Comment optimiser la détection des attaques Botnet à l'aide des approches en autoencodeur et CNN

Mémoire présenté par

Elodie WAN

Pour l'obtention du Master 1 MIAGE

De l'université

Paris 1 Panthéon - Sorbonne

Année Universitaire : **2022-2023**

Date de soutenance : **04 juillet 2023**

Directeur de mémoire : **Irina RYCHKOVA**

Membre du jury : **Nicolas HERBAUT**

Table des matières

Remerciements 2
Résumé 3
Abstract 3
Introduction 4
Background 6
1 – Définition 6
2 – Related works 16
Méthodologie de recherche 17
1 – Questions de recherche 17
2 – Sélection des articles 18
3 – Extraction de données 20
Autoencodeur 21
1 – Définition 21
2 – Modèles étudiés 24
Convolutional Neural Network (CNN) 30
1 – Définition 30
2 – Approches combinées 32
Limites des recherches 35
Conclusion 37
Glossaire 38
Table des Figures 40
Références 41

Remerciements

Je tiens tout d'abord à remercier madame Irina RYCHKOVA pour son soutien durant cette année et les précieux conseils qu'elle a pu me donner tout au long de la conception de ce mémoire d'état de l'art. J'exprime également ma gratitude envers tous les enseignants du master 1 MIAGE à l'université Paris 1 Panthéon-Sorbonne pour le professionnalisme et les connaissances académiques qu'ils m'ont apportées durant cette année.

Je souhaite également remercier la société Cloud Girafe de m'avoir offert un environnement propice au développement de mes compétences et à l'apprentissage.

Enfin, je tiens à exprimer ma gratitude envers toutes les personnes que j'ai rencontrées durant cette année pour leur sympathie et bienveillance à mon égard.

Résumé

L'Internet des Objets (en anglais Internet of Things ou IoT) est une technologie prometteuse présente dans de nombreux domaines aux usages variés tels que la e-santé ou la domotique. Cependant, leurs sécurités encore faibles posent des problèmes donc les risques de cybersécurité sont augmentés. L'objectif de cet état de l'art est de présenter différents moyens de détection de ces attaques. Pour cela, nous allons mettre en avant deux approches du Deep Learning : l'autoencodeur et le Convolutional Neural Network (CNN).

Après avoir présenté les concepts clés liés aux Botnets et à la détection d'intrusions, ce mémoire se concentrera sur les techniques de traitement de données qui permettent d'améliorer l'efficacité de la détection avec autoencodeur ainsi qu'une approche combinée avec CNN.

Abstract

The Internet of Things (IoT) is a promising technology that is present in many different fields such as e-health or home automation. However, their weak security poses a problem, which increases the risks of cybersecurity threats. The objective of this state of the art is to present different means of detecting these attacks. To achieve this, we will highlight two approaches of Deep Learning: autoencoders and Convolutional Neural Network (CNN). After presenting the key concepts related to Botnets and intrusion detection, this thesis will focus on data processing techniques that improve the effectiveness of autoencoders as well as a combined approach with CNN.

Introduction

J'ai choisi le sujet d'optimisation de la détection des attaques Botnet à l'aide des approches en autoencodeur et CNN pour mon mémoire en souhaitant explorer le domaine de la cybersécurité. Je suis partie au départ de la détection des attaques Botnet à l'aide du Deep Learning, mais après lecture de quelques articles secondaires qui m'ont donné plus d'informations sur les différentes techniques [3], j'ai décidé de me focaliser sur 2 approches : l'autoencodeur et le CNN.

Avec le progrès technologique, les moyens de communications ont fortement évolué et sont présents partout de nos jours. Ces outils sont utilisés dans de nombreux domaines : bancaires, marketing ou juste l'utilisation des courriels dans la vie quotidienne. L'implémentation des appareils Internet of Things[1] (IoT) dans le marché a grandi exponentiellement [1].

Les IoT désignent le processus de connexion d'objets physiques à Internet, des objets du quotidien tels que les ampoules, aux dispositifs médicaux, appareils portables, appareils intelligents ou encore feux de circulation routière dans les villes intelligentes. Cependant, cela entraîne malheureusement une augmentation de risque en matière de sécurité. En 2016, des millions d'appareils ont été attaqués par le botnet[2] Mirai[3][1].

Les IoT présentent généralement des contraintes en termes de ressources et d'environnements, tels que des capacités de calcul et de mémoire limitées. Par conséquent, ces contraintes posent des problèmes lors de la mise en œuvre d'une solution de sécurité sur les dispositifs IoT. Dû à ces vulnérabilités, divers types d'attaques sont possibles, l'attaque par botnet étant l'une des plus populaires.

[1] Internet of Things (IoT) : Désigne un nombre croissant d'objets connectés à Internet permettant ainsi une communication entre nos biens dit physiques et leurs existences numériques.
[2] Botnet (attaque) : Combinaison des mots « robot » et « réseau » (*network* en anglais). Désigne un groupe d'ordinateurs ou de dispositifs sous le contrôle d'un attaquant utilisé pour mener des activités malveillantes.
[3] Mirai (attaques botnet) : Logiciel malveillant capable de transformer des ordinateurs fonctionnant sous Linux en bots contrôlés à distance.

Pour résoudre ces problèmes de sécurité dans le réseau, des systèmes de détection d'intrusion[4] (Intrusion Detection System ou IDS) sont mis en place [2, 14]. Dans le domaine de la cybersécurité, l'IDS est une mesure de sécurité impérative permettant de reconnaître et parer les attaques malveillantes.

Il existe de nombreux algorithmes qui peuvent être utilisés pour mettre en place un IDS, dans ce mémoire, on se concentrera sur deux approches en Deep Learning : autoencodeur et CNN. Même en se limitant à ces deux sujets, la quantité de recherches scientifiques reste important. Elles sont en général composées de trois parties : présentation de l'algorithme, phase d'apprentissage avec des jeux de données tels que MedBIoT ou N-BaIoT puis une démonstration avec des résultats atteignant une détection proche de 100% [2].

Cette quantité importante d'algorithme de détection d'anomalies présente des différences selon leurs approches. Cependant, pourquoi une seule solution n'est pas adoptée pour prévenir tous ces problèmes ? Pour chaque algorithme, la durée de détection, la vitesse de traitement, d'apprentissage peuvent varier énormément. Cela nous amène à la problématique suivante : quels sont les avantages des différentes approches en autoencodeur et CNN afin de détecter des attaques botnet et leurs optimisations.

Dans une première partie, nous allons nous concentrer sur le concept des autoencodeurs suivi de celles des CNN puis des approches combinées. Dans une dernière partie, nous allons discuter des limites des recherches actuelles et donc pourquoi il n'y a pas de solutions universelles pour la détection des attaques botnet.

[4] Intrusion Detection System (IDS) : Mécanisme destiné à repérer des activités anormales ou suspectes sur un réseau ou un hôte

Background

Cette partie va définir les termes et concepts de base pour comprendre les attaques botnet sur les IoT puis discuter de certaines revues systématiques. Une définition plus détaillée des autoencodeurs et CNN seront introduites dans les prochaines sections.

1 – Définition

Un botnet d'après les recherches [17, 18] est un réseau d'ordinateurs ou d'appareils connectés à Internet qui sont infectés par des logiciels malveillants et contrôlés à distance par des personnes malintentionnées. Les appareils infectés, appelés « bot » ou « zombies » exécutent des tâches automatisées sans le consentement ni la connaissance de leurs propriétaires légitimes. Les botnets sont souvent utilisés pour mener des attaques coordonnées. Les attaques les plus courantes d'après une enquête de [19] sont les suivantes :

- Attaques par déni de services (Distributed Denial-of-Service ou DDoS) : Attaque ayant pour but de rendre indisponible un service en saturant le serveur par l'envoi de multiples requêtes ou provoquer une panne/dégradation du service par l'exploitation d'une faille de sécurité à l'aide d'un grand nombre de bot.
- Spam : Les botnets peuvent être utilisés pour envoyer massivement des courriels indésirables à grande échelle en utilisant les ordinateurs infectés comme relais pour distribuer ces courriels.
- Vol d'informations : Certains botnets sont conçus pour voler des informations sensibles, telles que des données personnelles, des informations de connexion ou des informations financières. Les bots peuvent être utilisés pour envoyer ces informations vers les serveurs des attaquants.
- Attaque brute-force et piratage de comptes : Attaque qui essaye de deviner le mot de passe de comptes en essayant différentes combinaisons jusqu'à trouver le bon.
- Mining de cryptomonnaie : Les bots sont utilisés pour faire des calculs complexes pour la génération de cryptomonnaie et ainsi générer des profits pour les attaquants.

Les botnets peuvent être très puissants et difficiles à éradiquer, car ils exploitent la force combinée des nombreux machines infectées pour leurs opérations malveillantes [3].

Pour les détecter nous allons utiliser des IDS [14] qui sont eux souvent fait à l'aide du Machine Learning et du Deep Learning qui sont des sous-ensembles de l'intelligence artificielle (IA).

L'IA peut être définie d'après les publications du CNIL [21], comme la théorie et le développement de systèmes informatiques qui visent à simuler l'intelligence « humaine », tels que le raisonnement, la planification et la créativité. L'IA est présente dans notre quotidien. Par exemple, elle est utilisée par en tant que Service Client Virtuel (SCV) pour répondre aux demandes courant des clients sans intervention humaine. UPS implémente ça et peut répondre à des questions d'utilisateur comme le suivi des colis avec une reconnaissance vocale et un simulateur de dialogue humain.

L'apprentissage automatique (Machine Learning ou ML)[5] [20, 21] est un sous-ensemble de l'IA et utilise des techniques statistiques pour permettre aux systèmes informatiques d'apprendre et de prendre des décisions ou de faire des prédictions sans être explicitement programmés. Les algorithmes sont entraînés à trouver des patterns et des corrélations dans de grands ensembles de données, ainsi qu'à prendre les meilleures décisions et à émettre les meilleures prévisions en s'appuyant sur leur analyse. Plus le volume de données auxquelles elles ont accès est important, plus elles deviennent précises.

