'entraînement a été suivie à travers la courbe de perte, permettant d'apprécier l'évolution de la capacité du modèle à reconstruire les images.

1.4.3 ÉVALUATION DES PERFORMANCES

Une fois le modèle appliqué, nous avons évalué ses performances à l'aide de métriques pertinentes pour la cryptographie des images. La similarité cosinus a été utilisée pour mesurer la fidélité de la reconstruction des images. L'entropie a servi à évaluer le degré de désordre et donc la sécurité potentielle du chiffrement. Les taux de corrélation de Kendall et Spearman ont été calculés pour examiner la préservation des relations entre les pixels. Enfin, l'erreur quadratique moyenne (EQM) a été enregistrée pour quantifier les différences entre les images originales et reconstruites.

1.4.4 COMPARAISON AVEC LES STANDARDS DE CRYPTOGRAPHIE

Pour contextualiser les résultats obtenus, nous avons effectué une comparaison quantitative avec trois algorithmes de cryptographie reconnus pour leur fiabilité : AES, Twofish et Chacha20. Cette analyse comparative nous a permis d'évaluer l'efficacité de notre modèle d'autoencodeur au regard des normes actuelles de la cryptographie.

En somme, cette méthodologie offre un cadre rigoureux pour explorer le potentiel des autoencodeurs convolutionnels profonds dans le renforcement de la sécurité des images numériques. Elle combine une approche systématique de l'entraînement et de l'évaluation du modèle avec une analyse comparative approfondie pour fournir des informations précieuses sur les avantages et les limites de cette technologie émergente en matière de cryptographie.

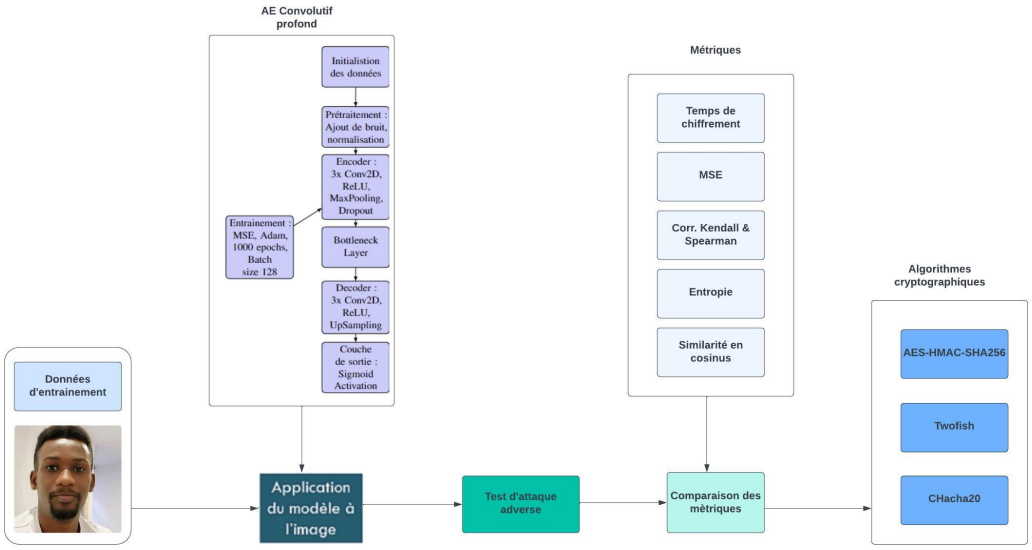

FIGURE 1.1 : schéma d'approche

©Abib Sy

1.5 ORGANISATION DU DOCUMENT

Ce mémoire est structuré en cinq chapitres principaux pour une exploration approfondie de la sécurisation des données sensibles à l'aide d'autoencodeurs convolutifs profonds pour les images :

1. **Chapitre II : Revue de la littérature** - Ce chapitre fournit les connaissances de base sur les algorithmes de cryptographie traditionnels tels que AES-HMAC-SHA256, Twofish et Chacha20, de même que sur les réseaux de neurones et les autoencodeurs, établissant le cadre théorique de cette recherche.
2. **Chapitre III : Travaux connexes** - Dans ce chapitre, une discussion approfondie sur les travaux connexes dans le domaine est présentée, mettant en évidence les avancées récentes et les lacunes existantes.

3. **Chapitre IV : Implémentations et tests** - Ce chapitre détaille les implémentations techniques et les tests effectués au cours de cette étude, démontrant la faisabilité et l'efficacité des approches proposées.
4. **Chapitre V : Évaluation des résultats** - Les résultats obtenus sont évalués dans ce chapitre, fournissant une analyse critique de la performance et de la sécurité du modèle proposé.
5. **Conclusion et perspectives futures** - Le document se conclut par une synthèse des découvertes majeures de la recherche et une discussion sur les directions futures potentielles dans ce domaine.

CHAPITRE II
REVUE DE LITTÉRATURE

Ce chapitre fournit un aperçu des algorithmes de cryptographie traditionnels comme AES, Twofish et ChaCha20, ainsi que des principes fondamentaux des réseaux de neurones artificiels, en particulier les réseaux de neurones convolutifs (CNN). Il explore également le concept des autoencodeurs et leur potentiel dans la cryptographie, notamment pour la sécurisation des images numériques. En outre, il aborde les vulnérabilités des autoencodeurs et les défis associés à leur utilisation dans la cryptographie.

2.1 LES ALGORITHMES DE CRYPTOGRAPHIE

Les algorithmes de cryptographie fournissent des services de sécurité de l'information tels que la confidentialité, l'intégrité des données, l'authentification et la non-répudiation Dunn Cavelty (2010).

2.1.1 ADVANCED ENCRYPTION STANDARD (AES)

L'Advanced Encryption Standard (AES) est une spécification de chiffrement établie par le National Institute of Standards and Technology (NIST) des États-Unis en 2001. AES est également connu sous le nom de Rijndael, d'après ses créateurs belges, Joan Daemen et Vincent Rijmen Joan & Vincent (2002). Ce standard est devenu une des méthodes de chiffrement les plus utilisées à travers le monde, reconnue pour sa solidité et sa fiabilité.

Le fonctionnement d'AES est un modèle de cryptographie symétrique, ce qui signifie qu'il utilise la même clé pour le chiffrement et le déchiffrement des données. La clé peut être de longueur 128, 192 ou 256 bits, offrant ainsi différentes intensités de sécurité. Plus la clé est

longue, plus le nombre de cycles de chiffrement est élevé, ainsi la sécurité de l'encodage. Voir tableau 2.1.

Le processus de chiffrement AES opère sur un bloc de données de 128 bits, appelé un état, qui est ensuite traité dans un nombre défini de cycles ou de tours, en fonction de la longueur de la clé : 10 tours pour 128 bits, 12 tours pour 192 bits et 14 tours pour 256 bits (Voir la représentation de l'entrée des clés AES256 figure 2.1). À chaque tour, plusieurs transformations sont appliquées, à savoir SubBytes, ShiftRows, MixColumns et AddRoundKey.Dworkin *et al.* (2001)

La première étape, SubBytes, est un processus de substitution non linéaire où chaque octet de l'état est remplacé par un autre selon une table de substitution prédéfinie (S-box). Cette étape introduit la confusion dans le texte chiffré, contribuant à sa sécurité.

La deuxième étape, ShiftRows, consiste à décaler les rangées de l'état de manière cyclique, ce qui disperse horizontalement les octets dans l'état. Cette transformation vise à éliminer les motifs et les structures dans les données claires, rendant ainsi le texte chiffré plus difficile à attaquer.

MixColumns est la troisième transformation, où les colonnes de l'état sont mélangées en utilisant une opération de mélange algébrique. Cette étape combine les octets entre eux, ajoutant de la complexité au processus de chiffrement et renforçant la diffusion des données.

Enfin, AddRoundKey est l'opération où une clé de tour est ajoutée à l'état en utilisant une opération XOR. La clé de tour est dérivée de la clé principale dans un processus appelé l'extension de clé. Cette étape assure que chaque bit du texte chiffré dépend de la clé de chiffrement.

Le déchiffrement avec AES utilise les mêmes étapes mais dans l'ordre inverse et avec des transformations inversées pour restaurer les données originales à partir du texte chiffré.

La sécurité de l'AES a été largement testée et approuvée par la communauté cryptographique mondiale. Son adoption par le gouvernement américain et son utilisation comme élément de chiffrement dans d'autres normes et protocoles de sécurité, comme le protocole de sécurisation des échanges sur Internet SSL/TLS, témoignent de sa robustesse et de sa fiabilité en tant que standard de chiffrement. of Standards & Technology (2001).

En résumé, AES est une méthode de chiffrement robuste qui repose sur une série de transformations cryptographiques pour sécuriser les données. Sa mise en œuvre dans divers logiciels et matériels est soutenue par son efficacité et sa capacité à résister aux attaques, ce qui en fait un choix prédominant dans la sécurisation des communications et des données sensibles à travers le monde.

Exemples d'utilisation de l'AES dans le chiffrement d'images : Pour le transfert sécurisé de données d'images, une méthode de cryptage d'images AES et basée sur le chaos est proposée par Lata & Saini (2020) et des images rapides et hautement sécurisées peuvent être transférées pour les satellites d'observation de la Terre (EOS) en utilisant cette approche Bentoutou *et al.* (2020) . Cette méthode de cryptage AES améliorée peut également être utilisée pour le transfert d'images sécurisé normal. Le cryptage des images utilisant AES et RSA a été réalisé par Alsaffar *et al.* (2020) et il s'avère que les performances AES sont meilleures pour RSA pour sécuriser les images.

Dans le domaine de la cryptographie, l'algorithme AES (Advanced Encryption Standard) joue un rôle crucial dans la sécurisation des communications numériques. Kumar *et al.* (2016) ont exploré l'implémentation de l'algorithme AES 128 bits dans MATLAB, démontrant son applicabilité pratique dans divers contextes de sécurité des données. Par ailleurs, Daemen &

TABLEAU 2.1 : - Tableau rectangulaire d'octets : - 4 lignes - 4, 6 ou 8 colonnes (clé de 128, 192 ou 256 bits)

k0,0	k0,1	k0,2	k0,3	k0,4	k0,5	k0,6	k0,7
k1,0	k1,1	k1,2	k1,3	k1,4	k1,5	k1,6	k1,7
k2,0	k2,1	k2,2	k2,3	k2,4	k2,5	k2,6	k2,7
k3,0	k3,1	k3,2	k3,3	k3,4	k3,5	k3,6	k3,7

Rijmen (2013) ont fourni une analyse approfondie de la conception de Rijndael, l'algorithme à la base d'AES, mettant en lumière ses caractéristiques de sécurité robustes. Enfin, Rijmen & Daemen (2001) ont discuté de l'adoption d'AES comme standard de cryptographie avancé, soulignant son importance dans la protection des informations sensibles.

2.1.2 TWOFISH

La sécurité des données est une préoccupation majeure dans le monde numérique d'aujourd'hui. Pour relever ce défi, l'algorithme Twofish s'est imposé comme une solution puissante. Twofish est un algorithme de cryptage symétrique qui offre une combinaison équilibrée de sécurité, d'efficacité et de flexibilité. Il s'agit donc d'un choix idéal pour protéger les informations sensibles.

L'une des principales caractéristiques de Twofish est l'utilisation de boîtes de substitution non linéaires (S box) et de permutations de clés dérivées. L'algorithme Twofish est un chiffrement par bloc de 128 bits qui accepte une clé de longueur variable jusqu'à 256 bits. L'algorithme est composé de 16 tours formés de la même manière que la structure du réseau Feistel. Ceci est avantageux puisque la structure du cryptage et du déchiffrement est très similaire ; les seules différences significatives résident dans les clés utilisées dans les deux procédures. De plus, les entrées et sorties sont XORées avec les 8 touches K0. . .. K7. Ces procédures sont connues sous le nom de blanchiment d'entrée et de blanchiment de sortie. La

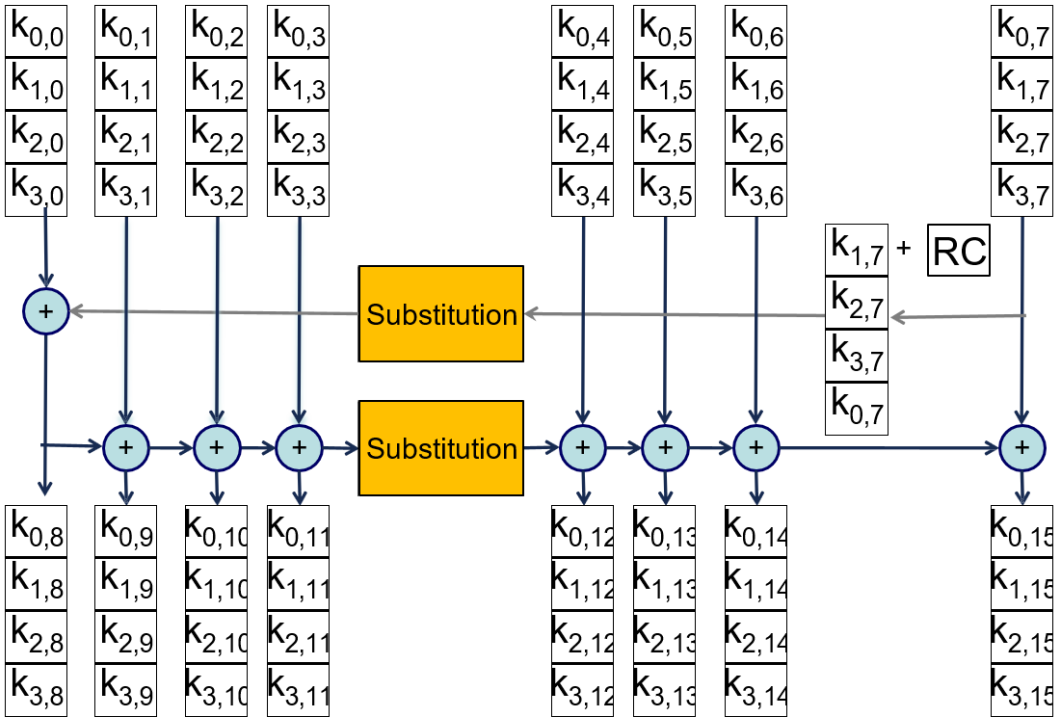

FIGURE 2.1 : Représentation de l'entrée des clés AES256

Daemen & Rijmen (2010)

plupart des chiffrements utilisent des opérations XOR car elles sont réversibles, permettant ainsi d'effectuer le déchiffrement. La figure 2.2 présente le schéma fonctionnel de l'algorithme Twofish Schneier *et al.* (1998) . Ces mécanismes introduisent une confusion et une diffusion efficaces dans le processus de cryptage, ce qui le rend résistant aux attaques cryptographiques.

