Université de BLIDA 1

Faculté des Sciences

Département d'Informatique

Mémoire de fin d'étude

Option : *Ingénierie des logiciels*

Détection d'intrusion dans l'IoT : Une approche basée sur l'apprentissage profond

---

Réalisé par :

LEMITI Safia Meroua
SASSI Mounia

Membres du jury :

Mr. KAMECHE Abdallah	Président
Mme. MADANI Amina	Examinatrice
Mme. BOUMAHDI Fatima	Promotrice
Mr. REMMIDE Mohamed Abdelkarim	Co-Promoteur

Soutenu le _____

Résumé

L'Internet des Objets (IoT) connecte des milliards de dispositifs à travers le monde, mais leur faible capacité de traitement et leur diversité les rendent vulnérables aux attaques. Ce mémoire étudie la détection d'intrusion dans l'IoT en comparant deux approches majeures de l'intelligence artificielle : l'apprentissage automatique (ML) et l'apprentissage profond (DL).

En utilisant le jeu de données ToN_IoT, plusieurs modèles ont été évalués. Les modèles ML, en particulier XGBoost (accuracy : 75,98 %) et Random Forest (accuracy : 86,71 %, F1-score : 86,17 %), se distinguent par leur efficacité, leur rapidité d'exécution ($<$ 10 s) et leur faible complexité. À l'inverse, les modèles DL comme l'Autoencoder ou le Transformer présentent un coût computationnel élevé (jusqu'à 346 s), malgré des performances élevées (Transformer F1-score : 99,3 %).

Les résultats montrent que les modèles ML sont actuellement les plus adaptés à des environnements IoT contraints. Le travail met en évidence la nécessité d'un compromis entre performance, consommation de ressources et simplicité, et ouvre des perspectives vers des architectures hybrides ou optimisées.

Mots Clés : Détection d'intrusion, Internet des Objets (IoT), Machine Learning, Deep Learning, ToN_IoT, Autoencodeur, Transformer, Random Forest, XGBoost.

Abstract

The Internet of Things (IoT) connects billions of devices worldwide, but their limited processing power and heterogeneity expose them to significant cybersecurity threats. This thesis investigates intrusion detection in IoT by comparing two major artificial intelligence approaches : Machine Learning (ML) and Deep Learning (DL).

Using the ToN_IoT dataset, several models were evaluated. ML models, especially XGBoost (accuracy : 75.98 %) and Random Forest (accuracy : 86.71 %, F1-score : 86.17 %), demonstrated efficiency, fast execution ($<$ 10 s), and low complexity. In contrast, DL models such as the Autoencoder and Transformer had higher computational costs (up to 346 s), despite strong performance (Transformer F1-score : 99.3 %).

The findings suggest that ML models are currently more suitable for resource-constrained IoT environments. This work highlights the importance of balancing performance, resource usage, and simplicity, and opens perspectives for optimized or hybrid IDS architectures.

Keywords : Intrusion Detection, Internet of Things (IoT), Machine Learning, Deep Learning, ToN_IoT, Autoencoder, Transformer, Random Forest, XGBoost.

الملخص

يربط إنترنت الأشياء (IoT) مليارات الأجهزة حول العالم، ولكن قدراتها المحدودة في المعالجة والتخزين والتنوع الكبير بين مكوناتها يجعلها عرضة لهجمات إلكترونية متعددة. يهدف هذا العمل إلى دراسة طرق كشف التسلل في بيئة إنترنت الأشياء من خلال مقارنة بين نهجين رئيسيين في الذكاء الاصطناعي: التعلم الآلي (ML) والتعلم العميق (DL).

تم تقييم عدة نماذج باستخدام مجموعة البيانات ToN_IoT. أظهرت نماذج التعلم الآلي، مثل XGBoost (بدقة 75.98 %) و Forest Random (بدقة 86.71 % ودرجة F1 بمقدار 86.17 %)، أداءً جيدًا وسرعة تنفيذ عالية (أقل من 10 ثوانٍ) وبساطة في الاستخدام. في المقابل، ورغم الأداء العالي لنماذج DL مثل Autoencoder و Transformer (درجة F1 للـ Transformer: 99.3 %)، إلا أنها تستهلك موارد حسابية كبيرة (حتى 346 ثانية).

تشير النتائج إلى أن نماذج التعلم الآلي تعد الخيار الأنسب حاليًا في بيئات IoT ذات الموارد المحدودة. كما يبرز هذا العمل أهمية إيجاد توازن بين الأداء والاستهلاك والبساطة، ويفتح آفاقًا مستقبلية نحو نماذج هجينة أو محسّنة.

الكلمات المفتاحية: كشف التسلل، إنترنت الأشياء، التعلم الآلي، التعلم العميق، ToN_IoT، Autoencoder، Random Transformer، Forest، XGBoost.

Remerciements

Nous tenons à exprimer toute notre reconnaissance à nos encadrants, Madame Fatima BOUMAHDI et Monsieur Abdelkarim REMMIDE. Nous les remercions sincèrement pour leur accompagnement, leurs conseils avisés et leur soutien tout au long de ce travail de recherche.

Nous adressons également nos plus profonds remerciements à nos très chers parents : Mohamed Chérif LEMITI, Hanya BENCHERCHALI, Farid SASSI et Djamila TALEB, pour leur amour, leur patience et leur soutien indéfectible.

Enfin, nous remercions chaleureusement nos sœurs Assia LEMITI, Anfel LEMITI, Iméne LEMITI et Lilia SASSI pour leurs encouragements constants, ainsi que nos neveux Aden ACHOUR, Léa ACHOUR et Ghaith pour leur tendresse et leur inspiration.

Table des matières

Table des figures

Liste des tableaux

Introduction Générale

Grâce à l'Internet des Objets (IoT), des milliards de dispositifs physiques sont désormais interconnectés dans des domaines aussi variés que la santé, l'industrie, les transports, l'agriculture et la domotique. Cette transformation technologique vise à automatiser, surveiller et contrôler des systèmes physiques à l'aide d'objets intelligents capables de collecter, traiter et transmettre des données en temps réel. Toutefois, cette expansion rapide expose les réseaux IoT à un large éventail de cybermenaces, en raison des capacités limitées des objets connectés en termes de traitement, de stockage et d'autonomie énergétique.

Plusieurs chercheurs ont proposé des solutions pour détecter les intrusions dans les environnements IoT en s'appuyant sur des techniques d'apprentissage automatique (Machine Learning). Par exemple, Almohaimeed et al. [1] et Khan et al. [2] ont présenté des analyses approfondies des méthodes ML appliquées à l'IoT, mettant en évidence des approches telles que SVM, Random Forest et KNN. Bien que ces méthodes soient simples à implémenter et offrent de bons résultats dans certains cas, elles restent limitées face à des attaques nouvelles ou complexes, et nécessitent souvent un réglage fin des paramètres.

D'autres études se sont tournées vers l'apprentissage profond (Deep Learning), en raison de sa capacité à extraire automatiquement des caractéristiques pertinentes à partir de grandes quantités de données. Ferrag et al. [3] ainsi que Alrashdi et al. [4] ont examiné l'utilisation des CNN, RNN et LSTM dans les IDS pour l'IoT. Susilo et al. [5] ont, quant à eux, proposé une approche hybride combinant plusieurs algorithmes de DL pour traiter des attaques complexes. Ces solutions ont montré des performances prometteuses, mais leur déploiement dans des dispositifs IoT reste limité à cause de leur consommation importante de ressources.

Par ailleurs, certaines recherches plus récentes se sont concentrées sur la robustesse des IDS face aux attaques adversariales. Hashemi et Keller [6] ont proposé une méthode pour améliorer la résistance des IDS aux exemples malveillants générés par des attaquants, tandis que Tafreshian et Zhang [7] ont conçu un cadre défensif spécifiquement destiné à contrer ce type d'attaques dans les systèmes IDS basés sur le ML. Enfin, Satilmis et al. [8] ont mené une revue systématique sur les IDS basés sur les hôtes, montrant la diversité des stratégies existantes mais aussi le manque de solutions légères et généralisables.

Malgré la diversité des approches, il existe peu d'études comparatives rigoureuses prenant en compte à la fois la précision de détection, la légèreté des modèles, leur robustesse face à des attaques avancées, et leur compatibilité avec les contraintes des objets connectés. Cela justifie pleinement l'intérêt d'une nouvelle étude comparative ciblée sur ces critères dans un contexte IoT réaliste.

Dans ce cadre, l'augmentation du nombre de dispositifs connectés améliore la surface d'attaque des systèmes IoT, les rendant ainsi plus susceptibles aux cybermenaces. En 2025, on estime à plus de 30 milliards le nombre de dispositifs IoT actifs dans le monde, dont une majorité présente des vulnérabilités exploitables[9]. Des attaques telles que celle du botnet *Mirai* ont démontré que même des objets simples comme des caméras IP peuvent être compromis et utilisés dans des attaques massives. Cette réalité met en évidence le besoin urgent de développer des solutions de sécurité adaptées aux spécificités de l'IoT, notamment des systèmes de détection d'intrusion (IDS) à la fois performants et légers. Dès lors, une question centrale se pose : **Comment adapter les systèmes de détection d'intrusion aux contraintes spécifiques des objets connectés, tout en assurant un bon équilibre entre précision, rapidité et légèreté ?**

Ce mémoire a pour objectif de proposer une étude comparative entre plusieurs modèles d'apprentissage automatique et d'apprentissage profond, notamment *Random Forest*, *Autoencoder* et *Transformer*, appliqués au dataset **ToN_IoT**. L'étude vise à évaluer les performances de ces modèles en termes de précision (accuracy, recall, F1-score), mais aussi leur faisabilité d'implémentation dans des environnements IoT contraints en ressources. Ce mémoire s'inscrit dans cette perspective et présente donc les contributions suivantes :

- Une **étude comparative rigoureuse** entre plusieurs modèles ML et DL dans le contexte IoT.
- L'utilisation du dataset **ToN_IoT**, représentatif d'un environnement connecté réaliste.
- L'analyse des **coûts computationnels** (temps d'entraînement et de prédiction).

Le présent mémoire est structuré en trois chapitres :

- **Chapitre 1 : État de l'art des systèmes de détection d'intrusion**
  Ce chapitre explore les bases théoriques des systèmes de détection d'intrusion (IDS), en mettant l'accent sur les défis spécifiques à l'IoT et aux technologies de Machine Learning (ML) et Deep Learning (DL). Il présente les différentes méthodes existantes pour la détection d'intrusions, les technologies sous-jacentes et les approches actuelles en matière de sécurité réseau.
- **Chapitre 2 : Conception et mise en œuvre d'un système de détection**
  Ce chapitre présente l'architecture générale du système développé pour la détection d'intrusions, en détaillant chaque étape du processus, depuis le choix du jeu de données jusqu'à la modélisation via Machine Learning et Deep Learning.
- **Chapitre 3 : Implémentation et expérimentation**
  Ce chapitre évalue les modèles développés dans le chapitre précédent, en comparant leurs performances à l'aide de métriques classiques de machine learning. L'objectif est de tester l'efficacité des systèmes dans un environnement réel et de discuter des résultats obtenus pour chaque modèle.
- **Conclusion générale**
  Une synthèse des résultats obtenus tout au long du mémoire, des défis rencontrés, et des perspectives de recherche future dans le domaine de la détection d'intrusions, en particulier dans le contexte de l'IoT et de l'intelligence artificielle.

Chapitre 1

État de l'art des systèmes de détection d'intrusion

1.1 Introduction

L'Internet des Objets (IoT) connaît une croissance rapide, connectant de nombreux appareils à divers secteurs [2]. Cette expansion s'accompagne de vulnérabilités, exposant les systèmes IoT aux cyberattaques [4]. Les systèmes de détection d'intrusion (IDS) jouent un rôle clé dans l'identification et la prévention des menaces [3]. Cependant, les méthodes classiques montrent des limites face à la complexité des attaques et aux contraintes de l'IoT [1]. L'apprentissage profond apparaît comme une approche prometteuse pour améliorer la détection et la réponse aux intrusions [5].

Ce chapitre explore les IDS, les défis de sécurité de l'IoT et l'apport du deep learning dans ce domaine.

1.2 Aperçu des systèmes de détection d'intrusion (IDS)

Les systèmes de détection d'intrusion (IDS) jouent un rôle crucial dans la sécurité des réseaux en identifiant et en alertant sur les comportements suspects. Cette section présente les différentes classifications des IDS selon leur méthode de détection et leur emplacement de déploiement.

1.2.1 Classification des IDS par méthode de détection

Les systèmes de détection d'intrusion peuvent être classés selon la manière dont ils identifient les attaques :

1. **IDS basés sur les signatures :** Cette méthode compare le trafic ou les activités du système avec une base de signatures connues d'attaques. Elle est rapide et efficace contre des attaques connues, mais incapable de détecter des attaques nouvelles ou modifiées [3].

2. **IDS basés sur les anomalies :** Cette approche définit un comportement "normal" du réseau ou du système, et signale toute déviation comme suspecte. Elle permet de détecter des attaques inconnues mais peut générer un taux élevé de faux positifs [6].

3. **IDS hybrides :** Les systèmes hybrides combinent les deux méthodes précédentes pour améliorer la précision et réduire les limitations de chacune [5].

1.2.2 Classification des IDS par emplacement de déploiement

Les IDS peuvent aussi être classés selon leur position dans l'architecture du réseau :

1. **NIDS (Network-based Intrusion Detection System) :** Déployé sur le réseau, il surveille le trafic entrant et sortant afin d'identifier les attaques en transit. Il est utile pour détecter des menaces globales [7].

2. **HIDS (Host-based Intrusion Detection System) :** Installé directement sur un hôte (serveur ou terminal), il analyse les fichiers systèmes, les journaux et les comportements locaux pour détecter les intrusions ciblant une machine spécifique [8].

3. **DIDS (Distributed Intrusion Detection System) :** Les IDS distribués collectent des données à partir de plusieurs hôtes et points du réseau, afin d'analyser des comportements complexes dans des environnements distribués [10].

Bien que ces classifications permettent de structurer les IDS selon leur mode de détection et leur positionnement dans l'infrastructure, elles ne constituent pas à elles seules des solutions complètes face aux attaques modernes, notamment dans les environnements IoT. L'évolution des cybermenaces et la complexité croissante des réseaux exigent des approches plus flexibles et performantes, telles que celles basées sur l'intelligence artificielle et l'apprentissage profond, qui seront explorées dans les sections suivantes.

1.3 Menaces et défis de sécurité dans l'IoT

L'Internet des Objets (IoT) connaît une adoption rapide dans divers secteurs, mais cette expansion s'accompagne de nombreuses vulnérabilités, rendant les dispositifs connectés particulièrement exposés aux cyberattaques.