Le machine learning est utilisé dans de nombreux domaines et industries pour résoudre une variété de problèmes. On peut prendre en exemple dans le marketing et la publicité : la recommandation de produits, la personnalisation des offres, etc…

Il existe différents types d'approches, notamment l'apprentissage supervisé, l'apprentissage non supervisé et d'autres. Dans le machine learning, il y a l'apprentissage profond qui est lui est basé sur les réseaux de neurones artificiels.

[5] Machine Learning : Sous-ensemble d'intelligence artificiel permettant aux ordinateurs d'apprendre sans avoir été programmés explicitement

Un réseau de neurones artificiels (Artificial Neural Networks ou ANN)[6] est un modèle mathématique et informatique qui tente de simuler le fonctionnement des réseaux de neurones biologiques présents dans le cerveau humain [5]. Un ANN est généralement constitué d'une succession de couches en entrée, les valeurs de sortie de la couche précédente (voir Figure 1).

La première couche est constituée de neurones d'entrée. Ces neurones envoient des données aux couches plus profondes, qui à leur tour enverront les données de sortie finales à la dernière couche de sortie pour obtenir une classification ou une prédiction. Toutes les couches internes sont cachées et sont formées par des unités qui modifient de manière adaptative les informations reçues d'une couche à l'autre par une série de matrice de transformations [22].

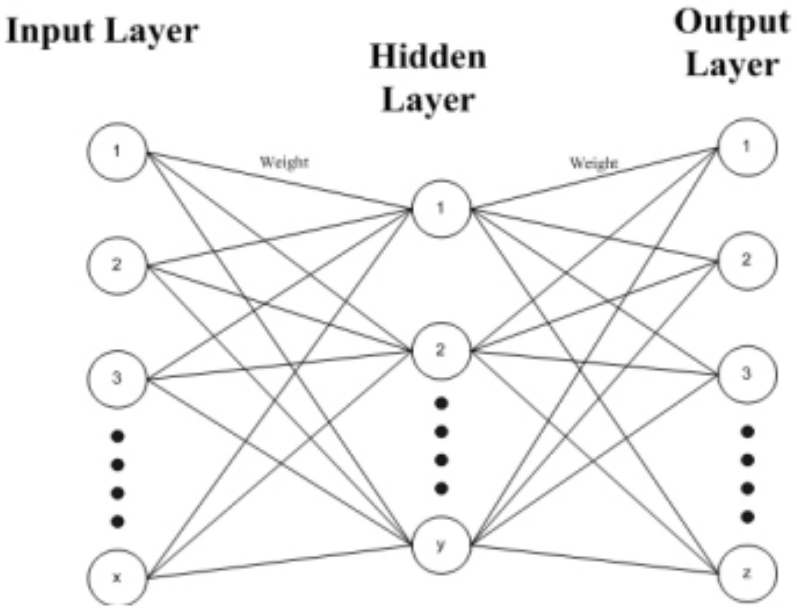

Figure 1 : Représentation d'un réseau de neurones artificiels [9]

Les ANN ont des paramètres ajustables comme les poids et les biais. Leurs ajustements vont permettre d'optimiser le réseau pour qu'il reconstruit mieux les données d'entrée. L'ajustement se fera lors de la phase d'apprentissage à l'aide d'un algorithme d'optimisation nommé rétropropagation du gradient.

[6] Réseau de neurones artificiels (Artificial Neural Network ou ANN) : Ensemble de neurones interconnectés.

L'apprentissage supervisé (Supervised Learning)[7] comme défini dans [20, 24] se composent de paires de données « d'entrée » et « de sortie » et est entrainé à partir de données étiquetées, dans lesquelles la sortie est étiquetée avec la valeur souhaitée pour qu'il apprenne à généraliser et faire des prédictions sur de nouvelles données. L'apprentissage supervisé est utilisé dans divers cas comme la classification, la détection d'anomalies, …

L'apprentissage non supervisé (Unsupervised Learning)[8] tel que décrit dans [20, 23], ne possède pas d'indice sur les données de sortie : elle est entraînée à partir de données non étiquetées. Dans ce type d'apprentissage, le modèle cherche à trouver des structures, patterns pour extraire les informations utiles en autonomie. Cette méthode d'apprentissage ressemble fortement à la réflexion humaine. L'apprentissage non supervisé est utilisé pour la réduction de dimensionnalité, la détection d'anomalies, …

En accord avec les définitions dans [20, 25], l'apprentissage profond (Deep Learning ou DL)[9] est basé sur de nombreuses couches des réseaux de neurones artificiels pour apprendre et effectuer des tâches complexes. C'est une méthode d'apprentissage automatique qui cherche à modéliser et simuler le fonctionnement du cerveau humain. La phase d'apprentissage demande un nombre très important de données. Le deep learning est couramment utilisé dans des comme les systèmes de recommandation, la bio-informatique, …

[7] Apprentissage supervisé (Supervised Learning) : Données d'entrée sont étiquetés et les algorithmes apprennent à prédire le résultat des données d'entrée
[8] Apprentissage non supervisé (Unsupervised Learning) : Données d'entrée non étiquetés et les algorithmes apprennent la structure inhérente à partir des données d'entrée
[9] Deep Learning : Procédé d'apprentissage automatique utilisant des réseaux de neurones possédant plusieurs couches de neurones cachées

Il existe deux grandes catégories d'IDS : la détection par signature et détection par anomalies [4].

La première méthode, les systèmes de détection d'intrusion par signature[10] (Signature-based Intrusion Detection System ou SIDS) se reposent sur les signatures (descriptions) des attaques. Lors de l'analyse du flux réseau, l'IDS analysera chaque événement et alerte lorsqu'une signature sera détectée.

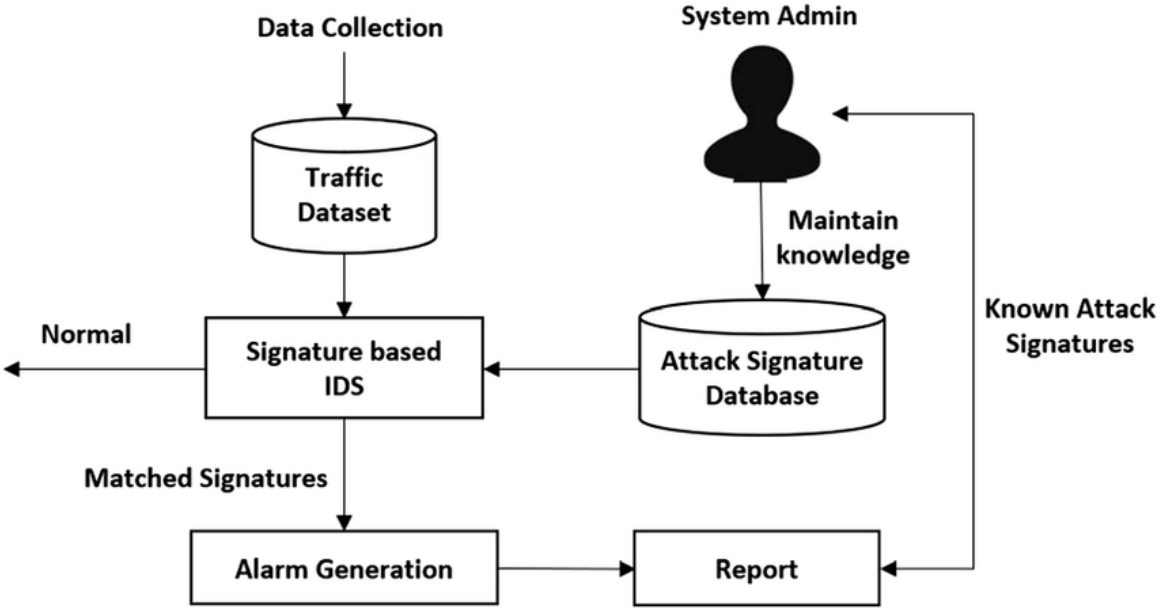

Figure 2 : Système de détection d'intrusion par signature [8]

Comme observé dans la figure 2, lorsque le trafic rentre dans le système, une alarme sera générée si elle connait déjà l'attaque, un rapport sera créé et l'attaque sera aussi sauvegardée dans la liste des attaques connues. Cependant, cette approche n'est efficace que si la base de signatures et maintenue à jour de manière régulière et ne fonctionne donc qu'avec des attaques connues. L'IDS ne sera pas déclenché dans le cas où l'attaque est inconnue. Il en existe plusieurs open-source tel que Snort, Suricata, …

[10] Signature-based Intrusion Detection System (SIDS) : Détection en faisant un matching du flux et de la signature (modèle qui correspond à une menace spécifique étudiée)

La deuxième méthode, les systèmes de détection d'intrusion par anomalies[11] (Anomaly-based Intrusion Detection System ou AIDS) passent par une phase d'apprentissage, où le système va étudier des comportements normaux de flux de réseau et créer un modèle statistique de ce dernier.

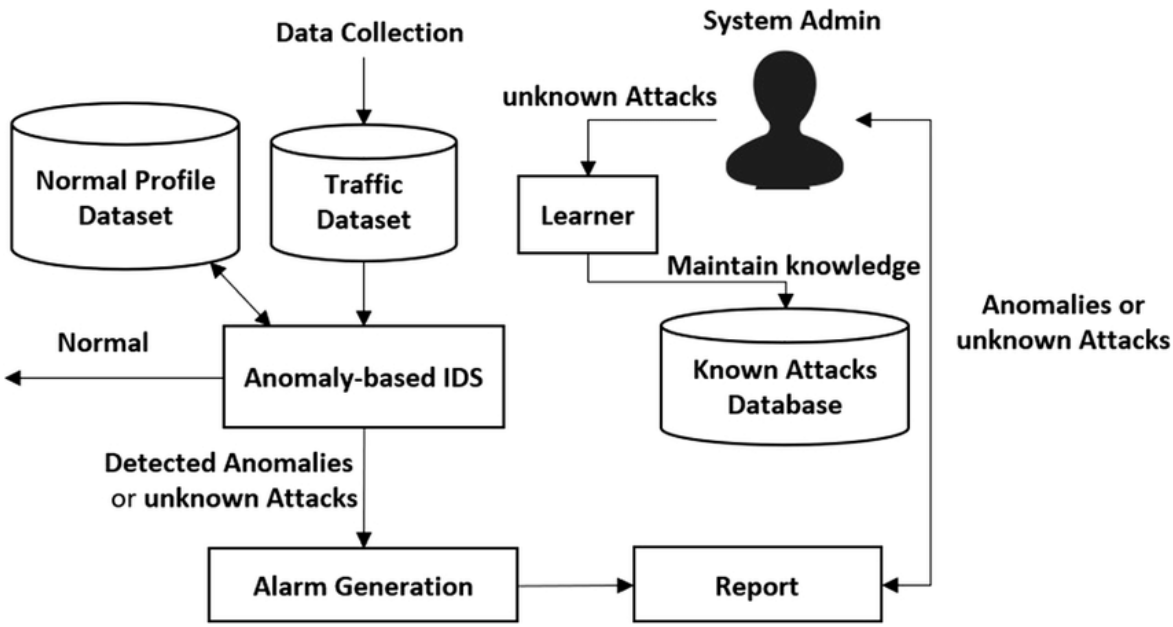

Figure 3 : Système de détection d'intrusion par anomalie [8]

Selon l'exemple donnée dans la figure 3, les données d'entrée sont comparées au trafic normal, s'il y'a une anomalie, une alarme sera générée. Dans le cas où l'attaque est inconnue, le modèle va apprendre les caractéristiques de l'attaque pour pouvoir y remédier plus rapidement la prochaine fois.

