Université Saad DAHLAB - Blida 1

Faculté des sciences

Département d'Informatique

Mémoire présenté par :

M^lles BELHOUT Rima et BOULAKDAM Meriem

Pour l'obtention du diplôme de Master

Domaine : Mathématique et Informatique

Filière : Informatique

Spécialité : Systèmes Informatiques et Réseaux

Sujet :

Proposition d'une Solution pour la Reconnaissance des Appareils de l'Internet des Objets grâce à l'Analyse du Trafic Réseau

Soutenu le : 19 Juin 2023, devant le jury composé de

Dr. Y. MANCER	Présidente du jury	Université de Blida 1
Dr. B. BOUTOUMI	Examinatrice	Université de Blida 1
Dr. M. MEZZI	Promotrice	Université de Blida 1

Résumé

Avec l'adoption à grande échelle de l'Internet des objets (IoT), les problèmes de sécurité, de gestion et de maintenance sont devenus de plus en plus proéminent. L'identification des appareils est un moyen efficace de garantir ces actions dans l'environnement IoT en identifiant la catégorie ou le modèle des appareils dans le réseau.

Le but de notre projet étant de faire une classification des appareils IoT avec l'analyse de trafic réseaux à l'aide des modèles d'apprentissage automatique et de Deep learning, nous avons utilisé neuf (09) algorithmes d'apprentissage supervisé et apprentissage profond (dont KNN, ANN, Arbre de décision, XGBoost…) sur un dataset public « Aalto_IoTDevID **» pour** classifier 27 appareils IoT. Selon les données utilisées, l'algorithme qui était le plus performant est l'algorithme des Arbres de Décision avec un taux d'accuracy égal à 99%. Ce dernier a donc été choisi pour développer une application web mettant en avant les objectifs du projet.

Mots clés :

Analyse du Trafic Réseau, Internet des Objets, Classification des Appareils du Réseau, Apprentissage automatique, Apprentissage profond.

Abstract

With the widespread adoption of the Internet of Things (IoT), security, management and maintenance issues have become increasingly prominent. Device identification is an effective way to ensure these actions in the IoT environment by identifying the category or model of devices in the network.

The goal of our project being to classify IoT devices with network traffic analysis using machine learning and deep learning models, we used nine (09) supervised learning and deep learning algorithms (including KNN, ANN, Decision tree, XGBoost…) on a public dataset "Aalto_IoTDevID" to classify 27 IoT devices. According to the data used, the algorithm that was the most efficient is the Decision Tree algorithm with an accuracy rate equal to 99%. The latter was therefore chosen to develop a web application highlighting the objectives of the project.

Key words:

Network Traffic Analysis, Internet of Things, Network Device Classification, Machine Learning, Deep Learning.

ملخص

مع التبني الواسع النطاق لإنترنت الأشياء (IoT)، أصبحت قضايا الأمن والإدارة والصيانة بارزة بشكل متزايد. يعد تعريف الجهاز طريقة فعالة لضمان هذه الإجراءات في بيئة إنترنت الأشياء من خلال تحديد فئة أو طراز الأجهزة في الشبكة.
الهدف من مشروعنا هو تصنيف أجهزة إنترنت الأشياء مع تحليل حركة مرور الشبكة باستخدام نماذج التعلم الآلي والتعلم العميق، استخدمنا تسعة (09) خوارزميات تعلم خاضع للإشراف وتعلم عميق (بما في ذلك KNN وANN وشجرة القرار و XGBoost ...) في مجموعة بيانات عامة "Aalto_IoTDevID" لتصنيف 27 جهاز إنترنت الأشياء. وفقًا للبيانات المستخدمة، كانت الخوارزمية الأكثر كفاءة هي خوارزمية شجرة القرار بمعدل دقة يساوي 99٪. لذلك تم اختيار الأخير لتطوير تطبيق ويب يسلط الضوء على أهداف المشروع.

الكلمات الدالة:
تحليل حركة مرور الشبكة، إنترنت الأشياء، تصنيف أجهزة الشبكة، التعلم الآلي، التعلم العميق.

Dédicaces

Je dédie ce travail à ma famille, qui m'a soutenu tout au long de cette aventure. Tout particulièrement, à ma mère, qui a été mon pilier et ma source de motivation durant toutes ces années. Je tiens également à remercier mes sœurs pour leur soutien et leurs encouragements.

Merci à tous pour votre soutien et votre confiance, je suis très reconnaissante de vous avoir à mes côtés.

MERIEM

Dédicaces

Je dédie ce travail à mes chers parents pour leur amour, soutien et encouragements tout au long de mes études, et à ma sœur qui était toujours avec moi pour me soutenir dans tous les pas de ma vie et aussi à mes frères pour leur présence à mes côtés lors des moments importants de ma vie

Merci du fond du cœur pour tout ce que vous avez fait pour moi.

Rima

Remerciement

Premièrement, nous remercions le dieu de nous avoir donné la volonté et la santé pour finaliser ce travail.

Nous tenons à exprimer toute notre gratitude à notre promotrice ***« Madame Mezzi Melyara »,*** *pour son soutien constant, sa patience et son expertise tout au long de nos recherches. Nous remercions également tous les participants à cette étude, sans qui ce travail n'aurait pas été possible.*

Nos vifs remerciements au membre du jury pour avoir accepté d'évaluer notre travail.

Nous souhaitons également remercier nos familles pour leur soutien infaillible, leurs encouragements et leur confiance en nous.

Enfin, nous tenons à remercier le corps professoral du département d'Informatique pour avoir veillé à notre formation et tous ceux qui ont contribué de près ou de loin à la réalisation de ce modeste travail. Votre aide, vos encouragements et votre soutien nous ont été précieux et nous serons à jamais très reconnaissantes pour cela.

Nous espérons que ce travail pourra contribuer positivement à la recherche dans ce domaine.

Table des matières

Liste des figures

Liste des tableaux

Liste des Abréviations

ANN: Artificial Neural Network.

CNN: Convolutional Neural Network.