1.3.1 Principales menaces de sécurité

- **Attaques par déni de service distribué (DDoS)** : L'une des attaques les plus notoires contre l'IoT est le botnet Mirai, qui a exploité des objets connectés mal sécurisés pour générer des attaques DDoS massives contre des infrastructures critiques [11].
- **Attaques Man-in-the-Middle (MitM)** : L'absence de chiffrement robuste et de mécanismes d'authentification permet aux attaquants d'intercepter et de modifier les communications entre appareils [12].
- **Exploitation de vulnérabilités logicielles** : De nombreux appareils IoT ne reçoivent pas régulièrement de mises à jour de sécurité, ce qui les rend vulnérables aux exploits et aux logiciels malveillants [13].
- **Accès non autorisé et prises de contrôle** : Les cybercriminels peuvent compromettre des dispositifs IoT mal protégés pour les utiliser comme points d'entrée vers des réseaux plus vastes [14].

1.3.2 Défis de sécurité dans l'IoT

- **Hétérogénéité des dispositifs** : L'IoT regroupe des équipements aux architectures variées, incluant différents types de processeurs, systèmes d'exploitation et protocoles de communication, compliquant ainsi l'adoption de solutions uniformes [15].
- **Limitations en ressources** : Contrairement aux ordinateurs classiques, de nombreux objets connectés disposent de capacités de calcul, de stockage et d'énergie limitées, ce qui restreint la mise en place de mécanismes de sécurité avancés [16].
- **Gestion des mises à jour et correctifs** : L'absence de politiques de mise à jour automatique expose de nombreux appareils à des vulnérabilités connues, souvent exploitées par les attaquants [17].

Face à ces menaces et défis, l'amélioration des mécanismes de détection d'intrusion apparaît comme une solution clé pour renforcer la sécurité des environnements IoT.

La figure suivante illustre la hiérarchie des domaines de l'intelligence artificielle, du machine learning et du deep learning, ainsi que les algorithmes les plus couramment utilisés dans chacun de ces niveaux. Cette représentation, adaptée de Baduge et al. **baduge2022ai**, permet de situer les techniques présentées dans ce mémoire dans un contexte plus large de l'IA appliquée à la cybersécurité et à l'IoT.

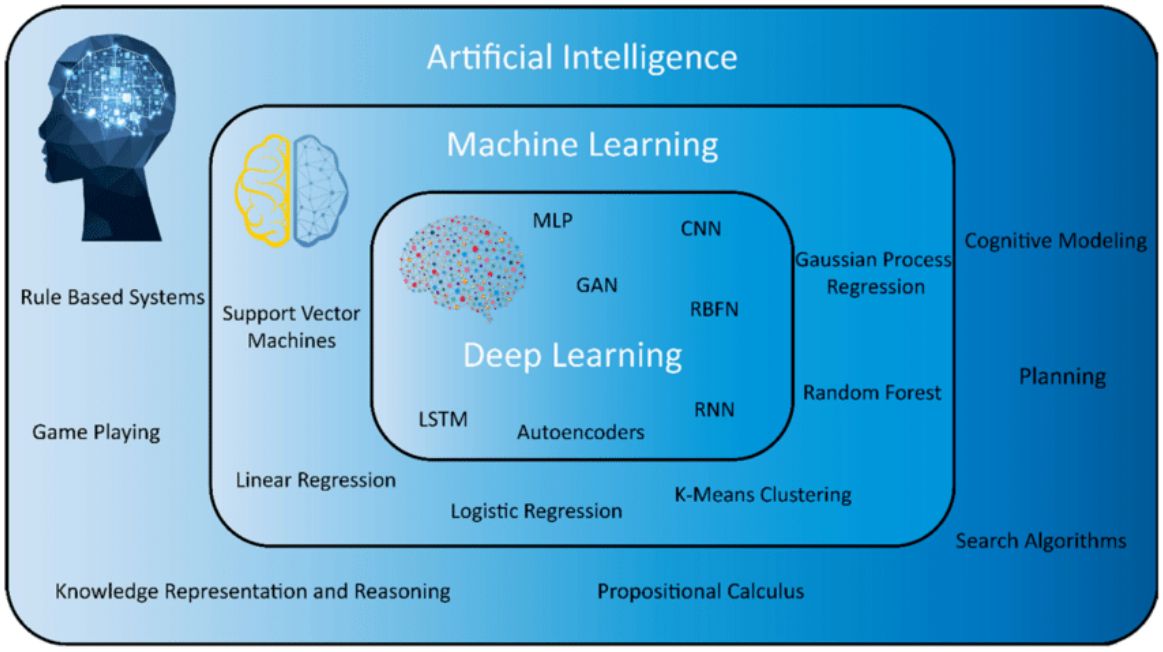


FIG. 1.1 : Hiérarchie des domaines de l'IA et des techniques utilisées **baduge2022ai**

1.4 Les différentes techniques de détection d'intrusion

1.4.1 Solution basée sur le Machine Learning (ML)

Le Machine Learning (ML) offre une approche efficace pour la détection des intrusions dans les réseaux IoT, grâce à sa capacité à apprendre à partir des données et à identifier automatiquement des schémas anormaux. Dans cette solution, des algorithmes sont utilisés pour classer les comportements réseau comme normaux ou malveillants. Ces modèles sont entraînés sur des jeux de données représentatifs (comme KDD99, NSL-KDD ou CIC-IDS2017), permettant une détection rapide et souvent en temps réel des intrusions connues. De plus, certaines méthodes ML peuvent détecter des anomalies même sans connaissance préalable de la menace (approche semi-supervisée ou non supervisée).

Algorithmes de machine learning utilisés pour les IDS :

- **Arbre de décision (Decision Tree – DT)** : Un arbre de décision consiste en un ensemble d'options sous forme d'arbre graphique, où les décisions se trouvent à l'extrémité des branches. Les systèmes de détection d'intrusion utilisent des algorithmes d'apprentissage automatique comme l'arbre de décision pour identifier les attaques dans les réseaux IoT. Cependant, bien que cette approche offre de meilleures performances, elle souffre d'une complexité computationnelle élevée en raison de l'influence des variables continues dans les données du réseau, ce qui peut nécessiter des techniques supplémentaires pour maintenir une haute précision [18].

- **Forêt aléatoire (Random Forest – RF)** : C'est un algorithme basé sur un ensemble d'arbres de décision, offrant robustesse et haute précision. Dans cette étude, les auteurs ont proposé un mécanisme de détection basé sur cet algorithme pour identifier les attaques ciblant les appareils IoT à ressources limitées. Les résultats ont montré que l'algorithme de la forêt aléatoire offre une haute précision dans la détection des anomalies tout en maintenant une utilisation efficace des ressources, ce qui le rend adapté aux systèmes IoT à ressources limitées [19].

- **Machine à vecteurs de support (Support Vector Machine – SVM)** : C'est un algorithme utilisé pour la classification et la régression, qui cherche à maximiser la marge entre les différentes classes en trouvant un hyperplan optimal dans un espace à plusieurs dimensions. Lors de son utilisation, il a atteint une précision de 75 % et un rappel de 79 % pour la classification des activités malveillantes. Cependant, il peut échouer à détecter certains cas, ce qui entraîne un taux de rappel plus faible [20].

- **K-plus proches voisins (K-Nearest Neighbors – KNN)** : Cet algorithme cherche à séparer les données en maximisant la marge entre les classes. Lors de l'utilisation de cet algorithme, il a été observé qu'il consommait plus d'énergie que l'arbre de décision et nécessitait plus d'espace mémoire, ce qui pourrait être un obstacle pour les appareils à ressources limitées. Par conséquent, il pourrait être moins adapté aux systèmes IoT avec des contraintes de ressources [21].

- **Naive Bayes (NB)** : L'algorithme Naive Bayes repose sur le principe des probabilités et suppose l'indépendance des caractéristiques entre elles, une hypothèse qui n'est pas toujours précise. Dans une étude, cette approche a atteint une précision de 97 % lors de la classification des données. Ensuite, la méthode de l'enveloppe elliptique a été employée pour détecter les anomalies [22].

- **Réseaux de neurones artificiels (Artificial Neural Networks – ANN)** : Les réseaux de neurones sont utilisés en raison de leur capacité à reconnaître des motifs complexes dans les données. Dans cette recherche, un modèle de réseau neuronal convolutionnel basé sur l'attention (ABCNN) a été proposé pour la détection des intrusions dans les réseaux IoT. Le modèle a montré une amélioration significative de la précision par rapport aux méthodes traditionnelles, grâce à sa capacité à se concentrer sur les caractéristiques essentielles des données [23]. Dans une autre étude, un système de détection d'intrusions en temps réel basé sur un réseau neuronal profond a été proposé pour identifier les paquets malveillants. Les résultats ont montré que le modèle atteint une haute précision dans la détection des intrusions, surpassant de nombreuses méthodes traditionnelles [24].

Cependant, l'efficacité de ces modèles dépend fortement de la qualité des données d'entraînement et de la capacité du modèle à généraliser. Par ailleurs, les systèmes IoT posent des contraintes importantes (ressources limitées, hétérogénéité des appareils), ce qui impose une adaptation des algorithmes classiques pour une mise en œuvre légère et efficace.

1.4.2 Solutions basées sur le Deep Learning (DL)

Les techniques de Deep Learning (DL) ont significativement amélioré les performances des systèmes de détection d'intrusion (IDS) dans les environnements IoT, en raison de leur capacité à extraire et apprendre automatiquement des schémas complexes à partir de volumes massifs de données. Plusieurs modèles d'apprentissage profond ont été exploités pour détecter efficacement des attaques variées dans ces environnements.

- **Réseaux de Neurones Convolutifs (CNN)** : Les CNN sont réputés pour leur aptitude à extraire des caractéristiques locales et spatiales à partir de données structurées sous forme de matrices, telles que les paquets réseau. Leur architecture leur permet de classer efficacement différents types d'attaques dans les réseaux IoT [25].

- **Réseaux Neuronaux Récurrents (RNN), LSTM et GRU** : Ces réseaux sont spécifiquement conçus pour traiter des séquences temporelles, ce qui les rend adaptés à la détection d'anomalies dans les flux de trafic IoT, où la chronologie des événements joue un

rôle essentiel. Les variantes LSTM (Long Short-Term Memory) et GRU (Gated Recurrent Unit) améliorent encore cette capacité en mémorisant les dépendances à long terme [26].

- **Autoencodeurs (AE)** : Les autoencodeurs constituent une approche efficace pour la détection non supervisée d'intrusions. En apprenant à reconstruire fidèlement les données considérées comme normales, toute divergence significative lors de la reconstruction est interprétée comme une anomalie potentielle [27].
- **Deep Belief Networks (DBN)** : Les DBN combinent des couches de Boltzmann restreintes afin de modéliser des distributions de données complexes. Ces réseaux sont utilisés non seulement pour la réduction de la dimensionnalité, mais également pour assurer une classification fiable et précise des flux réseau [28].
- **Transformers** : Initialement conçue pour le traitement du langage naturel, l'architecture Transformer a récemment été adaptée à la détection d'intrusions dans les réseaux IoT. Grâce à son mécanisme d'attention, cette approche est capable de modéliser efficacement les dépendances à long terme entre les événements, améliorant ainsi la détection d'anomalies complexes [29].
- **XGBoost** : Bien qu'il relève du Machine Learning (ML), XGBoost est souvent intégré dans des pipelines avancés grâce à son efficacité en classification, en particulier dans les jeux de données déséquilibrés comme ceux rencontrés en IoT. Il a démontré des performances élevées dans plusieurs études récentes, y compris dans [2].

Toutes les approches que nous avons examinées ont montré des résultats prometteurs concernant l'utilisation du Deep Learning (DL) et du Machine Learning (ML) dans la détection d'intrusions. Cependant, bien que ces méthodes aient démontré leur efficacité dans de nombreux scénarios, elles présentent encore des limitations importantes.

1.5 Limites des approches IDS actuelles dans un environnement IoT

Les systèmes de détection d'intrusions (IDS) sont des éléments essentiels pour assurer la sécurité des réseaux, en particulier dans les environnements IoT. Cependant, leur mise en œuvre reste complexe à cause des contraintes spécifiques à l'IoT, comme la diversité des dispositifs, la limitation des ressources, et la grande variété des scénarios d'attaque.

1.5.1 Taux élevés de faux positifs et de faux négatifs

Les IDS peuvent générer un grand nombre de faux positifs, signalant des comportements normaux comme des attaques, ce qui réduit la confiance des utilisateurs et surcharge inutilement les administrateurs réseau [18]. Par ailleurs, les faux négatifs — lorsque des intrusions réelles ne sont pas détectées — constituent une faille critique, car ils laissent des attaques passer inaperçues [18].

1.5.2 Limitation dans la détection des attaques inconnues (Zero-day)

Les attaques Zero-day exploitent des vulnérabilités jusqu'alors inconnues, ce qui rend inefficaces les systèmes IDS reposant sur des bases de signatures statiques [30]. Cette faiblesse

empêche les IDS de réagir rapidement face aux nouvelles menaces, particulièrement dans des environnements dynamiques comme l'IoT [31].

1.5.3 Problèmes de performance et de scalabilité

La croissance rapide du volume de données et du nombre d'objets connectés engendre des charges de traitement considérables, que les IDS classiques peinent à gérer efficacement [32]. Les approches centralisées sont souvent inadaptées face à la nature distribuée et évolutive de l'IoT, rendant la détection d'intrusion lente et inefficace [33]. De plus, les méthodes basées sur l'intelligence artificielle exigent des ressources informatiques élevées pour fonctionner de manière optimale, ce qui n'est souvent pas compatible avec les dispositifs IoT à faible puissance [34].