[11] Anomaly based Intrusion Detection System (AIDS) : Détection d'anomalie sur le flux réseau en se basant sur l'apprentissage des comportements normaux du flux réseau

Grâce à cela lors de l'analyse du flux réseau, si le flux dérive du modèle, il pourra être classifié comme une anomalie et être reconnu comme une intrusion. Cette méthode est reconnue par la communauté comme étant très efficace et de plus en plus de recherches scientifiques sont publiés dessus depuis 2019 [4]. Elle repose généralement sur l'apprentissage autonome des intelligences artificielles, voici quelques exemples :

- Machine Learning :
  - Forêt d'arbres décisionnels (Random Forest ou RF)
  - Réseau de neurones artificiels, la méthode des k plus proches voisins (k-nearest neighbors ou K-NN)
  - Machine à vecteurs de support (Support Vector Machine ou SVM)
- Deep Learning :
  - Réseau de neurones convolutifs (Convolutional Neural Networks ou CNN)
  - Autoencoder
  - Cellule de longue mémoire à court terme (Long Short-Term Memory ou LSTM)

Un autoencodeur est un réseau de neurones artificiels. L'autoencodeur est généralement utilisé pour l'apprentissage non supervisé de caractéristiques discriminantes[12]. L'apprentissage va donc se faire en regroupant en autonomie des informations non triées en fonction des similitudes et différences qu'elles ont entre elles [4, 10].

L'objectif d'un autoencodeur est d'apprendre une représentation (encodage) d'un ensemble de données et une reconstruction de ces dernières (décodage) pour donner une représentation la plus similaire à la donnée de base. Cela va permettre une réduction de la dimension[13] de cet ensemble [6].

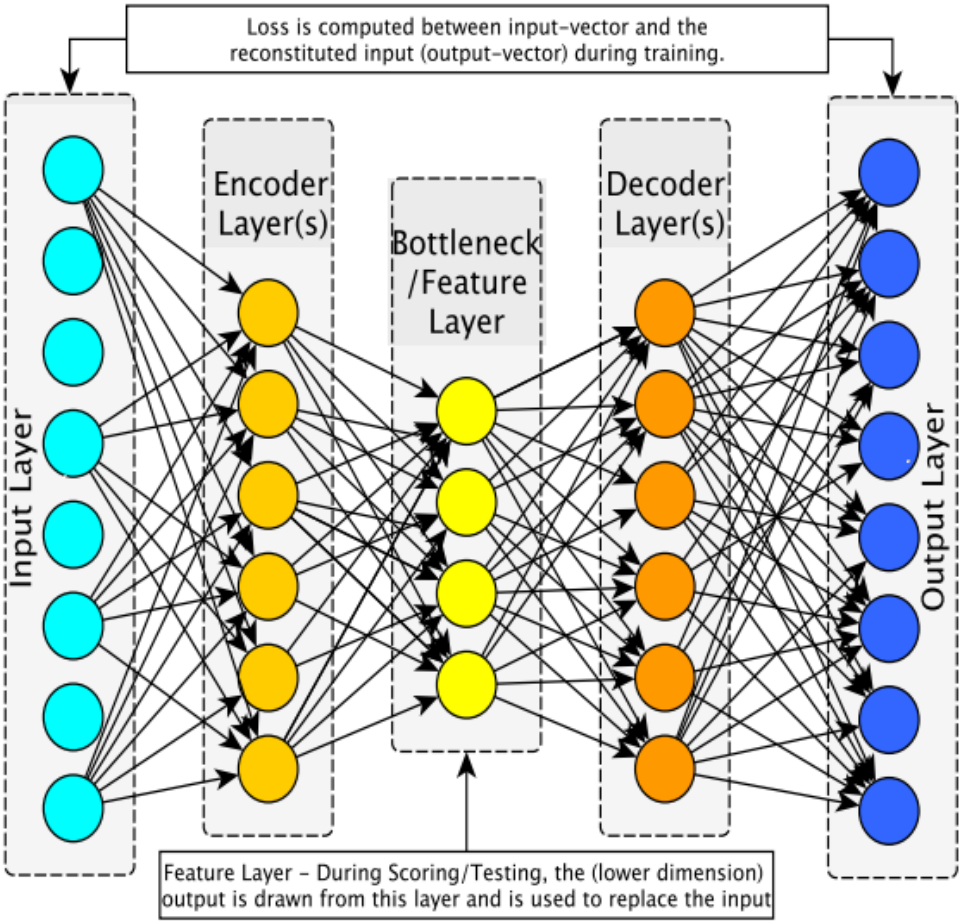

Figure 4 : Représentation des couches d'un autoencoder [6]

[12] Caractéristiques (features) : Extraction de caractéristiques. Les caractéristiques sont généralement numériques mais peut être des chaînes de caractères, de graphes ou d'autres quantités encore.
[13] Réduction de la dimensionnalité (Dimensionality Reduction) : Réduction du nombre de variable ou caractéristiques permettant d'entrainer le modèle d'IA

Elle est généralement constituée d'une couche d'entrée, d'une ou plusieurs couches d'encodage, une couche latente (couche bottleneck) qui représente une version compressée des données d'entrée, une ou plusieurs couches de décodage et une dernière de sortie (voir Figure 4).

On peut prendre en exemple d'attaque, un bot de botnet fait une attaque de type commande et contrôle[14] (Command and Control ou C2 ou C&C) sur un système IoT de gestion de température et d'humidité dans une serre. Le bot va recevoir des instructions de son opérateur à distance et les exécuter.

Il pourra par exemple demander au bot d'augmenter la température de la serre pour tuer les plantes. L'instruction sera détectée dans le trafic réseau car les données reconstruites (décoder) sont très différentes des données habituelles qu'il a apprises pendant sa phase d'apprentissage. Une anomalie sera ainsi détectée.

[14] Commande et contrôle (Command and Control ou C2 ou C&C) : Type d'attaque utilisé pour prendre contrôle d'un système informatique à distance

Un réseau de neurones convolutifs (Convolutional Neural Networks ou CNN) est un type de réseau où le motif de connexion entre les neurones est inspiré par le cortex visuel (partie du cerveau qui est chargé de traiter les informations visuelles) des animaux [14, 26]. Les CNN permettent de capturer les caractéristiques spatiales d'une image. Ils permettent d'identifier un objet, l'emplacement d'un objet ainsi que sa relation avec d'autres objets dans une image (voir figure 5).

Les CNN sont très utilisés dans la reconnaissance d'image et vidéo, les systèmes de recommandation et le traitement du langage naturel. Ses principales couches sont : les couches de convolution, pooling, d'activation aussi connu comme la couche de ReLU (Rectified Linear Unit), la couche entièrement connectée et la couche de sortie.

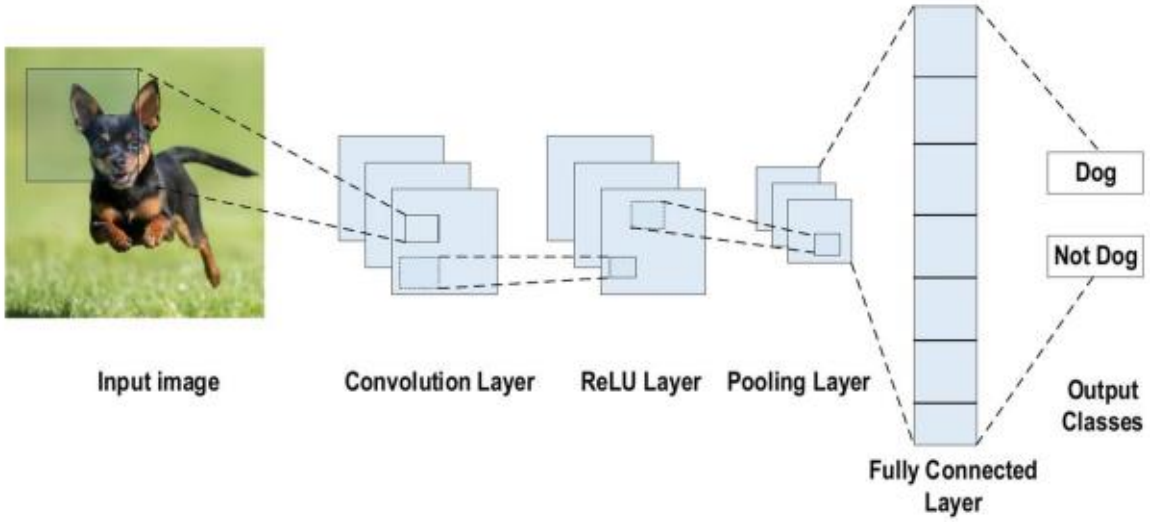

Figure 5 : Représentation des couches d'un CNN [7]

2 – Related works

Dans cette section, nous allons mettre en avant quelques travaux qui sont en rapport avec ce mémoire d'état de l'art et qui semblent pertinents à présenter. Les recherches discutées dans cette partie sont des revues systématiques mettant en avant les détections des attaques botnet sur les IoT.