DL: Deep Learning.

DT: Decision tree.

GNB: Gaussian Naïve Bayes

KNN: K-Nearest Neighbor.

LDA: Linear Discriminant Analysis

LSTM: long short-term memory.

ML: Machine Learning.

MLP: Multilayer Perceptron

QDA: Quadratic Discriminant Analysis

RF: Random Forest.

SVM: Support Vector Machine.

XGBoost: Extreme Gratient boosting.

Introduction générale

1. Contexte général

L'internet des objets (IoT) est un ensemble de choses qui intègrent des systèmes capables de communiquer entre eux par l'intermédiaire de réseaux câblés ou sans fil. L'IoT contribue à la vie humaine dans de nombreux domaines importants, notamment les maisons et villes intelligentes, le commerce de détail, les soins de santé, les transports, l'agriculture, l'armée et l'industrie manufacturière. D'ici 2026, le nombre d'appareils IoT dans le monde devrait atteindre 80 milliards[1].

De plus en plus de personnes installent des appareils compatibles IP et des appareils électroménagers chez eux afin de bénéficier de l'amélioration de la capacité à être informé et à contrôler les caractéristiques de leurs maisons. Parmi les exemples de systèmes IoT émergents on trouve : le chauffage et la climatisation automatisés, les systèmes de sécurité et la surveillance à domicile, l'éclairage ou les équipements ménagers traditionnels, mais aussi les appareils électroménagers avec connectivité Wi-Fi supplémentaire.

En raison des ressources limitées de ses appareils (le processeur, la batterie et la bande passante), il peut être difficile de les protéger, gérer et maintenir avec des solutions de sécurité traditionnelles. Par ailleurs, l'hétérogénéité des appareils crée des défis supplémentaires. Par exemple, un appareil peut avoir de nombreux capteurs (température, humidité, mouvement, lumière, etc.) et avoir des exigences très différentes pour le canal sur lequel, il communique avec d'autres appareils. [1]

2. Problématique

Avec le développement des dispositifs IoT dans tous les domaines à travers le monde, Nous avons besoin de toute urgence d'une classification des appareils IoT hétérogènes pour fournir fiabilité, sécurité et qualité de service améliorée aux applications en amont. Cependant, la classification des appareils en analysant le trafic réseau est une tâche non triviale en raison de la nature dynamique et complexe du

[1] https://www.alliedmarketresearch.com/

trafic réseau dans l'IoT. Le trafic réseau d'un appareil peut varier considérablement à différents moments associés aux interactions de l'utilisateur ou aux communications client-serveur. Il est donc difficile de caractériser le trafic réseau de l'appareil en un schéma fixe. Il faut connaître le type d'appareil connecté au réseau pour aider à renforcer la sécurité du réseau aussi. [2]

3. Objectif

L'objectif principal de notre projet est de concevoir et développer une solution pour reconnaitre les appareils IoT avec une classification de ces derniers grâce à l'analyse du trafic réseau. Pour atteindre cet objectif, nous nous sommes particulièrement penchées sur les modèles d'apprentissage automatique et d'apprentissage profond pour la classification des appareils IoT et nous nous sommes concentrées sur les points suivants :

- Utiliser les techniques de machine Learning et Deep Learning pour la classification des appareils selon un jeu de données.
- Evaluer les modèles par des mesures de performances.
- Réalisation d'une application web décrivant le fonctionnement de notre système.

4. Organisation de mémoire

Pour atteindre les objectifs susmentionnés, nous avons organisé notre mémoire comme suit :

- ***Chapitre 1 « Etat de l'art »*** : présente les concepts de base de l'Analyse du Trafic Réseau et l'Internet des Objets (IoT) aussi les travaux connexes dans le domaine.
- ***Chapitre 2 « Architectures neuronales et*** **Conception de la solution »** : nous y présentons l'architecture globale de notre système, le dataset utilisé, les algorithmes testés, ainsi que la conception de plateforme finale.
- ***Chapitre 3 « Implémentation de la solution » :*** nous y présentons les implémentations de notre système, ainsi que les résultats obtenus par les algorithmes utilisés, et enfin l'application web crée.

Chapitre I : Etat de l’art

1. Introduction

Le nombre total d'appareils de l'Internet des objets (IoT) devrait atteindre 80 milliards d'ici 2026. Avec l'adoption généralisée de l'Internet des objets (IoT), des milliards d'objets du quotidien sont connectés à Internet à chaque seconde.

De nouvelles techniques d'analyse du réseau sont nécessaires pour découvrir quels appareils IoT sont connectés au réseau. Dans ce contexte, les techniques d'analyse de données peuvent être exploitées pour découvrir des modèles spécifiques qui peuvent aider à reconnaître les types d'appareils.

Nous allons organiser ce chapitre comme suit : la section 2 présente la notion de l'*Analyse de Trafic Réseau*, les techniques utilisées et les outils et plateformes. La section 3 définit l'*Internet des Objets*, son fonctionnement, la connexion des appareils, ainsi que ses avantages et inconvénients. Ensuite, la section 4 parle sur la classification supervisée et non supervisée ainsi que les principaux algorithmes d'apprentissage. Enfin, nous présentons une étude comparative des travaux récents sur classification des appareils IoT.

2. Analyse du trafic réseau

2.1. Définition

L'analyse du trafic réseau est le processus de capture du trafic réseau et d'inspection minutieuse pour déterminer ce qui se passe sur le réseau. Un analyseur de réseau décode ou dissèque les paquets de données des protocoles communs et affiche le trafic du réseau dans un format lisible par l'homme.

L'analyse du trafic réseau est également connue par plusieurs autres noms : analyse de trafic, analyse de protocole, spoffing (reniflage), analyse de paquets. Le Spoffing ayant tendance à être l'un des termes les plus populairement utilisé aujourd'hui.