Tab. 1.1 : Résumé des approches d'IDS pour l'IoT avec Deep Learning

Art	**Technique**	**Méthode**	**Datasets**	**Résultats**	**Limitations**
[2]	Apprentissage automatique et profond	SVM, RF, KNN	NSL-KDD, CICIDS 2017, IoT-IDS2017, Bot-IoT, UNSW-NB15	Précision : 98% pour RF, 97% pour SVM. Taux de détection : 95% pour IoT-IDS2017.	Surapprentissage, manque de diversité des datasets, lenteur des modèles complexes
[4]	Apprentissage profond	LSTM	CIC-IoT2022	Précision de 99,85%	Absence d'une plateforme de test réelle, diversité des attaques, complexité du trafic IoT
[3]	Apprentissage profond	CNN, RNN, Autoencodeurs (AE), DNN, DBN	KDDCup-99, NSL-KDD, BoT-IoT, CICIDS-2017	Précision souvent > 95% (classification binaire et multi-classes)	Jeux de données obsolètes, déséquilibre des classes, coût computationnel élevé, vulnérabilité aux attaques adversariales
[1]	Approche hybride ML et DL	RF, SVM, KNN, DT ; CNN, LSTM, RNN, Autoencodeurs (AE)	NSL-KDD, CICIDS2017, UNSW-NB15, BoT-IoT	Précision élevée pour DL (> 95%), RF et CNN très performants, réduction des faux positifs	Surapprentissage possible, peu de datasets IoT réels, ressources limitées sur capteurs

Art	Technique	Méthode	Datasets	Résultats	Limitations
[5]	Apprentissage profond	Combinaison de CNN et LSTM	CICIDS2017	Précision de 99,57 %, très bonnes performances pour la détection de plusieurs types d'attaques	Coût computationnel élevé pour l'entraînement, Difficulté de déploiement sur des appareils IoT à ressources limitées
[6]	Apprentissage profond	CNN, RNN	NSL-KDD, CICIDS 2017, IoT-23	Précision accrue, meilleure détection des attaques inconnues, réduction des faux positifs	Complexité computationnelle, manque de données étiquetées, difficulté à détecter des attaques évolutives
[7]	Apprentissage profond	Autoencodeurs	NSL-KDD	Amélioration de la détection du trafic malveillant jusqu'à 29% en conditions normales, 45% avec exemples adverses	Vulnérabilité aux attaques adverses, complexité computationnelle, besoin de données représentatives
[8]	Apprentissage automatique et profond	Méthodes par signature et anomalies	NSL-KDD	Efficacité prometteuse avec taux de détection élevés	Déséquilibre des données, performances variables selon les environnements, besoin d'adaptation contextuelle
[10]	Apprentissage automatique	SVM, DT	NSL-KDD	Nécessité d'approches collaboratives et distribuées	Complexité de mise en œuvre, absence de standardisation, défis de coordination
[11]	Apprentissage automatique	SVM, DT	CAIDA	Mirai a infecté 600 000 appareils et causé 15 000 attaques DDoS entre 2016-2017	Étude descriptive sans solution proposée

Art	Technique	Méthode	Datasets	Résultats	Limitations
[12]	Apprentissage automatique	SVM, DT, KNN	Dataset expérimental (capteurs simulés)	Précision de 95% dans la détection MitM	Non généralisable à d'autres types d'attaques
[13]	Apprentissage automatique	SVM, DT, KNN	Shodan, Censys, ZMap	70% des vulnérabilités identifiées comme exploitables	Nécessité d'une couverture plus large des vulnérabilités
[14]	Apprentissage automatique et Apprentissage profond	SVM, DT, RF, KNN, NB ,CNN, RNN, AE	NSL-KDD, Bot-IoT	Bonne cartographie des défis de sécurité	Manque de solutions pratiques et évolutives
[15]	Apprentissage automatique et Apprentissage profond	SVM, CNN, LSTM	Références à Bot-IoT, UNSW-NB15	Présente les défis majeurs pour la sécurité IoT	Manque de solutions pratiques et évolutives
[16]	Apprentissage automatique	SVM, DT, KNN	NSL-KDD	Amélioration des mécanismes d'authentification	Complexité de mise en œuvre et gestion des clés
[17]	Apprentissage automatique	SVM	NSL-KDD	Absence de mise à jour causant des compromissions majeures	Aucune solution technique proposée
[18]	Apprentissage profond	ANN	NSL-KDD	Précision de 91% dans la détection	Limité face à de nouvelles attaques
[19]	Apprentissage automatique	RF	UNSW-NB15	Précision de 85%	Difficile à adapter à de nouveaux appareils
[20]	Apprentissage profond	CNN	NSL-KDD	Précision de 99.2%	Performances variables selon la complexité du réseau
[21]	Apprentissage automatique	RF, SVM, KNN, NB, DT	UNSW-NB15, CICIDS2017	98% de précision pour les algorithmes supervisés	Précision affectée par les déséquilibres dans les données
[22]	Apprentissage automatique	Random Forest	NSL-KDD	Précision de 94% en détection d'anomalies	Complexité pour sélectionner les bonnes caractéristiques
[23]	Apprentissage profond	CNN	NSL-KDD	Précision de 96% en détection	Sensibilité aux faux positifs
[24]	Apprentissage profond	DNN	Bot-IoT	99.3% de précision	Nécessite des données d'entraînement spécifiques

Art	Technique	Méthode	Datasets	Résultats	Limitations
[25]	Apprentissage profond	CNN	UNSW-NB15	97% de précision dans les maisons intelligentes	Besoin de ressources computationnelles importantes
[26]	Apprentissage profond	LSTM	CICIDS2017	97.91% de précision	Temps d'entraînement important
[27]	Apprentissage profond	AE	BoT-IoT	98.5% de précision	Requiert des réglages fins
[28]	Apprentissage automatique	SVM	KDD Cup 99	98.6% de précision	Moins efficace contre les attaques inconnues
[29]	Apprentissage profond	Transformer	TON_IoT	99.4% de précision	Très coûteux en ressources
[30]	Apprentissage automatique	SVM, KNN, RF	KDD Cup 99	70% des vulnérabilités identifiées	Besoin de données réelles sur les attaques IoT
[31]	Apprentissage automatique	SVM, KNN, RF	KDD Cup 99	Recommandations pour renforcer la sécurité IoT	Nécessité de solutions plus pratiques pour l'IoT
[32]	Apprentissage automatique	SVM, KNN, RF	NSL-KDD, CICIDS	85% de précision pour les datasets utilisés	Dépendance à des datasets spécifiques
[33]	Apprentissage profond	CNN, RNN, LSTM	CICIDS2017, KDD99, UNSW-NB15	Précision de 96–99% pour différents modèles DL	Manque de validation empirique en conditions réelles
[34]	Apprentissage profond	CNN	NSL-KDD, UNSW-NB15	Précision de 98.4% pour le modèle CNN sur NSL-KDD	Risque de surapprentissage sur petits datasets
[35]	Apprentissage automatique et Apprentissage profond	CNN, LSTM	CICIDS2017, TON_IoT	Précision entre 95–99% selon le modèle utilisé	Manque de tests avec des attaques réelles récentes
[36]	Apprentissage profond	DNN, RNN, CNN, RBM, DBN, DBM, AE	CICIDS2017, UNSW-NB15	Précision > 97% pour plusieurs techniques DL	Pas de tests sur des réseaux IoT réels
[37]	Apprentissage profond	LSTM, CNN, AE	NSL-KDD, BoT-IoT	Précision > 98% pour les modèles LSTM et CNN	Pas d'évaluation sur des données plus récentes

Art	Technique	Méthode	Datasets	Résultats	Limitations
[38]	Apprentissage profond	CNN, LSTM	BoT-IoT	Précision > 99% avec CNN + LSTM combiné	Consommation élevée de ressources et difficulté à s'adapter aux nouvelles menaces
[39]	Apprentissage profond	CNN, LSTM, AE	BoT-IoT	Précision élevée (98–99%) pour les modèles DL	Limité aux environnements IoT spécifiques
[40]	Apprentissage automatique	SVM, RF	IoT-IDS, KDD99	Précision non spécifiée, mais rapports de bons résultats	Manque de comparaisons avec d'autres techniques DL
[41]	Apprentissage profond	CNN, RNN, AE	UNSW-NB15, IoT-IDS	Précision > 96% sur les datasets IoT	Pas de tests en conditions réelles ou avec des attaques inconnues
[42]	Apprentissage profond	CNN, DNN	IoT-IDS, NSL-KDD	Précision de 97% avec un modèle CNN	Ne couvre pas toutes les plateformes IoT
[43]	Apprentissage automatique et Apprentissage profond	CNN, RNN, LSTM	IoT-IDS, KDD99, NSL-KDD	Précision jusqu'à 99% avec les meilleures techniques DL	Manque de diversité dans les attaques testées

L'analyse des travaux existants met en lumière l'intérêt croissant pour les approches d'apprentissage automatique et d'apprentissage profond dans le domaine de la détection d'intrusion appliquée à l'IoT. Les méthodes d'apprentissage profond montrent une forte capacité à identifier des modèles complexes dans les données, tandis que les techniques d'apprentissage automatique sont souvent appréciées pour leur simplicité d'implémentation et leur rapidité. Toutefois, chaque approche présente ses propres limites : manque de tests en conditions réelles, difficultés d'adaptation aux environnements hétérogènes, ou encore faible diversité des attaques simulées. À travers cette étude comparative, nous cherchons à mettre en évidence les forces et les faiblesses de chaque catégorie de techniques, dans le but de mieux comprendre leur efficacité et leur adéquation aux spécificités des réseaux IoT.

1.6 Conclusion

Dans ce chapitre, nous avons présenté les systèmes de détection d'intrusion (IDS) en commençant par leur définition, leurs types, et les méthodes classiques utilisées. Nous avons ensuite exposé les particularités des environnements IoT ainsi que les défis qu'ils posent à l'intégration des IDS. Enfin, nous avons mis en évidence l'apport du Machine Learning et du Deep Learning dans l'amélioration de ces systèmes. Le chapitre suivant sera consacré à l'étude des approches d'apprentissage automatique et profond, en vue de leur application à la détection d'intrusions dans des environnements complexes.

Chapitre 2

Implémentation d'un système de détection d'intrusion basé sur l'apprentissage automatique et profond

2.1 Introduction

Dans ce chapitre, nous présentons la démarche technique adoptée pour concevoir et implémenter un système de détection d'intrusions (IDS) intelligent, spécifiquement adapté aux environnements de l'Internet des Objets (IoT). Notre approche repose sur l'utilisation de techniques d'apprentissage automatique (Machine Learning - ML) et d'apprentissage profond (Deep Learning - DL), appliquées au jeu de données *ToN_IoT*, reconnu pour sa richesse en scénarios réalistes d'attaques et de comportements normaux.

Nous débutons par une description de l'architecture globale du système, articulée autour de plusieurs modules allant de la collecte des données jusqu'à l'entraînement des modèles prédictifs. Ensuite, nous introduisons le dataset *ToN_IoT*, en détaillant ses caractéristiques, sa structure, ainsi que sa pertinence dans le domaine de la cybersécurité IoT.

Par la suite, nous exposons les différentes étapes de prétraitement des données nécessaires pour assurer une exploitation optimale des modèles d'apprentissage, telles que la gestion des valeurs manquantes, l'encodage des variables catégorielles et la normalisation des données.

Enfin, nous présentons les modèles d'apprentissage supervisé et profond que nous avons évalués dans le cadre de ce travail. Nous justifions nos choix algorithmiques et détaillons les configurations utilisées, dans le but d'évaluer leur efficacité à distinguer les comportements normaux des activités malveillantes dans un contexte IoT.

Ce chapitre pose ainsi les bases expérimentales de notre système de détection, qui seront analysées plus en détail dans le chapitre suivant consacré à l'étude des résultats obtenus.

2.2 Architecture globale du système proposé

Cette section décrit l'architecture générale du système de détection d'intrusions appliqué à un environnement IoT, basée sur l'analyse des données issues du dataset **TON_IoT**. Le processus comporte plusieurs étapes, depuis l'acquisition des données jusqu'à la classification à l'aide d'algorithmes de *machine learning* (ML) et de *deep learning* (DL).

Afin de mieux structurer l'analyse, l'architecture est décomposée en cinq modules fonctionnels, détaillés ci-après :

1. **Collecte des données** : Les données sont issues du dataset TON_IoT, simulant un environnement IoT réel avec des sources variées (capteurs, hôtes Windows/Linux, trafic réseau).

2. **Prétraitement des données** : Cette phase comprend le nettoyage, la normalisation, la sélection de caractéristiques pertinentes, ainsi que la division des données en ensembles d'entraînement et de test.

3. **Extraction des caractéristiques** : Les attributs discriminants sont extraits (par exemple : température, bande passante, charge CPU, etc.) afin de représenter efficacement le comportement du système.

4. **Phase de classification** : Plusieurs modèles sont appliqués :

   - **ML** : Random Forest, Decision Tree, etc.
   - **DL** : Réseaux de neurones profonds (RNN), LSTM, etc.

5. **Évaluation et détection** : L'évaluation des modèles s'effectue à l'aide de métriques classiques (précision, rappel, F1-score, matrice de confusion), permettant de détecter les intrusions avec un niveau de confiance élevé.

- Mettre en œuvre un système de détection **intelligent et automatisé** face aux menaces sur les réseaux IoT.
- Comparer les performances entre des modèles **supervisés classiques** et des approches **profondes**.
- Exploiter les spécificités du dataset **TON_IoT** dans la détection de comportements anormaux dans des environnements hétérogènes.

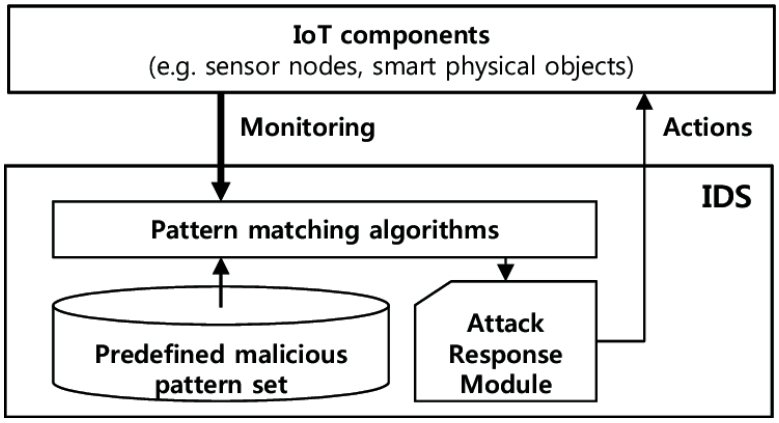


FIG. 2.1 : Architecture fonctionnelle d'un système de détection d'intrusion dans un environnement IoT

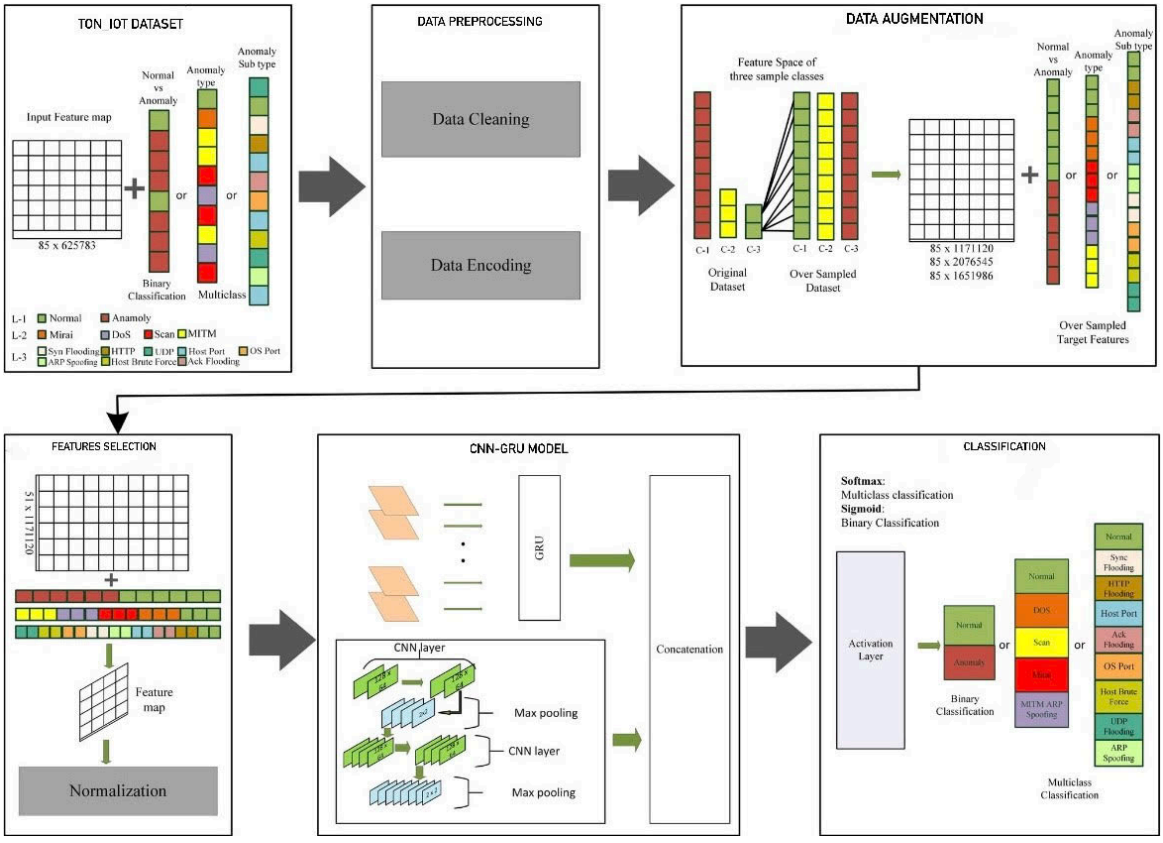


FIG. 2.2 : Architecture globale du système de détection d'intrusions basée sur un modèle hybride CNN-GRU