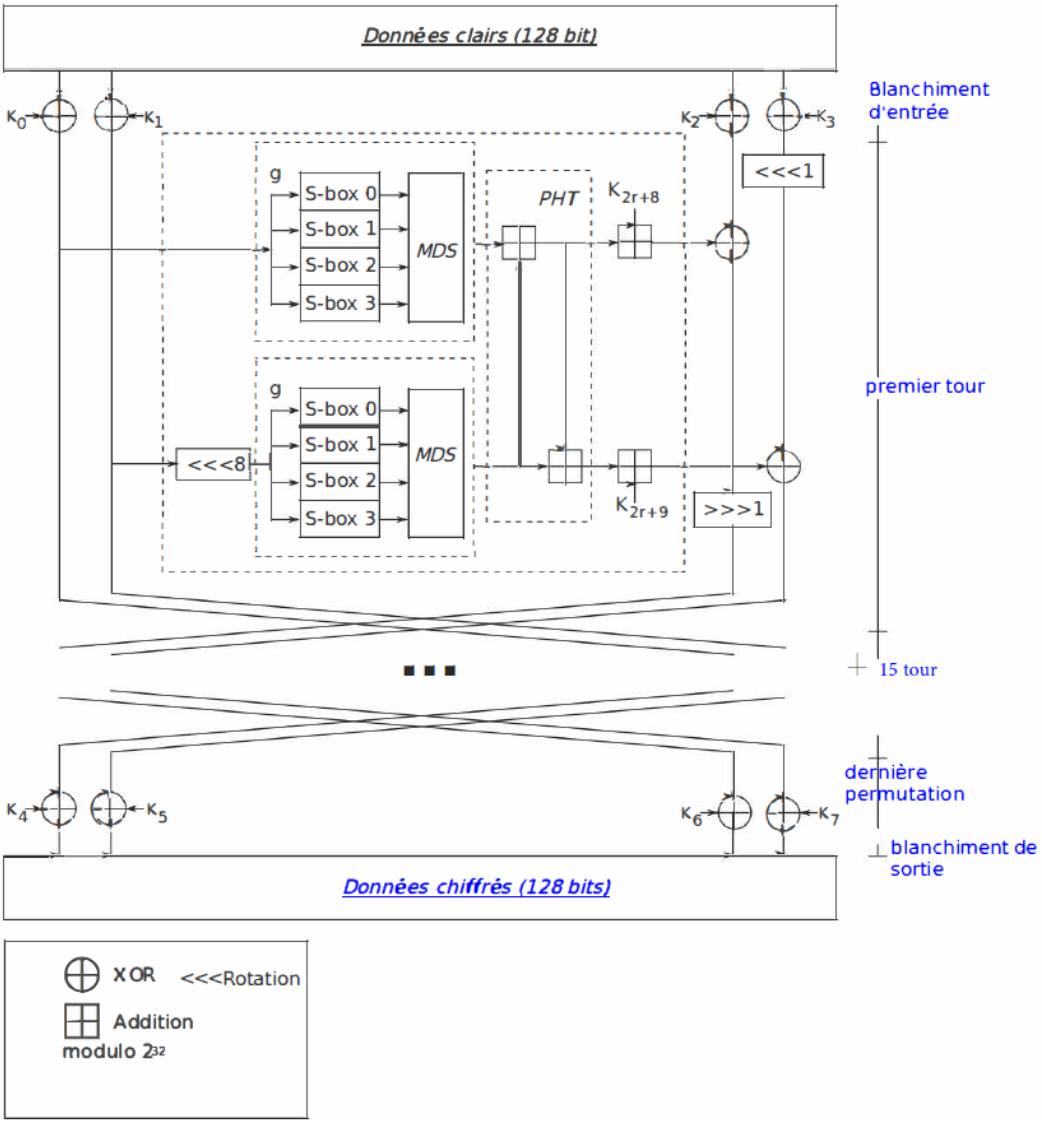

FIGURE 2.2 : Schéma fonctionnel de l'algorithme Twofish

Haq *et al.* (2021)

En outre, Twofish utilise une gestion dynamique des clés. Cela implique des opérations de matrice de diffusion maximale (MDS), qui renforcent la sécurité en permettant la création de sous-clés basées sur la clé principale. Cette approche rend l'algorithme résistant aux attaques par force brute.

L'algorithme se compose d'un certain nombre d'éléments de base, décrits comme suit : Des matrices séparables à distance maximale (MDS), une transformation pseudo-Hadamard, des rotations au niveau des bits (PHT) et des boîtes S 8 x 8 bits dépendant de la clé sont également incluses dans l'approche.

MDS : Les matrices séparables à distance maximale multiplient une valeur d'entrée de 32 bits par des constantes de 8 bits avec toutes les multiplications effectuées dans le champ de Galois GF (octet par octet) (256) Vaudenay (1994) .

PHT : Le PHT Gehlot *et al.* (tion) est une fonction d'addition de base dont les équations sont comme en suit(1) :

$$\begin{aligned} c' &= c + d \bmod 32 \\ d' &= c + 2^{*}d \bmod 32 \end{aligned} \tag{1}$$

— **Exemple d'utilisation de l'algorithme Twofish dans la securisation de données sensible notamment d'images**

L'algorithme proposé par Amdouni *et al.* (2022) assure le cryptage et le décryptage des images médicales selon les propriétés de diffusion et de confusion de Shannon. Pour améliorer le schéma de chiffrement par blocs, un générateur de nombres pseudo-aléatoires basé sur le chaos a été implémenté pour générer des clés de chiffrement. L'algorithme a été implémenté et évalué sur le kit de développement FPGA-Zynq

Zedboard en utilisant le langage VHDL. Les algorithmes conçus satisfont une fréquence d'horloge importante avec un débit élevé et une consommation minimale de ressources matérielles, selon l'évaluation précise de l'implémentation cible du FPGA. La force du système de chiffrement par blocs suggéré par rapport à l'éventail de menaces connues a été minutieusement testée à l'aide de diverses images médicales. Les résultats montrent que l'architecture offre une bonne sécurité et de bonnes performances.

Dans l'ensemble, l'algorithme de cryptage Twofish constitue une contribution importante à la cryptographie symétrique. Sa conception robuste, ses mécanismes sophistiqués de substitution et de permutation et sa gestion dynamique des clés en font un choix intéressant pour les applications nécessitant un niveau de sécurité élevé. Son adoption dans des normes de sécurité telles que le protocole Secure Shell (SSH) démontre son efficacité dans des environnements où la confidentialité des données est cruciale.

2.1.3 CHACHA20

ChaCha20 est un type de chiffrement par flux qui génère une séquence de bits qui semblent aléatoires à l'aide d'une clé secrète. Ce type de chiffrement est bien adapté aux communications en continu et aux protocoles à haut débit.

La structure algorithmique de ChaCha20 est basée sur des opérations de substitution, de permutation et d'addition modulo (2^{32}), ce qui le rend simple mais efficace et contribue à sa vitesse et à sa sécurité.

L'algorithme ChaCha20 commence avec un état initial de 16 mots de 32 bits chacun. Il comprend une constante de 128 bits, une clé privée de 256 bits, un compteur de blocs de 32 bits incrémenté à partir de zéro et un nom occasionnel de 96 bits unique pour un flux de clés. L'état initial est organisé sous la forme d'un4×4 matrice comme le montre le tableau 2.2 .

TABLEAU 2.2 : Matrice des états

constante [1]	constante [2]	constante [3]	constante [4]
clé [4]	clé [5]	clé [6]	clé [7]
clé [8]	clé [9]	clé [10]	clé [11]
compteur de bloc [13]	nonce [14]	nonce [15]	nonce [16]

TABLEAU 2.3 : Tours

Aamir *et al.* (2021)

Tours pairs					Tours impairs				
Quart de tour(QR)				Col	Quart de tour(QR)				Diag
0	5	10	15	1	0	4	8	12	1
1	6	11	12	2	1	5	9	13	2
2	7	8	13	3	2	6	10	14	3
3	4	9	14	4	3	7	11	15	4

Une série de 20 tours, alternant entre tours pairs et impairs, est appliquée à la matrice d'état initiale, générant un flux de clés de 512 bits. Le flux de l'algorithme ChaCha20 est illustré à la Figure 2.3 . Chaque tour contient quatre fonctions quart de tour (QR). Les tours pairs et impairs diffèrent par leurs entrées, comme le montre le tableau 2.3 . Chaque QR prend 4 mots de 32 bits chacun en entrée. Il utilise des ajouts, des XOR et des rotations pour mettre à jour les mots, comme le montre l'organigramme de la figure 2.3 .

ChaCha20 est une avancée dans le domaine de la cryptographie symétrique, connue pour sa conception élégante, ses performances élevées et sa sécurité robuste. Il est couramment utilisé dans les protocoles de sécurité tels que TLS (Transport Layer Security) et IPSec (Internet Protocol Security) pour améliorer la confidentialité des communications sur l'internet. Les développeurs sont également attirés par cette technologie en raison de son adoption dans des bibliothèques de chiffrement comme OpenSSL.Serrano *et al.* (2021)

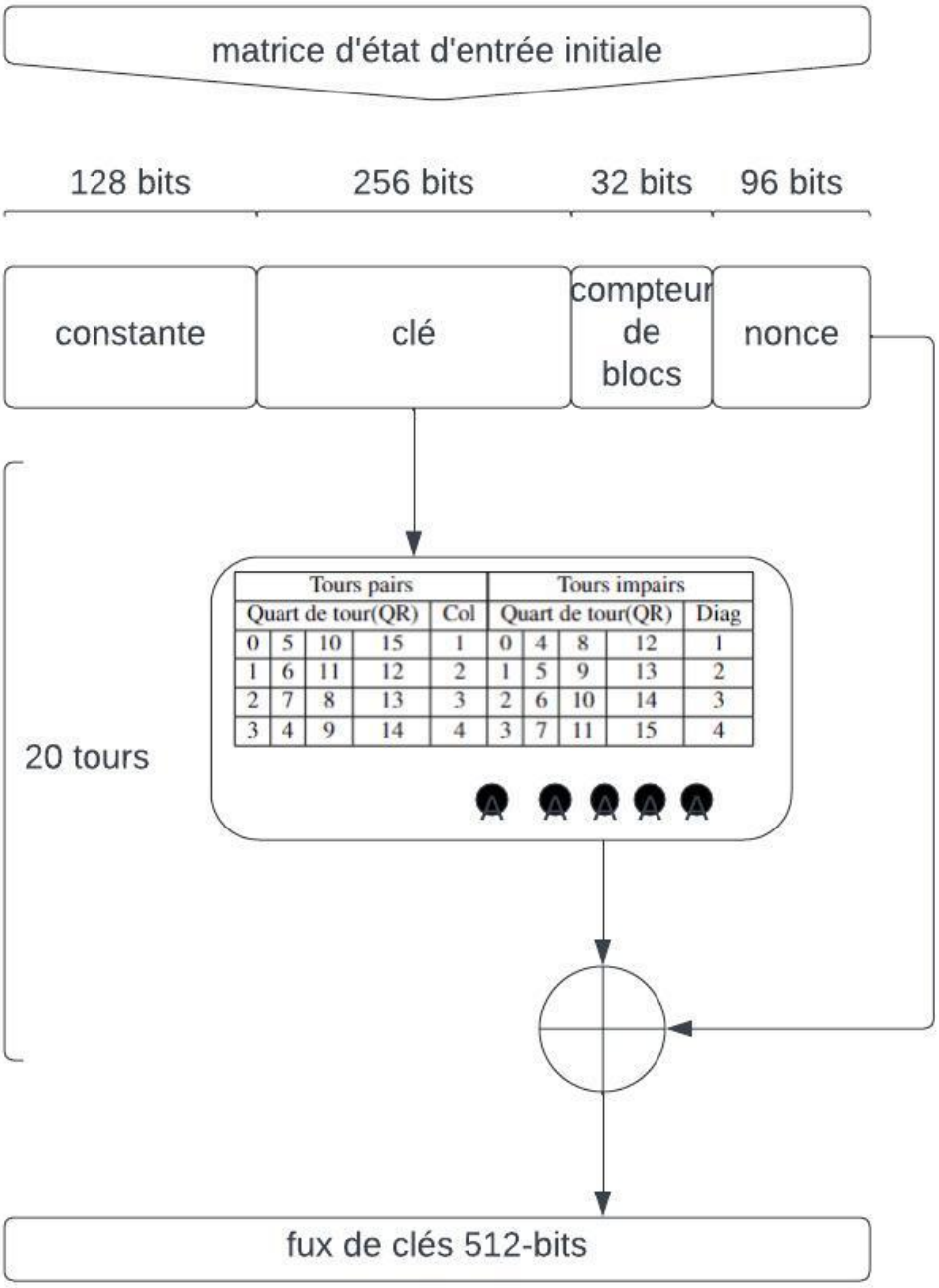

FIGURE 2.3 : Schéma fonctionnel de l'algorithme ChaCha20

Aamir *et al.* (2021)

2.2 RÉSEAUX DE NEURONES ARTIFICIELS

Les réseaux de neurones artificiels (RNA) sont des modèles informatiques inspirés du fonctionnement du cerveau humain Goodfellow *et al.* (2016a). Ils sont composés d'un ensemble de neurones artificiels, également appelés nœuds ou unités, organisés en couches. Chaque neurone reçoit des entrées, effectue des calculs et transmet des sorties à d'autres neurones. Les RNA sont largement utilisés dans divers domaines tels que la reconnaissance d'images, la reconnaissance vocale, le traitement du langage naturel et la prédiction de séries temporelles LeCun *et al.* (2015).

2.2.1 COMPOSANTS D'UN RÉSEAU DE NEURONES ARTIFICIELS

1. **Neurones :** Les unités de base d'un RNA, qui reçoivent des entrées, effectuent des calculs et transmettent des sorties.
2. **Poids :** Des valeurs numériques attribuées aux connexions entre les neurones, qui déterminent l'importance de l'entrée d'un neurone sur un autre.
3. **Biais :** Un terme ajouté à la somme pondérée des entrées pour ajuster le niveau d'activation du neurone.
4. **Fonction d'activation :** Une fonction mathématique appliquée à la somme pondérée des entrées et du biais pour déterminer la sortie du neurone. Des exemples courants incluent la fonction sigmoïde, la fonction ReLU (Rectified Linear Unit) et la fonction tangente hyperbolique.

2.2.2 TYPES DE RÉSEAUX DE NEURONES ARTIFICIELS

1. **Réseaux de neurones feedforward :** Les signaux se déplacent uniquement dans une direction, de l'entrée vers la sortie, sans boucles de rétroaction.

2. **Réseaux de neurones récurrents (RNN) :** Ils possèdent des connexions récurrentes, permettant de traiter des séquences de données et de conserver des informations sur les états précédents.
3. **Réseaux de neurones convolutifs (CNN) :** Spécialement conçus pour traiter des données structurées en grille, comme les images, en utilisant des opérations de convolution.

2.2.3 RÉSEAUX DE NEURONES CONVOLUTIFS (CNN)

Les réseaux de neurones convolutifs sont un type de réseau de neurones artificiels spécialement conçus pour traiter des données structurées en grille, telles que les images. Ils sont particulièrement efficaces pour capturer des caractéristiques spatiales et hiérarchiques dans les données.