L'analyse du trafic fournit des données détaillées et exploitables sur le trafic réseau et la bande passante, ce qui permet de définir et mettre en œuvre des stratégies d'utilisation de la bande passante, de contrôle des frais de FAI (Fournisseur d'accès à

Internet), de sécurité pour le réseau. Elle offre ainsi une vision d'une rare précision sur le trafic [3].

L'analyse du trafic réseau (NTA pour Network Trafic Analysis en Anglais) est un terme qui a été inventé par Gartner qui a publié le premier manuel sur le marché pour analyser le trafic réseau après avoir instauré la "NTA" comme domaine à la fin de 2018. Cette technologie s'appuie sur l'Intelligence Artificielle (IA) via l'apprentissage automatique (ML) et l'analyse comportementale [4].

2.2. Techniques utilisées

Il existe au moins deux façons d'effectuer une analyse du trafic réseau : ***l'analyse des Paquets et l'analyse du flux du trafic réseau.***

Dans les deux techniques, bien entendu, l'objectif est le même : obtenir des informations sur le trafic réseau qui peuvent être présentées sur une interface facilitant son évaluation.

Les différences entre l'une et l'autre se concentrent dans la méthodologie utilisée.

L'analyse des paquets offre la possibilité d'évaluer le trafic réseau paquet par paquet, tandis que l'analyse de flux vise à collecter des métadonnées ou des informations sur le trafic et à travers elles faciliter l'analyse statistique [5].

2.3.1. L'analyse des paquets

L'analyse des paquets commence par l'application de techniques de capture, telles que la configuration des ports SPAN (Switch Port Analyzer) ou l'installation d'équipements tels que les TAP (Terminal Network TAP) pour accéder au trafic réseau.

En réalité, les dispositifs TAP ont été développés pour couvrir certaines lacunes qui surviennent lors de l'application de ports SPAN, comme la dépendance aux ressources de traitement du commutateur où ils sont configurés et la relation délicate entre la quantité de trafic qu'on a l'intention de capturer et la capacité du port SPAN elle-même.

Une fois le problème de capture résolu, deux problèmes très importants se posent [5]:

- **Le stockage du trafic :** il s'agit de savoir si on peut faire une analyse en temps réel ou en temps différé et le coût de stockage que l'analyse implique.
- **La sélection des paquets que l'on veut évaluer :** pour résoudre ce problème, les outils qui implémentent l'analyse des packages offrent généralement de nombreuses fonctionnalités qui permettent de choisir et de sélectionner les packages à évaluer.

Les variables de choix sont généralement nombreuses, dès les adresses IP source et de destination jusqu'à la présence d'une certaine séquence d'octets dans les paquets. Un autre point important à mentionner concernant l'analyse des paquets est celui correspondant au traitement qui est donné à la partie de données des paquets.

2.3.2. L'analyse du flux

L'analyse des flux de trafic propose les éléments suivants :

- **Évaluez le trafic réseau en fonction des caractéristiques communes.** En d'autres termes, on part d'une abstraction – appelée « flux de trafic » – qui correspond à tout le trafic qui partage certaines caractéristiques communes et se déplace d'un hôte réseau à un autre.

Par exemple, si l'on considère tout le trafic qu'une station et un serveur peuvent partager, le trafic faisant partie de la même conversation ou ayant le même objectif sera considéré comme un flux.

- **Le flux n'est pas stocké en tant que tel, uniquement les métadonnées.** L'idée est d'utiliser les périphériques impliqués dans le passage du trafic réseau pour générer des informations sur le flux du trafic ou ses métadonnées, sont stockées dans les paquets qui composent le flux du trafic.

Ces métadonnées doivent ensuite être stockées et retraitées pour être finalement affichées avec l'idée de permettre l'analyse, quels qu'ils soient : supervision, sécurité, facturation, etc.

L'analyse des flux de trafic a été établie sur la base d'un ensemble de protocoles qui permettent la mise en œuvre des processus de génération, de transport, de stockage et de prétraitement des métadonnées.

Il est important de préciser que ces protocoles ne spécifient pas comment l'analyse doit être effectuée. Cela est laissé aux outils qui utilisent les métadonnées pour atteindre leurs objectifs.

Il existe deux protocoles qui représentent deux approches différentes pour mettre en œuvre l'analyse du flux de trafic [5] : ***NetFlow*** et ***sFlow***.

2.3. Outils et plateformes

L'analyse du trafic réseau joue un rôle majeur dans la surveillance de la disponibilité du réseau. Il est également essentiel de surveiller l'activité pour identifier les anomalies. Cela aide à améliorer les performances du réseau. Il peut identifier les goulots d'étranglement d'un réseau et trouver la raison du ralentissement du réseau. Il existe plusieurs outils pour effectuer cette tâche, dans le tableau qui suit nous en présentons quelques-uns :

Outil	Définition
SolarWinds	SolarWinds fournit la solution d'analyse du trafic réseau, NetFlow Traffic Analyzer. Il peut effectuer une analyse approfondie du trafic réseau avec précision. Ses rapports et alertes personnalisables vous aideront à rationaliser l'analyse du trafic réseau. Il peut identifier les points de terminaison et les applications qui génèrent un trafic réseau important et créent des goulots d'étranglement.
Paessler	PRTG Network Analyzer est une solution puissante et conviviale. Il peut analyser tous les éléments de votre réseau. Cela accélérera le dépannage et évitera les goulots d'étranglement. Cela vous aidera à planifier efficacement vos ressources. Il utilise les technologies SNMP, Packet Sniffing, Flow et WMI pour l'analyse. PRTG Network Analyzer vous aidera à identifier rapidement les goulots d'étranglement. Vous pouvez les éliminer et éviter les goulots d'étranglement. Il peut fournir un enregistrement à long terme de vos données réseau.
Wireshark	Wireshark est un analyseur de protocole réseau qui vous donnera des informations détaillées sur ce qui se passe sur votre réseau. De nombreuses entreprises commerciales et à but non lucratif, agences gouvernementales et établissements d'enseignement ont fait de Wireshark une norme de facto. Il effectue une inspection approfondie de centaines de protocoles. Il peut capturer en direct et effectuer une analyse hors ligne. Il prend en charge Windows, Mac, Linux, Solaris, FreeBSD, NetBSD, etc.
NetFort LANGuar	LANGuardian de NetForts est un outil d'inspection approfondie des paquets. Il peut surveiller le réseau et l'activité des utilisateurs. Il dispose de fonctionnalités pour la surveillance des fichiers, la surveillance Web, le dépannage de la bande passante, la capture de paquets, etc. Il peut être un point de référence unique pour la surveillance du réseau et de l'activité des utilisateurs.
Analyseur ManageEngine	ManageEngine est un outil d'analyse du trafic en temps réel. Cela vous donnera une visibilité sur les performances de la bande passante du réseau. Il a effectué une analyse approfondie du trafic. Il utilise la technologie de flux pour fournir une visibilité en temps réel. Il peut collecter, analyser et rendre compte de la bande passante de votre réseau. Il vous aide à optimiser l'utilisation de la bande passante. ManageEngine NetFlow Analyzer vous permettra de suivre les anomalies réseau qui dépassent votre pare-feu réseau. Il identifie les anomalies contextuelles. Il peut collecter et analyser les flux des principaux appareils tels que Cisco, 3COM, Juniper, Foundry Networks, Hewlett-Packard, etc.