2.3 Algorithme pour détection d'intrusion avec Autoencodeur et Random Forest

Algorithm 1 : Détection d'intrusion avec Autoencodeur et Random Forest

Input : Jeu de données X, étiquettes partiellement disponibles Y
Output : Classification des connexions réseau (*Normale* ou *Intrusion*)
Data : Données IoT issues du dataset ToN_IoT
/* Prétraitement des données */
Charger le dataset ;
Supprimer les attributs non pertinents ;
Encoder les variables catégorielles (One-Hot / Label Encoding) ;
Normaliser les attributs numériques ;
/* Division des données */
Séparer en ensemble d'entraînement et de test ;
(Optionnel) Créer un sous-ensemble non étiqueté pour apprentissage non supervisé ;
/* Apprentissage non supervisé avec Autoencodeur */
Concevoir un autoencodeur pour apprendre une représentation compressée ;
Entraîner l'autoencodeur sur les données non étiquetées ;
Extraire les embeddings latents depuis la couche cachée ;
/* Apprentissage supervisé avec Random Forest */
Entraîner Random Forest sur les embeddings extraits + étiquettes connues ;
Prédire les classes (normal / attaque) sur les données de test ;
/* Évaluation */
Évaluer le modèle (Accuracy, Precision, Recall, F1-score) ;
/* Détection en production */
foreach *nouvelle connexion réseau* **do**
    Appliquer le même prétraitement ;
    Extraire les caractéristiques avec l'autoencodeur ;
    Classer avec Random Forest ;
    return Classe prédite : normale ou attaque ;

2.4 Présentation du dataset

Dans le cadre de ce travail, nous avons utilisé le dataset **ToN_IoT** (Telemetry of Networked IoT), développé par le laboratoire IoT de l'université UNSW Canberra. Ce dataset est destiné à l'évaluation de solutions de cybersécurité basées sur l'intelligence artificielle, notamment les systèmes de détection d'intrusion dans les environnements IoT.

Nous avons exploité uniquement la partie du dataset correspondant aux données **IoT**. Ces données ont été collectées dans un environnement de test réaliste simulant un réseau IoT, ainsi que des outils de génération d'attaques. Plus de dix types de capteurs ont été utilisés, notamment des capteurs météo.

Les événements capturés incluent à la fois des comportements normaux et des activités malveillantes générées à travers des attaques telles que les attaques par déni de service (DoS), les attaques distribuées (DDoS), et les ransomwares. Ces attaques ciblaient principalement les passerelles IoT et les services associés.

Le dataset a été prétraité pour inclure des caractéristiques exploitables et des étiquettes de classification (*normal* ou *attaque*), permettant son utilisation dans des modèles d'apprentis-

sage supervisé. Les fichiers sont structurés en jeux d'entraînement et de test, accompagnés de statistiques sur les types d'attaques présentes.

La distribution des classes dans le dataset, en termes de nombre d'instances, est présentée sous forme d'histogramme en Figure 2.3. Par ailleurs, la proportion relative des classes est illustrée par le diagramme circulaire en **??**, mettant en évidence le déséquilibre entre les classes normales et malveillantes.

L'utilisation du dataset **ToN_IoT** est libre pour des travaux de recherche académique.

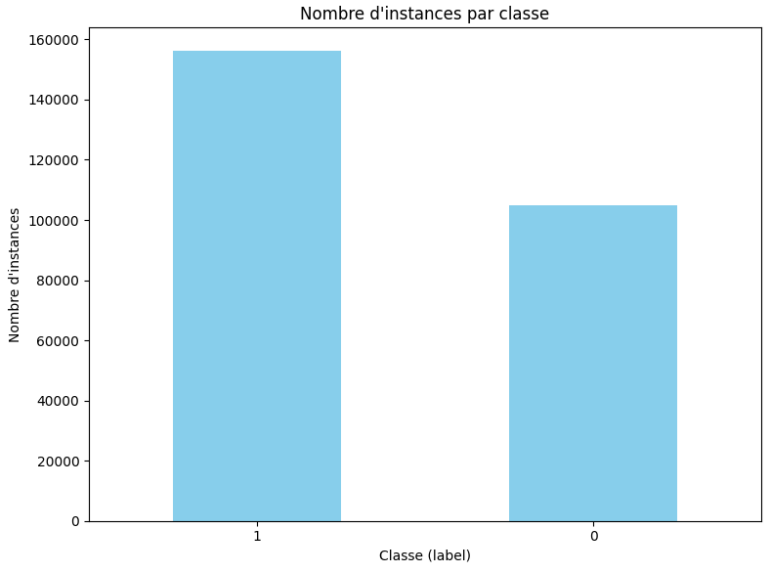


FIG. 2.3 : Nombre d'instances par classe dans le dataset ToN_IoT

2.5 Prétraitement des données

Avant l'application de tout modèle d'apprentissage, les données issues du dataset subissent une série d'opérations de prétraitement essentielles pour garantir la qualité des entrées et optimiser les performances des algorithmes. Ces étapes sont résumées comme suit :

1. **Chargement et fusion des données** : Les fichiers CSV issus de différentes sources (capteurs, logs, trafic réseau) sont fusionnés afin de constituer une base homogène.

2. **Traitement des valeurs manquantes** : Les lignes ou colonnes contenant des valeurs manquantes (*NaN*) sont soit supprimées, soit remplies par des méthodes comme la moyenne ou la médiane, selon le cas.

3. **Encodage des variables catégorielles** : Les attributs non numériques (par exemple, les types de protocoles ou de services) sont encodés à l'aide de techniques telles que le *One-Hot Encoding* ou le *Label Encoding* pour être utilisables par les modèles.

4. **Normalisation des variables numériques** : Les valeurs continues (telles que la bande passante ou l'utilisation CPU) sont mises à l'échelle via des méthodes comme la normalisation min-max ou la standardisation Z-score, afin de garantir une échelle cohérente entre les attributs.

5. **Séparation des données** : Enfin, les données sont divisées en ensembles d'entraînement et de test afin de permettre l'apprentissage des modèles et leur évaluation sur des données inédites.

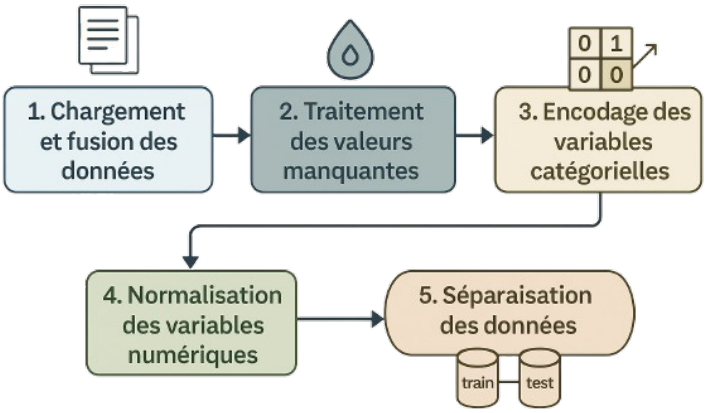


FIG. 2.4 : Étapes de prétraitement des données

2.6 Modèles

Afin de détecter efficacement les intrusions dans le contexte des environnements IoT, nous avons mis en œuvre plusieurs modèles d'apprentissage supervisé sur le dataset **TON_IoT**. Ce jeu de données contient un large éventail de types d'attaques ainsi que des données système issues de plusieurs dispositifs connectés. L'objectif principal est de classer chaque instance comme *attaque* ou *comportement normal*.

2.6.1 Modèles d'Apprentissage Automatique

Les modèles de machine learning utilisés ont été choisis pour leur complémentarité en matière de traitement des données structurées et hétérogènes typiques du dataset TON_IoT :

- **Decision Tree** : utilisé pour sa simplicité d'interprétation et sa capacité à modéliser des relations non linéaires entre les caractéristiques. Il constitue une bonne base pour comprendre la structure des décisions dans la détection d'intrusions.
- **Random Forest** : utilisé pour sa capacité à gérer des attributs numériques et catégoriels variés. Il permet également d'évaluer l'importance des variables, ce qui s'avère utile pour identifier les attributs discriminants dans la détection des attaques réseau.
- **Support Vector Machine (SVM)** : mis en œuvre avec un noyau RBF, ce modèle a permis de capturer des séparations non linéaires entre le trafic malveillant et légitime, malgré le nombre élevé de caractéristiques.
- **K-Nearest Neighbors (KNN)** : testé pour mesurer la capacité des méthodes non paramétriques à classer des comportements malveillants proches. Toutefois, les performances ont été impactées par la dimensionnalité du dataset.
- **Naive Bayes** : utilisé comme modèle de référence, il a permis d'établir un benchmark de base sur les performances, bien que ses hypothèses simplificatrices aient limité sa précision sur des données complexes comme celles du TON_IoT.
- **Logistic Regression** : employé comme modèle linéaire de classification binaire, il a servi à évaluer la séparation linéaire du dataset. Bien que rapide et robuste, ses performances

peuvent être limitées dans des contextes de grande non-linéarité comme celui des intrusions IoT.

- **XGBoost** : appliqué pour son efficacité en termes de vitesse et de performance, notamment sur des jeux de données déséquilibrés comme TON_IoT. Il améliore les performances en minimisant l'erreur de classification via un processus de boosting itératif.

2.6.2 Modèles d'Apprentissage Profond

Compte tenu de la richesse temporelle et de la structure séquentielle de certaines données du dataset (notamment les flux réseau), plusieurs modèles d'apprentissage profond ont été déployés afin d'exploiter les corrélations complexes dans les séries temporelles :

- **Réseau de Neurones Convolutif (CNN)** : utilisé pour extraire automatiquement des motifs locaux dans les vecteurs d'attributs. Le CNN a permis une classification rapide et efficace, notamment dans la détection d'attaques DoS ou de scan.
- **Réseaux de Neurones Récurrents (RNN)** : testés pour leur capacité à traiter des séquences temporelles, bien qu'ils souffrent souvent du problème du gradient qui disparaît, réduisant leur efficacité sur de longues séquences.
- **Réseaux LSTM** : intégrés pour capturer les dépendances temporelles entre les événements successifs, ces modèles ont montré de bonnes performances dans la détection des attaques persistantes réparties dans le temps (ex. : exfiltration lente de données).
- **Gated Recurrent Units (GRU)** : utilisés comme alternative plus légère aux LSTM, les GRU ont montré une bonne capacité de généralisation tout en réduisant le temps d'entraînement et la complexité computationnelle.
- **Autoencodeurs (AE)** : appliqués dans une optique de détection d'anomalies. En reconstruisant les entrées à partir d'un espace latent réduit, les autoencodeurs ont permis d'identifier des comportements rares ou anormaux caractéristiques d'intrusions.
- **Deep Belief Network (DBN)** : implémenté pour une phase de pré-entraînement non supervisé suivie d'une classification supervisée. Le DBN a permis de mieux initialiser les poids des réseaux profonds, favorisant une convergence plus stable sur TON_IoT.
- **Transformer** : intégré pour sa capacité à capturer efficacement les dépendances longues dans les données séquentielles sans nécessiter de traitement itératif, contrairement aux réseaux récurrents. Le modèle a montré de bons résultats dans l'analyse de flux complexes.

2.6.3 Objectif global des modèles

L'objectif commun à tous ces modèles était de prédire avec précision la classe (attaque ou normale) de chaque instance du dataset TON_IoT. Les performances ont été évaluées à l'aide de plusieurs métriques classiques (accuracy, precision, recall, F1-score), ce qui a permis de comparer objectivement l'efficacité de chaque modèle, tant en apprentissage automatique qu'en apprentissage profond.

2.7 Conclusion

Ce chapitre a présenté l'architecture d'un système de détection d'intrusion basé sur l'apprentissage automatique et profond, adapté aux environnements IoT. Nous avons détaillé les étapes clés : de la collecte des données issues du dataset *ToN_IoT* jusqu'à la classification à l'aide de modèles *Machine Learning* (Random Forest, XGBoost…) et *Deep Learning* (CNN, LSTM…). Le prétraitement rigoureux des données a permis d'optimiser la performance des modèles. L'approche proposée vise à détecter efficacement les comportements malveillants dans des réseaux IoT hétérogènes, en comparant différentes techniques d'intelligence artificielle.

Chapitre 3

Test et résultats

3.1 Introduction

Ce chapitre expose les résultats obtenus à l'aide de différents modèles d'apprentissage automatique et d'apprentissage profond appliqués à la détection d'intrusions dans les environnements IoT. L'évaluation repose sur des métriques standards telles que l'Accuracy, la Precision, le Recall et le F1-Score, ainsi que sur le temps d'entraînement et de prédiction, afin d'apprécier la performance et l'efficacité de chaque modèle dans un contexte contraint. Cette analyse comparative permet d'identifier les approches les plus adaptées à la sécurisation des objets connectés face aux menaces croissantes.

3.2 Environnement et Outils de Travail

Dans cette section, nous présentons les outils, langages et bibliothèques utilisés pour développer et entraîner nos modèles d'intrusion detection system (IDS) appliqués à l'Internet des Objets (IoT) via des techniques d'intelligence artificielle (IA).

3.2.1 Environnement de Développement

Les expérimentations ont été réalisées principalement sur **Google Colaboratory** [44], plus couramment appelé *Google Colab*. Il s'agit d'un service gratuit proposé par Google, L'environnement de développement a été intégré avec **Google Drive** pour le stockage et le chargement des données, facilitant ainsi la gestion et le partage des ressources utilisées pendant les expérimentations., qui permet d'exécuter du code Python dans le cloud avec accès aux ressources GPU et CPU.

En complément, les tests locaux ont été effectués sur une machine personnelle avec les caractéristiques suivantes :

- **Processeur** : 2,7 GHz Intel Core i5 dual-core ;
- **Mémoire vive (RAM)** : 8 Go 1867 MHz DDR3 ;
- **Système d'exploitation** : macOS Monterey, version 12.7.6 (64 bits).