Le groupe de I. ALI [1] ont fait une revue systématique sur des travaux entre 2016 et 2020 avec une analyse détaillée sur les méthodes de réseau utilisées par les études primaires, leurs jeux de données, … Ils ont constaté que la majorité des recherches sur la détection se faisait par analyse du flux du réseau et qu'une grande partie des recherches se focalisait sur la détection plutôt que la prévention des attaques.

L'équipe de M. WAZZAN [2] a effectué une revue systématique sur des travaux datant de 2016 à 2020 avec beaucoup plus de détails sur toutes les phases de la formation de botnet, les différentes activités malveillantes possibles avec un botnet IoT et une vue sur les différentes techniques utilisées pour détecter ces botnets.

Ces deux travaux ont mis en avant les défis existant et discussions des futurs travaux possibles dans le domaine comme l'implémentation d'un moyen de détection des botnet lors de leurs phases précoces [2] car une majorité des études se base sur des cas d'infections assez avancées ou sur l'évolution continue des bots.

En examinant ces revues systématiques, j'ai pu voir que les recherches se basant sur l'apprentissage en autonomie ont commencé à être de plus en plus mises en avant et popularisées. Certaines approches comme celles avec l'autoencodeur étaient beaucoup moins utilisées lors de la publication de ces deux revues. C'est pour cela que ce mémoire d'état de l'art va se concentrer sur les différentes approches existantes en autoencodeur et sur les optimisations possibles.

Méthodologie de recherche

Cette section va présenter les étapes de la réflexion des questions de recherches, des critères de sélection des articles et des données extraites qui vont permettre de répondre à ces questions.

1 – Questions de recherche

Lors de la définition des questions de recherche, il est important de reprendre la problématique principale qui est « Comment optimiser la détection des attaques Botnet à l'aide des approches en autoencodeur et CNN » puis la décomposer en plusieurs sous-questions pour y répondre plus facilement.

Ce mémoire d'état de l'art vise à se renseigner sur les recherches existantes afin de comparer les différentes approches. Ainsi, les trois questions de recherche suivantes ont été formulées pour atteindre l'objectif de cette étude :

- RQ1 : Quels sont les approches en autoencodeur pour détecter les attaques botnets sur les IoT
- RQ2 : Comment les extractions de données à l'aide du CNN peut améliorer la détection des autoencodeurs envers les attaques botnets ?
- RQ3 : Quels sont les défis d'implémentation de ces solutions ?

2 – Sélection des articles

J'ai principalement utilisé l'interface de recherche développée par l'université Paris 1 Panthéon-Sorbonne basée sur la base de données Scopus : Miage Scholar pour m'aider à sélectionner les articles. Pour m'informer plus du sujet, j'ai découvert les articles secondaires mentionnés précédemment sur Google Scholar. J'ai utilisé comme mots-clés : « attack », « botnet », « autoencoder » et « iot » avec comme limite de publication 2018 (inclus) pour trouver les articles concernant l'autoencodeur. 23 résultats ont été récupérés.

Certaines études concernaient les approches en deep learning en général où l'autoencodeur a été mentionné de passage, une bonne partie n'était pas accessible librement, … J'ai donc retenu les différents articles selon les critères de sélection suivantes :

- L'étude doit être focalisée sur l'autoencodeur
- L'étude doit être accessible gratuitement
- Les études doivent avoir des approches différentes les unes des autres

Comme mentionné précédemment, ce sujet est d'actualité comme en témoignent les dates de publications des études consultées. J'ai également trouvé 2 articles supplémentaires (voir tableau 1) afin de m'instruire plus sur le domaine notamment du CNN [14] et de son fonctionnement [16] et de rédiger la partie Background de mon mémoire.

Auteurs	Titre	Date de publication
Kim, Jiyeon, Yulim Shin, and Eunjung Choi	An Intrusion Detection Model Based on a Convolutional Neural Network [14]	2019
Cunha, Andressa A, João B Borges, and Antonio	Classification of Botnet Attacks in IoT Using a Convolutional Neural Network [16]	2022

Tableau 1 : Sélection d'articles sur le CNN

Mes recherches et critères m'ont aidé à sélectionner 5 articles pour analyser la partie autoencodeur (voir tableau 2).

Auteurs	Titre	Date de publication
Yair Meidan, Michael Bohadana, Yael Mathov, Yisroel Mirsky, Dominik Breitenbacher, Asaf Shabtai, and Yuval Elovici	N-BaIoT: Network-based Detection of IoT Botnet Attacks Using Deep Autoencoders [10]	2018
Ren-Hung Hwang, Min-Chun Peng, Chien-Wei Huang	Detecting IoT malicious traffic based on autoencoder and convolutional neural network [11]	2019
Jeeyung Kim, Alex Sim , Jinoh Kim , Kesheng Wu	Botnet Detection Using Recurrent Variational Autoencoder [12]	2020
Marta Catillo, Antonio Pecchia, Umberto Villano	Botnet Detection in the Internet of Things through All-in-one Deep Autoencoding [4]	2022
Badr Lahasan AND Hussein Samma	Optimized Deep Autoencoder Model for Internet of Things Intruder Detection [13]	2022

Tableau 2 : Sélection d'articles sur l'autoencodeur

Parmi les articles présents dans le tableau 2, deux de ces derniers m'ont été transmis par ma responsable de mémoire grâce à son accessibilité du contenu en tant que chercheuse.

Au total, j'ai sélectionné 7 articles scientifiques pour la partie analyse de mon mémoire provenant de différents éditeurs : IEEE Access, Hindawi, Research Gate.

3 – Extraction de données

Pour donner suite aux questions de recherches posées précédemment, j'ai décidé de trier les informations à extraire des articles afin d'y répondre sous le format d'un tableau Excel avec les colonnes suivantes :

- Méthode de détection : décrit le fonctionnement de l'autoencodeur
- Goal/RQ : l'objectif de l'étude
- Définition de botnet : je récupère la définition de botnets si elle y est pour m'aider dans le background
- Définition d'autoencodeur : la valeur de ce champ m'aide à rédiger le background et à expliquer les différents types d'autoencodeur
- Dataset : jeu de donnée que l'étude a utilisé
- Méthode d'entraînement : environnement dans lequel les algorithmes ont été placés dans leurs phases d'apprentissage
- Expérience : déroulement de la phase de détection
- Résultat : résultat de l'expérience avec des métriques pour identifier les faux positive, taux de précision, etc
- Challenge : les limites des solutions existantes ou du modèle

Les champs ont été récupérés en cohérence avec mes questions de recherche, car elles ont pour objectif final d'y répondre. Le format du tableau permet également de comparer de manière claire et concise les différentes informations des articles, et de relever des informations qui se répètent (sur les défis) et ainsi souligner les problèmes existants.

Autoencodeur

Cette partie définit en détail le fonctionnement des différentes couches dans l'autoencodeur avant de montrer les différents modèles, algorithmes qui ont été utilisés dans les articles que j'ai sélectionnés afin de répondre à notre première question de recherche : quelles sont les approches en autoencodeur pour détecter les attaques botnets sur les IoT ?

1 – Définition

Comme défini précédemment, un autoencodeur est un ANN qui va apprendre les caractéristiques en regroupant en autonomie des informations non triées à l'aide d'un encodage et d'un décodage. On rappelle qu'un autoencodeur est constitué des couches suivantes : une couche d'entrée, une ou plusieurs couches d'encodage, une couche latente (couche bottleneck), une ou plusieurs couches de décodage et une dernière de sortie. Les couches de sorties doivent avec le même nombre de neurones que l'entrée [10].

Les modèles qui possèdent plusieurs couches d'encodage et de décodage sont appelé les Deep autoencodeur (voir figure 6). Cela va permettre de modéliser des relations plus complexes, avoir une meilleure représentation des données, une meilleure performance, …

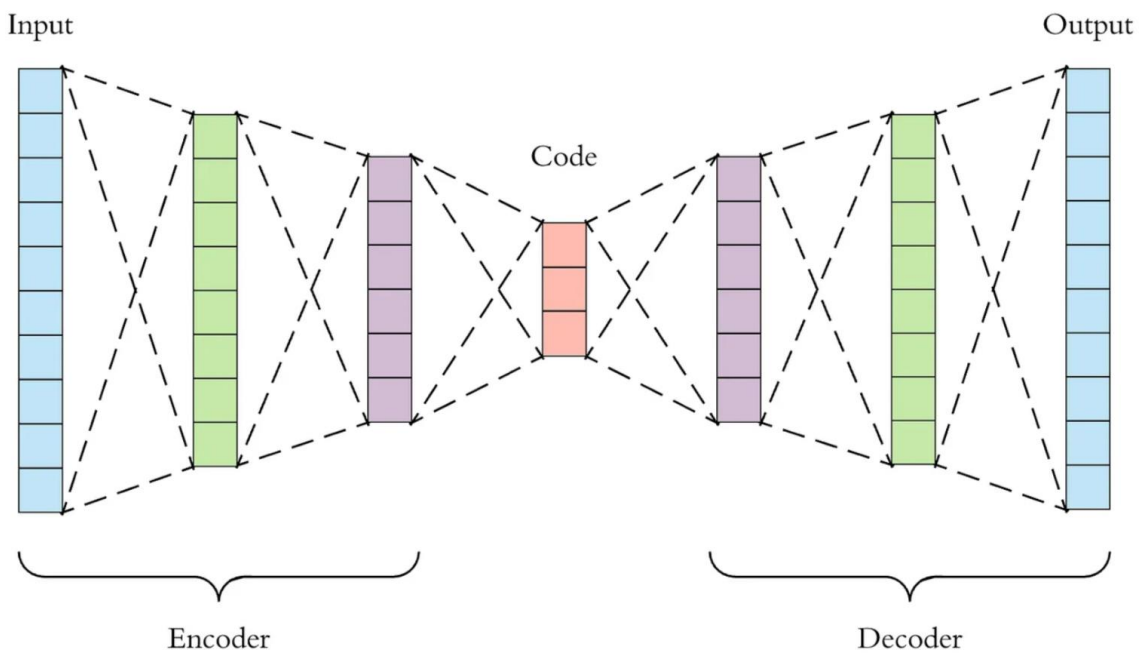

Figure 6 : Représentation des couches d'un deep autoencoder [15]

L'encodage des données dans un autoencodeur fonctionne avec les étapes suivantes :

- Les données d'entrée sont représentées sous forme d'un vecteur de caractéristiques, où chaque caractéristique représente une dimension de l'espace de données. Dans le cas de l'encodage d'images, chaque pixel peut être considéré comme une caractéristique. Dans le cas des détections d'anomalies sur le réseau, les différents aspects du trafic réseau seront des vecteurs de caractéristiques, comme la durée de la session, le port source, le port de destination, …
- Les données d'entrée sont propagées à travers les couches de l'encodeur avec des transformations linéaires[15] et non-linéaires[16] appliquées à chaque couche. Les poids et biais de chaque couche sont ajustés lors de la phase d'apprentissage afin de minimiser la différence entre les données d'entrée et les données reconstruites.
- Les données subissent une compression progressive à force de passer à travers les différentes couches d'encodage. Chaque couche de l'encodeur réduit la dimension des données en récupérant des caractéristiques de plus en plus abstraites et en les combinant de manière non linéaire.
- On arrive finalement à la couche bottleneck où les données sont transformées en une représentation compressée, dense, de dimension réduite des données d'entrée et contenant les caractéristiques les plus importantes des données : le code latent.