EVOLUTION DE LA STRUCTURE DE CNN

Les travaux pionniers de LeCun *et al.* (1998) ont introduit le réseau LeNet pour la reconnaissance de caractères manuscrits, marquant la première utilisation des CNN dans l'extraction d'informations. LeNet se compose de sept couches cachées, dont deux couches de convolution, deux couches de sous-échantillonnage et deux couches denses, aboutissant à une couche de sortie formant un perceptron multicouche (MLP) final. Les performances initiales de ces réseaux convolutifs étaient contraintes par la capacité de calcul des ordinateurs de l'époque, en particulier pour les tâches impliquant de nombreuses classes ou de grandes images. Ce n'est qu'avec l'avènement d'AlexNet en 2012 Krizhevsky *et al.* (2012), les CNN sont devenus la norme pour les tâches de vision par ordinateur. Cette avancée est en partie due à l'implémentation efficace de la rétropropagation sur les unités de traitement graphique (GPU).d'Acremont (2020)

Les CNN modernes sont devenues plus profondes, comme le VGGNet de Simonyan & Zisserman (2014). Avec davantage de couches de convolution et de pooling maximal, les reseau de neurones convolutifs peuvent extraire des caractéristiques plus complexes des images. Cependant, l'augmentation de la profondeur a introduit de nouveaux défis, notamment un risque accru de surapprentissage dû à un nombre croissant de paramètres et de couches cachées. De nouvelles architectures de CNN ont émergé pour faciliter l'apprentissage, abandonnant la séquence classique de couches de convolution et de sous-échantillonnage au profit de chemins parallèles. Parmi ces réseaux figurent la famille Inception Szegedy *et al.* (2014) Szegedy *et al.* (2015) **?**, ResNet He *et al.* (2015) et DenseNet Huang *et al.* (2017). Ces modèles utilisent également la normalisation par lots Ioffe & Szegedy (2015) pour optimiser leur apprentissage.d'Acremont (2020)

ARCHITECTURE D'UN CNN

Les réseaux de neurones convolutifs sont composés de plusieurs types de couches :

Couches de convolution : Ces couches appliquent un ensemble de filtres convolutifs aux données d'entrée pour extraire des caractéristiques locales. Chaque filtre se déplace à travers l'image et effectue une opération de convolution, produisant ce qu'on appelle une carte des caractéristiques.

$$C(i,j) = (I * N)(i,j) = \sum_{m=0}^{x-1} \sum_{n=0}^{y-1} I(i-m, j-n) \times N(m,n) \tag{2.1}$$

La couche de convolution utilise des filtres appelés aussi noyeaux pour effectuer l'opération de convolution (2.1) sur l'entrée. Chaque filtre est une petite matrice de poids qui glisse sur l'entrée pour produire une carte de caractéristiques (feature map). Les filtres sont conçus

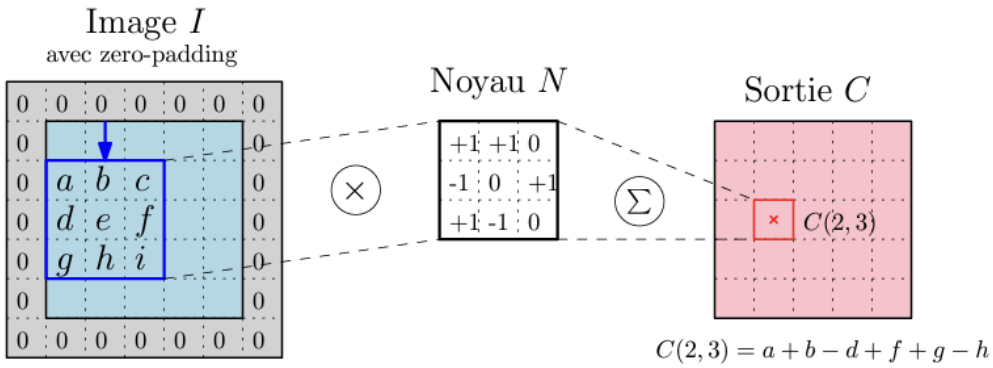

FIGURE 2.4 : Opération de convolution

©Abib Sy

pour détecter des caractéristiques spécifiques, telles que des bords, des textures ou des motifs. la figure 2.4 illustre une opération de convolotion.

Les paramètres principaux d'une couche de convolution sont :

— Taille du filtre :La taille du filtre est un paramètre essentiel dans les couches de convolution des réseaux de neurones. Un filtre de plus grande dimension est capable d'englober des caractéristiques plus larges et plus générales de l'entrée, permettant ainsi une perception plus globale des motifs. À l'inverse, un filtre de plus petite taille se focalise sur des caractéristiques plus précises et détaillées, capturant ainsi des informations fines et subtiles.

— Quantité de filtres : Cette valeur représente le nombre de filtres uniques employés dans une couche donnée. Chaque filtre individuel génère une carte de caractéristiques distinctes, permettant ainsi l'extraction de diverses caractéristiques de l'entrée.

COUCHES D'ACTIVATION :

La fonction d'activation ReLU (Rectified Linear Unit) est largement répandue et fréquemment utilisée dans les architectures de réseaux de neurones convolutifs. Elle est appliquée à la suite de chaque couche de convolution afin d'introduire une non-linéarité essentielle au traitement et à la modélisation des données complexes. Elle est définie comme suit :

$$f(x) = \max(0, x)$$

où x est l'entrée de la couche d'activation.

Propriétés de la ReLU

- **Non-linéarité :** La ReLU introduit une non-linéarité simple dans le réseau, ce qui est essentiel pour apprendre des représentations complexes des données.
- **Calcul efficace :** La fonction ReLU est très simple à calculer, ce qui la rend efficace pour l'entraînement et l'inférence dans les réseaux de neurones profonds.
- **Réduction du problème de disparition du gradient :** Contrairement à d'autres fonctions d'activation comme la sigmoïde ou la tangente hyperbolique, la ReLU atténue le problème de disparition du gradientest qui est un défi rencontré lors de l'entraînement des réseaux de neurones profonds. Il se produit lorsque les gradients (valeurs utilisées pour ajuster les poids du réseau) deviennent très petits, presque nuls, au fur et à mesure qu'ils sont propagés à travers les couches du réseau. Cela rend difficile la mise à jour efficace des poids des couches initiales, ralentissant ou arrêtant même l'apprentissage du réseau.

COUCHES DE POOLING :

Les couches de pooling, souvent appelées sous-échantillonnage, réduisent la dimensionnalité des cartes de caractéristiques en regroupant plusieurs valeurs adjacentes en une seule. Le pooling le plus courant est le max-pooling, qui conserve la valeur maximale dans chaque groupe.

COUCHES ENTIÈREMENT CONNECTÉES :

Après plusieurs couches de convolution et de pooling, les CNN incluent généralement une ou plusieurs couches entièrement connectées. Ces couches combinent les caractéristiques extraites pour effectuer la classification ou la régression.

COUCHE DE SORTIE :

Produit la sortie finale, souvent avec une fonction softmax pour la classification. La fonction softmax est une fonction mathématique utilisée dans les réseaux de neurones pour transformer les scores bruts (appelés logits) en probabilités. Elle est souvent employée dans la couche de sortie des réseaux de neurones pour la classification multi-classes. La fonction softmax s'assure que toutes les probabilités sont positives et que leur somme est égale à 1, permettant ainsi d'interpréter les scores comme des probabilités pour chaque classe.

2.3 FONCTIONNEMENT

Le fonctionnement d'un reseau de neurones convolutif peut être résumé en plusieurs étapes comme suit :

1. **Convolution :** Les filtres glissent sur l'entrée et effectuent une opération de convolution.
2. **Activation :** Application de la fonction d'activation aux cartes de caractéristiques.
3. **Pooling :** Réduction de la dimensionnalité des cartes de caractéristiques.
4. **Propagation avant :** Les étapes de convolution, d'activation et de pooling se répètent.
5. **Propagation arrière :** Les erreurs sont propagées en arrière pour ajuster les poids et les biais.

2.4 TYPES DE PARAMÈTRES

Les principaux types de paramètres dans un CNN sont :

— **Poids des filtres :** Appris pendant l'entraînement pour détecter des caractéristiques spécifiques.
— **Biais :** Associés à chaque carte de caractéristiques et appris pendant l'entraînement.
— **Hyperparamètres :** Définis avant l'entraînement, comme le nombre et la taille des filtres, le pas de convolution, etc.

2.4.1 APPRENTISSAGE DANS LES RÉSEAUX DE NEURONES ARTIFICIELS

Les CNN sont souvent entraînées à l'aide de l'apprentissage supervisé pour effectuer des tâches telles que la classification d'images, la détection d'objets et la segmentation sémantique. Ils ont révolutionné le domaine de la vision par ordinateur en atteignant des performances remarquables dans diverses applications.

L'apprentissage dans les réseau de neurones artificiels implique l'ajustement des poids et des biais pour minimiser l'erreur entre les sorties prédites et les sorties réelles. Cela se fait généralement à l'aide d'algorithmes d'optimisation tels que la descente de gradient.

L'algorithme de rétropropagation est couramment utilisé pour calculer les gradients nécessaires à la mise à jour des poids dans les réseaux de neurones feedforward Rumelhart *et al.* (1986).

2.4.2 DÉFIS ET PERSPECTIVES

Bien que les réseaux de neurones artificiels aient montré des performances remarquables dans diverses tâches, ils présentent certains défis, tels que la nécessité de grandes quantités de données pour l'entraînement, la vulnérabilité aux attaques adverses et la difficulté d'interprétation des modèles. La recherche continue dans le domaine vise à surmonter ces défis et à étendre l'application des RNA à de nouveaux domaines Schmidhuber (2015).

Exemple d'application : Les autoencodeurs convolutifs sont une variante des réseau de neurones convolutifs utilisée pour la reconstruction d'images. Contrairement aux encodeurs automatiques traditionnels qui utilisent des couches entièrement connectées, les CAE exploitent la structure 2D des images à l'aide de couches de convolution et de pooling, permettant ainsi une meilleure capture des caractéristiques spatiales **?**.

2.5 LES AUTOENCODEURS

Un autoencodeur est un type de réseau de neurones utilisé pour l'apprentissage non supervisé, qui vise à apprendre une représentation compressée (encodage) des données d'entrée, généralement dans le but de réduire la dimensionnalité ou de réaliser de la détection d'anomalies . Il se compose de deux parties principales : un encodeur, qui transforme les données d'entrée en une représentation de dimension réduite, et un récepteur, qui tente de reconstruire les données d'origine à partir de cette représentation compressée Goodfellow *et al.* (2016a).

2.5.1 STRUCTURE DES AUTOENCODEURS

Un autoencodeur se compose de deux parties principales :

1. Encodeur : Cette partie du réseau prend en entrée des données et les transforme en une représentation de dimension réduite, souvent appelée code ou espace latent. L'encodeur apprend à conserver les caractéristiques importantes des données d'entrée tout en particulier leur dimensionnalité.
2. Décodeur : Le récepteur prend le code produit par l'encodeur et tente de reconstruire les données d'origine à partir de cette représentation réduite. L'objectif est de minimiser la différence entre les données d'origine et les données reconstruites, souvent enregistrées par une fonction de perte comme l'erreur quadratique moyenne.

2.5.2 TYPES D'AUTOENCODEURS

Il existe plusieurs variantes d'autoencodeurs, chacune avec des caractéristiques et des applications spécifiques :

1. **Autoencodeurs traditionnels :** Ils utilisent généralement des couches entièrement connectées pour l'encodeur et le boîtier.
2. **Autoencodeurs convolutionnels :** Ces autoencodeurs utilisent des couches convolutionnelles dans l'encodeur et des couches de convolution transposées (ou déconvolutionnelles) dans le terminal. Ils sont particulièrement adaptés au traitement des images et des données spatiales Masci *et al.* (2011).
3. **Autoencodeurs à bruit (Denoising Autoencoders) :** Ils sont entraînés à reconstruire des données d'entrée corrompues par du bruit, ce qui les rend utiles pour la détection d'anomalies et la restauration d'images Vincent *et al.* (2008).

4. **Autoencodeurs variationnels (Variational Autoencoders, VAE) :** Cette variante introduit une couche stochastique dans l'espace latent, ce qui permet de générer de nouvelles données similaires à celles d'entrée. Les VAE sont utilisés dans la génération de données et l'apprentissage de représentations probabilistes Kingma & Welling (2013).

2.5.3 APPLICATIONS DES AUTOENCODEURS

Les autoencodeurs ont une large gamme d'applications, notamment :

— **Réduction de dimensionnalité :** Ils peuvent être utilisés pour compresser des données tout en préservant leurs caractéristiques essentielles.

— **Détection d'anomalies :** En apprenant à reconstruire les données normales, les autoencodeurs peuvent identifier des données anormales ou atypiques.

— **Génération de données :** Les VAE et d'autres variantes peuvent générer de nouvelles données similaires aux données d'entrée.

— **Prétraitement pour l'apprentissage supervisé :** Les représentations apprises par les autoencodeurs peuvent servir de caractéristiques d'entrée pour des modèles d'apprentissage supervisé.

2.5.4 AUTOENCODEUR CONVOLUTIF

2.5.5 AUTOENCODEUR CONVOLUTIF POUR LA RECONSTRUCTION D'IMAGES

L'encodeur automatique traditionnel utilisait une couche entièrement connectée comme encodeur et décodeur, tandis que CAE peut capturer la structure 2D de l'image. Les modèles CAE incluent des couches de convolution, de pooling, entièrement connectées, de déconvolution, de non-pooling et de perte.Li & Yeh (2019) Le flux de travail de l'apprentissage des fonctionnalités non supervisé à l'aide de CAE est illustré à la figure 2.5 . La couche de

convolution utilise des noyaux de filtre qui se déplacent de gauche à droite et descendent jusqu'à la ligne suivante avec le même pas jusqu'à la fin de l'image et extraient les cartes de caractéristiques latentes. La couche de pooling réduit l'espace des cartes de caractéristiques après la couche de convolution. De plus, il est utilisé pour extraire des fonctionnalités supérieures, conservant ainsi le processus efficace du modèle de formation. Dans la partie encodeur, nous apprenons les cartes de caractéristiques en utilisant la couche de convolution et la couche de pooling. La déconvolution signifie que le processus inverse de convolution et de dé-pooling est appelé reverse max pooling. La couche de dé-pooling restaure la fonctionnalité max-pooling au bon endroit et remplit zéro dans les autres positions. Nous reconstruisons l'image par des processus de déconvolution et de suréchantillonnage et la rendons aussi proche que possible des données d'entrée.

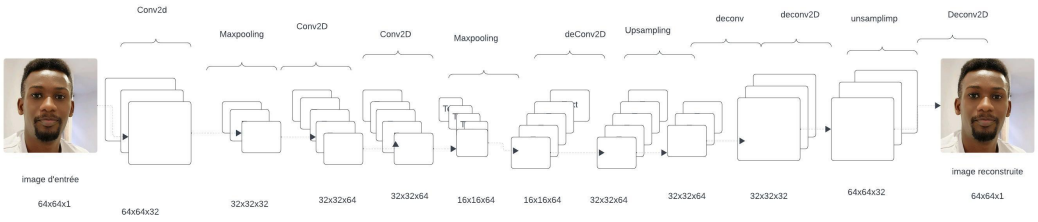

FIGURE 2.5 : Illustration d'un Autoencodeur convolutif

©Abib Sy

2.5.6 POTENTIEL DES AUTOENCODEURS DANS LA CRYPTOGRAPHIE

Les autoencodeurs, grâce à leurs capacités exceptionnelles d'apprentissage de représentations, ont été explorés comme une méthode potentiellement révolutionnaire pour la cryptographie, notamment pour la sécurisation des images numériques Doe & Smith (2020b). Les autoencodeurs convolutionnels profonds offrent une voie prometteuse pour la sécurisation des données, notamment grâce à leur capacité à générer des représentations compactes et

difficiles à inverser des données d'entrée. Cette propriété est particulièrement attrayante pour la cryptographie des images, où il est essentiel de préserver la qualité tout en assurant la confidentialité.