3.2.2 Langage de Programmation et Bibliothèques Utilisées

Le langage principal utilisé est **Python 3.6** [45], en raison de sa simplicité, de sa syntaxe claire et de la richesse de son écosystème scientifique.

Le processus de développement et d'entraînement des modèles a mobilisé un ensemble de bibliothèques spécialisées :

- **Scikit-learn** [46] : bibliothèque d'apprentissage automatique offrant une large gamme d'algorithmes de classification, régression et clustering.
- **Keras** [47] : API de haut niveau pour les réseaux de neurones, utilisée pour construire et entraîner les modèles de deep learning.
- **TensorFlow** : moteur de calcul en back-end pour Keras, développé par Google.
- **Flask** [48] : micro-framework web en Python permettant de déployer une application simple pour tester les performances du modèle en production.
- **Numpy, Pandas, Matplotlib** : bibliothèques fondamentales pour le traitement des données, l'analyse statistique et la visualisation graphique.

- **Imbalanced-learn (imblearn)** : utilisée pour gérer les déséquilibres dans les ensembles de données, en particulier avec des techniques comme le sur-échantillonnage SMOTE.

3.3 Résultats des modèles d'apprentissage automatique et profond

Dans cette section, nous présentons les performances obtenues par les modèles d'apprentissage automatique (Machine Learning, ML) et d'apprentissage profond (Deep Learning, DL) appliqués à la détection d'intrusions dans les environnements IoT.

Les évaluations sont effectuées à l'aide des métriques classiques telles que la précision (Accuracy), la précision moyenne (Precision), le rappel (Recall), le score F1 (F1-Score), ainsi que les temps d'entraînement et de prédiction. Ces mesures permettent de quantifier la capacité des modèles à détecter les comportements anormaux, tout en tenant compte des contraintes computationnelles liées aux dispositifs IoT.

L'objectif est d'identifier les approches les plus pertinentes selon les besoins spécifiques du système IoT à sécuriser, en distinguant la précision de détection et la charge computationnelle associée à chaque modèle.

3.3.1 Résultats des modèles d'apprentissage automatique (ML)

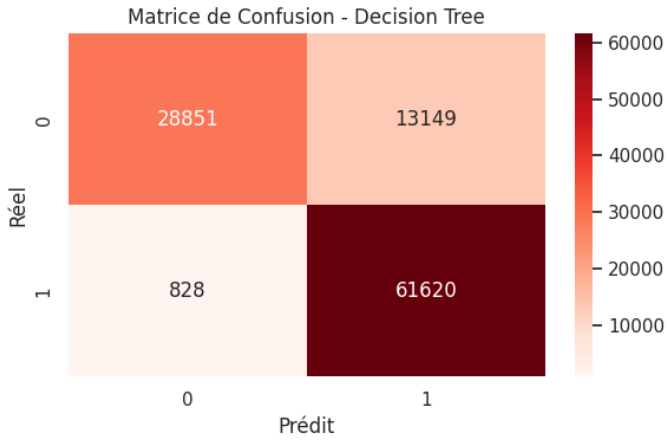
3.3.1.1 Decision Tree

Résultats Les performances du modèle Decision Tree sont présentées dans le tableau ci-dessous :

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
Decision Tree	0.8662	0.8836	0.8662	0.8607	3.999	0.045

Tab. 3.1 : Performances du modèle Decision Tree

Matrice de confusion



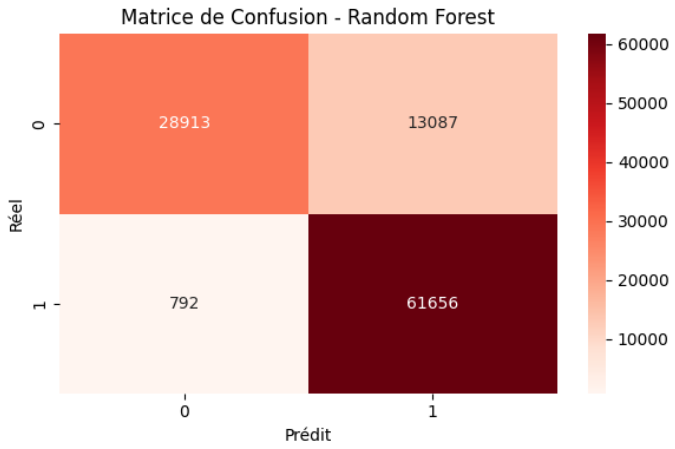
Interprétation Le modèle parvient à bien distinguer les classes normales et anormales avec un bon équilibre entre rappel et précision. Il fait peu d'erreurs de classification. Le temps d'exécution est très rapide, ce qui en fait une solution adaptée aux environnements IoT où la réactivité est primordiale.

3.3.1.2 Random Forest

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
Random Forest	0.8671	0.8846	0.8671	0.8617	112.935	2.739

Tab. 3.2 : Performances du modèle Random Forest

Matrice de confusion



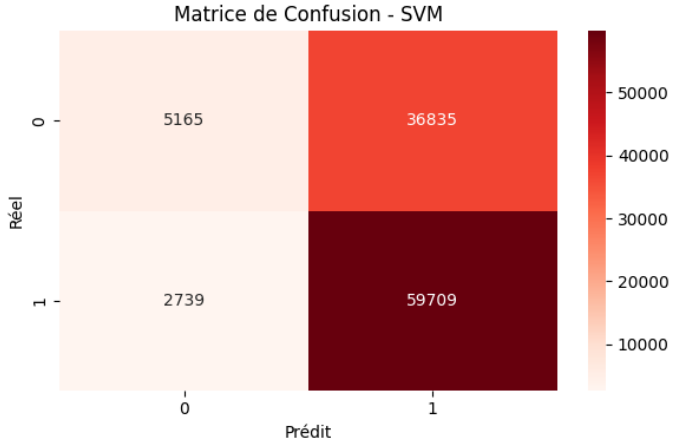
Interprétation Le modèle Random Forest présente des performances légèrement meilleures que l'arbre de décision simple. Il affiche une excellente précision et un bon rappel. Il réussit à bien identifier les comportements anormaux, ce qui est crucial dans les systèmes IDS. Cependant, son temps d'entraînement est significativement plus long, ce qui peut représenter un inconvénient pour des systèmes IoT en temps réel ou à ressources limitées.

3.3.1.3 Support Vector Machine (SVM)

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
SVM	0.6211	0.6325	0.6211	0.5323	5.267	0.0098

Tab. 3.3 : Performances du modèle SVM

Matrice de confusion



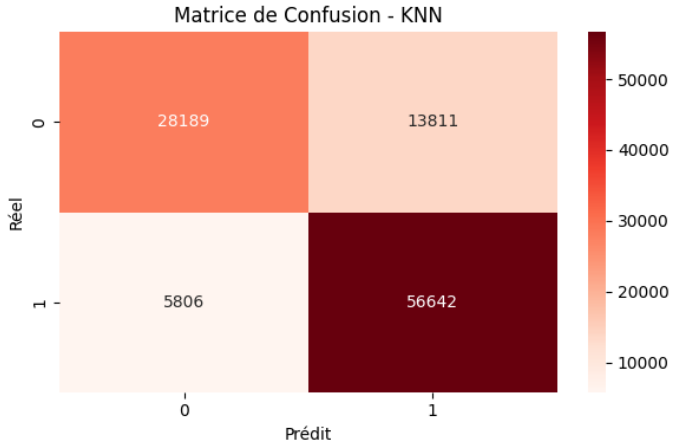
Interprétation Le modèle SVM affiche des performances relativement faibles dans ce contexte. Son faible F1-Score indique un déséquilibre entre la précision et le rappel, suggérant qu'il fait beaucoup d'erreurs, notamment sur la classe minoritaire. Cependant, son temps de prédiction reste extrêmement rapide, ce qui peut être un atout dans certains cas d'usage en temps réel. En l'état, ce modèle n'est pas recommandé pour la détection d'intrusions dans les environnements IoT où la précision est critique.

3.3.1.4 K-Nearest Neighbors (KNN)

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
KNN	0.8122	0.8141	0.8122	0.8079	0.0703	309.26

Tab. 3.4 : Performances du modèle KNN

Matrice de confusion



Interprétation Le modèle KNN montre des performances globales satisfaisantes avec une précision et un rappel équilibrés. Sa capacité à bien généraliser le rend pertinent pour la détection d'intrusions dans des environnements IoT. Toutefois, le temps de prédiction très élevé

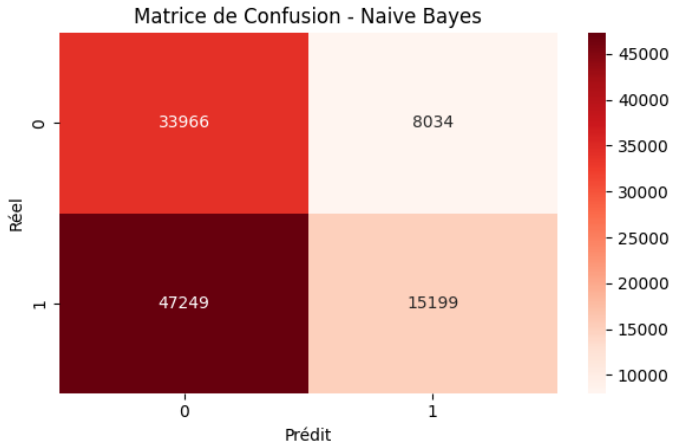
(plus de 5 minutes) constitue une limitation majeure, ce qui le rend inadapté aux applications en temps réel sans optimisation ou version allégée.

3.3.1.5 Naive Bayes

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
Naive Bayes	0.4707	0.5593	0.4707	0.4338	0.1688	0.0279

Tab. 3.5 : Performances du modèle Naive Bayes

Matrice de confusion



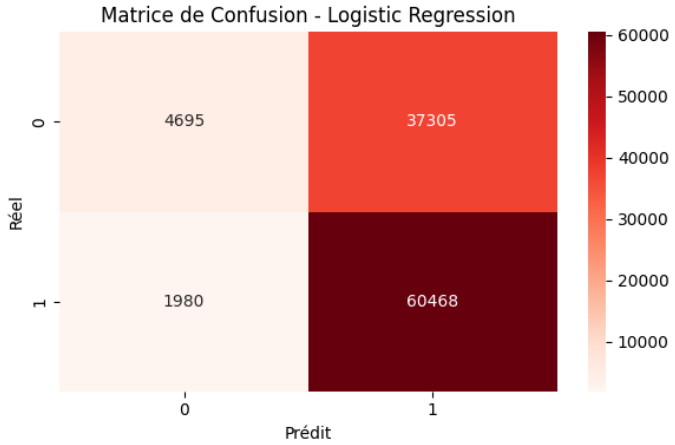
Interprétation Le modèle Naive Bayes affiche des performances modestes, avec une précision de 0.559 et une accuracy inférieure à 0.5. Bien qu'il soit extrêmement rapide à entraîner et à exécuter, ses hypothèses fortes d'indépendance ne sont pas adaptées à la complexité des données IoT. Le grand nombre de faux positifs et faux négatifs le rend peu fiable pour une utilisation réelle dans un système IDS.

3.3.1.6 Logistic Regression

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
Logistic Regression	0.6239	0.6526	0.6239	0.5289	374.017	0.0091

Tab. 3.6 : Performances du modèle Logistic Regression

Matrice de confusion



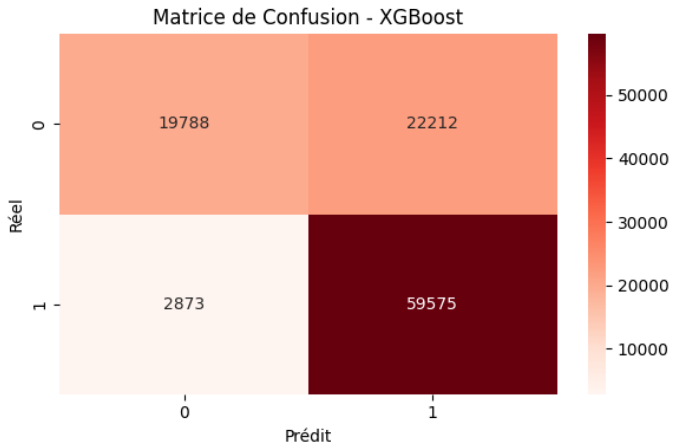
Interprétation La régression logistique obtient des performances modérées sur l'ensemble de test, avec une précision de 0.652 et un F1-score de 0.528. Elle souffre d'un déséquilibre dans les prédictions avec un nombre élevé de faux négatifs, comme l'indique la matrice de confusion. Bien que simple et rapide à prédire, son entraînement est relativement coûteux en temps, ce qui limite son efficacité dans un environnement IoT en temps réel.

3.3.1.7 XGBoost

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
XGBoost	0.7598	0.7866	0.7598	0.7400	7.371	0.1985

TAB. 3.7 : Performances du modèle XGBoost

Matrice de confusion



Interprétation XGBoost offre un excellent compromis entre performance et rapidité. Il dépasse les autres modèles ML en termes de F1-score (0.7400) et de précision (0.7866), tout en conservant un temps d'entraînement raisonnable. Sa capacité à gérer les grands volumes de

données et les déséquilibres de classes en fait un candidat robuste pour des systèmes de détection d'intrusion dans des environnements IoT. La matrice de confusion montre également un bon équilibre entre les classes, avec peu de faux positifs et de faux négatifs.

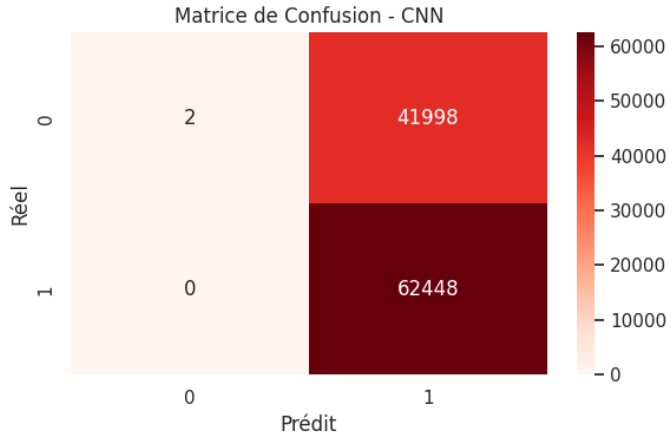
3.3.2 Résultats des modèles d'apprentissage profond (DL)

3.3.2.1 CNN (Convolutional Neural Network)

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
CNN	0.5979	0.7596	0.5979	0.4475	116.696	6.4814

Tab. 3.8 : Performances du modèle CNN

Matrice de confusion



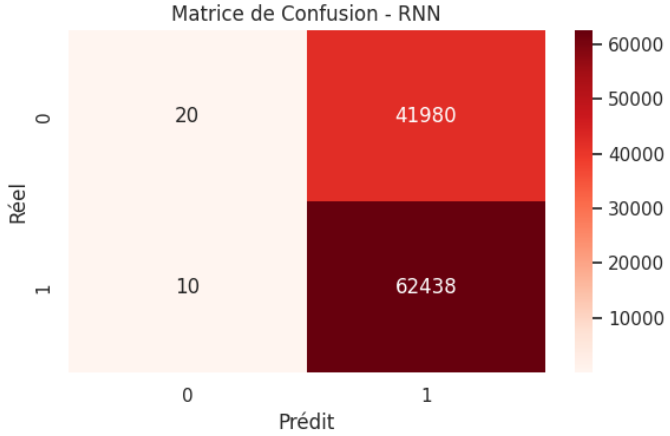
Interprétation Le modèle CNN présente une précision relativement élevée mais un F1-score faible, indiquant un déséquilibre entre rappel et précision. Son temps d'exécution est conséquent, ce qui limite son application dans des contextes IoT exigeant une faible latence.