Tableau 1 : Les outils de l'analyse de trafic réseau.

3. Internet des objets

Dans cette section, nous allons parler de la notion de l'Internet des Objets :

3.1. Définition

IEEE décrit l'Internet des Objets comme un réseau d'éléments embarqués avec des capteurs se connectant à Internet.

L'Union Internationale des Télécommunications (UIT) définit l'IoT comme une infrastructure globale qui offre des services avancés en interconnectant des objets avec les technologies de communication actuelles. UIT a également mis à jour la définition du système de télécommunication pour l'IoT en ajoutant « Everything », IoE. Cette définition signifie tous types de communication entre les humains, les ordinateurs et les objets (appareils intelligents).

L'Institut national des normes et de la technologie (NIST) définit l'IoT comme une technologie de systèmes cyber-physiques (CPS) afin de connecter des appareils intelligents dans divers secteurs tels que les transports, les soins de santé et l'énergie.

Enfin, Cisco dans le secteur commercial définit l'Internet des Objets « IoT » dans le cadre de l'« IoE » comme la technologie pour connecter les personnes, les processus, les données et les choses qui échangent de l'information, des services ou des expériences [6].

Les paradigmes de l'internet des objets sont résumés dans la figure qui suit :

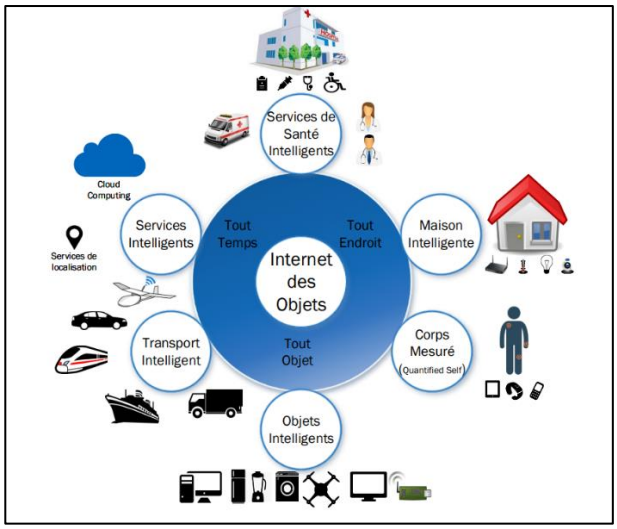

Figure 1 : Les paradigmes de l'Internet des Objets. [7]

3.2. Fonctionnement de l'Internet des Objets

Un système IoT complet se compose de quatre éléments distincts. Capteurs ou appareils, connectivité, traitement des données et interface utilisateur. Examinons-les tour à tour :

- **Capteurs/Appareils :(Collecter)**

Ceux-ci collectent des données à partir de leur environnement. Un appareil peut comporter plusieurs capteurs : par exemple, un smartphone contient un GPS, un appareil photo, un accéléromètre, etc. Essentiellement, le ou les capteurs collectent des données de l'environnement dans un but précis.

- **Connectivité :(Communiquer)**

Une fois que l'appareil a collecté les données, il doit les envoyer au Cloud. Il le fait de différentes façons : Wi-Fi, Bluetooth, satellite, réseaux étendus à faible puissance (LPWAN) ou connexion directe à Internet via Ethernet. L'option de connectivité spécifique dépendra de l'application de l'IoT.

- **Traitement des données :(Exécuter)**

Une fois que les données ont atteint le Cloud, le logiciel les traite et peut décider d'effectuer une action. Il peut s'agir d'envoyer une alerte ou de régler automatiquement les capteurs ou l'appareil sans intervention de l'utilisateur. Cependant, l'utilisateur doit

parfois intervenir, et c'est là que l'interface utilisateur entre en jeu.

- **Interface utilisateur :(Visualiser)**

Si l'utilisateur doit intervenir ou s'il veut vérifier le système, une interface utilisateur le lui permet. Toute action effectuée par l'utilisateur est envoyée en sens inverse par le système. De l'interface utilisateur au Cloud et de nouveau aux capteurs/appareils pour effectuer le changement demandé.