Leur fonctionnement repose sur l'apprentissage de caractéristiques essentielles des données pour produire un encodage compact, rendant la tâche de déchiffrement sans la clé appropriée extrêmement difficile. Cette technologie pourrait surpasser les méthodes traditionnelles en offrant une sécurité renforcée pour les données en mouvement et au repos, grâce à sa capacité à générer des encodages uniques et complexes. Cependant, l'implémentation des autoencodeurs présente des défis techniques, notamment la nécessité d'un grand volume de données pour l'entraînement et la gestion de la complexité computationnelle, qui doivent être pris en compte pour assurer leur efficacité dans des environnements réels Doe & Smith (2020b).

2.5.7 VULNÉRABILITÉS DES AUTOENCODEURS

Les autoencodeurs, malgré leur utilité diverses dans les applications d'apprentissage automatique, présentent certaines vulnérabilités qui peuvent affecter leurs performances et leur sécurité.

SURAPPRENTISSAGE

L'une des principales vulnérabilités des autoencodeurs est le risque de surapprentissage, en particulier lorsque la dimension de l'espace latent est trop grande par rapport à la complexité des données. Dans ce cas, l'autoencodeur peut apprendre à reproduire parfaitement les données d'entraînement, y compris le bruit, au lieu de capturer les caractéristiques générales sous-

jacentes. Cela peut conduire à une mauvaise généralisation sur de nouvelles données Hawkins (2004).

SENSIBILITÉ AUX ANOMALIES

Les autoencodeurs sont souvent utilisés pour la détection d'anomalies en supposant que les anomalies seront mal reconstruites par rapport aux données normales. Cependant, si les anomalies sont similaires aux données normales ou si l'autoencodeur est trop puissant, il peut apprendre à reconstruire également les anomalies, ce qui réduit son efficacité pour détecter les comportements atypiques Zhou & Paffenroth (2017).

VULNÉRABILITÉS AUX ATTAQUES ADVERSES

Les autoencodeurs, comme d'autres modèles de réseaux de neurones, peuvent être vulnérables aux attaques indésirables. Des exemples perturbés intentionnellement peuvent être conçus pour induire des erreurs dans la reconstruction, ce qui peut nuire à la sécurité des applications basées sur des autoencodeurs, telles que la détection de fraudes ou de logiciels malveillants Goodfellow *et al.* (2014). Dans Szegedy *et al.* (2013) , les auteurs ont prouvé que les autoencodeur sont extrêmement vulnérables aux attaques adverses. Cela entraîne de sérieux défis en matière de sécurité et de sérieux problèmes de fiabilité. Les exemples contradictoires sont définis comme des entrées malveillantes obtenues en perturbant ou en perturbant légèrement l'entrée d'origine, de telle manière que le classificateur ou la fonction discriminante ne parvient pas à classer les signaux reçus Goodfellow *et al.* (2014). Les attaques contradictoires sont classées en attaques physiques et numériques qui dépendent des informations connues par l'attaquant et de sa capacité à accéder aux entrées du système Kurakin *et al.* (2016) . Dans une attaque numérique, l'attaquant peut former complètement l'entrée

du modèle du système, alors que, dans une attaque physique, l'attaquant peut implicitement appliquer l'entrée au modèle du système Kurakin *et al.* (2016) . Dans les attaques physiques antagonistes, l'attaquant vise à perturber le modèle de sortie du récepteur en ajoutant une perturbation optimisée (p) à l'entrée du message, ce qui entraîne une mauvaise classification des sorties Goodfellow *et al.* (2014) Sadeghi & Larsson (2018) . Contrairement aux attaques de brouillage classiques, la perturbation ici n'est pas seulement du bruit mais plutôt un vecteur optimisé bien conçu dans l'espace des caractéristiques du domaine d'entrée qui pourrait tromper le modèle correspondant.

DÉPENDANCE AUX DONNÉES D'ENTRAÎNEMENT

La performance des autoencodeurs dépend fortement de la qualité et de la diversité des données d'entraînement. Si les données d'entraînement ne sont pas représentatives de l'ensemble de données complet, l'autoencodeur peut ne pas être en mesure de capturer correctement la structure des données, ce qui entraîne une performance médiocre sur les données non vues Bengio *et al.* (2013).

CHAPITRE III

TRAVAUX CONNEXES

Ce chapitre examine diverses recherches liées à l'utilisation des réseaux de neurones artificiels pour le chiffrement, en particulier les approches basées sur des réseaux neuronaux et des techniques de brouillage pour sécuriser la transmission des données d'images. Il met en évidence les avantages et les inconvénients de ces méthodes, ainsi que leur efficacité pour renforcer la sécurité des images numériques.

3.1 CHIFFREMENT PAR RESEAU DE NEURONES ARTIFICIEL

Dans leur étude, Lian *et al.* (2004) explorent l'utilisation d'un réseau neuronal chaotique pour le cryptage d'images encodées JPEG2000. L'algorithme proposé intègre des techniques de chaos dans le processus de cryptage pour améliorer la sécurité des images numériques. Cette approche innovante montre le potentiel d'associer des réseaux neuronaux et la théorie du chaos pour renforcer la protection des données visuelles dans des applications diverses.

Le processus de cryptage étant une fonction à sens unique, les réseaux neuronaux artificiels sont considérés comme les mieux adaptés à cette fin, car ils possèdent des caractéristiques telles qu'une sécurité élevée, l'absence de distorsion et la capacité d'exécuter des caractéristiques d'entrée-sortie non linéaires. Ainsi, il est possible d'éliminer le besoin d'échange de clés, qui est une condition préalable pour la plupart des algorithmes utilisés aujourd'hui. À titre d'exemple, Lian (2007) a étudié les propriétés des réseaux neuronaux pour proposer une authentification à faible coût pour les images ou les vidéos. L'auteur affirme que l'approche a la capacité intégrée de détecter si les données sont modifiées de manière malveillante. L'auteur a enfin mis en évidence plusieurs questions en suspens dans ce domaine, notamment :

quelle propriété des réseaux neuronaux doit être exploitée pour la protection des données ; quels modèles de réseaux neuronaux conviennent à la protection des données ; et la capacité d'apprentissage des réseaux neuronaux. Dans un autre travail, Munukur & Gnanam (2009) ont utilisé un réseau neuronal dans le récepteur à des fins de décryptage en exploitant l'algorithme de rétropropagation dans le récepteur pour l'entraîner avec un texte chiffré de 12 bits comme entrée et un texte en clair de 8 bits comme sortie cible. Le texte en clair à l'entrée comprend également une certaine impureté basée sur une clé prédéterminée pour afin d'induire en erreur toute personne susceptible d'écouter les conversations.

De même, les travaux de Joshi *et al.* (2012) ont ciblé la sécurisation de la transmission de données d'images à l'aide d'un algorithme aléatoire en introduisant de la confusion dans les données, et l'ajout d'impuretés pour tromper le cryptanalyste. Dans une autre recherche, Bigdeli *et al.* (2012a) ont proposé un algorithme de cryptage/décryptage d'images basé sur un réseau neuronal chaotique. Le réseau utilisé comprend deux couches : la couche de neurones chaotiques (CNL) et la couche de neurones de permutation (PNL), chacune avec trois couches. L'approche utilise un code d'authentification de 160 bits pour générer les conditions initiales et les paramètres des deux couches. Le processus global a été répété plusieurs fois afin de le rendre robuste et d'en accroître la complexité. La méthode proposée utilise deux clés supplémentaires et une légère différence dans l'une d'entre elles ne permet pas de décrypter une image avec succès. Dans un autre travail, les mêmes auteurs Bigdeli *et al.* (2012b) ont proposé une méthode de cryptage basée sur un algorithme de cryptage hybride basé sur le chaos. L'algorithme utilise une architecture de permutation-diffusion qui utilise des paramètres de contrôle chaotiques pour la permutation. Une carte logistique génère ces paramètres de contrôle pour l'étape de permutation. Dans la phase de diffusion, une autre carte logistique chaotique avec des conditions initiales et des paramètres différents a été utilisée pour générer des conditions initiales pour un réseau neuronal Hopfield hyper-chaotique afin de générer un

flux de clés pour l'homogénéisation de l'image mélangée. Zirra *et al.* (2011) a essayé des techniques différentes des réseaux neuronaux chaotiques, où le brouillage a été utilisé pour transformer l'information en un ensemble d'équations linéaires et le déchiffrement a été réalisé en résolvant les systèmes d'équations linéaires avec les principes du schéma de codage delta, une formule et une table de recherche. Les lecteurs sont invités à se référer à Memon *et al.* (2007) et à Memon & Khoja (2009) pour de plus amples informations sur ce sujet.

Notre étude s'inscrit dans la continuité de ces recherches, en explorant le potentiel des autoencodeurs pour le cryptage d'images. Nous nous distinguons des approches antérieures en mettant l'accent sur l'évaluation de la performance et de la sécurité du modèle à l'aide de métriques telles que la similarité de cosinus, l'entropie, les taux de Kendall et de Spearman, ainsi que l'erreur quadratique moyenne (MSE). Ces métriques, combinées aux temps de cryptage et de décryptage, nous permettent d'offrir une analyse assez complète de la robustesse de notre modèle face aux défis posés par la protection des données visuelles dans les environnements intelligents.

CHAPITRE IV
IMPLÉMENTATIONS ET TESTS

Dans le chapitre 4, nous plongeons au cœur des implémentations pratiques et des tests de comparaisons de divers modèles de cryptographie, avec un accent particulier sur un autoencodeur convolutif profond (CAE). Une vaste collection de 233,218 images a été mobilisée pour l'entraînement, mettant en lumière les subtilités du traitement des données et la configuration minutieuse du CAE. L'efficacité de l'autoencodeur est comparée à celle des méthodes cryptographiques traditionnelles comme AES-SHA-HMAC256, Twofish, et Chacha20, en se basant sur des mesures de performance telles que l'entropie, le MSE, et les temps de chiffrement/ déchiffrement. La robustesse du CAE est également testée dans des scénarios d'attaques adverses, démontrant sa capacité à reconstruire avec précision les images après compression. Le chapitre conclut en discutant des résultats obtenus et des défis à relever, en reconnaissant la complexité du réseau et l'importance de trouver un équilibre entre sécurité et performance.

4.1 IMPLÉMENTATION DU MODEL D'AUTOENCODEUR CONVOLUTIF PROFOND

4.1.1 JEUX DE DONNÉES

Pour entraîner notre modèle d'autoencodeur convolutionnel profond, nous avons utilisé un ensemble de données de 233 218 images couleur de format JPEG redimensionnées à 24x24 pixels. Les 233,218 images proviennent de la base de donnée de photos Dreamstime [1] qui est une banque de 39,902,701 images libre de droits. Nous avons choisi le format JPEG plus

1. https ://www.megapixl.com/stock-photos

précisément car, pour la compression sans perte, la meilleure technique disponible dans la littérature est le JPEG en mode sans perte (JPEG-LS) Weinberger *et al.* (2000). Les fichiers JPEG contiennent moins de données, donc sont plus légers que les PNG.

Dans le contexte d'un autoencodeur utilisé pour l'encodage et la reconstruction d'images comme notre modele CAE, le balancement des données n'est généralement pas nécessaire. L'objectif principal est de capturer les caractéristiques générales des données, indépendamment de la distribution des classes. Il est cependant important de s'assurer que l'ensemble de données d'entraînement est diversifié pour permettre une représentation fidèle des variations des données.

4.1.2 PRÉTRAITEMENT DES DONNÉES

Avant l'entraînement, les données ont fait l'objet d'un prétraitement méticuleux. Un bruit gaussien a été ajouté aux données afin d'accroître la résistance du modèle aux perturbations potentielles, et les valeurs des pixels ont été normalisées pour se situer dans une fourchette de 0 à 1. Ces étapes sont cruciales pour préparer les données à être traitées efficacement par le réseau neuronal.

4.1.3 CONFIGURATION DE NOTRE MODEL D'AUTOENCODEUR CONVOLUTIF PROFOND

Comme indiqué précédemment, ce modèle utilise un autoencodeur profond en Python via la bibliothèque Keras de Tensorflow. Le modèle dispose d'éléments spécifiques pour assurer sa robustesse face aux perturbations, notamment le bruit. Le modèle est conçu pour l'encription et la decription d'images. Voici les principales caractéristiques du modèle et son utilisation :

ENCODAGE

Nous utilisons ici trois couches Conv2D avec activation ReLU, taille de noyau de (3, 3), remplissage 'same' et régularisation L2. Nous ajoutons également des couches MaxPooling pour réduire les dimensions spatiales et Dropout pour la régularisation.

— **Couche d'entrée :** La couche d'entrée prend des images de taille (height, width, channels) où height et width sont les dimensions de l'image et channels est le nombre de canaux de couleur (3 pour les images RGB).

— **Première couche de convolution :** Une couche de convolution avec 64 filtres de taille (3, 3), une activation ReLU, un padding 'same' pour conserver la taille de l'image, et une régularisation L2 avec un facteur de 0.01 pour réduire le surapprentissage.

— **Première couche de dropout :** Une couche de dropout avec un taux de 0.2 pour réduire le surapprentissage en désactivant aléatoirement 20% des neurones pendant l'entraînement.

— **Première couche de MaxPooling :** Une couche de MaxPooling de taille (2, 2) pour réduire la dimensionnalité de l'image tout en conservant les caractéristiques importantes.

— **Deuxième couche de convolution :** Une couche de convolution similaire à la première mais avec 128 filtres.

— **Deuxième couche de dropout :** Une couche de dropout similaire à la première.

— **Deuxième couche de MaxPooling :** Une couche de MaxPooling similaire à la première.

— **Troisième couche de convolution :** Une couche de convolution similaire à la première mais avec 256 filtres.

— **Troisième couche de MaxPooling :** Une couche de MaxPooling similaire à la première. Après cette couche, l'image est fortement réduite en taille mais contient des caractéristiques importantes encodées.

TABLEAU 4.1 : Représentation de notre Encodeur

Couche	Filtres	Taille du filtre	Activation	Padding	Régularisation	Taux de Dropout	Pooling
Conv2D (1ère couche)	64	(3, 3)	ReLU	same	L2 (0.01)	0.2	-
MaxPooling2D (1ère couche)	-	-	-	same	-	-	(2, 2)
Conv2D (2ème couche)	128	(3, 3)	ReLU	same	L2 (0.01)	0.2	-
MaxPooling2D (2ème couche)	-	-	-	same	-	-	(2, 2)
Conv2D (3ème couche)	256	(3, 3)	ReLU	same	L2 (0.01)	-	-
MaxPooling(3ème couche)	-	-	-	same	-	-	(2, 2)

DÉCODAGE

L'architecture du modèle comprend trois couches de convolution avec activation ReLU, chacune ayant une taille de noyau de (3, 3) et un remplissage réglé sur "même". Le modèle comporte également des couches de régularisation L2 et d'échantillonnage ascendant pour augmenter les dimensions spatiales. La dernière couche du modèle utilise une fonction d'activation "sigmoïde" pour générer des valeurs de sortie entre 0 et 1.

— **Première couche de convolution :** Une couche de convolution avec 256 filtres, similaire à la dernière couche de l'encodeur.