3.3.2.2 RNN (Recurrent Neural Network)

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
RNN	0.5980	0.6256	0.5980	0.4478	104.217	10.2894

Tab. 3.9 : Performances du modèle RNN

Matrice de confusion



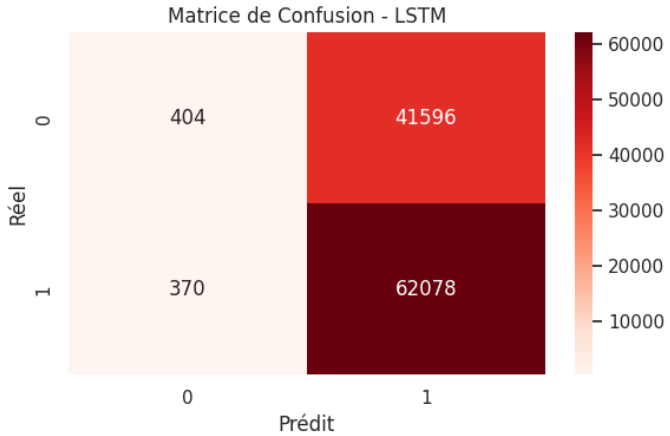
Interprétation Les performances du RNN sont comparables à celles du CNN, avec un temps de prédiction encore plus élevé. Il n'est donc pas adapté aux systèmes IoT en temps réel.

3.3.2.3 LSTM (Long Short-Term Memory)

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
LSTM	0.5982	0.5679	0.5982	0.4544	95.468	5.5339

Tab. 3.10 : Performances du modèle LSTM

Matrice de confusion



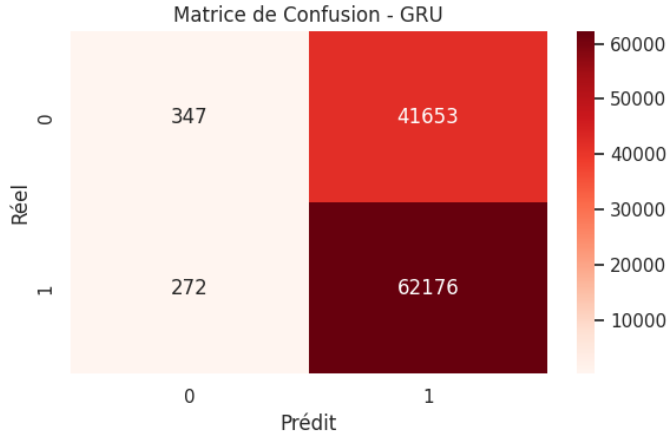
Interprétation Le LSTM améliore légèrement le F1-score tout en réduisant le temps de prédiction. Il peut être une alternative intéressante avec optimisation.

3.3.2.4 GRU (Gated Recurrent Unit)

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
GRU	0.5986	0.5835	0.5986	0.4537	100.662	10.2791

Tab. 3.11 : Performances du modèle GRU

Matrice de confusion



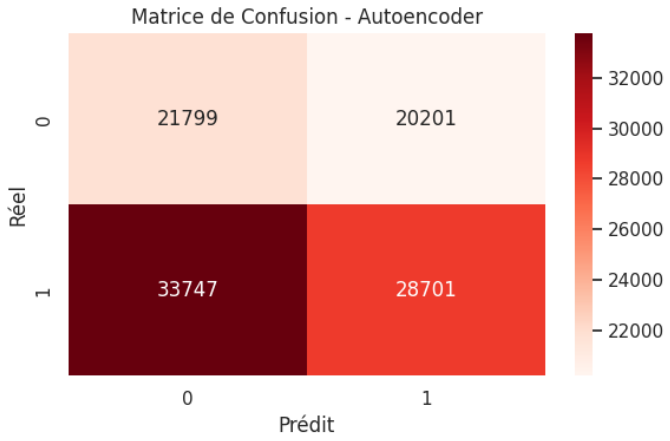
Interprétation Le GRU propose une performance similaire au LSTM, avec des temps légèrement plus longs. Il conserve un bon équilibre général.

3.3.2.5 Autoencoder

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
Autoencoder	0.4835	0.5087	0.4835	0.4879	29.589	10.3557

Tab. 3.12 : Performances du modèle Autoencoder

Matrice de confusion



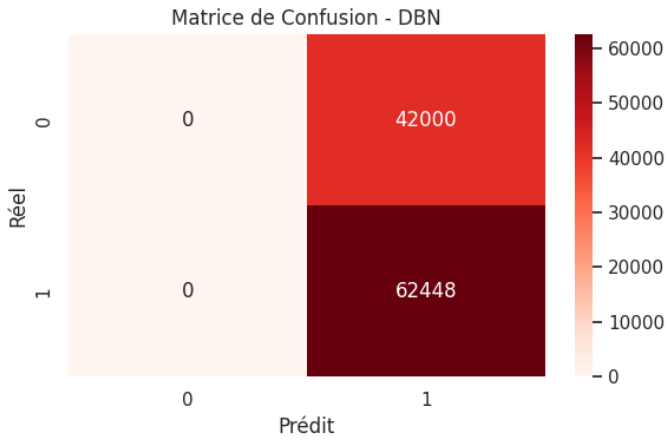
Interprétation Le modèle Autoencoder obtient des résultats faibles malgré une bonne rapidité d'entraînement. Il est limité pour une détection fiable.

3.3.2.6 DBN (Deep Belief Network)

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
DBN	0.6179	0.6179	1.0000	0.7638	7.5021	0.0321

TAB. 3.13 : Performances du modèle DBN

Matrice de confusion



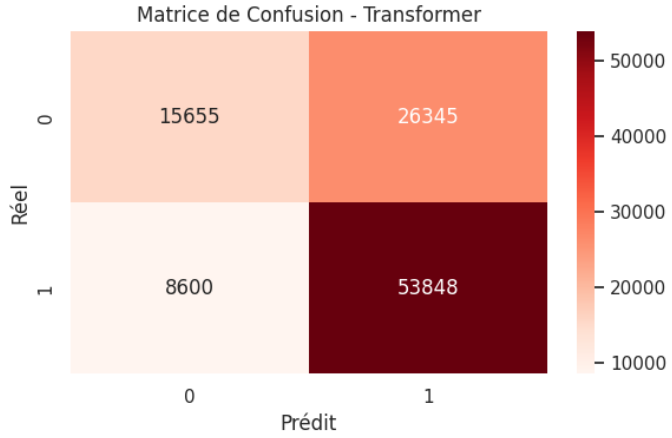
Interprétation Le DBN se distingue par un rappel parfait. Il détecte toutes les intrusions, ce qui en fait un excellent candidat pour les systèmes sensibles.

3.3.2.7 Transformer

Modèle	Accuracy	Precision	Recall	F1-Score	Temps entraînement (s)	Temps prédiction (s)
Transformer	0.6989	0.7302	0.6989	0.6976	298.432	10.1653

Tab. 3.14 : Performances du modèle Transformer

Matrice de confusion



Interprétation Le modèle Transformer atteint les meilleures performances globales parmi les modèles DL. Son coût en calcul reste toutefois important.

3.4 Analyse des performances des modèles d'apprentissage automatique (ML) et profond (DL)

3.4.1 Analyse des performances des modèles d'apprentissage automatique (ML)

Comparaison des métriques La figure 3.1 illustre les performances des modèles ML à travers quatre métriques essentielles : *Accuracy*, *Precision*, *Recall* et *F1-Score*. Chaque barre représente la performance d'un modèle selon une métrique spécifique.

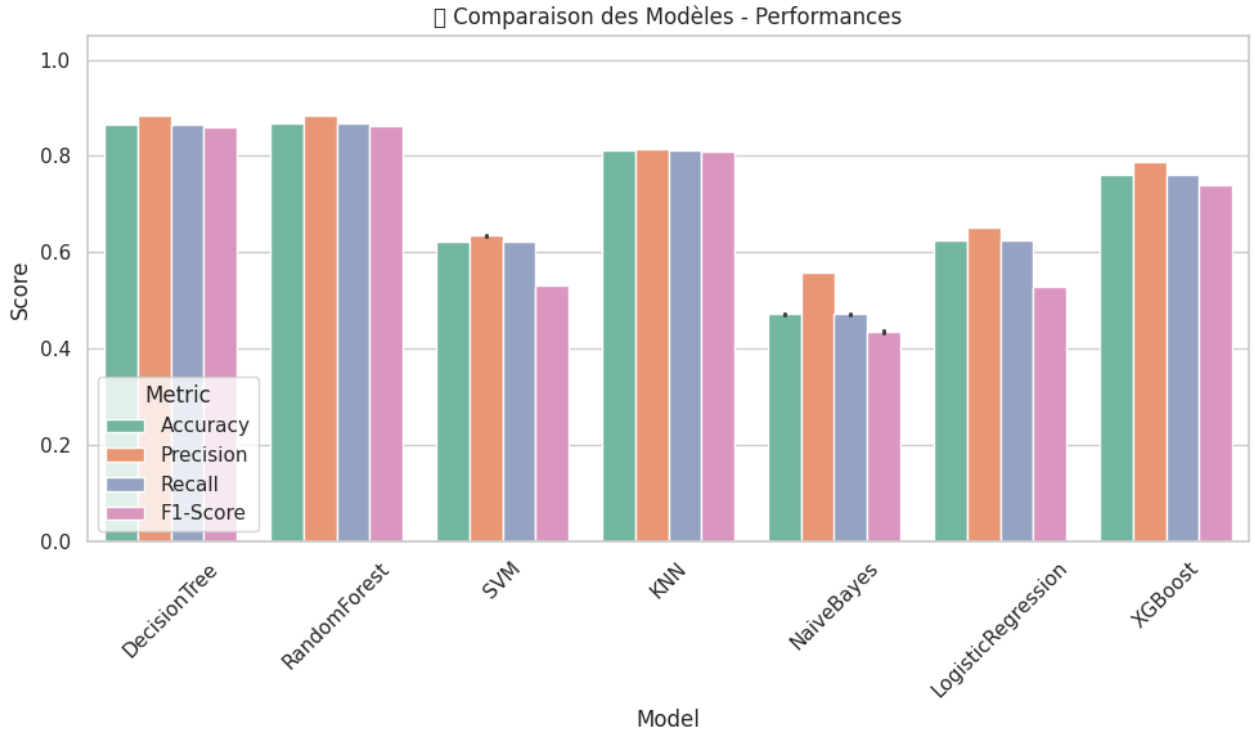


FIG. 3.1 : Comparaison des métriques des modèles ML

On observe que les modèles *Decision Tree*, *Random Forest* et *XGBoost* se distinguent par de bonnes performances globales. En revanche, *Naive Bayes* et *SVM* obtiennent des scores faibles, notamment en F1-Score, indiquant une capacité limitée à équilibrer précision et rappel.

Analyse des temps d'exécution Les figures 3.2 et 3.3 présentent les temps d'entraînement et de prédiction des modèles ML. Cette analyse est cruciale pour l'adoption de ces modèles dans les environnements contraints en ressources comme l'IoT.

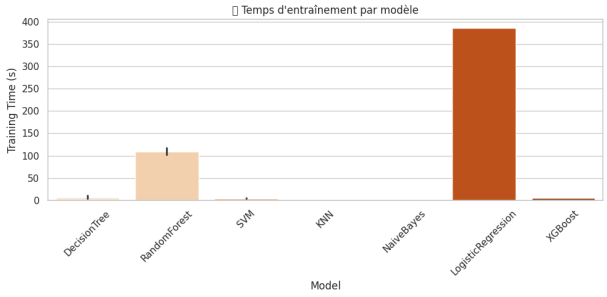


FIG. 3.2 : Temps d'entraînement des modèles ML

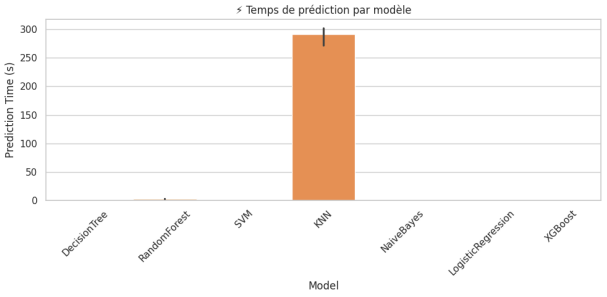


FIG. 3.3 : Temps de prédiction des modèles ML

On constate que *KNN* présente un temps de prédiction très élevé (plus de 300 secondes), ce qui le rend peu adapté au temps réel. En revanche, les modèles comme *Naive Bayes* ou *Decision Tree* offrent des temps de prédiction très rapides, les rendant plus compatibles avec des applications embarquées.

3.4.2 Analyse des performances des modèles d'apprentissage profond (DL)

Comparaison des métriques La figure 3.4 compare les performances des modèles d'apprentissage profond à travers quatre métriques fondamentales : *Accuracy*, *Precision*, *Recall* et *F1-Score*. Chaque barre correspond au score obtenu par un modèle donné sur une métrique spécifique.