Les protocoles précis de connectivité, de mise en réseau et de communication utilisés par les appareils compatibles avec le Web varient en fonction de chaque application de l'IoT. De plus en plus, l'IoT utilise l'intelligence artificielle (IA) et l'apprentissage automatique pour faciliter et accélérer les processus de collecte de données. [8]

Un système IoT se décompose en 4 fonctionnalités comme la montre la figure ci-dessou

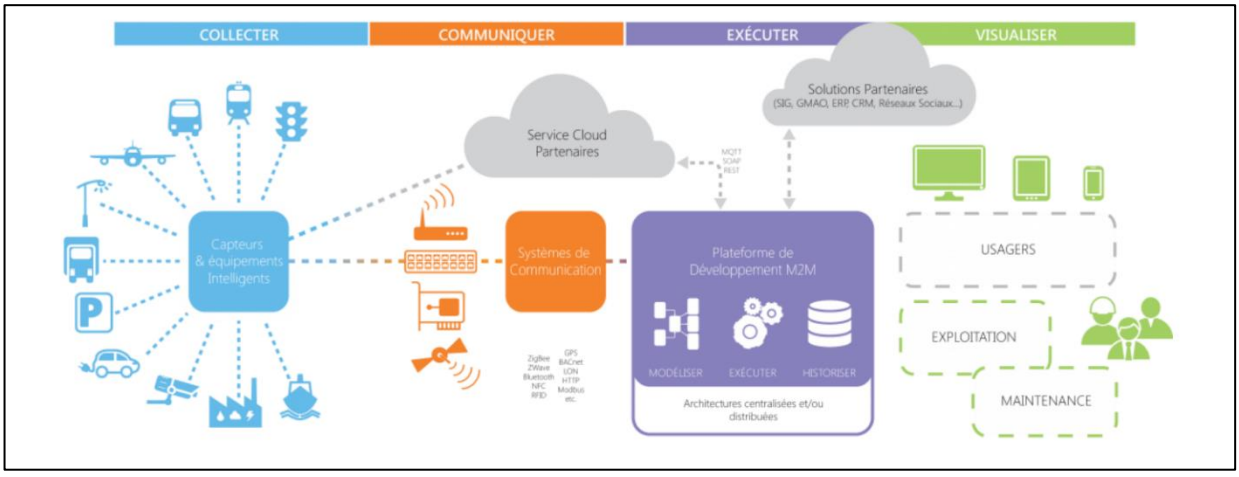

Figure 2 : Les fonctionnalités d'un système IoT. [8]

3.3. Connexion des appareils IoT

Un objet connecté est un appareil composé de capteurs qui envoient des données vers une application mobile ou un service web, pour de multiples domaines d'application. Les objets connectés s'appuient sur les innovations technologiques majeures tel que le Cloud computing, et l'avancée en générale des techniques informatiques. De plus en plus performants, ils peuvent être reliés à des services pour mesurer, partager, suivre, analyser les données quotidiennes de l'individu et lui donner des conseils. Depuis peu

les objets connectés commencent à prendre part à notre vie quotidienne, et nous les retrouvons partout où nous allons [9].

Grâce à l'Internet des objets, des appareils ayant accès à Internet se connectent les uns aux autres et exploitent les données qu'ils s'échangent. Les appareils communiquent via le cloud et se connectent à Internet par le biais d'un réseau Wi-Fi, une connexion cellulaire (3G ou 4G), Bluetooth, NFC, ZigBee, 6LoWPAN, Z-Wave et RFID.

3.4. Avantages et inconvénients

Les objets connectés offrent, sans conteste, de nombreux avantages aux utilisateurs. Notamment, dans certains domaines comme la santé, les objets connectés vont faire des économies aux patients, dans le domaine de la domotique, les objets connectés améliorent considérablement la sécurisation et le contrôle des habitats.

Le principal doute des utilisateurs des objets connectés demeure dans la sécurisation des données personnelles confidentielles, circulant sur Internet, auquel les éditeurs et concepteurs continuent de focaliser leurs efforts et innovations [9].

Avantages	Inconvénients
Efficacité : les interactions entre machines améliorent l'efficacité et permettent aux personnes de gagner du temps pour se concentrer sur d'autres tâches.	**Compatibilité :** en l'absence de normes de compatibilité internationales, les appareils de différents fabricants sont susceptibles de mal communiquer entre eux.
Automatisation : l'automatisation entraîne une uniformisation des tâches, ce qui peut améliorer la qualité des services et réduire le besoin d'intervention humaine.	**Moins d'emplois :** l'accélération de l'automatisation par l'IdO pourrait entraîner le retrait de postes qualifiés du marché du travail.
Réduction des coûts : une efficacité et une automatisation accrues peuvent réduire à la fois les déchets et les coûts de main-d'œuvre, ce qui rend la fabrication et la livraison des marchandises moins coûteuses.	**Complexité :** en raison de la taille du réseau de l'IdO, avec de nombreux appareils qui en dépendent, une seule défaillance logicielle ou matérielle pourrait avoir des conséquences disproportionnées.
Contrôle de la qualité : l'IdO améliore la communication entre les appareils, ce qui permet un meilleur contrôle de la qualité.	**Confidentialité et sécurité :** avec un si grand nombre d'appareils courants connectés à Internet, une quantité considérable d'informations se retrouve en ligne. Il en résulte des risques pour la vie privée et la sécurité.

Tableau 2 : Avantages et Inconvénients de l'Internet des objets.

4. Les algorithmes de classification

Dans cette section, nous allons particulièrement nous intéresser aux différentes techniques informatiques pour la classification.

4.1. Classification automatique

De manière générale, les systèmes d'apprentissage automatique sont des systèmes de classification ou de reconnaissance de formes. Un système d'apprentissage automatique est une boite noire avec une entrée, par exemple une image, un son, ou un texte, et une sortie qui peut représenter la classe de l'objet dans l'image, la parole ou le sujet dans le texte.

Au début, l'IA fait à peine la différence entre l'image d'un chat et celle d'un chien, pour que le programme comprenne ce qu'est un chat, un superviseur apprend au programme que cette image contient un chat. Mais pour progresser, il faut développer le bon algorithme d'apprentissage, et cela prend un temps très important. Aujourd'hui l'apprentissage automatique se développe de plus en plus ; particulièrement dans les systèmes experts ou les super calculateurs avec le grand saut qu'il a fait grâce aux réseaux de neurones artificiels et au Deep Learning.

L'objectif de la Classification automatique est de classer de façon automatique les documents dans des catégories qui ont été définies soit préalablement par un expert, il s'agit alors de classification supervisée ou catégorisation, soit de façon automatique, il s'agit alors de classification non supervisée ou encore clustering [10].