TABLEAU 4.2 : Représentation de notre Decodeur

Couche	Filtres	Taille du filtre	Activation	Padding	Régularisation	UpSampling
Conv2D (1ère couche)	256	(3, 3)	ReLU	same	L2 (0.01)	-
UpSampling2D (1ère couche)	-	-	-	-	-	(2, 2)
Conv2D (2ème couche)	128	(3, 3)	ReLU	same	L2 (0.01)	-
UpSampling2D (2ème couche)	-	-	-	-	-	(2, 2)
Conv2D (3ème couche)	64	(3, 3)	ReLU	same	L2 (0.01)	-
UpSampling (3ème couche)	-	-	-	-	-	(2, 2)
Conv2D (Couche finale)	Channels	(3, 3)	Sigmoid	same	-	-

— **Première couche d'UpSampling :** Une couche d'UpSampling de taille (2, 2) pour augmenter la dimensionnalité de l'image.

— **Deuxième couche de convolution :** Une couche de convolution avec 128 filtres, similaire à la deuxième couche de l'encodeur.

— **Deuxième couche d'UpSampling :** Une couche d'UpSampling similaire à la première.

— **Troisième couche de convolution :** Une couche de convolution avec 64 filtres, similaire à la première couche de l'encodeur.

— **Troisième couche d'UpSampling :** Une couche d'UpSampling similaire à la première.

— **Couche de convolution finale :** Une couche de convolution avec un nombre de filtres égal au nombre de canaux de l'image d'entrée, une activation 'sigmoid' pour obtenir des valeurs entre 0 et 1, et un padding 'same' pour conserver la taille de l'image.

4.1.4 ROBUSTESSE DE L'AUTOENCODEUR

La robustesse d'un autoencodeur, en particulier dans le contexte des images, fait référence à sa capacité à apprendre des représentations utiles et à reconstruire des images avec précision, même en présence de bruit ou de perturbations. L'ajout de bruit aux données d'entraînement est une technique couramment utilisée pour améliorer la robustesse des autoencodeurs. Voici quelques points clés sur la robustesse et l'ajout de bruit :

Généralisation : Un autoencodeur robuste est capable de généraliser à partir des données d'entraînement bruitées et de reconstruire des images propres. Cela signifie qu'il apprend à ignorer le bruit et à se concentrer sur les caractéristiques importantes des données.

Régularisation : La robustesse est améliorée grâce à des techniques de régularisation comme la régularisation L2 (aussi appelée régularisation de poids) et le dropout. La régularisation L2 pénalise les poids élevés dans le modèle, encourageant ainsi des poids plus petits et plus stables. Le dropout désactive aléatoirement une partie des neurones pendant l'entraînement, ce qui aide à prévenir le surapprentissage et améliore la robustesse du modèle.

Capacité de débruitage : Un autoencodeur robuste peut être utilisé comme un débruiteur d'images, c'est-à-dire qu'il peut apprendre à supprimer le bruit des images d'entrée. Cela est particulièrement utile dans des applications telles que l'amélioration de la qualité d'image ou la restauration d'image.

AJOUT DE BRUITS

Bruit gaussien : Le bruit gaussien est un type courant de bruit ajouté aux images. Il est caractérisé par une distribution normale et peut être contrôlé par deux paramètres : la moyenne (généralement fixée à 0) et l'écart-type (qui détermine l'intensité du bruit).

nous avons choisi le bruit gaussien etant donné que la sécurité et la robustesse sont les principaux défis en raison des faiblesses des auto-encodeurs face aux attaques physiques adverses. Certains travaux comme Sadeghi & Larsson (2019) et Albaseer *et al.* (2020) ont été consacrés à résoudre ces problèmes en utilisant uniquement le modèle de canal Additive White Gaussian Noise (AWGN).

Augmentation de données : L'ajout de bruit aux images d'entraînement peut être considéré comme une forme d'augmentation de données. Cela augmente la diversité des données d'entraînement et aide le modèle à apprendre à reconnaître et à reconstruire des images dans des conditions moins idéales.

Contrôle du niveau de bruit : Le facteur de bruit (noise_factor) détermine l'intensité du bruit ajouté aux images. Un facteur de bruit plus élevé signifie plus de bruit et peut rendre l'apprentissage plus difficile, mais peut également conduire à un modèle plus robuste si l'entraînement est réussi.

Clipping : Après l'ajout de bruit, les valeurs des pixels peuvent dépasser la plage valide (généralement [0, 1] pour les images normalisées). Le clipping est utilisé pour limiter ces valeurs à la plage valide, en s'assurant que les images bruitées restent valides pour l'entraînement.

4.1.5 ENTRAINEMENT

Le modèle est entraîné avec les images bruitées en utilisant la fonction de perte mean_-squared_error et l'optimiseur Adam comme indiqué dans l'algorithme 4.1.

1 [1] Initialiser les paramètres du modèle θ
2 Initialiser les premiers moments $m \leftarrow 0$ et les seconds moments $v \leftarrow 0$
3 Définir le taux d'apprentissage α, les hyperparamètres $\beta_1 = 0.9$ et $\beta_2 = 0.999$
4 Définir $t \leftarrow 0$ et $\varepsilon \leftarrow 10^{-8}$ **tant que** *non convergent* **faire**
5 **fin**
6 $t \leftarrow t + 1$
7 Calculer le gradient $\nabla_\theta L(\theta)$ de la fonction de perte
8 Mettre à jour les premiers moments : $m \leftarrow \beta_1 \cdot m + (1 - \beta_1) \cdot \nabla_\theta L(\theta)$
9 Mettre à jour les seconds moments : $v \leftarrow \beta_2 \cdot v + (1 - \beta_2) \cdot (\nabla_\theta L(\theta))^2$
10 Corriger le biais pour les premiers moments : $\hat{m} \leftarrow m/(1 - \beta_1^t)$
11 Corriger le biais pour les seconds moments : $\hat{v} \leftarrow v/(1 - \beta_2^t)$
12 Mettre à jour les paramètres : $\theta \leftarrow \theta - \alpha \cdot \hat{m}/(\sqrt{\hat{v}} + \varepsilon)$

Algorithme 4.1 : Optimiseur Adam pour notre Autoencodeur convolutif Profond

Le modèle est entraîné sur les données bruitées, en accordant une attention particulière aux paramètres tels que le nombre d'époques 1000 et la taille du lot 128. Nous avons utilisé le Backend Google Compute Engine Python 3 (TPU) comme ressource pour entraîner notre modèle avec une RAM de 32,5 Go et un disque de 256 Go. Les images sont redimensionnées et normalisées à partir d'un répertoire spécifié.

Paramètres d'entraînement : L'entraînement est effectué sur un nombre défini d'époques (par exemple, 1000) avec une taille de lot spécifiée (par exemple, 128). Une époque correspond à une passe complète sur l'ensemble des données d'entraînement. La taille du lot détermine le nombre d'échantillons traités avant la mise à jour des poids du modèle.

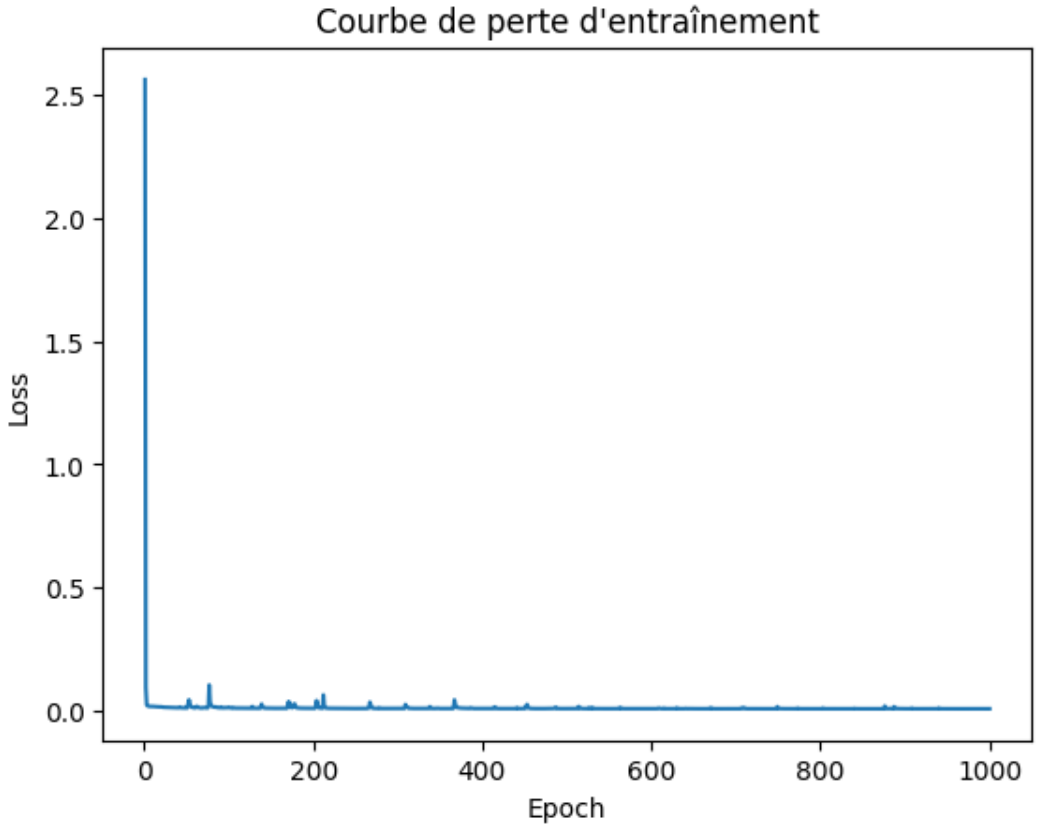

FIGURE 4.1 : Courbe de perte d'entrainement de notre model d'Autoencodeur

©Abib Sy

Boucle d'entraînement : À chaque époque, le modèle est entraîné sur les images bruitées (X_train_noisy) avec les images originales (X_train) comme cibles. Cela signifie que le modèle apprend à reconstruire les images originales à partir des images bruitées.

Suivi de la performance : La perte d'entraînement est enregistrée à chaque époque pour surveiller la performance du modèle. Comme le montre la figure 4.1 qui est la courbe de perte et d'entrainement de notre model d'autoencodeur, nous notons une baisse de la perte indique que le modèle s'améliore dans la reconstruction des images.

POST-ENTRAÎNEMENT :

Évaluation : Après l'entraînement, le modèle peut être évalué sur un ensemble de données de test pour vérifier sa capacité à reconstruire des images non vues pendant l'entraînement.

Sauvegarde : Le modèle entraîné est sauvegardé sur le disque pour une utilisation future, par exemple pour la débruitage d'images ou comme partie d'un système de traitement d'images plus large.

En résumé, l'entraînement de notre autoencodeur profond implique la préparation minutieuse des données, la configuration appropriée du modèle et une boucle d'entraînement bien définie, avec un suivi attentif de la performance et des ajustements en cours de route pour optimiser les résultats.

4.2 ÉVALUATION DES MÉTRIQUES

Dans cette section, nous présentons les métriques utilisés pour cette étude comparatives et les motivations de leur choix.

4.2.1 ENTROPY

L'entropie est une mesure de l'incertitude ou du désordre dans un ensemble de données. Plus l'entropie est élevée, plus l'ensemble de données est diversifié et moins prévisible Shannon (1948). Dans le contexte des prédictions d'un modèle, l'entropie peut être utilisée pour mesurer l'incertitude associée à ces prédictions, ce qui est crucial pour garantir la robustesse de l'approche de cryptage. L'objectif principal est d'analyser comment l'entropie des images cryptées par un modèle d'autoencodeur convolutif évolue dans le temps et comment elle peut

être utilisée comme indicateur de la sécurité du processus de cryptage. Pour calculer l'entropie des prédictions d'un modèle, vous pouvez utiliser la formule de l'entropie de l'information. Supposons que vous disposiez d'un ensemble de prédictions pour un échantillon, représenté par la distribution de probabilité $P(y_i)$

$$-\sum_i P(y_i).log_2(P(y_i)$$

Cela signifie que plus la distribution de probabilité des prédictions est uniforme, plus l'entropie est élevée. Si toutes les prédictions sont concentrées sur une seule classe, l'entropie est faible. L'entropie est un concept fondamental en cryptographie car elle est liée à la qualité du caractère aléatoire utilisé pour générer les clés cryptographiques. Une bonne gestion de l'entropie est essentielle pour garantir la sécurité des systèmes cryptographiques et prévenir les attaques basées sur les prédictions de clés. Une bonne valeur d'entropie pour les données cryptées doit être proche de la limite maximale possible, ce qui indique une distribution de probabilité uniforme. Par exemple, une entropie de 8 bits est considérée comme excellente dans le contexte des systèmes de cryptographie, car elle suggère que chaque bit de sortie a une probabilité égale d'être 0 ou 1.

4.2.2 MSE

L'erreur quadratique moyyenne est une mesure couramment utilisée pour évaluer les performances des autoencodeurs, qui sont des modèles de réseaux neuronaux souvent utilisés pour la compression et la reconstruction de données. Dans le contexte des autoencodeurs, l'erreeur quadratique moyenne (EQM) mesure la différence entre les données d'entrée et les données reconstruites par le modèle. L'importance de l'EQM dans la sécurité des autoencodeurs peut être abordée sous plusieurs angles :

Évaluation de la reconstruction L'EQM est utilisée pour mesurer la qualité de la reconstruction des données en comparant les données d'entrée à celles reconstruites par l'autoencodeur. Une EQM proche de 0 est idéale, mais la valeur acceptable dépend fortement du contexte de l'application Ali & Abead (2016). Par exemple, une MSE inférieure à 0,001 peut être considérée comme excellente pour la reconstruction de données d'image.

L'image dégradée Î est toujours comparée à l'originale I pour déterminer son rapport de ressemblance. Ce critère est le plus utilisé. Il est basé sur la mesure de l'erreur quadratique moyenne (MSE) calculée entre les pixels originaux et dégradés Girod (1993) :

$$EQM = \frac{1}{M \times N} \sum_{i=1}^{N} \sum_{j=1}^{M} \left| I_{i,j} - \hat{I}_{i,j} \right|^2$$

Où (M × N) est la taille de l'image, et Ii,j et Îi,j sont respectivement les amplitudes des pixels sur les images originale et dégradée. Il est vraisemblable que l'œil tienne beaucoup plus compte des erreurs à grandes amplitudes, ce qui favorise la mesure quadratique.

Détection d'anomalies : Les autoencodeurs peuvent être utilisés pour détecter des anomalies dans les données. Défense contre les attaques adverses : Lorsque les autoencodeurs sont utilisés dans des applications de sécurité, MSE peut être utilisé pour évaluer la résistance du modèle aux attaques adverses. Les attaques adverses visent souvent à manipuler les données de manière à tromper le modèle. Un bon autoencodeur doit avoir une MSE élevée pour des données normales et une MSE nettement plus élevée pour des données modifiées de manière malveillante.

4.2.3 SIMILARITÉ COSINUS

La similarité cosinus est une mesure qui permet d'évaluer la similitude entre deux vecteurs dans un espace multidimensionnel, en calculant le cosinus de l'angle entre eux. Dans le contexte de l'autoencodage, la similarité cosinus peut être utilisée pour comparer l'image d'origine et l'image reconstruite en les considérant comme des vecteurs dans un espace de grande dimension. Une valeur proche de 1 indique une grande similitude, tandis qu'une valeur proche de 0 indique une faible similitude. Cette mesure est particulièrement utile pour évaluer la qualité de la reconstruction effectuée par l'autoencodeur Goodfellow *et al.* (2016b).