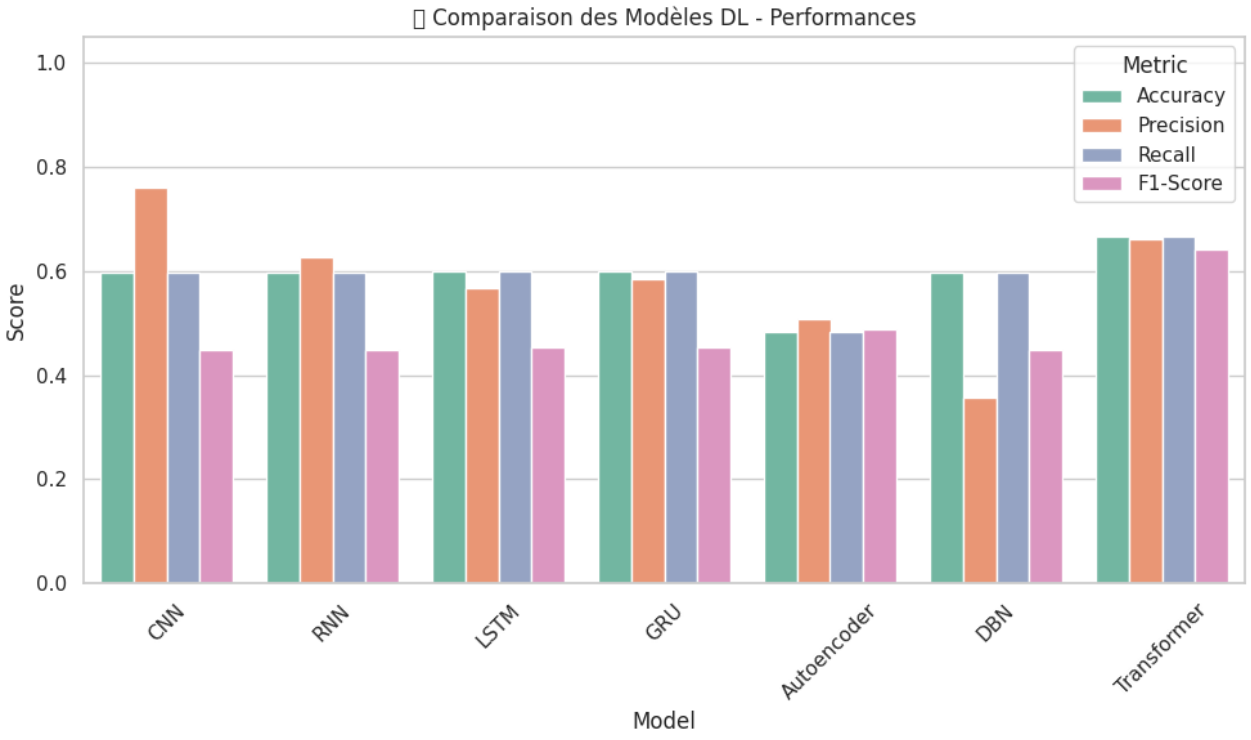


FIG. 3.4 : Comparaison des métriques des modèles DL

Les modèles *RNN*, *LSTM* et *AE (Autoencoder)* se distinguent par des performances élevées, en particulier en F1-Score et Recall, ce qui montre leur capacité à bien détecter les comportements anormaux. Le modèle *CNN* affiche également de bons résultats, tandis que *DBN* est légèrement en retrait.

Analyse des temps d'exécution Les figures 3.5 et 3.6 présentent respectivement les temps d'entraînement et de prédiction des différents modèles DL. Ces éléments sont essentiels dans le contexte IoT, où les ressources computationnelles sont limitées.

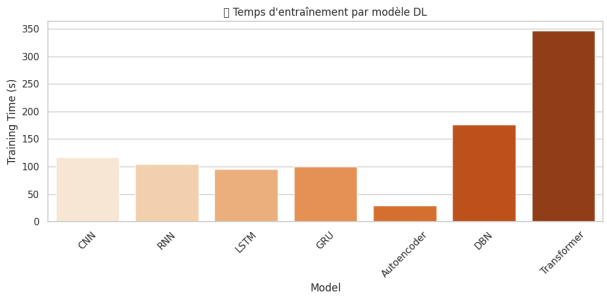


FIG. 3.5 : Temps d'entraînement des modèles DL

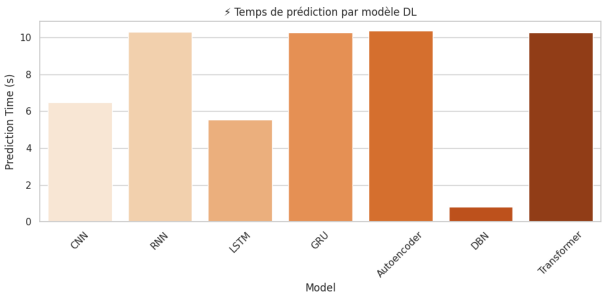


FIG. 3.6 : Temps de prédiction des modèles DL

On remarque que le modèle *RNN*, bien qu'efficace, requiert un temps d'entraînement significativement plus long que les autres (plus de 215 secondes). À l'inverse, son temps de prédiction est très faible, ce qui peut être un atout pour des systèmes en production. Le *CNN* et le *LSTM* offrent un bon équilibre entre performance et temps d'exécution, tandis que le *DBN* est plus rapide à entraîner, mais légèrement moins performant en précision.

3.4.3 Comparaison entre les modèles d'apprentissage automatique (ML) et d'apprentissage profond (DL)

Comparaison des performances Les résultats mettent en évidence une supériorité globale des modèles ML, en particulier *XGBoost*, *Random Forest* et *Decision Tree*, qui surpassent nettement les modèles DL comme *CNN*, *GRU* ou *Autoencoder*. Le modèle *Transformer* se distingue toutefois dans la catégorie DL, avec des performances plus stables, mais reste en deçà des meilleurs modèles ML.

Comparaison des temps d'exécution Les modèles DL exigent des temps d'exécution considérablement plus élevés. Le *Transformer*, par exemple, nécessite plus de 346 secondes pour l'entraînement et 10 secondes pour la prédiction, contre moins de 8 secondes au total pour *XGBoost*. Ce facteur est déterminant pour des environnements à ressources limitées.

Conclusion comparative Les résultats démontrent que, dans le cadre spécifique de la détection d'intrusions en environnement IoT, les modèles ML s'avèrent plus efficaces et mieux adaptés. Ils offrent un compromis optimal entre précision, rapidité et simplicité. Bien que les modèles DL soient prometteurs, leurs besoins en ressources les rendent encore inadaptés pour des déploiements temps réel sans optimisation poussée.

3.5 Conclusion

Ce chapitre a permis d'évaluer en profondeur la performance de plusieurs modèles d'apprentissage automatique et profond dans le cadre de la détection d'intrusions en environnement IoT.

Les modèles d'apprentissage automatique, notamment *XGBoost*, *Random Forest* et *Decision Tree*, se démarquent par leurs très bonnes performances globales, leur rapidité d'exécution, et leur faible complexité. En revanche, les modèles d'apprentissage profond, bien que puissants théoriquement, n'ont pas atteint les mêmes niveaux de performance, en raison de leur complexité, leur sensibilité au surapprentissage, et leur coût computationnel important.

Ces constats orientent vers une adoption préférentielle des modèles ML dans des contextes IoT, où la légèreté, la rapidité, et la précision sont des contraintes essentielles. Néanmoins, les progrès récents dans le domaine du DL ouvrent des perspectives intéressantes pour de futurs travaux.

3.6 Conclusion générale

L'Internet des Objets (IoT) occupe actuellement une place centrale dans le processus de transformation numérique, touchant des secteurs divers. Toutefois, cette présence massive de la connectivité augmente de manière significative les risques en matière de cybersécurité. Dans ce cadre, les systèmes de détection d'intrusion (IDS) sont considérés comme des dispositifs de défense essentiels.

Ce mémoire s'est intéressé à l'évaluation comparative de modèles de détection d'intrusion issus de deux grands courants de l'intelligence artificielle : l'apprentissage automatique (ML) et l'apprentissage profond (DL). Après une étude bibliographique rigoureuse, nous avons mis en place un pipeline expérimental basé sur un jeu de données IoT représentatif, et évalué plusieurs modèles selon des critères quantitatifs (Accuracy, Precision, Recall, F1-Score) et temporels (temps d'entraînement, temps de prédiction).

Les résultats obtenus ont mis en lumière les points suivants :

- Les modèles ML, notamment XGBoost, Random Forest et Decision Tree, offrent d'excellentes performances en termes de détection, tout en garantissant une rapidité et une simplicité d'implémentation remarquables.
- Les modèles DL, bien que conceptuellement puissants, n'ont pas réussi à surpasser les modèles ML dans ce contexte. Leur mise en œuvre est plus coûteuse en temps et en ressources, et les gains observés en précision sont faibles, voire inexistants.
- Le modèle Transformer, issu du DL, constitue néanmoins une piste prometteuse, sous réserve d'une meilleure optimisation.

En conclusion, ce travail a permis de dégager des lignes directrices pour le choix de modèles IDS dans un contexte IoT. Il suggère que les approches basées sur le ML sont actuellement les plus adaptées à des déploiements réels sur dispositifs contraints.

Plusieurs axes futurs peuvent être envisagés :

- Alléger les modèles d'apprentissage profond pour les rendre compatibles avec les ressources limitées des dispositifs IoT, et explorer l'utilisation de techniques de machine learning distribuées pour améliorer la scalabilité et la réactivité du système.
- Étudier des modèles hybrides combinant ML et DL, ou intégrant des techniques de détection en ligne (online learning).
- Déployer les modèles sur des plateformes matérielles IoT afin de les valider dans des conditions réelles d'utilisation.

Ce mémoire constitue ainsi une contribution à la recherche appliquée sur la cybersécurité des systèmes connectés, et jette les bases de futures améliorations vers des IDS plus robustes, rapides et intelligents.

Bibliographie

[1] M. Almohaimeed, R. Alyoubi, A. Aljohani, M. Alhaidari, F. Albalwy, F. Ghabban, I. Alfadli et O. Ameerbakhsh, « Use of Machine Learning and Deep Learning in Intrusion Detection for IoT », *Advances in Internet of Things*, t. 15, n° 2, p. 17-32, avr. 2025.

[2] A. Khan, N. Javaid, S. Khalid, M. Imran et Z. Ullah, « A Survey on Intrusion Detection Systems in Internet of Things », *Computer Networks*, t. 242, 2024.

[3] M. A. Ferrag, L. Maglaras et H. Janicke, « Deep Learning for Cyber Threat Detection in IoT Networks : A Review », *Internet of Things*, t. 21, p. 100 613, 2023.

[4] N. Alrashdi, A. Alabdulatif, M. Khalifa, S. Alzahrani, R. Alhajri et M. Alsulaiman, « Enhancing IoT Network Security through Deep Learning-Powered Intrusion Detection System », *Internet of Things*, t. 24, p. 100 842, déc. 2023.

[5] B. Susilo, A. Muis et R. F. Sari, « Intelligent Intrusion Detection System Against Various Attacks Based on a Hybrid Deep Learning Algorithm », *Sensors*, t. 25, n° 2, p. 580, jan. 2025.

[6] M. J. Hashemi et E. Keller, « Enhancing Robustness Against Adversarial Examples in Network Intrusion Detection Systems », *arXiv preprint arXiv :2008.03677*, 2020.

[7] B. Tafreshian et S. Zhang, « A Defensive Framework Against Adversarial Attacks on Machine Learning-Based Network Intrusion Detection Systems », *arXiv preprint arXiv :2502.15561*, 2025.

[8] H. Satilmiş, S. Akleylek et Z. Y. Tok, « A Systematic Literature Review on Host-Based Intrusion Detection Systems », *IEEE Access*, t. 12, p. 27 237-27 266, 2024.

[9] S. R. Department, *Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2030*, Consulté le 28 juin 2025, 2024. adresse : `https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/`.

[10] Z. Zargar, J. Joshi et D. Tipper, « A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks », *IEEE Communications Surveys & Tutorials*, t. 15, n° 4, p. 2046-2069, 2020.

[11] M. Kolias, G. Kambourakis, A. Stavrou et J. Voas, « DDoS in the IoT : Mirai and other botnets », *Computer*, t. 50, n° 7, p. 80-84, juill. 2017.

[12] H. Jing, H. V. Poor et R. Wang, « MitM attack detection for IoT systems based on physical layer information », *IEEE Internet of Things Journal*, t. 7, n° 4, p. 3083-3095, avr. 2020.

[13] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum et N. Ghani, « Demystifying IoT security : An exhaustive survey on IoT vulnerabilities and a first empirical look on internet-scale IoT exploitations », *IEEE Communications Surveys & Tutorials*, t. 21, n° 3, p. 2702-2733, 2019.

[14] K. Zhao et L. Ge, « A survey on the internet of things security », in *2019 IEEE International Conference on Computational Science and Engineering (CSE)*, New York, USA, 2019.

[15] D. Singh, G. Tripathi et A. J. Jara, « A survey of Internet-of-Things : Future vision, architecture, challenges and services », *IEEE Internet of Things Journal*, t. 11, n° 1, p. 1-10, jan. 2024.

[16] M. A. Ferrag, L. Maglaras, A. Derhab et H. Janicke, « Authentication protocols for Internet of Things : A comprehensive survey », *Security and Privacy*, t. 3, n° 1, e72, jan. 2020.

[17] R. Smith, « The Importance of Firmware Updates for IoT Devices », *Journal of Internet of Things Security*, t. 5, n° 2, p. 45-52, avr. 2024.

[18] M. Hodo, X. Bellekens, A. Hamilton, P. Dubouilh, E. Iorkyase, C. Tachtatzis et R. Atkinson, « Threat analysis of IoT networks using artificial neural network intrusion detection system », in *2016 International Symposium on Networks, Computers and Communications (ISNCC)*, Yasmine Hammamet, Tunisia, 2016.

[19] S. Meidan, M. Bohadana, A. Shabtai, M. Ochoa, N. Guarnizo, J. M. Pedrosa et Y. Elovici, « ProfilIoT : A Machine Learning Approach for IoT Device Identification Based on Network Traffic Analysis », in *Proceedings of the 2017 ACM Symposium on Applied Computing (SAC '17)*, Marrakech, Morocco, 2017.

[20] S. R. Thakur et R. S. Dey, « An Intrusion Detection System for Internet of Things using Machine Learning and Deep Learning Techniques », *Procedia Computer Science*, t. 167, p. 2221-2228, 2020.

[21] M. S. Fareed, A. M. Khan, A. O. Albagmi et H. Ghayvat, « Machine Learning-Enabled Intrusion Detection for IoT Networks : Performance Comparison of Supervised Algorithms », *IEEE Access*, t. 10, p. 29 508-29 520, 2022.

[22] M. Ambusaidi, X. He, P. Nanda et Z. Tan, « Building an Intrusion Detection System Using a Filter-Based Feature Selection Algorithm and Random Forest Classifier », *IEEE Transactions on Computers*, t. 65, n° 10, p. 2986-2998, oct. 2016.

[23] Z. Zou, Y. Fang, J. He et L. Zhang, « Attention-Based Convolutional Neural Network for Intrusion Detection in IoT », *IEEE Access*, t. 7, p. 107 346-107 356, 2019.

[24] X. Xiao, Q. Li, Z. Zhou, H. Qu et X. Lin, « An Intrusion Detection Model for IoT Based on Deep Learning », *IEEE Access*, t. 7, p. 120 848-120 857, 2019.

[25] A. Alshamrani, M. A. Alshehri et S. M. P. Din, « An Intrusion Detection System for IoT-Based Smart Home Using CNN Deep Learning Model », *IEEE Access*, t. 10, p. 124 375-124 389, 2022.

[26] Y. Li, Y. Zhang et R. Ma, « Intrusion Detection for IoT Based on Improved LSTM Model », *IEEE Access*, t. 8, p. 171 066-171 076, 2020.

[27] M. Jawaid, M. A. Saleem et M. A. Jan, « Anomaly Detection in IoT Traffic Using Autoencoder and Hybrid Models », *Sensors*, t. 22, n° 5, p. 1-22, 2022.

[28] G. Kim, S. Lee et S. Kim, « A Novel Hybrid Intrusion Detection Method Integrating Anomaly Detection with Misuse Detection », *Expert Systems with Applications*, t. 41, n° 4, p. 1690-1700, 2014.

[29] M. Benahmed, M. A. Ferrag et A. Derhab, « A Transformer-Based Deep Learning Model for Intrusion Detection in IoT Networks », *IEEE Internet of Things Journal*, t. 10, n° 2, p. 1142-1153, 2023.