Les différents types d'apprentissage sont résumés dans la (Figure 3).

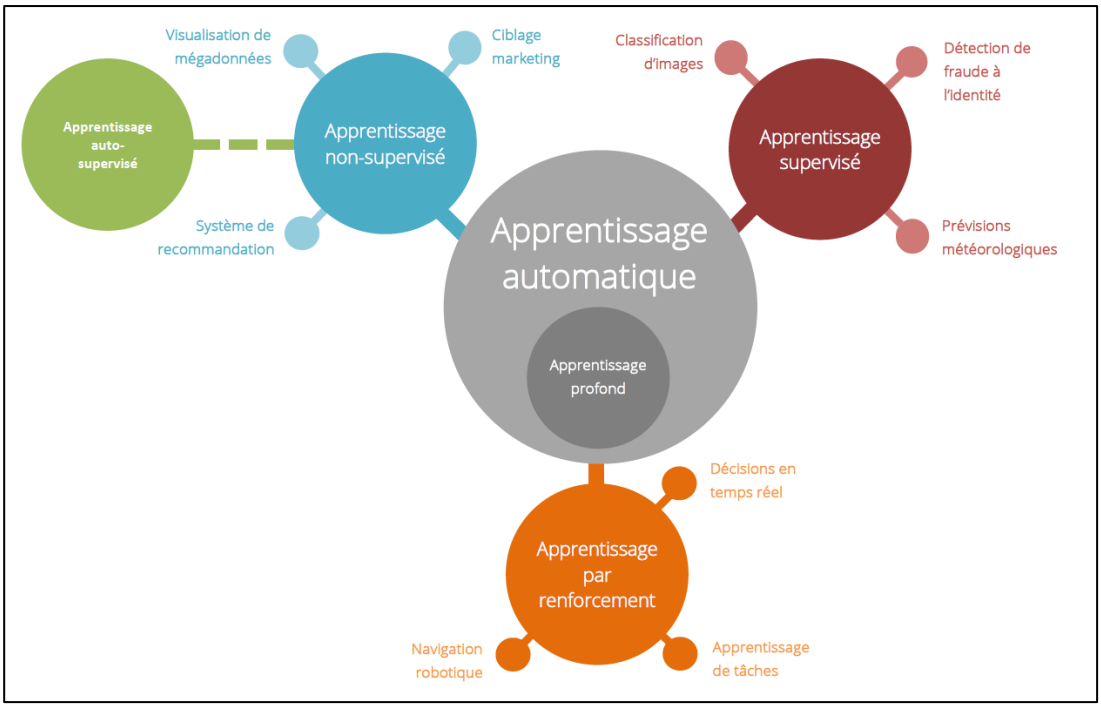

Figure 3 : Les types de l'apprentissage automatique. [11]

4.1.1. Apprentissage supervisé

Dans le cas de l'apprentissage supervisé, le système est guidé dans son apprentissage. On lui indique le type de résultat à atteindre en le nourrissant d'exemples. Pour cela, on lui fournit des données d'entrée pour lesquelles le résultat est connu et communiqué au système.

Le but est qu'il puisse ensuite généraliser ce qu'il a appris pour des données non connues. Par exemple, si le système doit apprendre à reconnaitre des feuilles de vigne dans une image, on lui fournit des images ou la feuille est signalée et ou le label « feuille de vigne » est associé. On parle ainsi de données étiquetées. Le jeu de données d'entrainement annotées permet au système de calculer ses erreurs en comparant ses résultats avec les résultats connus et ainsi d'ajuster le modèle pour progresser. Une partie des données annotées (non utilisées pendant l'entrainement) pourra également servir à vérifier l'efficacité du modèle, une fois l'apprentissage terminé.

La figure 4 décrit le fonctionnement d'un système d'apprentissage supervisé.

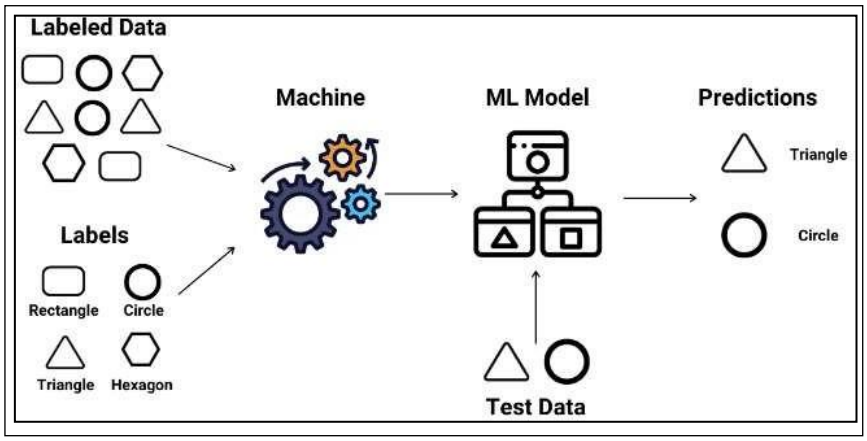

Figure 4 : Schéma général d'un système d'apprentissage supervisé. [12]

4.1.2. Apprentissage non supervisé

Dans le cas de l'apprentissage non supervisé, on ne donne pas d'exemples de résultats attendus au système. Seules les données d'entrée sont fournies et le système doit apprendre, de façon autonome, la meilleure façon d'explorer les données. Il doit chercher à identifier dans le jeu de données une façon de les structurer (trouver des modèles ou 'patterns') ou encore à extraire des caractéristiques. La performance permettant l'ajustement du modèle est alors appréciée grâce à des indicateurs objectifs, par exemple, des calculs de variabilité intra ou interclasses.