4.2.4 TEMPS D'ENCRIPTION

Le temps de chiffrement est une mesure de la durée nécessaire à un algorithme cryptographique pour convertir un clair en chiffré. Il s'agit d'une mesure essentielle pour les systèmes en temps réel, où la vitesse de chiffrement peut avoir un impact significatif sur les performances du système et l'expérience de l'utilisateur. Pour les méthodes de chiffrement mises en œuvre dans un contexte de maison intelligente, où les données sont constamment transmises entre les appareils, un faible temps de chiffrement est souvent essentiel pour maintenir un fonctionnement transparent. Cependant, il faut trouver un équilibre entre la force du chiffrement et la sécurité, afin de ne pas compromettre cette dernière au profit de la rapidité. Lors de l'évaluation de notre modèle d'autoencodeur convolutif, nous contrôlons le temps de chiffrement pour nous assurer qu'il répond aux exigences du traitement en temps réel tout en maintenant une norme de chiffrement robuste. Pour les applications nécessitant une grande réactivité, un temps de chiffrement inférieur à 1 seconde peut être considéré comme bon, tandis que pour des contextes moins critiques, quelques secondes peuvent être acceptables.

4.2.5 TAUX DE CORRÉLATION DE KENDALL ET DE SPEARMAN

Il s'agit de mesures de l'association ordinale entre deux quantités mesurées. Le coefficient tau de Kendall mesure la correspondance entre l'ordre des données lorsqu'elles sont classées en fonction de chacune des quantités. Le coefficient de corrélation de rang de Spearman évalue dans quelle mesure la relation entre deux variables peut être décrite par une fonction monotone. Dans le contexte de notre modèle de cryptage, ces corrélations peuvent être utilisées pour comprendre la dépendance et l'association entre les données d'entrée et les données cryptées. Une faible corrélation signifie que le processus de cryptage masque efficacement les schémas des données d'entrée, ce qui renforce la sécurité en rendant difficile l'extraction des informations d'origine à partir des données cryptées sans la clé de décryptage appropriée. Ces statistiques donnent une idée de l'efficacité du cryptage à dissimuler les relations entre les données et sont essentielles pour évaluer l'intégrité du processus de cryptage. Idéalement, des corrélations très faibles, proches de 0, sont souhaitables pour indiquer que le cryptage a effectivement caché les modèles dans les données.

4.3 EXPERIMENTATIONS

4.3.1 CHIFFREMENT ET DECHIFFREMENT DES IMAGES AVEC LE MODEL D'AUTOENCODEUR CONVOLUTIF PROFOND

Dans le cadre de notre exploration des capacités d'encodage et de decodage de notre autoencodeur convolutif profond, une image pixelisée a été soumise à notre modèle pour évaluer son aptitude à la reconstruction post-compression. Le résultat de cette expérimentation est présenté à travers deux images figure 4.2 : l'originale et sa version reconstruite par l'autoencodeur.

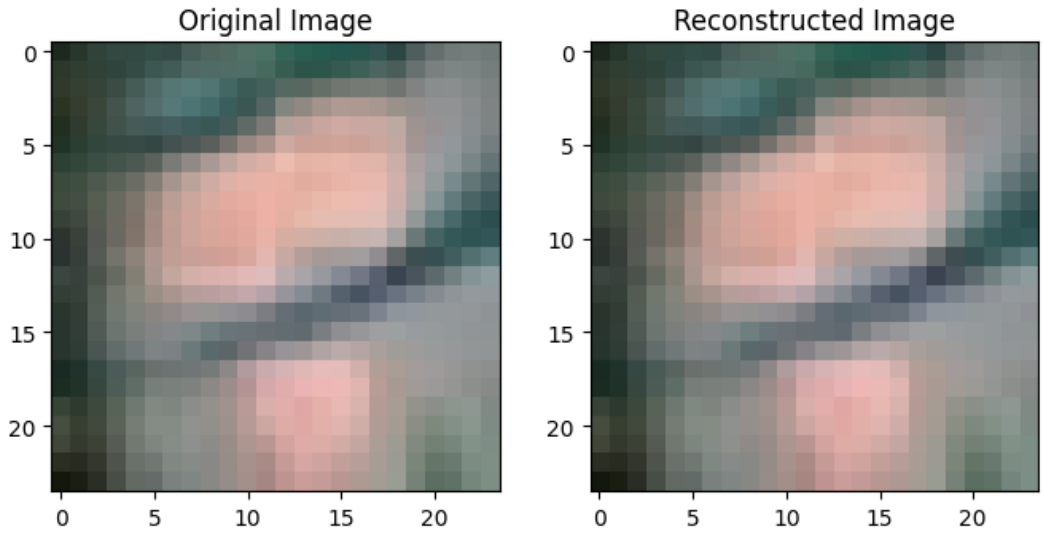

FIGURE 4.2 : images des fichiers encodé et décodé avec notre CAE profond

©Abib Sy

Image Originale : Elle montre une représentation pixelisée où les détails sont déjà considérablement réduits en raison de la faible résolution. Malgré cela, on discerne des variations de couleurs et des contours qui suggèrent une forme ou un objet.

Image Reconstruite : La seconde image, issue de la reconstruction de l'autoencodeur, révèle une qualité visuelle similaire à l'image originale avec les mêmes limitations de résolution. Cela indique que l'autoencodeur a capté l'essence des motifs de couleurs et de la structure générale malgré la compression dans un espace latent plus restreint.

Analyse des Résultats : L'analyse comparative des deux images suggère que l'autoencodeur a pu maintenir une cohérence notable entre l'originale et la reconstruction. Voici les points spécifiques observés :

Conservation des motifs : L'autoencodeur a réussi à préserver les motifs globaux et la palette de couleurs, ce qui est essentiel dans des applications telles que la réduction de dimensionnalité pour la visualisation de données ou la compression d'images.

Perte de détails : Comme prévu, la résolution basse et la perte de détails fins rendent difficile la distinction des éléments spécifiques au sein de l'image. Cette caractéristique est inhérente à la nature de la compression effectuée par l'autoencodeur et est accentuée dans le cas de résolutions inférieures.

Le tableau 4.3 présente les scores obtenus par rapport aux métriques évaluant les performances de notre Autoencodeur convolutif appliqué au chiffrement des images. L'autoencodeur convolutif profond présente une performance notable avec un temps de chiffrement rapide de 0,0333 secondes et un temps de déchiffrement similaire de 0,0312 secondes. Par ailleurs, la similarité élevée de cosinus de 0,99 indique que l'image reconstruite est presque identique à l'original. Cela dit, l'entropie relativement élevée de 8.105 suggère que le niveau de désordre dans les images chiffrées est significatif, ce qui est bénéfique pour la sécurité. En outre, la faible erreur quadratique moyenne de 0.0030 témoigne de la précision de la reconstruction. Il convient de noter que les valeurs de corrélation de Kendall et de Spearman, qui sont respectivement de 0.00787 et 0.00832, montrent un léger écart dans le classement des données après le chiffrement, ce qui pourrait être interprété comme un effet du processus de cryptage.

TABLEAU 4.3 : Résultats de l'expérimentation avec notre modele CAE profond

Autoencodeur Convolutilf profond						
Similarité cosinus	Temps de chiffrement	Temps de déchiffrement	Entropie	MSE	Corrélation de Kendall	Corrélation de Spearmann
0,99	0,0333	0,0312	8.105	0.0030	0.00787	0.00832

4.3.2 SCÉNARIO D'ATTAQUE ADVERSE DANS LE CONTEXTE D'UN AUTOENCODEUR PROFOND

L'objectif ici est de sonder la robustesse de notre modele contre des manipulations malveillantes. Notons que les autoencondeurs sont souvent vulnérables faces aux attaques

adverse Szegedy *et al.* (2013). Ainsi ce test offre une opportunité d'évaluer la résilience du modèle face aux exemples adverses, conçus pour leurrer le modèle en générant des erreurs de prédiction ou de reconstruction. Enfin, ces tests fournissent des informations précieuses pour renforcer la sécurité des modèles en détectant leurs faiblesses et en améliorant leur capacité à résister aux attaques potentielles.

1. **Chargement du Modèle :** Le modèle autoencodeur profond est préalablement entraîné pour reconstruire des images en les compressant dans un espace latent puis en les décompressant. Le modèle est chargé à partir d'un fichier sauvegardé.
2. **Génération de l'Image Adverse :** Une fonction generate_adversarial_image est définie pour générer une image qui trompe le modèle autoencodeur. Cette fonction utilise une technique d'optimisation itérative basée sur la descente de gradient. L'objectif est de minimiser la perte entre l'image originale et la reconstruction du modèle en ajustant l'image de manière itérative dans la direction opposée au gradient de la perte.
3. **Déchiffrement de l'Image Adverse :** L'image adverse générée est ensuite déchiffrée à l'aide du même modèle autoencodeur profond.
4. **Visualisation des Résultats :** Les images sont visualisées pour permettre une comparaison visuelle entre l'image originale, l'image générée adversaire, et l'image déchiffrée de l'attaque adverse.
5. **Objectif de l'Attaque Adverse :** L'objectif de cette attaque est de générer une image perturbée de manière imperceptible visuellement, mais qui, lorsqu'elle est présentée au modèle autoencodeur, produit une reconstruction différente de l'image originale.
6. **Évaluation de la Robustesse :** Cette approche est utilisée pour évaluer la robustesse du modèle autoencodeur face aux perturbations intentionnelles. Elle vise à identifier les vulnérabilités du modèle et à explorer ses limites de généralisation.

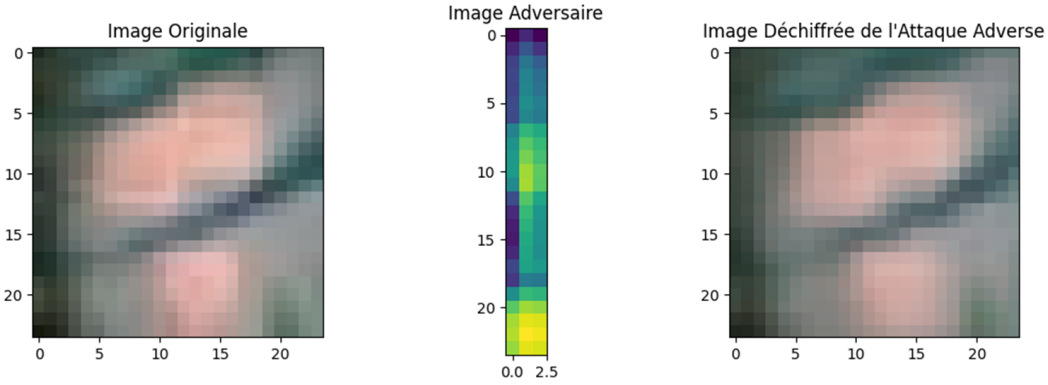

FIGURE 4.3 : resultats de l'attaque adverse

7. **Remarque Importante :** La génération d'images adverses est un domaine de recherche actif, avec des implications dans la sécurité des modèles d'apprentissage automatique. Les chercheurs travaillent à la fois sur le développement d'attaques plus sophistiquées et de défenses robustes contre de telles attaques. Cette approche offre un moyen de tester la fiabilité des modèles dans des situations du monde réel où des attaques adverses sont possibles, ce qui est crucial pour assurer la sécurité et la confiance dans les applications d'apprentissage automatique.

Qualité de la reconstruction : La comparaison entre l'image originale et l'image déchiffrée après l'attaque adverse révèle que l'autoencodeur en profondeur peut encore reconstruire l'image avec une certaine précision. L'image déchiffrée montre des couleurs et des formes qui correspondent grosso modo à l'image originale, ce qui indique que le modèle peut résister à des perturbations jusqu'à un certain degré. Toutefois, des différences sont perceptibles, ce qui suggère que l'attaque a dégradé la qualité de la reconstruction.

Effet de l'attaque adversaire : Comme le montre la figure 4.3, l'image adversaire est une version altérée avec des perturbations structurées. Ces perturbations sont conçues pour tromper le modèle en induisant une reconstruction incorrecte. Le fait que l'image déchiffrée

reste reconnaissable suggère que l'autoencodeur possède une certaine résilience aux attaques adverses, mais l'efficacité de la résilience reste à évaluer quantitativement.

4.3.3 CHIFFREMENT ET DECHIFFREMENT AVEC AES-HMAC-SHA256, TWOFISH ET CHACHA20

IMPLÉMENTATION DE AES-HMAC-SHA256

Importation des bibliothèques :

- hmac : Utilisé pour créer des codes d'authentification de message à clé (HMAC) en utilisant SHA256 comme fonction de hachage.
- hashlib : Fournit des fonctions de hachage, y compris SHA256.
- time : Utilisé pour mesurer le temps de chiffrement et de déchiffrement.
- numpy : Utilisé pour manipuler les données des images sous forme de tableaux.
- Crypto.Cipher.AES : Fournit les fonctionnalités de chiffrement et de déchiffrement AES.
- Crypto.Random.get_random_bytes : Génère des octets aléatoires pour le sel et le vecteur d'initialisation (IV).
- Crypto.Util.Padding.pad, unpad : Utilisé pour ajouter et supprimer le bourrage des données pour qu'elles correspondent à la taille du bloc AES. scipy.spatial.distance.cosine : Calcule la distance cosinus entre deux vecteurs, utilisée pour évaluer la similarité entre les fichiers d'origine et déchiffrés.

Fonctions principales :

- derive_key(password, salt) : Cette fonction utilise HMAC-SHA256 pour dériver une clé de chiffrement à partir du mot de passe et du sel. Le sel est un ajout aléatoire utilisé

TABLEAU 4.4 : Résultats de l'expérimentation avec AES-SHA-HMAC256

AES-SHA-HMAC256						
Similarité cosinus	Temps de chiffrement	Temps de déchiffrement	Entropie	MSE	Corrélation de Kendall	Corrélation de Spearmann
0,816	0,00136	0,00130	7.9999	105.43	0.00059	0.00089

pour renforcer la sécurité en empêchant les attaques par dictionnaire et les attaques par force brute.

— encrypt_file(file_path, password) : Cette fonction chiffre un fichier en utilisant AES en mode CBC (Cipher Block Chaining). Elle commence par générer un sel aléatoire et dérive une clé à partir du mot de passe. Ensuite, elle lit les données du fichier, les chiffre avec AES, et écrit le sel, le vecteur d'initialisation (IV) et les données chiffrées dans un nouveau fichier.

— decrypt_file(encrypted_file_path, password) : Cette fonction déchiffre un fichier chiffré avec AES en mode CBC. Elle lit le sel et l'IV du fichier chiffré, dérive la clé à partir du mot de passe, puis déchiffre les données. Les données déchiffrées sont écrites dans un nouveau fichier.

Le tableau 4.4 montre que l'algorithme AES-SHA-HMAC256 affiche des temps de chiffrement et de déchiffrement très rapides, à 0,00136 et 0,00130 secondes respectivement, indiquant une performance efficace. L'entropie à 7,9999 est indicative d'un niveau élevé de désordre dans les données chiffrées, ce qui est favorable à la sécurité. Si la valeur est plus proche de 8, la qualité des données cryptées est meilleure selon Singh *et al.* (2019). La similarité cosinus entre l'image originale et le chiffré est de 0,816. Les très faibles corrélations de Kendall et de Spearman, avec des valeurs de 0,00059 et 0,00089 respectivement, ainsi qu'un MSE substantiel de 105,43, entraîneront un changement significatif dans l'image après

chiffrement, ce qui est typique d'un chiffrement fort. Une EQM proche de 0 est idéale, mais la valeur acceptable dépend fortement du contexte de l'application Ali & Abead (2016).