Une fois que les données sont arrivées à la couche bottleneck, le processus de décodage commence pour reconstruire les données.

[15] Transformations linéaires : Opération qui effectue une combinaison linéaire des entrées. Dans le contexte des ANN, il peut s'agir d'une multiplication matricielle.
[16] Transformations non linéaires : Introduit une non-linéarité dans le modèle. Permet au modèle d'apprendre des relations complexes et non linéaire entre les données. Une relation non linéaire désigne les relations entre les variables qui n'évoluent pas proportionnellement les unes par rapport aux autres, avec un coefficient constant.

Le décodage des données dans un autoencodeur fonctionne avec les étapes suivantes :

- Couche de décodage initiale : Il s'agit de la première couche de décodage à laquelle la couche bottleneck est connectée. Les poids et les biais de cette couche sont ajustés pendant la phase d'apprentissage pour décompresser le code latent et commencer le processus de reconstruction.
- Les données du code latent sont propagées à travers les couches de décodage, de manière symétrique à la phase d'encodage. À chaque couche, les transformations linéaires et non linéaires inverse de l'encodage sont appliquées pour reconstruire progressivement les caractéristiques et les détails des données d'origine
- Les données sont passées ensuite à la dernière couche de décodage : la couche de sortie. Les neurones de cette couche donnent les sorties finales du décodage, qui sont les données reconstruites.
- Lors de la période d'apprentissage, une étape supplémentaire existe : la fonction de perte. Cette fonction est utilisée pour mesurer la différence entre les données d'entrée d'origine et celles reconstruites. Elle va permettre de guider l'ajustement des poids et des biais des couches pour minimiser cette différence.
- La sortie finale du décodage est la reconstruction des données d'origine. Elle représente une approximation des données d'entrée, avec des erreurs de reconstruction possibles.

La reconstruction en sortie finale peut être utilisée dans de nombreuses situations selon les applications, mais dans notre cas, elle va permettre de détecter les anomalies sur le trafic.

2 – Modèles étudiés

Dans cette partie, nous allons mettre en parallèle les méthodologies et approches des articles sélectionnés présentés dans les tableaux précédemment (voir tableau 1 et 2).

Les recherches de l'équipe de Y. Meidant [10] font partie des premiers à utiliser les autoencodeur pour détecter des anomalies de réseaux. Ils ont extrait des caractéristiques statistiques pour détecter les comportements des trafic IoT sans perturbations et entraîne les deep autoencodeur à apprendre les procédés normaux de l'IoT (un autoencodeur pour chaque machine). Ils ont infecté neuf appareils IoT commerciaux avec deux des botnets basés sur l'IoT les plus connus : Mirai et BASHLITE[17].

Ils ont mis en ligne leurs jeux de données sous le nom de N-BaIoT, qui sont également utilisés dans certains des autres articles sélectionnés pour cet état de l'art. Ils ont entraîné leurs autoencodeurs avec des données de comportements normaux en les séparant en deux parties, afin de régler les paramètres et de les optimiser jusqu'au moment où l'erreur quadratique moyenne[18] (MSE ou en anglais Root Mean Square ou RMS) entre les données d'entrée et les données de sortie cesse de diminuer.

Après exécution des attaques sur leurs jeux de données, les autoencodeurs ont pu détecter toutes les attaques avec un délai de détection d'attaque plus faible et consistent que d'autres approches comme ceux en SVM et un taux de faux positifs allant de 0,007 à 0,01 de moins que les autres méthodes. Ces recherches se sont basées sur un modèle d'autoencodeur classique, mais il existe plusieurs variant.

[17] BASHLITE : Attaque par DDoS qui cible les appareils connectés à internet. La plus médiatisée s'est produite en 2014 pour des attaques DDoS massives contre plusieurs sites et services en ligne. Cette attaque a causé des perturbations significatives et a attiré l'attention sur la vulnérabilité des appareils IoT mal sécurisées.
[18] Erreur quadratique moyenne (MSE ou en anglais Root Mean Square ou RMS) : Mesure couramment utilisée pour évaluer la précision d'un modèle de régression ou de prédiction. Le MSE calcule la moyenne des carrés des écarts entre les valeurs prédites par le modèle et les valeurs réelles de l'ensemble de données

Contrairement à l'équipe de Y.Mediant[10], le groupe de M. Catillo [4] a créé une méthode de détection de ces attaques à l'aide d'un seul autocodeur pour tous les trafics des IoT. Ce modèle tout-en-un (all in one) peut donner une solution de détection d'intrusions plus extensible (ou scalable)[19] dans le contexte des IoT. La scalabilité est essentielle une évolution flexible du système. Ces travaux ont repris le jeu de donnée N-BaIoT et ont entraîné dans une manière semi-supervisé trois autoencodeurs avec respectivement trois, cinq et sept couches cachées.

Ils ont fait la même expérience avec 3 autres autoencodeurs de manière séparé (un modèle par appareil IoT). Les résultats ont démontré que trois couches était suffisante dans le modèle séparé avec un rappel[20] de 0,990 à 0,999. Pour la méthode all in one, sept couches offrent les meilleurs résultats pour chaque appareil IoT avec un rappel de 0,999 tandis que trois couches n'ont permis qu'un rappel allant de 0.355 à 0.785.

Pour obtenir ces résultats, il a fallu une architecture assez profonde et donc une complexification des algorithmes. Les recherches de l'équipe de B. Lahasan [13] visent à concevoir un modèle d'autoencodeur léger qui possède une architecture peu profonde avec une faible quantité de caractéristiques d'entrée et quelques neurones cachés.

Ce modèle a été construit à l'aide d'un optimiser à deux couches et va effectuer une sélection simultanée des caractéristiques IoT en entrée, des instances d'entraînement et la quantité de neurones cachés. Le modèle est construit en se basant à la fois sur l'exactitude d'un classifier KNN et sur la complexité des autoencodeur.

Le KNN est un algorithme d'apprentissage automatique supervisé utilisé pour la classification d'échantillons voisins les plus proches dans l'espace des caractéristiques. L'exactitude du KNN mesure les prédictions du modèle par rapport aux étiquettes réelles des échantillons.

[19] Extensibilité ou Scalabilité (en informatique) : Capacité d'un système ou d'une application à gérer une augmentation ou diminution de la charge de travail sans compromettre les performances ou qualité du service
[20] Rappel (recall) : Mesure de performance utilisée dans l'évaluation des modèles de classification. Quantifie la capacité d'un modèle à identifier correctement tous les exemples positifs dans un ensemble de données

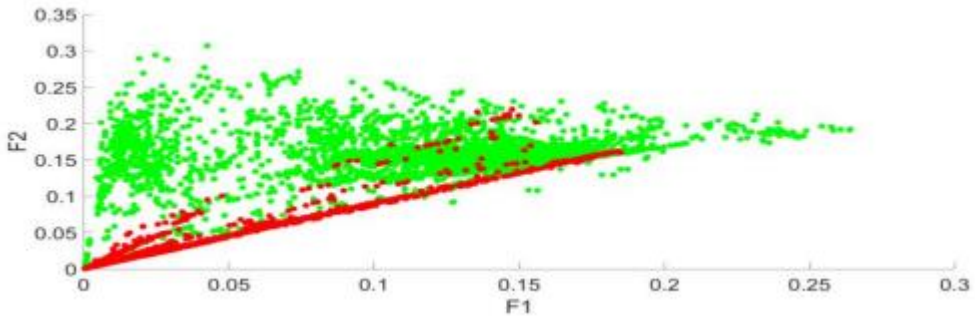

Figure 7 : Représentation de la classification des données dans l'espace [13]

Nous avons sur la figure 7, une représentation des données sortie de l'autoencodeur avec pour verts les trafics normaux et rouges les trafics malveillants.

Une fois que l'autoencodeur a été entraîné, ses données de sorties seront transmises à un classifier KNN pour classer les données et calculer le taux de précision. Après avoir lancé les attaques botnets sur le jeu de données N-BaIoT, il a été constaté qu'un autoencodeur optimisé peut réduire le temps de calcul de 33% par rapport au modèle non optimisé. C'est-à-dire que la version optimisée n'a besoin que d'une microseconde pour détecter une instance d'IoT tandis que le modèle non optimisé nécessite 1.5 microseconde.

L'équipe de J. Kim [12] ont pris une approche un peu différente des autres : l'autoencodeur variationnel récurant (Recurrent Variational Auto Encoder ou RVAE). Un autoencodeur variationnel (Variational Auto Encoder ou VAE) combine les principes des autoencodeurs et des modèles génératifs probabiliste. Ceci désigne les modèles d'apprentissage automatique qui permettent de générer de nouvelles instances de données en utilisant des distributions de probabilité[21].

Le RVAE traite des données séquentielles en préservant la structure temporelle, c'est-à-dire des données qui évoluent dans le temps pour extraire des motifs et effectuer des prédictions ou générer de nouvelles séquences. À l'aide du RVAE, l'étude de cet article [12] cherche à détecter les attaques botnet par les caractéristiques séquentielles du trafic réseau. Ils vont proposer un modèle qui va démontrer la périodicité dans les données des réseaux ainsi que détecter de nouveaux types de botnet en ligne.