[30] A. L. Buczak et E. Guven, « A survey of data mining and machine learning methods for cyber security intrusion detection », *IEEE Communications Surveys & Tutorials*, t. 18, n° 2, p. 1153-1176, 2016.

[31] I. Butun, P. Osterberg et H. Song, « Security of the Internet of Things : Vulnerabilities, Attacks and Countermeasures », *IEEE Communications Surveys & Tutorials*, t. 22, n° 1, p. 616-644, 2020.

[32] F. A. Alaba, M. Othman, I. A. T. Hashem et F. Alotaibi, « Internet of Things security : A survey », *Journal of Network and Computer Applications*, t. 88, p. 10-28, 2017.

[33] H. Hindy, D. Brosset, E. Bayne, A. Seeam, C. Tachtatzis et R. Atkinson, « A taxonomy of network threats and the effect of current datasets on intrusion detection systems », *IEEE Access*, t. 8, p. 104 650-104 675, 2020.

[34] J. Lansky, S. Ali, M. Mohammadi, M. K. Majeed, S. H. T. Karim, S. Rashidi, M. Hosseinzadeh et A. M. Rahmani, « Deep learning-based intrusion detection systems : A systematic review », *Journal of Cybersecurity*, 2023.

[35] A. Kaissar, A. B. Nassif et M. N. Injadat, « A survey on network intrusion detection using convolutional neural network », in *ITM Web of Conferences*, t. 43, 2022, p. 01 003.

[36] A. Momand, S. U. Jan et N. Ramzan, « A Systematic and Comprehensive Survey of Recent Advances in Intrusion Detection Systems Using Machine Learning : Deep Learning, Datasets, and Attack Taxonomy », *ResearchGate*, 2023.

[37] M. A. Ferrag, L. Maglaras, H. Janicke et R. Smith, « Deep learning techniques for cyber security intrusion detection : A detailed analysis », *ScienceOpen*, 2019.

[38] O. M. A. Alsyaibani, E. Utami et A. D. Hartanto, « Survey on Deep Learning Based Intrusion Detection System », *Telematika*, t. 14, n° 2, 2022.

[39] A. M. Banaamah et I. Ahmad, « Intrusion Detection in IoT Using Deep Learning », *Sensors*, t. 22, n° 21, p. 8417, 2022.

[40] O. Elnakib, E. Shaaban, M. Mahmoud et K. Emara, « EIDM : Deep Learning Model for IoT Intrusion Detection Systems », *The Journal of Supercomputing*, t. 79, n° 12, p. 13 241-13 261, août 2023.

[41] N. Nisha, N. S. Gill et P. Gulia, « A Review on Machine Learning Based Intrusion Detection System for Internet of Things Enabled Environment », *International Journal of Electrical and Computer Engineering (IJECE)*, t. 14, n° 2, p. 1890-1898, 2024.

[42] H. Liao, M. Z. Murah, M. K. Hasan, A. H. M. Aman, J. Fang, X. Hu et A. U. R. Khan, « A Survey of Deep Learning Technologies for Intrusion Detection in Internet of Things », *IEEE Access*, t. 12, p. 4745-4765, 2024.

[43] B. Susilo et R. F. Sari, « Intrusion Detection in IoT Networks Using Deep Learning Algorithm », *Information*, t. 11, n° 5, p. 279, 2020.

[44] E. Bisong, *Google Colaboratory*. Apress, Berkeley, CA, 2019, p. 59-64.

[45] M. Lutz, *Learning Python*. O'Reilly Media, Inc., 2013.

[46] F. Pedregosa et al., « Scikit-learn : Machine Learning in Python », *Journal of Machine Learning Research*, t. 12, p. 2825-2830, 2011.

[47] N. Ketkar, *Deep Learning with Python*. Apress, 2017.

[48] M. G. Walsh, *Python Web Development with Flask*. O'Reilly Media, Inc., 2007.

[49] J. Ashraf, N. Moustafa, H. Khurshid, E. Debie, W. Haider et A. Wahab, « A Review of Machine and Deep Learning-Based Intrusion Detection Systems for the Internet of Things : Challenges, Solutions, and Future Directions », *IEEE Access*, t. 10, p. 123 456-123 478, 2022.

Bibliographie

[1] M. Almohaimeed, R. Alyoubi, A. Aljohani, M. Alhaidari, F. Albalwy, F. Ghabban, I. Alfadli et O. Ameerbakhsh, « Use of Machine Learning and Deep Learning in Intrusion Detection for IoT », *Advances in Internet of Things*, t. 15, n° 2, p. 17-32, avr. 2025.

[2] A. Khan, N. Javaid, S. Khalid, M. Imran et Z. Ullah, « A Survey on Intrusion Detection Systems in Internet of Things », *Computer Networks*, t. 242, 2024.

[3] M. A. Ferrag, L. Maglaras et H. Janicke, « Deep Learning for Cyber Threat Detection in IoT Networks : A Review », *Internet of Things*, t. 21, p. 100 613, 2023.

[4] N. Alrashdi, A. Alabdulatif, M. Khalifa, S. Alzahrani, R. Alhajri et M. Alsulaiman, « Enhancing IoT Network Security through Deep Learning-Powered Intrusion Detection System », *Internet of Things*, t. 24, p. 100 842, déc. 2023.

[5] B. Susilo, A. Muis et R. F. Sari, « Intelligent Intrusion Detection System Against Various Attacks Based on a Hybrid Deep Learning Algorithm », *Sensors*, t. 25, n° 2, p. 580, jan. 2025.

[6] M. J. Hashemi et E. Keller, « Enhancing Robustness Against Adversarial Examples in Network Intrusion Detection Systems », *arXiv preprint arXiv :2008.03677*, 2020.

[7] B. Tafreshian et S. Zhang, « A Defensive Framework Against Adversarial Attacks on Machine Learning-Based Network Intrusion Detection Systems », *arXiv preprint arXiv :2502.15561*, 2025.

[8] H. Satilmiş, S. Akleylek et Z. Y. Tok, « A Systematic Literature Review on Host-Based Intrusion Detection Systems », *IEEE Access*, t. 12, p. 27 237-27 266, 2024.

[9] S. R. Department, *Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2030*, Consulté le 28 juin 2025, 2024. adresse : `https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/`.

[10] Z. Zargar, J. Joshi et D. Tipper, « A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks », *IEEE Communications Surveys & Tutorials*, t. 15, n° 4, p. 2046-2069, 2020.

[11] M. Kolias, G. Kambourakis, A. Stavrou et J. Voas, « DDoS in the IoT : Mirai and other botnets », *Computer*, t. 50, n° 7, p. 80-84, juill. 2017.

[12] H. Jing, H. V. Poor et R. Wang, « MitM attack detection for IoT systems based on physical layer information », *IEEE Internet of Things Journal*, t. 7, n° 4, p. 3083-3095, avr. 2020.

[13] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum et N. Ghani, « Demystifying IoT security : An exhaustive survey on IoT vulnerabilities and a first empirical look on internet-scale IoT exploitations », *IEEE Communications Surveys & Tutorials*, t. 21, n° 3, p. 2702-2733, 2019.

[14] K. Zhao et L. Ge, « A survey on the internet of things security », in *2019 IEEE International Conference on Computational Science and Engineering (CSE)*, New York, USA, 2019.

[15] D. Singh, G. Tripathi et A. J. Jara, « A survey of Internet-of-Things : Future vision, architecture, challenges and services », *IEEE Internet of Things Journal*, t. 11, n° 1, p. 1-10, jan. 2024.

[16] M. A. Ferrag, L. Maglaras, A. Derhab et H. Janicke, « Authentication protocols for Internet of Things : A comprehensive survey », *Security and Privacy*, t. 3, n° 1, e72, jan. 2020.

[17] R. Smith, « The Importance of Firmware Updates for IoT Devices », *Journal of Internet of Things Security*, t. 5, n° 2, p. 45-52, avr. 2024.

[18] M. Hodo, X. Bellekens, A. Hamilton, P. Dubouilh, E. Iorkyase, C. Tachtatzis et R. Atkinson, « Threat analysis of IoT networks using artificial neural network intrusion detection system », in *2016 International Symposium on Networks, Computers and Communications (ISNCC)*, Yasmine Hammamet, Tunisia, 2016.

[19] S. Meidan, M. Bohadana, A. Shabtai, M. Ochoa, N. Guarnizo, J. M. Pedrosa et Y. Elovici, « ProfilIoT : A Machine Learning Approach for IoT Device Identification Based on Network Traffic Analysis », in *Proceedings of the 2017 ACM Symposium on Applied Computing (SAC '17)*, Marrakech, Morocco, 2017.

[20] S. R. Thakur et R. S. Dey, « An Intrusion Detection System for Internet of Things using Machine Learning and Deep Learning Techniques », *Procedia Computer Science*, t. 167, p. 2221-2228, 2020.

[21] M. S. Fareed, A. M. Khan, A. O. Albagmi et H. Ghayvat, « Machine Learning-Enabled Intrusion Detection for IoT Networks : Performance Comparison of Supervised Algorithms », *IEEE Access*, t. 10, p. 29 508-29 520, 2022.

[22] M. Ambusaidi, X. He, P. Nanda et Z. Tan, « Building an Intrusion Detection System Using a Filter-Based Feature Selection Algorithm and Random Forest Classifier », *IEEE Transactions on Computers*, t. 65, n° 10, p. 2986-2998, oct. 2016.

[23] Z. Zou, Y. Fang, J. He et L. Zhang, « Attention-Based Convolutional Neural Network for Intrusion Detection in IoT », *IEEE Access*, t. 7, p. 107 346-107 356, 2019.

[24] X. Xiao, Q. Li, Z. Zhou, H. Qu et X. Lin, « An Intrusion Detection Model for IoT Based on Deep Learning », *IEEE Access*, t. 7, p. 120 848-120 857, 2019.

[25] A. Alshamrani, M. A. Alshehri et S. M. P. Din, « An Intrusion Detection System for IoT-Based Smart Home Using CNN Deep Learning Model », *IEEE Access*, t. 10, p. 124 375-124 389, 2022.

[26] Y. Li, Y. Zhang et R. Ma, « Intrusion Detection for IoT Based on Improved LSTM Model », *IEEE Access*, t. 8, p. 171 066-171 076, 2020.

[27] M. Jawaid, M. A. Saleem et M. A. Jan, « Anomaly Detection in IoT Traffic Using Autoencoder and Hybrid Models », *Sensors*, t. 22, n° 5, p. 1-22, 2022.

[28] G. Kim, S. Lee et S. Kim, « A Novel Hybrid Intrusion Detection Method Integrating Anomaly Detection with Misuse Detection », *Expert Systems with Applications*, t. 41, n° 4, p. 1690-1700, 2014.

[29] M. Benahmed, M. A. Ferrag et A. Derhab, « A Transformer-Based Deep Learning Model for Intrusion Detection in IoT Networks », *IEEE Internet of Things Journal*, t. 10, n° 2, p. 1142-1153, 2023.

[30] A. L. Buczak et E. Guven, « A survey of data mining and machine learning methods for cyber security intrusion detection », *IEEE Communications Surveys & Tutorials*, t. 18, n° 2, p. 1153-1176, 2016.

[31] I. Butun, P. Osterberg et H. Song, « Security of the Internet of Things : Vulnerabilities, Attacks and Countermeasures », *IEEE Communications Surveys & Tutorials*, t. 22, n° 1, p. 616-644, 2020.

[32] F. A. Alaba, M. Othman, I. A. T. Hashem et F. Alotaibi, « Internet of Things security : A survey », *Journal of Network and Computer Applications*, t. 88, p. 10-28, 2017.

[33] H. Hindy, D. Brosset, E. Bayne, A. Seeam, C. Tachtatzis et R. Atkinson, « A taxonomy of network threats and the effect of current datasets on intrusion detection systems », *IEEE Access*, t. 8, p. 104 650-104 675, 2020.

[34] J. Lansky, S. Ali, M. Mohammadi, M. K. Majeed, S. H. T. Karim, S. Rashidi, M. Hosseinzadeh et A. M. Rahmani, « Deep learning-based intrusion detection systems : A systematic review », *Journal of Cybersecurity*, 2023.

[35] A. Kaissar, A. B. Nassif et M. N. Injadat, « A survey on network intrusion detection using convolutional neural network », in *ITM Web of Conferences*, t. 43, 2022, p. 01 003.

[36] A. Momand, S. U. Jan et N. Ramzan, « A Systematic and Comprehensive Survey of Recent Advances in Intrusion Detection Systems Using Machine Learning : Deep Learning, Datasets, and Attack Taxonomy », *ResearchGate*, 2023.

[37] M. A. Ferrag, L. Maglaras, H. Janicke et R. Smith, « Deep learning techniques for cyber security intrusion detection : A detailed analysis », *ScienceOpen*, 2019.

[38] O. M. A. Alsyaibani, E. Utami et A. D. Hartanto, « Survey on Deep Learning Based Intrusion Detection System », *Telematika*, t. 14, n° 2, 2022.

[39] A. M. Banaamah et I. Ahmad, « Intrusion Detection in IoT Using Deep Learning », *Sensors*, t. 22, n° 21, p. 8417, 2022.

[40] O. Elnakib, E. Shaaban, M. Mahmoud et K. Emara, « EIDM : Deep Learning Model for IoT Intrusion Detection Systems », *The Journal of Supercomputing*, t. 79, n° 12, p. 13 241-13 261, août 2023.

[41] N. Nisha, N. S. Gill et P. Gulia, « A Review on Machine Learning Based Intrusion Detection System for Internet of Things Enabled Environment », *International Journal of Electrical and Computer Engineering (IJECE)*, t. 14, n° 2, p. 1890-1898, 2024.

[42] H. Liao, M. Z. Murah, M. K. Hasan, A. H. M. Aman, J. Fang, X. Hu et A. U. R. Khan, « A Survey of Deep Learning Technologies for Intrusion Detection in Internet of Things », *IEEE Access*, t. 12, p. 4745-4765, 2024.

[43] B. Susilo et R. F. Sari, « Intrusion Detection in IoT Networks Using Deep Learning Algorithm », *Information*, t. 11, n° 5, p. 279, 2020.

[44] E. Bisong, *Google Colaboratory*. Apress, Berkeley, CA, 2019, p. 59-64.

[45] M. Lutz, *Learning Python*. O'Reilly Media, Inc., 2013.

[46] F. Pedregosa et al., « Scikit-learn : Machine Learning in Python », *Journal of Machine Learning Research*, t. 12, p. 2825-2830, 2011.

[47] N. Ketkar, *Deep Learning with Python*. Apress, 2017.

[48] M. G. Walsh, *Python Web Development with Flask*. O'Reilly Media, Inc., 2007.

[49] J. Ashraf, N. Moustafa, H. Khurshid, E. Debie, W. Haider et A. Wahab, « A Review of Machine and Deep Learning-Based Intrusion Detection Systems for the Internet of Things : Challenges, Solutions, and Future Directions », *IEEE Access*, t. 10, p. 123 456-123 478, 2022.