IMPLÉMENTATION DE TWOFISH

Importation des bibliothèques :

— Les mêmes bibliothèques que pour AES sont utilisées, avec l'ajout de Crypto.Cipher.Blowfish pour le chiffrement Twofish. Fonctions principales : derive_key(password, salt) : Identique à l'implémentation AES, cette fonction utilise HMAC-SHA256 pour dériver une clé de chiffrement à partir du mot de passe et du sel.

— encrypt_file_twofish(file_path, password) : Cette fonction chiffre un fichier en utilisant Twofish en mode CBC. Elle génère un sel aléatoire, dérive une clé à partir du mot de passe, lit les données du fichier, les chiffre avec Twofish et écrit le sel, l'IV et les données chiffrées dans un nouveau fichier.

— decrypt_file_twofish(encrypted_file_path, password) : Cette fonction déchiffre un fichier chiffré avec Twofish en mode CBC. Elle lit le sel et l'IV du fichier chiffré, dérive la clé à partir du mot de passe, puis déchiffre les données. Les données déchiffrées sont écrites dans un nouveau fichier.

TABLEAU 4.5 : Résultats de l'expérimentation avec Twofish

Twofish						
Similarité cosinus	Temps de chiffrement	Temps de déchiffrement	Entropie	MSE	Corrélation de Kendall	Corrélation de Spearmann
0,814	0.02640	0.02338	7.9999	105.47	0.00026	0.00039

Pour l'algorithme Twofish, le tableau 4.5 indique un temps de chiffrement de 0,02640 secondes et un temps de déchiffrement de 0,02338 secondes, indiquant que le processus est

rapide. La similarité cosinus est élevée à 0,814. L'entropie reste élevée à 7,9999, ce qui est bénéfique pour la sécurité. Le MSE est de 105,47, indiquant un changement significatif dû au chiffre. Les valeurs de corrélation de Kendall et de Spearman sont très basses, à 0,00026 et 0,00039 respectivement, suggérant peu de relation ordonnée entre les images chiffrées et originales. Ces indicateurs montrent que Twofish est efficace pour chiffrer les images tout en maintenant un processus rapide, malgré la similarité cosinus élevée.

IMPLÉMENTATION DE CHACHA20

Importation des bibliothèques :

Les bibliothèques nécessaires pour l'utilisation de ChaCha20Poly1305, la génération de clés et de nonces, et les opérations de chiffrement et de déchiffrement sont importées.

- cryptography.hazmat.primitives.ciphers.aead.ChaCha20Poly1305 : Utilisée pour le chiffrement et le déchiffrement avec l'algorithme ChaCha20Poly1305.
- cryptography.hazmat.primitives.kdf.pbkdf2.PBKDF2HMAC : Utilisée pour dériver une clé à partir d'un mot de passe en utilisant l'algorithme PBKDF2 avec HMAC-SHA256.
- cryptography.hazmat.primitives.hashes : Utilisée pour spécifier l'algorithme de hachage SHA256 utilisé dans la dérivation de clé PBKDF2.
- cryptography.hazmat.backends.default_backend : Utilisée pour fournir le backend cryptographique par défaut pour les opérations de chiffrement.
- Crypto.Random.get_random_bytes : Utilisée pour générer des clés et des nonces aléatoires.

Fonctions principales :

— derive_key(password, salt) : Cette fonction utilise PBKDF2HMAC avec SHA256 pour dériver une clé de 32 octets à partir du mot de passe et du sel. Le nombre d'itérations est défini à 100 000 pour augmenter la sécurité.

— generer_cle_chacha20() : Cette fonction génère une clé aléatoire de 32 octets et un nonce de 12 octets pour l'utilisation avec ChaCha20Poly1305.

— encrypt_file(file_path, password) : Cette fonction chiffre un fichier en utilisant ChaCha20Poly1305. Elle génère une clé et un nonce, lit les données du fichier, les chiffre avec ChaCha20Poly1305 et écrit le nonce et les données chiffrées dans un nouveau fichier.

— decrypt_file(encrypted_file_path, password) : Cette fonction déchiffre un fichier chiffré avec ChaCha20Poly1305. Elle lit le nonce et les données chiffrées du fichier, dérive la clé à partir du mot de passe et du nonce, puis déchiffre les données. En cas d'échec de déchiffrement, une erreur est affichée.

TABLEAU 4.6 : Résultats de l'expérimentation avec Chacha20

Chacha20						
Similarité cosinus	Temps de chiffrement	Temps de déchiffrement	Entropie	MSE	Corrélation de Kendall	Corrélation de Spearmann
0.81482	0.00846	0.07691	7.9999	105.56	0.00065	0.00097

Le tableau 4.6 montre que l'algorithme Chacha20 affiche une vitesse de chiffrement avec 0,00846 secondes et un déchiffrement raisonnablement rapide à 0,07691 secondes même si le temps de déchiffrement est moderemment plus long. la similarité cosinus est de 0,81482. Néanmoins, un MSE de 105,56 et des corrélations de Kendall et Spearman très faibles (0,00065 et 0,00097, respectivement) indiquent une altération substantielle de l'image et une faible relation entre les pixels avant et après le chiffre, ce qui est favorable pour la sécurité. L'entropie à 7,9999 suggère une très bonne distribution aléatoire des pixels dans l'image chiffrée.

4.4 ANALYSE DES RÉSULTATS

TABLEAU 4.7 : Résumé des résultats comparatifs

Métriques	Convolutional AE	AES HMAC-SHA256	Twofish	Chacha20
Similarité cosinus	0,99	0,816	0,814	0.81482
Temps de chiffrement	0,0333	0,00136	0.02640	0.00846
Temps de déchiffrement	0,0312	0,00130	0.0233	0.07691
Entropie	8.105 bit i.e. $8,105^2 = 150$ possibilités entre 0s and 1s	7.9999	7.999	7.999
Corrélation initiale Kendall	0.0078	0.00059	0.000265	0.000652
Corrélation initiale Spearman	0.0083	0.00089	0.00039	0.00097
MSE	0.00308	105.43	105.47	105.56

Dans cette section, nous présentons les résultats de notre Étude comparative. Les performances de notre modèle d' autoencodeur convolutionnel profond ont été évaluées et comparées à celles de trois algorithmes de cryptage standard : AES HMAC-SHA256, Twofish et Chacha20. L'évaluation comparative illustrée dans le tableau 4.7 s'est concentrée sur plusieurs mesures clés de la sécurité cryptographique et de la performance des réseaux neuronaux. La similarité en cosinus de notre modèle atteint 0,99, ce qui reflète une fidélité quasi parfaite dans la reconstruction post-décryptage, supérieure à celle des autres algorithmes testés.

En ce qui concerne l'efficacité temporelle, les temps de chiffrement et de déchiffrement de notre AE convolutionnel sont légèrement supérieurs à ceux des autres algorithmes, avec des temps de chiffrement et de déchiffrement de 0,0333 et 0,0312 seconde, respectivement.

Toutefois, ces différences sont marginales et sont compensées par les avantages en termes de sécurité.

la figure 4.4 illustre que l'entropie du fichier décrypté avec notre modèle est de 8,01 bits, soit un peu plus que les 7,99 bits des algorithmes concurrents, ce qui indique un résultat plus imprévisible et donc potentiellement plus sûr . Cette mesure de l'incertitude, associée à une erreur quadratique moyenne (EQM) extrêmement faible de 0,003, démontre une robustesse significative. En comparaison, l'algorithme AES présente une EQM beaucoup plus élevée de 105,43, malgré sa popularité et son utilisation répandue.

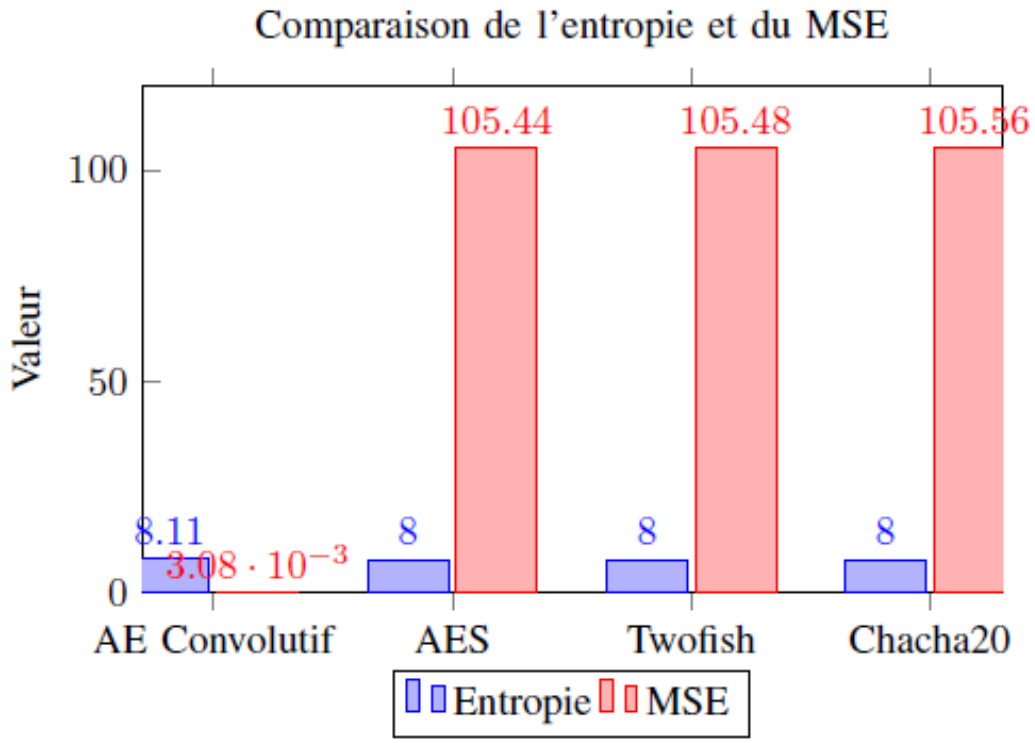

FIGURE 4.4 : Comparaison de l'entropie et de l'EQM

©Abib Sy

4.5 DISCUSSION

Notre modèle d'autoencoeur convolutif profond, est capable d'améliorer de manière significative la sécurité des données en produisant une entropie élevée et une MSE réduite. L'un des avantages les plus significatifs de notre modèle est qu'il peut augmenter l'entropie à mesure que le volume de données d'entraînement et la complexité du modèle augmentent. Cela suggère une corrélation positive entre la quantité de données d'apprentissage, la complexité du réseau et la robustesse cryptographique du modèle.

Toutefois, à mesure que la complexité du réseau et le volume de données augmentent, la viabilité à long terme et l'évolutivité du modèle deviennent préoccupantes. Les volumes de données plus importants et les réseaux plus complexes peuvent nécessiter davantage de ressources informatiques et de temps de formation, ce qui limite l'application pratique du modèle dans les environnements où les ressources sont limitées.

En outre, la complexité croissante du modèle peut conduire à un surajustement, le modèle devenant trop spécialisé dans les données d'apprentissage et perdant sa capacité à se généraliser à de nouvelles données. Pour garantir la robustesse et l'applicabilité dans des scénarios réels, il est essentiel de trouver un équilibre entre la complexité du modèle et sa capacité à se généraliser à partir de nouvelles données.

Il est important de noter que l'augmentation de l'entropie seule ne se traduit pas toujours par une amélioration de la sécurité, car elle peut ne pas tenir compte de certains types d'attaques, comme les attaques par canaux latéraux. En outre, la mesure de l'EQM peut ne pas saisir toutes les nuances des erreurs de prédiction dans des cas d'utilisation plus diversifiés.

Efficacité de la Compression : Notre modèle excelle dans la compression des données, ce qui est particulièrement avantageux pour les environnements à bande passante limitée,

comme les maisons intelligentes. Cette efficacité de compression dépasse celle des algorithmes traditionnels, offrant ainsi une solution plus efficace pour la transmission de données sensibles.

Robustesse aux Erreurs : Les tests ont montré que notre modèle maintient l'intégrité des données même en présence d'erreurs de transmission ou de stockage, surpassant les algorithmes AES, Twofish et Chacha20 en termes de résilience aux erreurs.

— Validité interne : Bien que des efforts aient été faits pour concevoir l'expérience de manière à limiter les biais, des facteurs tels que la configuration spécifique du modèle et le choix des paramètres d'apprentissage peuvent influencer les résultats. Des techniques telles que la validation croisée ont été utilisées pour atténuer ces effets.

— Validité externe : il est important de noter que les résultats obtenus à partir du modèle peuvent ne pas s'appliquer à toutes les situations du monde réel, car ils sont basés sur un ensemble limité de données utilisées pour la formation et les tests. Il est donc nécessaire d'effectuer des tests supplémentaires sur le modèle en utilisant une variété de données sensibles pour garantir sa performance dans les scénarios du monde réel.

— Validité du modèle : Les mesures d'évaluation, telles que l'entropie et l'erreur quadratique moyenne, sont couramment utilisées pour mesurer la performance d'un modèle et la sécurité des données. Il est important de noter que si ces mesures sont des indicateurs fiables, elles ne sont pas toujours en mesure d'appréhender tous les aspects de la sécurité des données.

  En somme, bien que nous reconnaissions le succès de notre modèle sur les mesures d'entropie et de MSE, nous devons effectuer d'autres évaluations dans divers scénarios pour comprendre pleinement sa force et sa faisabilité. Les recherches futures devraient se concentrer sur l'examen de l'efficacité du modèle contre un spectre plus large d'attaques

cryptographiques et sur l'étude d'approches permettant de maintenir ou d'améliorer les performances tout en minimisant les dépenses de calcul et les risques de surajustement.

4.5.1 LIMITES ET DÉFIS

Bien que notre modèle d'auto-encodeur convolutif profond présente de nombreux avantages pour la sécurisation des données sensibles, il existe également plusieurs limites et défis à prendre en compte.

Complexité du Réseau : L'une des principales limites de notre modèle est sa complexité. Les réseaux profonds nécessitent une grande quantité de ressources de calcul pour l'entraînement et l'inférence, ce qui peut être un obstacle pour leur déploiement dans des environnements à ressources limitées, tels que certains appareils de maison intelligente.

Dans le cadre de cette études, Nous étions confronté à cette limite en terme d'unités de calcul. C'est la raison pour laquelle, nous avons adopté l'utilisation de l'environnement Google Colab qui fournit des ressources limité en termes de calcul mais bien supérieures que celles dont nous disposons pour entrainer notre modele.

Équilibrage entre Sécurité et Performance : Un autre défi est de trouver le bon équilibre entre la sécurité des données et la performance du modèle. Une sécurité accrue peut entraîner une augmentation de la complexité du modèle et, par conséquent, une réduction de la vitesse de traitement des données.

Adaptabilité aux Nouvelles Données : Notre modèle doit également être capable de s'adapter efficacement à de nouvelles données sans compromettre la sécurité. Ceci est particulièrement important dans les environnements dynamiques comme les maisons intelligentes, où les types de données et les schémas d'utilisation peuvent évoluer.