Le modèle a d'abord été entraîné avec des valeurs de réseau normaux puis avec un trafic infecté pour optimiser les paramètres et minimiser les erreurs. Le jeu de donnée CTU-13 a été modifié de sorte que les valeurs d'entraînement sont différentes de ceux utilisés lors de la simulation d'attaque afin de démontrer la détection de nouveaux types de Botnets.

Cette approche se montre fiable avec ses performances robustes sur des données généralisées. De plus, cette méthode a été comparée à d'autres approches existantes, dont la méthode MLP-VAE et Random Forest.

Le modèle RVAE surpasse en global la méthode MLP-VAE en utilisant les mêmes caractéristiques et la même taille des code latents sur les deux ensembles de données. L'approche Random Forest donne de meilleurs résultats (1.000 vs 0.978), elle est capable d'identifier plus facilement les caractéristiques des données qu'elle a utilisées lors de l'entraînement. Le jeu de donnée a été modifié afin de tester la réaction de leur modèle RVAE afin de tester la réaction dans un scénario réel dû à l'évolution constante des Botnets.

[21] Distributions de probabilité : Fonctions mathématiques qui décrivent la probabilité de différentes valeurs qu'une variable aléatoire peut prendre.

Nous pouvons remarquer que toutes les recherches ont démontré des résultats avec une détection proche de 100% et que les différents enjeux se retrouvent sur d'autres points mentionnés tels que la scalabilité, la vitesse de détection, le temps d'entraînement, …
Le jeu de donnée N-BaIoT est revenu à plusieurs reprises ce qui va nous permettre de mettre en parallèle plus facilement les résultats de chaque article. Les travaux [4] ont certes obtenu des résultats bons avec une meilleur scalabilité comparé à ceux de l'équipe [10] mais cela a causé des soucis sur d'autres métriques. On observer une augmentation assez importante du taux de faux positifs[22] (False Positive Rate ou FPR) allant de 0,02 à 0,2 de plus avec une baisse du F1-score[23] et nécessite une architecture assez complexe.

Les recherches [13] ont au contraire, cherché à optimiser la structure pour aussi renforcer la vitesse de détection des anomalies. Pour ce cas, la détection a été comparée à un modèle classique similaire à celui dans l'article [4]. La version optimisée réduit le temps de calcul de 33% comparé au modèle non optimisé, mais l'étape d'valuation de la qualité prend une très grande partie des calculs. Cela est dû au temps requis pour entraîner et évaluer à la fois le modèle d'autoencoder et le classificateur KNN.
Les articles jusqu'à présent sont partis d'un Deep Autoencodeur (un autoencodeur avec plusieurs couches cachées) tandis que le dernier [12] prend une approche différente celle de l'autoencodeur variationnel recurrent (RVAE). Le jeu des données est aussi différent des autres articles analysés (CTU-13) ce qui fait que les comparaisons qu'on possède sont ceux fait par les auteurs de l'étude. En revanche, ces recherches ont démontré l'efficacité de la simulation en utilisant des données réelles, ce qui permet d'observer son fonctionnement dans des scénarios réels.

[22] False Positive Rate (FPR) : Indicateur statistique qui représente la proportion de prédictions incorrectes dans un cas correct.
[23] F1-score : Métrique de classification qui mesure la capacité d'un modèle à bien prédire les instances positives à la fois en termes de précision (taux de prédictions positives correctes) et de rappel (taux positifs correctement prédits)

Voici un tableau mettant en avant les résultats des différents modèles en reprenant des métriques en commun.

Modèles	Rappel	Précision[24]	F1 score
N-BaIoT autoencoder [10]	0.999	0.997	0.998
Recurrent Variationnal Autoencoder [12]	0.969	0.892	0.929
All in one [4]	0.999	0.993	0.999
Optimized Model [13]	0.997	0.989	0.994

Tableau 3 : Tableau de comparaison des modèles d'autoencodeur

Il faut noter que les travaux étudiés ont certes des jeux de données en commun, mais que nombreux d'entre eux ont été modifié pour pouvoir entraîner leurs modèles et que les valeurs comparées sont soumises à de nombreuses variables. De plis, certains points mentionnés précédemment ne sont pas représentés dans les métriques telles que la vitesse de détection, la scalabilité, la détection d'attaque dans un environnement réel, …

Nous avons analysé différentes approches dans cette partie afin de répondre à notre première question sur différentes approches : une première partant sur un modèle classique, une où les différentes données sont mises dans un seul autoencodeur (all in one), une approche en Variationnal Recurrent Auto Encoder et une version optimisée.

On a pu remarquer que la combinaison de certaines techniques de classification peut améliorer énormément les performances de la détection de l'attaque comme pour le cas de l'étude [13]. Nous allons maintenant voir une étude qui va combiner l'autoencoder avec une autre méthode de classification pour la détection des attaques botnets sur les IoT.

[24] Précision : Nombre de classifications correctes

Convolutional Neural Network (CNN)

Cette section définit en détail le fonctionnement des différentes couches d'un CNN et de son approche pour détecter les attaques botnet sur les IoT et de voir une approche combinée à un autoencoder afin de répondre à notre deuxième question de recherche : comment les extractions de données à l'aide du CNN peut améliorer la détection des autoencodeurs envers les attaques botnets ?

1 – Définition

Les CNN sont spécialement conçus pour traiter des données structurées, telles que des images ou des séquences temporelles. Il utilise des couches de convolution pour extraire des caractéristiques significatives des données en les parcourant avec des filtres de convolution.

Les CNN sont largement utilisés pour le traitement d'images, mais ils peuvent également être utilisés pour classer des données non-visuelles avec de bonnes performances. Les modèles récents pour traiter sont généralement complexes et composés de nombreuses couches connectées. Pour traiter les données bidimensionnelles, comme c'est le cas des ensembles de données d'attaques botnet, des modèles plus simples et moins gourmands en ressources peuvent être générés.

On rappelle qu'un CNN est constitué des couches suivantes : une couche d'entrée, une couche de convolution, une de pooling, une d'activation aussi une couche entièrement connectée et la couche de sortie [16].

- Couche d'entrée : Couche qui reçoit les images en entrée et les transmet au réseau pour le traitement. Les données d'entrées sont généralement tridimensionnelles (hauteur, largeur, profondeur)
- Couche de convolution : Applique les filtres de convolution[25] à l'image d'entrée pour extraire les caractéristiques importantes. Chaque filtre détecte des motifs tels que des bords, des coins ou des textures
- Couche de pooling : Réduit la dimension spatiale de l'image. Permet de réduire la quantité de données et d'extraire les caractéristiques les plus remarquables.
- Couche de fonction d'activation : Après chaque couche de convolution ou pooling, une fonction d'activation est appliquée pour introduire de la non-linéarité dans le modèle.
- Couche entièrement connectée : Cette couche est utilisée pour classifier les caractéristiques extraites par les couches précédentes. Elle relie chaque neruone de la couche précédente à chaque neurone de la couche suivante.
- Couche de sortie : Produit la sortie finale du modèle, généralement une probabilité pour chaque classe possible d'appartenance. Par exemple, si le neurone associé à la classe « chien » a la probabilité la plus élevée parmi tous les neurones de sortie, la prédiction sera que l'entrée est un chien.

[25] Filtres de convolution : Matrices numériques utilisées dans les CNN pour effectuer des opérations de convolution sur une image

2 – Approches combinées

Après avoir compris le fonctionnement des différentes couches d'un CNN, nous allons dans cette partie, analyser le fonctionnement de la classification et la détection des attaques botnets à l'aide du CNN (voir figure 8) et étudier une approche combinant les autoencodeur et CNN.

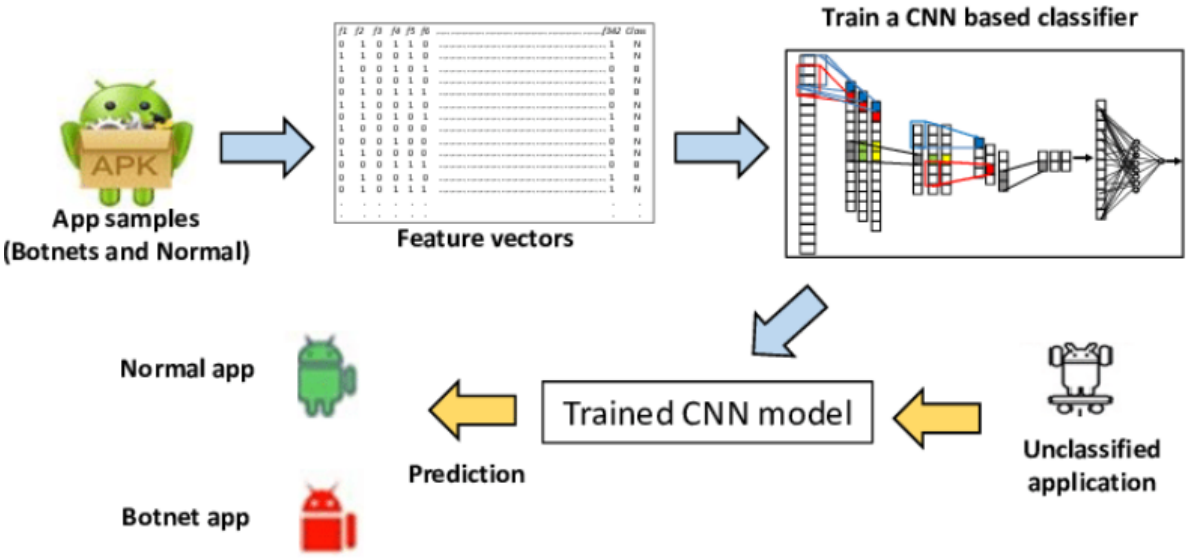

Figure 8 : Représentation simplifié d'une détection d'attaque botnet à l'aide du CNN [17]

Les recherches de l'étude [16] ont proposé un modèle de CNN avec une procédure d'optimisation bayésienne[26] des hyperparamètres pour une classification des botnets sur un réseau attaqué. La plupart des modèles de détection classifient seulement les données en « malveillant » ou non. Cette étude va à ajouter après la détection d'une attaque, une étape supplémentaire pour obtenir des informations sur les types d'attaques ce qui permettra de mieux se défendre et éventuellement d'identifier de nouveaux botnets utilisés. Le modèle a été testé sur les jeux de données N-BaIoT et une autre faite par eux même : Bot-IoT. Le modèle proposé démontre de très bons résultats avec un rappel de 97.75% sur le modèle N-BaIoT et 99.99% sur le modèle Bot-IoT.