La figure 6 décrit le fonctionnement d'un système d'apprentissage non supervisé :

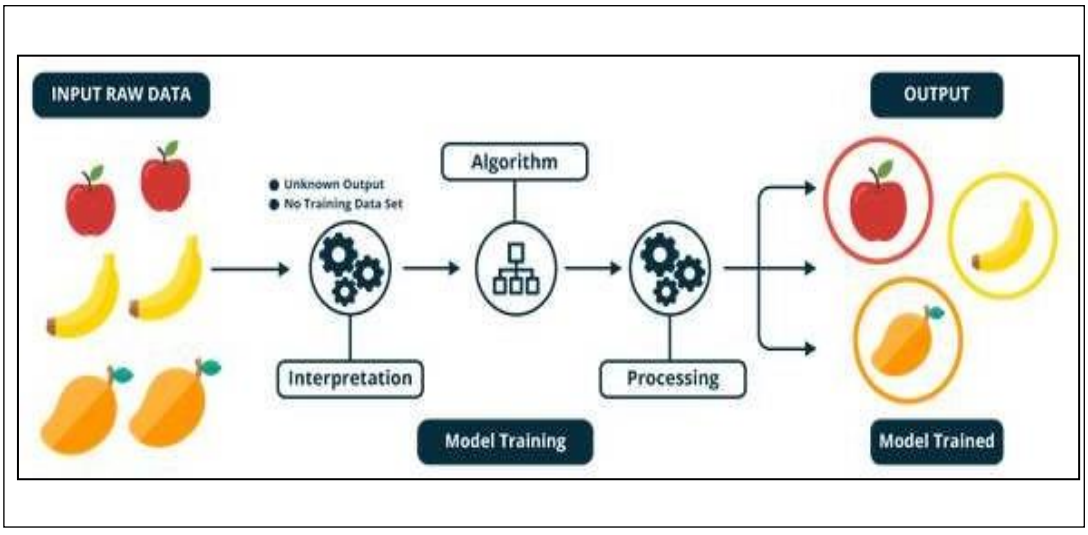

Figure 5 : Schéma général d'un système d'apprentissage non supervisé. [12]

4.1.3. Avantages et inconvénients

Parmi les avantages et inconvénients liés aux deux approches, on peut citer [10]:

- Les groupes ou clusters obtenus par la technique supervisée est de meilleure qualité et plus précise que la technique non-supervisée.
- Dans la technique supervisée, on sait ce qui est attendu favorisant de meilleurs résultats par rapport au non supervisée.
- Un avantage des techniques non supervisées, est qu'elles accomplissent la tâche de similarité sans avoir besoin des données expertisées.
- Un inconvénient des approches supervisées, repose sur le fait qu'il peut être difficile de se procurer des données expertisées.
- L'inconvénient majeur des approches non supervisées est qu'elles demandentdans l'étape d'évaluation des résultats l'intervention d'un expert.

4.1.4. Algorithmes d'apprentissage

Pour créer un modèle d'apprentissage supervisé, on peut recourir à différents algorithmes, par exemple :

- Les arbres de décision avec différentes variables de sortie,
- Les machines à vecteur de support (SVM).

On peut également évoquer l'algorithme k-NN pour réaliser des tests et la classification naïve bayésienne pour catégoriser des volumes de données importants.

L'apprentissage non supervisé comprend deux principales catégories d'algorithmes : les algorithmes de clustering et d'association.

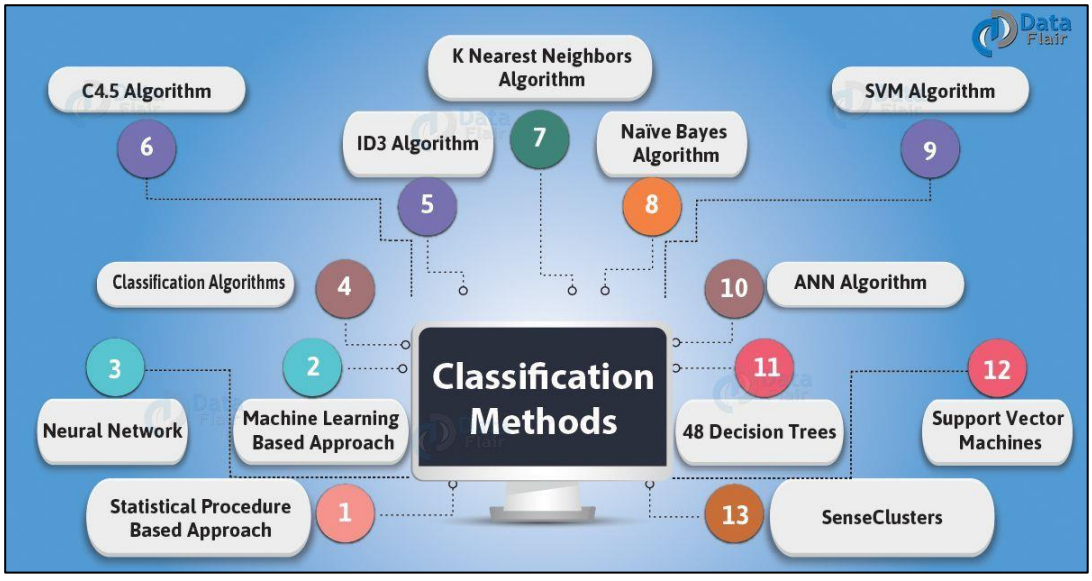

Figure 6 : les algorithmes de classification.

5. Travaux connexes

Le marché croissant de l'Internet des objets (IoT) introduit de nouveaux défis pour la surveillance de l'activité du réseau. La surveillance du travail en réseau héritée n'est pas conçue pour faire face à la grande diversité des appareils intelligents. De nouvelles techniques de découverte de réseau sont nécessaires pour découvrir quels appareils IoT sont connectés au réseau. Dans ce contexte, les techniques d'analyse de données peuvent être exploitées pour découvrir des modèles spécifiques qui peuvent aider à reconnaître les types d'appareils [13].

Après avoir présenté l'analyse du trafic réseau, l'internet des objets et les notions de base de la classification. Cette section présente une synthèse des travaux antérieurs qui s'articule sur l'identification des appareils IoT au sein d'un réseau en analysant et en classant les données du trafic du réseau.