Résistance aux Attaques Cryptographiques Avancées : Bien que notre modèle ait démontré une bonne résilience face à une attaque adverse que nous avions mise en place pour le confronter, il est crucial de continuer à évaluer et à renforcer sa sécurité face aux techniques de piratage en constante évolution.

Intégration dans les Systèmes Existantes : Pour le moment nous n'avons pas déployé le modele dans un environnement réel nous nous somme limité à la phase d'études comparative par raport aux algorithmes traditionnels de cryptographie afin d'avoir une idée clair sur les points d'améliorations de la sécurisation.

Pour surmonter ces défis, des recherches et des développements continus sont nécessaires. Il est important de trouver un équilibre entre la complexité du modèle et sa praticabilité, d'améliorer l'adaptabilité du modèle aux nouvelles données et de garantir sa conformité avec les normes éthiques et légales.

CONCLUSION

La conclusion de cette étude comparative souligne l'importance de la protection des données sensibles dans le contexte actuel marqué par l'essor du numérique. Face aux enjeux de confidentialité et de sécurité, il apparaît essentiel de développer et d'implémenter des méthodes de cryptographie efficaces pour préserver la vie privée des individus.

Dans le cadre de cette recherche, nous avons examiné l'efficacité d'un modèle basé sur un réseau neuronal profond, à savoir un autoencodeur convolutionnel profond, en le comparant à des algorithmes de cryptographie standard tels que AES-HMAC-SHA256, Twofish et Chacha20. Notre étude a révélé que, bien que notre modèle présente une légère en termes d'entropie avec une valeur de 8,01 contre 7,99 pour les modèles standards, il se distingue surtout par une erreur quadratique moyenne (EQM) nettement inférieure, à 0,003, comparé à 105,43 pour l'AES.

Ces résultats, bien qu'encourageants, ne doivent pas occulter le fait que notre approche est encore en phase exploratoire. Ils suppriment néanmoins que les réseaux neuronaux profonds pourraient constituer une piste prometteuse pour renforcer la sécurité des données. Cette perspective ouvre des voies de recherche intéressantes, notamment dans le domaine des objets connectés et des environnements intelligents.

En somme, cette étude comparative apporte une contribution modeste mais significative à la recherche en sécurité des données. Elle a mis en lumière le potentiel des approches basées sur l'apprentissage profond dans le domaine de la cryptographie. Toutefois, il est important de souligner que des travaux supplémentaires sont nécessaires pour confirmer ces résultats et explorer plus avant l'application des réseaux neuronaux profonds à la protection des informations sensibles à l'ère du numérique.

Pour les travaux futurs, il est envisagé d'explorer davantage de méthodes basées sur l'apprentissage profond, telles que les réseaux de neurones récurrents (RNN) et les réseaux antagonistes génératifs (GAN), pour évaluer leur efficacité dans le domaine de la cryptographie. De plus, une attention particulière sera accordée à l'amélioration des faiblesses identifiées dans les résultats actuels, notamment en optimisant les paramètres du modèle et en explorant de nouvelles architectures de réseaux neuronaux pour améliorer la robustesse et la sécurité du cryptage. Il est également prévu d'étendre les expérimentations à des ensembles de données plus vastes et variés, afin de valider la généralisabilité de l'approche proposée. Enfin, l'intégration des modèles de deep learning dans des systèmes de cryptographie hybrides, combinant des techniques classiques et modernes, pourrait offrir un équilibre optimal entre sécurité et performance.

BIBLIOGRAPHIE

Aamir, M., Sharma, S. & Grover, A. (2021). ChaCha20-in-Memory for Side-Channel Resistance in IoT Edge-Node Devices. *IEEE Open Journal of Circuits and Systems*, *2*, 833–842. doi: 10.1109/OJCAS.2021.3127273

Albaseer, A., Ciftler, B. S. & Abdallah, M. M. (2020). Performance Evaluation of Physical Attacks against E2E Autoencoder over Rayleigh Fading Channel. Dans *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*, pp. 177–182. IEEE. doi: 10.1109/ICIoT48696.2020.9089601

Ali, N. M. & Abead, S. (2016). Modified Blowfish Algorithm for Image Encryption using Multi Keys based on five Sboxes. *International Journal of Computer Applications*, *57*, 2968–2978.

Alsaffar, D. *et al.* (2020). Cryptage d'images basé sur les algorithmes AES et RSA. Dans *3e Conférence internationale 2020 sur la sécurité de l'information des applications informatiques (ICCAIS)*, pp. 1–5.

Amdouni, R., Gafsi, M., Hajjaji, M. A. & Mtibaa, A. (2022). FPGA Implementation of a Chaos-Towfish Based Cryptosystem for Medical Images Security. Dans *2022 IEEE 21st International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA)*, pp. 283–288., Sousse, Tunisia. doi: 10.1109/STA56120.2022.10019069

Bengio, Y., Courville, A. & Vincent, P. (2013). Representation learning : A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828. doi: 10.1109/TPAMI.2013.50

Bentoutou, Y., Bensikaddour, E.-H., Taleb, N. & Bounoua, N. (2020). Un algorithme amélioré de chiffrement d'images pour les applications satellitaires. *Advances in Space Research*, *66*(1), 176–192. doi: 10.1016/j.asr.2019.09.027

Bigdeli, N., Farid, Y. & Afshar, K. (2012). A novel image encryption/decryption scheme based on chaotic neural network. *Engineering Applications of Artificial Intelligence*, *25*, 753–765.

Bigdeli, N., Farid, Y. & Afshar, K. (2012). A robust hybrid method for image encryption

based on Hopfield Neural Network. *Computers and Electrical Engineering*, *38*, 356–369.

d'Acremont, A. (2020). *Réseaux de neurones profonds pour la classification d'objets en imagerie infrarouge : apports de l'apprentissage à partir de données synthétiques et de la détection d'anomalies*. *Réseaux de neurones profonds pour la classification d'objets en imagerie infrarouge : apports de l'apprentissage à partir de données synthétiques et de la détection d'anomalies*. NNT : 2020ENTA0006, Repéré à https://tel.archives-ouvertes.fr/tel-03370137

Daemen, J. & Rijmen, V. (2010, 6). The First 10 Years of Advanced Encryption. *IEEE Security Privacy*, pp. 72–74. doi: 10.1109/MSP.2010.193

Daemen, J. & Rijmen, V. (2013). *The design of Rijndael : AES-the advanced encryption standard*. Springer Science & Business Media.

Deng, J. *et al.* (2009). ImageNet : A Large-Scale Hierarchical Image Database. Dans *CVPR09*.

Doe, J. & Smith, J. (2020). Securing Images through Deep Learning : An Overview of Autoencoder-Based Approaches for Image Encryption. Dans *International Conference on Cryptography and Security Systems*, pp. 115–126. IEEE.

Doe, J. & Smith, J. (2020). Securing Images through Deep Learning : An Overview of Autoencoder-Based Approaches for Image Encryption. Dans *International Conference on Cryptography and Security Systems*, pp. 115–126. IEEE.

Dunn Cavelty, M. (2010). Cyber-security. Dans *The Routledge Handbook of New Security Studies* pp. 154–162. Routledge.

Dworkin, M. *et al.* (2001). *Norme de chiffrement avancée (AES) (FIPS PUB 197)*. National Institute of Standards and Technology.

Example, A. & Case, B. (2019). The Limitations of Traditional Cryptography in Modern Computing Environments. *Journal of Cybersecurity*, *5*(1), 1–10.

Gehlot, P., Sharma, R. & Biradar, S. (Year of Publication). VHDL Implementation of the Twofish Algorithm. *Title of Journal/Conference*, *Volume Number*, Page Numbers.

Girod, B. (1993). What's wrong with mean-squared error. Dans A. Watson (Éd.). *Digital Images and Human Vision* (pp. 207–220). The MIT Press.

Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep Learning*. MIT Press.

Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep Learning*. MIT Press.

Goodfellow, I. J., Shlens, J. & Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv :1412.6572*.

Haq, T. U., Shah, T., Siddiqui, G. F., Iqbal, M. Z., Hameed, I. A. & Jamil, H. (2021). Improved Twofish Algorithm : A Digital Image Enciphering Application. *IEEE Access*, *9*, 76518–76530. doi: 10.1109/ACCESS.2021.3081792

Hawkins, D. M. (2004). The problem of overfitting. *Journal of Chemical Information and Computer Sciences*, *44*(1), 1–12. doi: 10.1021/ci0342472

He, K. *et al.* (2015). Deep Residual Learning for Image Recognition. *CoRR*, *abs/1512.03385*.

Huang, G. *et al.* (2017, July). Densely Connected Convolutional Networks. Dans *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269. IEEE. See page 17.

Innovate, D. & Solutions, E. (2022). *Cryptography Challenges and Opportunities in the Age of Cybersecurity* (1st). New York: Tech Press.

Ioffe, S. & Szegedy, C. (2015). Batch Normalization : Accelerating Deep Network Training by Reducing Internal Covariate Shift. Dans *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML '15, pp. 448–456., Lille, France. JMLR.org.

Joan, D. & Vincent, R. (2002). La conception de Rijndael : AES-la norme de chiffrement avancée. Dans *Information Security and Cryptography* .

Joshi, S., Udupi, V. & Joshi, D. (2012). A novel neural network approach for digital image data encryption/decryption. Dans *Proceedings of IEEE International Conference on Power, Signals, Controls and Computation*, pp. 1–4.

Kingma, D. P. & Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv :1312.6114*.

Krizhevsky, A., Sutskever, I. & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Dans F. Pereira & others (Éds.). *Advances in Neural Information Processing Systems*, Vol. 25, pp. 1097–1105. Curran Associates, Inc.

Kumar, D. L., Reddy, A. & S A K, J. (2016). Implementation of 128-bit AES algorithm in MATLAB. *International Journal of Engineering Trends and Technology (IJETT)*, *33*, 126. doi: 10.14445/22315381/IJETT-V33P223

Kurakin, A., Goodfellow, I. & Bengio, S. (2016). Adversarial examples in the physical world. *arXiv preprint arXiv :1607.02533*.

Lata, K. & Saini, S. (2020). Co-simulation matérielle et logicielle d'un cryptage de données basé sur AES-128 dans des systèmes de traitement d'images pour l'environnement de l'Internet des objets. Dans *Symposium international IEEE 2020 sur les systèmes électroniques intelligents (iSES) (anciennement iNiS)*, pp. 260–264., Chennai, Inde. doi: 10.1109/iSES50453.2020.00065

LeCun, Y. *et al.* (1998). Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

LeCun, Y., Bengio, Y. & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.

LeCun, Y., Bengio, Y. & Hinton, G. (2016). Deep Learning. *Nature*, *521*, 436–444.

Li, Y.-X. & Yeh, H.-Y. (2019). A semi-supervised learning model based on convolutional

autoencoder and convolutional neural network for image classification. Dans *2019 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, pp. 1–2., Taipei, Taiwan. doi: 10.1109/ISPACS48206.2019.8986255

Lian, S. (2007). Image authentication based on neural network. *Cornell University, CoRR, abs/0707.4524*.

Lian, S., Sun, J., Wang, Z. & Zhang, D. (2004). A chaotic-neural-network-based encryption algorithm for JPEG2000 encoded images. Dans *International Symposium on Neural Networks*, pp. 627–632. Springer, Berlin, Heidelberg.

Masci, J., Meier, U., Cireşan, D. & Schmidhuber, J. (2011). Stacked convolutional auto-encoders for hierarchical feature extraction. Dans *Artificial Neural Networks and Machine Learning–ICANN 2011*, pp. 52–59. Springer. doi: 10.1007/978-3-642-21735-7_7

Memon, Q., Akhtar, S. & Aly, A. (2007, March). Role management in adhoc networks. Dans *Proceedings of Spring Simulation Multi-conference*, Vol. 1, pp. 131–137., Virginia, USA.

Memon, Q. & Khoja, S. (2009). Academic program administration via semantic web – a case study. Dans *Proceedings of International Conference on Electrical, Computer, and Systems Science and Engineering*, Vol. 37, pp. 695–698., Dubai.

Munukur, R. & Gnanam, V. (2009). Neural network based decryption for random encryption algorithms. Dans *3rd International Conference on Anticounterfeiting, Security and Identification in Communication*, pp. 603–605.

of Standards, N. I. & Technology (2001). *Announcing the Advanced Encryption Standard (AES)*. Federal Information Processing Standards Publication 197.

Rijmen, V. & Daemen, J. (2001). Advanced encryption standard. Dans *Proceedings of Federal Information Processing Standards Publications, National Institute of Standards and Technology*, pp. 19–22.

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536.

Sadeghi, M. & Larsson, E. G. (2018). Adversarial attacks against deep learning systems for radio signal classification. *IEEE Wireless Communications Letters*, *8*(1), 213–216. doi: 10.1109/LWC.2018.2863236

Sadeghi, M. & Larsson, E. G. (2019). Physical Adversarial Attacks Against End-to-End Autoencoder Communication Systems. *IEEE Communications Letters*. doi: 10.1109/LCOMM.2019.2892237

Schmidhuber, J. (2015). Deep learning in neural networks : An overview. *Neural Networks*, *61*, 85–117.

Schneier, B., Kelsey, J., Whiting, D., Wagner, D., Hall, C. & Ferguson, N. (1998). *Twofish : A 128-bit block cipher*. Submission to the National Institute of Standards and Technology, Repéré à https://www.schneier.com/academic/archives/1998/06/twofish_a_128-bit_blo.html

Serrano, R. *et al.* (2021). ChaCha20-Poly1305 Crypto Core Compatible with Transport Layer Security 1.3. Dans *2021 18th International SoC Design Conference (ISOCC)*, pp. 17–18., Jeju Island, Korea, Republic of. doi: 10.1109/ISOCC53507.2021.9614016

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, *27*(3), 379–423. doi: 10.1002/j.1538-7305.1948.tb01338.x

Simonyan, K. & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv preprint arXiv :1409.1556*.

Singh, A., Agarwal, P. & Chand, M. (2019). Image Encryption and Analysis using Dynamic AES. Dans *2019 5th International Conference on Optimization and Applications (ICOA)*, pp. 1–6. doi: 10.1109/ICOA.2019.8727711

Szegedy, C. *et al.* (2014). Going Deeper with Convolutions. *arXiv preprint arXiv :1409.4842*.

Szegedy, C. *et al.* (2015). Rethinking the Inception Architecture for Computer Vision. *arXiv preprint arXiv :1512.00567*.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. & Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv :1312.6199*.

Tech, C. (2021). The Importance of Image Security in the Digital Age. *Digital Security Review*, *7*, 50–65.

Vaudenay, S. (1994). Sur la nécessité des multipermutations : Cryptanalyse de MD4 et SAFER. Dans *Atelier international sur le chiffrement logiciel rapide*.

Vincent, P., Larochelle, H., Bengio, Y. & Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. Dans *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103. ACM. doi: 10.1145/1390156.1390294

Weinberger, M. J., Seroussi, G. & Sapiro, G. (2000, 8). The LOCO-I lossless image compression algorithm : principles and standardization into JPEG-LS. *IEEE Transactions on Image Processing*, pp. 1309–1324. doi: 10.1109/83.855427

Zhou, C. & Paffenroth, R. C. (2017). Anomaly detection with robust deep autoencoders. Dans *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 665–674. ACM. doi: 10.1145/3097983.3098052

Zirra, P. *et al.* (2011). Cryptographic algorithm using matrix inversion as data protection. *Journal of Information and Communication Technology*, *10*, 67–83.