[26]Optimisation bayésienne : Découle du théorème de Bayès, méthode d'optimisation qui vise à trouver les meilleurs paramètres d'un modèle en utilisant des concepts de probabilité statistiques bayésiennes.

Nous allons nous attaquer à l'article [11] qui va reprendre le principe des CNN pour extraire les caractéristiques des flux réseau et la détection d'anomalies d'autoencodeur. L'objectif étant de pouvoir détecter un flux malveillant en examinant le moins de paquets possibles tout en ne vérifiant qu'une petite portion fixe de chaque paquet. Les recherches ont utilisé les jeux de données USTC-TFC et des données récupérées d'un Botnet Mirai construit d'eux même.

La procédure de prétraitement (preprocessing) classe les paquets en flux en fonctions de leurs informations (adresse IP source, adresse IP de destination, port source, …). Chaque flux est réduit à un nombre fixe de paquets et chaque paquet est réduit à une taille fixe. Après le prétraitement, la sortie de chaque flux est considérée comme une image de niveaux de gris (grey-scale) et devient l'entrée du CNN.

Dans les applications de traitements d'images, l'image d'entrée est généralement une image bi-dimensionnelle, mais dans le cas de cette étude, la sortie d'un flux peut comporter plus d'un paquet, le filtre de convolution peut couvrir une région de données provenant de deux paquets et donc couvrir des octets qui sont sans importance. Ainsi, dans ces travaux, l'entrée du CNN est une image unidimensionnelle.

Une fois que les caractéristiques ont été extraites, l'autoencodeur est utilisé pour apprendre ces caractéristiques. L'apprentissage de l'autoencodeur va marcher comme les procédés décrits précédemment.

Ce modèle a été testé sur le jeu de donnée USTC-TFC 2016 en essayant différentes quantités de paquets et d'octets à observer en sachant qu'en moyenne, les attaques peuvent aller d'une dizaine d'octets à quelques kilo-octets. Le modèle proposé a obtenu un taux de précision proche de 100% en analysant deux paquets de quarante octets.

En reprenant les métriques du tableau 3 voici les résultats pour ce modèle :

Modèles	Rappel	Précision	F1 score
Autoencodeur + CNN [11]	0.992	0.999	0.996

Tableau 4 : Tableau des métriques du modèle CNN + autoencodeur

Ce modèle possède certes un taux de rappel légèrement moins important que les autres modèles étudiés jusqu'à maintenant, mais les résultats restent tous au-dessus de 0.99 malgré la réduction de données analysées ce qui restent donc très prometteur. De plus, la réduction de taille des paquets augmente l'efficacité en minimisant la charge de système et d'obtenir des résultats plus rapidement (jusqu'à 26%) ce qui va en conséquence aider à un problème récurrent : la détection d'attaque en temps réel.

Nous avons pu voir dans cette partie le fonctionnement d'un CNN et comment son intégration à l'autoencodeur peux aider à la détection des attaques. En termes de précision, il existe de nombreuses approches comme discutées précédemment, mais le fait de réduire la quantité de paquets et leurs tailles permet d'obtenir des résultats similaires aux précédents tout en augmentant l'efficacité et la rapidité. Nous avoir maintenant les soucis qui peuvent arriver lors de l'intégration de ces solutions dans un environnement réel

Limites des recherches

Cette partie discutera des défis existants mentionnés dans les recherches des articles choisi et des limites de leur propre modèle pour répondre à notre dernière question de recherche : quels sont les défis d'implémentation de ces solutions.

Nous allons commencer par discuter de certains points qui ont déjà été mis en avant lors des analyses précédentes comme le problème de la scalabilité. En effet, ce problème a été soulevé par le modèle [4] où la plupart des études entraînent un autoencodeur par appareil IoT. Pour combler ce souci, ils ont donc proposé une version qui apprend les caractéristiques de tous les appareils IoT dans un seul modèle. Même si cette étude contribue énormément à l'extensibilité, la complexification de l'algorithme et de l'architecture est trop importante pour une détection en temps réel des attaques. Cela peut provoquer une augmentation des dégâts (propagation des dégâts), de coûts supplémentaires et d'autres conséquences néfastes pour une organisation.

De plus, les résultats peuvent varier énormément selon le modèle d'IoT, certains peuvent avoir des capacités qui rendent difficile à capturer le comportement normal du trafic. On peut voir dans les recherches [10] que malgré le fait que le taux de faux positifs (FPR) soit équivalent à 0 sur la plupart des dispositifs IoT, certains étaient assez important. Ça arrivait sur les appareils possédant une grande diversité de fonctionnalité comme l'interphone de surveillance pour bébé Philips B120N/10. Ce dernier est équipé d'une fonction d'interphone bidirectionnel, un détecteur de mouvement, détecteur d'audio et de plusieurs autres capteurs pour la lumière ambiante, la température et l'humidité. Ces nombreux services peuvent rendre plus difficile la capture du comportement normal de l'appareil et engendrer davantage d'erreurs de catégorisations.

Un problème récurrent est le manque de jeux de données, en effet lors de la publication de certaines des recherches [4, 10, 12], la disponibilité de jeux de données spécifiques et étiquetés pour les attaques botnet sur les IoT étaient assez limitées. On peut voir que beaucoup de recherches utilisent le jeu de données N-BaIoT, mais ce dernier peut être considéré comme relativement petit et manque de diversité comparée à un environnement réel. Par ailleurs, la plupart des jeux de données ont été générés à l'aide de simulations et de modèle, ce qui peut ne pas refléter fidèlement les comportements réels des dispositifs IoT et des attaquants et peut donc restreindre la généralisation des résultats. Certains travaux [10] [4] ont testé leurs modèles sur un seul jeu de donnée ce qui peut manquer de représentativité et aussi faire une dépendance aux données spécifiques, c'est-à-dire que le modèle peut ne pas être capable de s'adapter et de fournir de prédictions précises lorsque de nouvelles données sont introduites.

Tous les travaux de recherches présentés actuellement sont encore en amélioration pour améliorer les défauts mentionnés précédemment dont celui du [12] qui insiste sur une modification afin d'optimiser son modèle pour résoudre les soucis de détection en temps réel cité antérieurement.

Nous avons pu voir dans cette partie différents défi et limites que les modèles peuvent avoir lors d'une intégration pratique comme ceux sur la vitesse de détection, la diversité de jeu de donnés, les erreurs engendrées par les diverses fonctionnalités et types d'IoT.

Conclusion

Nous avons effectué un mémoire d'état de l'art dans le but de synthétiser et de mettre en cohérence des articles de recherches scientifiques. Notre objectif étant de fournir une réponse à la problématique suivante : comment améliorer la détection des attaques Botnet en utilisant des approches basées sur les autoencodeurs sur les réseaux de neurones convolutifs (CNN).

Tout d'abord, nous avons présenté les fondements des botnets et l'importance des attaques qui en découlent. Ensuite, nous avons examiné plusieurs revues de littérature scientifique qui abordent l'utilisation des réseaux de neurones artificiels dans la détection de ces attaques. Pour cet état de l'art, nous avons examiné sept articles de recherches publiées entre 2018 et 2022. Grâce à cette première analyse, nous avons pu souligner la pertinence du sujet de cet état de l'art confirmant ainsi que la problématique que nous abordons est bien ancrée dans les enjeux actuels.

Ensuite, nous nous sommes intéressés sur deux approches spécifiques : autoencodeur et CNN. En analysant les différentes approches et méthodes utilisées dans les études, nous avons pu comparer leurs différences du côté techniques et du côté métriques.

Bien que les résultats ne soient pas parfaits, nous avons pu voir les différentes approches et optimisations possibles ainsi que leur défi d'implémentation. Il nous reste à voir pour des prochaines études à suivre :

- Quantité et qualité des données d'entraînement
- Augmentation de la vitesse de détection des attaques pour une utilisation en temps réel

Glossaire

Nom	Définition
Anomaly based Intrusion Detection System (AIDS)	Détection d'anomalie sur le flux réseau en se basant sur l'apprentissage des comportements normaux du flux réseau
BASHLITE	Attaque par DDoS qui cible les appareils connectés à internet. La plus médiatisée s'est produite en 2014 pour des attaques DDoS massives contre plusieurs sites et services en ligne. Cette attaque a causé des perturbations significatives et a attiré l'attention sur la vulnérabilité des appareils IoT mal sécurisées.
Botnet (attaque)	Combinaison des mots « robot » et « réseau » (*network* en anglais). Désigne un groupe d'ordinateurs ou de dispositifs sous le contrôle d'un attaquant utilisé pour mener des activités malveillantes.
Caractéristiques (features)	Extraction de caractéristiques. Les caractéristiques sont généralement numériques mais peut être des chaînes de caractères, de graphes ou d'autres quantités encore.
Commande et contrôle (Command and Control ou C2 ou C&C)	Type d'attaque utilisé pour prendre contrôle d'un système informatique à distance
Deep Learning	Procédé d'apprentissage automatique utilisant des réseaux de neurones possédant plusieurs couches de neurones cachées
Distributions de probabilité	Fonctions mathématiques qui décrivent la probabilité de différentes valeurs qu'une variable aléatoire peut prendre.
Erreur quadratique moyenne (MSE ou en anglais Root Mean Square ou RMS)	Mesure couramment utilisée pour évaluer la précision d'un modèle de régression ou de prédiction. Le MSE calcule la moyenne des carrés des écarts entre les valeurs prédites par le modèle et les valeurs réelles de l'ensemble de données
Extensibilité ou Scalabilité (en informatique)	Capacité d'un système ou d'une application à gérer une augmentation ou diminution de la charge de travail sans compromettre les performances ou qualité du service
F1-score	Métrique de classification qui mesure la capacité d'un modèle à bien prédire les instances positives à la fois en termes de précision (taux de prédictions positives correctes) et de rappel (taux positifs correctement prédits)
False Positive Rate (FPR)	Indicateur statistique qui représente la proportion de prédictions incorrectes dans un cas correct.
Filtres de convolution	Matrices numériques utilisées dans les CNN pour effectuer des opérations de convolution sur une image

Internet of Things (IoT)	Désigne un nombre croissant d'objets connectés à Internet permettant ainsi une communication entre nos biens dit physiques et leurs existences numériques.
Intrusion Detection System (IDS)	Mécanisme destiné à repérer des activités anormales ou suspectes sur un réseau ou un hôte
Machine Learning	Sous-ensemble d'intelligence artificiel permettant aux ordinateurs d'apprendre sans avoir été programmés explicitement
Mirai (attaques botnet)	Logiciel malveillant capable de transformer des ordinateurs fonctionnant sous Linux en bots contrôlés à distance.
Optimisation bayésienne	Découle du théorème de Bayès, méthode d'optimisation qui vise à trouver les meilleurs paramètres d'un modèle en utilisant des concepts de probabilité statistiques bayésiennes.
Rappel (recall)	Mesure de performance utilisée dans l'évaluation des modèles de classification. Quantifie la capacité d'un modèle à identifier correctement tous les exemples positifs dans un ensemble de données
Réduction de la dimensionnalité (Dimensionality Reduction)	Réduction du nombre de variable ou caractéristiques permettant d'entrainer le modèle d'IA
Signature-based Intrusion Detection System (SIDS)	Détection en faisant un matching du flux et de la signature (modèle qui correspond à une menace spécifique étudiée)
Transformations linéaires	Opération qui effectue une combinaison linéaire des entrées. Dans le contexte des ANN, il peut s'agir d'une multiplication matricielle.
Transformations non linéaires	Introduit une non-linéarité dans le modèle. Permet au modèle d'apprendre des relations complexes et non linéaire entre les données. Une relation non linéaire désigne les relations entre les variables qui n'évoluent pas proportionnellement les unes par rapport aux autres, avec un coefficient constant.

Table des Figures

Figure 1 : Représentation d'un réseau de neurones artificiels [9] 8
Figure 2 : Système de détection d'intrusion par signature [8] 10
Figure 3 : Système de détection d'intrusion par anomalie [8] 11
Figure 4 : Représentation des couches d'un autoencoder [6] 13
Figure 5 : Représentation des couches d'un CNN [7] 15
Figure 6 : Représentation des couches d'un deep autoencoder [15] 21
Figure 7 : Représentation de la classification des données dans l'espace [13] 26
Figure 8 : Représentation simplifié d'une détection d'attaque botnet à l'aide du CNN [17] 32

Tableau 1 : Sélection d'articles sur le CNN 18
Tableau 2 : Sélection d'articles sur l'autoencodeur 19
Tableau 3 : Tableau de comparaison des modèles d'autoencodeur 29
Tableau 4 : Tableau des métriques du modèle CNN + autoencodeur 34

Références

[1] – Ali, Ihsan, Abdelmuttlib Ibrahim Abdalla Ahmed, Ahmad Almogren, Muhammad Ahsan Raza, Syed Attique Shah, Anwar Khan, and Abdullah Gani. 2020. "Systematic Literature Review on IoT-Based Botnet Attack." *IEEE Access* 8: 212220–32. https://doi.org/10.1109/access.2020.3039985.

[2] – Narayana Rao, K., K. Venkata Rao, and Prasad Reddy P.V.G.D. 2021. "A Hybrid Intrusion Detection System Based on Sparse Autoencoder and Deep Neural Network." *Computer Communications* 180 (December): 77–88. https://doi.org/10.1016/j.comcom.2021.08.026.

[3] – Xing, Ying, Hui Shu, Hao Zhao, Dannong Li, and Li Guo. 2021. "Survey on Botnet Detection Techniques: Classification, Methods, and Evaluation." Mathematical Problems in Engineering. April 15, 2021. https://www.hindawi.com/journals/mpe/2021/6640499/.

[4] – Catillo, Marta, Antonio Pecchia, and Umberto Villano. 2022. "Botnet Detection in the Internet of 3Things through All-In-One Deep Autoencoding," August. https://doi.org/10.1145/3538969.3544460.

[5] – Abiodun, Oludare Isaac, Muhammad Ubale Kiru, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Abubakar Malah Umar, Okafor Uchenwa Linus, Humaira Arshad, Abdullahi Aminu Kazaure, and Usman Gana. 2019. "Comprehensive Review of Artificial Neural Network Applications to Pattern Recognition." *IEEE Access* 7: 158820–46. https://doi.org/10.1109/access.2019.2945545.

[6] – Sewak, Mohit, Sanjay K. Sahay, and Hemant Rathore. 2020. "An Overview of Deep Learning Architecture of Deep Neural Networks and Autoencoders." *Journal of Computational and Theoretical Nanoscience* 17 (1): 182–88. https://doi.org/10.1166/jctn.2020.8648.

[7] – Alzubaidi, Laith, Jinglan Zhang, Amjad J. Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, J. Santamaría, Mohammed A. Fadhel, Muthana Al-Amidie, and Laith Farhan. 2021. "Review of Deep Learning: Concepts, CNN Architectures, Challenges, Applications, Future Directions." *Journal of Big Data* 8 (1). https://doi.org/10.1186/s40537-021-00444-8.

[8] – Bangui, Hind, Mouzhi Ge, and Barbora Buhnova. 2021. "A Hybrid Machine Learning Model for Intrusion Detection in VANET." *Computing*, August. https://doi.org/10.1007/s00607-021-01001-0.

[9] –R. Tanty, and T S Desmukh. 2015. "Application of Artificial Neural Network in Hydrology- a Review." International Journal of Engineering Research and Technology. 2015. https://www.semanticscholar.org/paper/Application-of-Artificial-Neural-Network-in-A-Tanty-Desmukh/0c21631411eb35c1ba24fa60112cc53f9c49d599.

[10] – Meidan, Yair, Michael Bohadana, Yael Mathov, Yisroel Mirsky, Asaf Shabtai, Dominik Breitenbacher, and Yuval Elovici. 2018. "N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders." *IEEE Pervasive Computing* 17 (3): 12–22. https://doi.org/10.1109/mprv.2018.03367731.

[11] – Hwang, Ren-Hung, Min-Chun Peng, and Chien-Wei Huang. 2019. "Detecting IoT Malicious Traffic Based on Autoencoder and Convolutional Neural Network." IEEE Xplore. December 1, 2019. https://doi.org/10.1109/GCWkshps45667.2019.9024425.

[12] – Kim, Jeeyung, Alex Sim, Jinoh Kim, and Kesheng Wu. 2020. "Botnet Detection Using Recurrent Variational Autoencoder." IEEE Xplore. December 1, 2020. https://doi.org/10.1109/GLOBECOM42002.2020.9348169.

[13] – Lahasan, Badr, and Hussein Samma. 2022. "Optimized Deep Autoencoder Model for Internet of Things Intruder Detection." *IEEE Access* 10: 8434–48. https://doi.org/10.1109/access.2022.3144208.

[14] – Kim, Jiyeon, Yulim Shin, and Eunjung Choi. 2019. "An Intrusion Detection Model Based on a Convolutional Neural Network." Journal of Multimedia Information System 6 (4): 165–72. https://doi.org/10.33851/JMIS.2019.6.4.165.

[15] – Dertat, Arden. 2017. "Applied Deep Learning - Part 3: Autoencoders." Medium. Towards Data Science. October 3, 2017. https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798.

[16] – Cunha, Andressa A, João B Borges, and Antonio. 2022. "Classification of Botnet Attacks in IoT Using a Convolutional Neural Network," October. https://doi.org/10.1145/3551661.3561374.

[17] – Kambourakis, Georgios, Constantinos Kolias, and Angelos Stavrou. 2017. "The Mirai Botnet and the IoT Zombie Armies." *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*, October. https://doi.org/10.1109/milcom.2017.8170867.

[18] – Feily, Maryam, Alireza Shahrestani, and Sureswaran Ramadass. 2009. "A Survey of Botnet and Botnet Detection." IEEE Xplore. June 1, 2009. https://doi.org/10.1109/SECURWARE.2009.48.

[19] – Xing, Ying, Hui Shu, Hao Zhao, Dannong Li, and Li Guo. 2021. "Survey on Botnet Detection Techniques: Classification, Methods, and Evaluation." Mathematical Problems in Engineering. April 15, 2021. https://www.hindawi.com/journals/mpe/2021/6640499/.

[20] – "What Is Machine Learning? | Definition, Types, and Examples | SAP Insights." n.d. SAP. https://www.sap.com/products/artificial-intelligence/what-is-machine-learning.html.

[21] – IBM. 2023. “What Is Machine Learning?” Www.ibm.com. IBM. 2023. https://www.ibm.com/topics/machine-learning.

[22] – “Artificial Intelligence: The CNIL Publishes a Set of Resources for Professionals.” n.d. Www.cnil.fr. Accessed June 20, 2023. https://www.cnil.fr/en/artificial-intelligence-cnil-publishes-set-resources-professionals.

[22] – Zupan, J. 1994. “Introduction to Artificial Neural Network (ANN) Methods: What They Are and How to Use Them*.” Semantic Scholar. 1994. https://www.semanticscholar.org/paper/Introduction-to-Artificial-Neural-Network-(ANN)-Are-Zupan/a98fb04c9bdf7c52f53658bc5d8219482bf7d646.

[23] – “What Is Unsupervised Learning? | IBM.” n.d. Www.ibm.com. https://www.ibm.com/topics/unsupervised-learning#:~:text=the%20next%20step-.

[24] – IBM. n.d. “What Is Supervised Learning? | IBM.” Www.ibm.com. https://www.ibm.com/topics/supervised-learning.

[25] – IBM. 2023. “What Is Deep Learning? | IBM.” Www.ibm.com. 2023. https://www.ibm.com/topics/deep-learning.

[26] – IBM. n.d. “What Are Convolutional Neural Networks? | IBM.” Www.ibm.com. https://www.ibm.com/topics/convolutional-neural-networks.