Les travaux s'articulant sur la classification du trafic réseau sont assez nombreux. Parmi les solutions potentielles dans ce cadre, on trouve celles proposée par [14] [15] [2] [13] [16], ils utilisent les caractéristiques de flux, l'apprentissage supervisé ***(KNN, DT, RF, XGboost, SVM et NB)*** et l'apprentissage profond ***(ANN, LSTM-CNN et AdaBoost).***

D'autres travaux récents dans notre domaine, proposent des algorithmes personnalisés de classification des appareils IOT à l'aide de l'analyse du trafic réseau.

Chacun des travaux auxquels nous nous sommes intéressées propose une classification pour leur dataset et leurs appareils connectés. Par exemple **(Mustafizur, R. Shahid et autres)** dans [13] proposent une classification d'apprentissage utilisant les caractéristiques de flux bidirectionnels à travers les algorithmes ***(RF, DT, SVM, KNN, ANN et GNB)*** pour un ensemble réduit de types d'appareils ***(détecteurs de mouvement, caméra de sécurité Nest, ampoule intelligente et prise intelligente).***

Dans le cadre de nos recherches, nous avons étudié plusieurs travaux et avons cinq (05) Travaux récents qui nous ont particulièrement intéressés :

Travaux connexes :

Recherche	*Année*	*Caractéristiques*	*Classifieur*	*Topologie réelle*	*Performances*
Traffic Classification of Home Network Devices using Supervised [14]	2022	Les caractéristiques de flux	-DT -KNN -LDA -MLP -QDA -Adaboost -GB -RT -XGBoost	-PC -Appareils mobiles (smartphones, tablettes) -Appareils intelligents IoT (appareils électroménagers, appareils photos, Concentrateurs de contrôleur, Moniteur de santé…)	Exactitude : 99% F1-mesure : 99%
Automatic Device Classification from Network [2]	2016	Les caractéristiques de flux	-SVM - KNN -DT -RF -AdaBoost -LDA -LSTM-CNN	-Hubs -Switches & triggers -Electronics -Baby Monitor -Cameras	Exactitude : 74.8%
A Group-Based IoT Devices Classification Through Network Traffic Analysis Based on Machine Learning Approach [15]	2020	Les caractéristiques de flux	-ANN -DT -RF - SVM -KNN -GNB	-Multimédias -Hubs -switches et triggers -Cameras -Air quality sensor -Bulbs -Healthcare	Exactitude : 99.98%

IoT Devices Recognition Through Network Traffic Analysis [13]	2018	Les caractéristiques de flux bidirectionnels	- RF -DT -SVM -KNN -ANN -GNB	-montion-sensors -nest-security-camers -smart-bulb -smart-pulg	Exactitude : 99.9%
ProfilIoT: a machine learning approach for IoT device identification based on network traffic analysis [16]	2017	Les caractéristiques de flux	-XGBoost -RF	-pc /laptop -smartphones -appareils connectées avec Wi-Fi (moniteurs, impriment, capteurs, montres intelligentes, Réfrigérateur, TV, camera de sécurité …)	Exactitude : 99.281%

Tableau 3 : travaux sur classification des appareils IoT grâce a l'analyse de trafic réseau.

6. Conclusion

Dans ce chapitre, nous avons défini les notions connexes à notre sujet de recherche, à savoir : l'Analyse du trafic réseau, l'Internet des objets, la classification automatique, et finir par les travaux connexes similaires à notre projet.

Dans le chapitre suivant, nous allons parler sur la conception de notre solution.

Chapitre II : Architectures neuronales et Conception de la solution

1. Introduction

Les appareils IoT grand public comprennent les maisons intelligentes, les appareils électroménagers intelligents, les jouets intelligents et les appareils portables intelligents alors pour cela il est nécessaire de faire une classification pour indiquer le type d'appareil grâce à l'analyse de trafic réseau.

Dans ce chapitre, nous présenterons l'architecture globale du système, le jeu de données utilisé, les algorithmes qui on a testés et les mesures de performances pour la classification des appareils IoT avec le trafic réseau. Premièrement, dans la section 2, nous allons décrire l'architecture globale du système, dataset, algorithmes testés et mesures de performances, après dans la section 3, nous allons définir la conception sommaire de notre plateforme finale.

2. Architecture globale du système

Voici une architecture représentative de notre système :

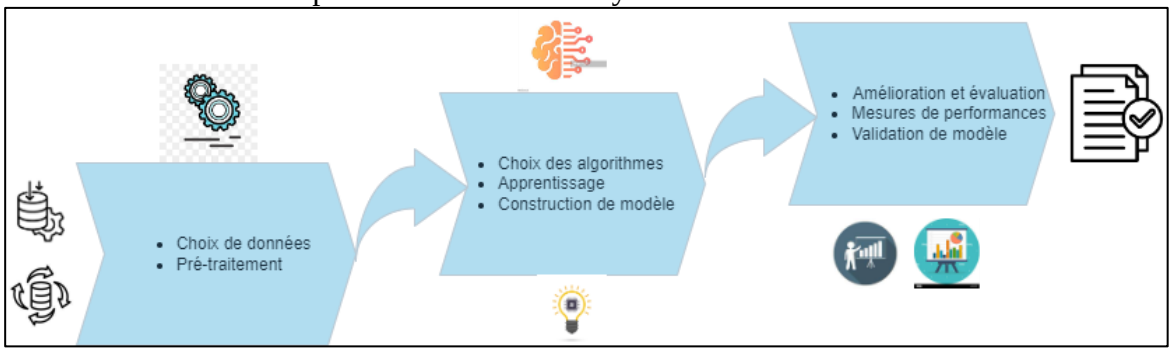

Figure 7 : architecture de système.

L'objectif de ce projet est faire une classification qui indique les types d'appareil IoT.

Pour nos tests, nous avons retenus les algorithmes les plus cités dans la littérature en termes d'apprentissage supervisé (***KNN, DT, RF, XGboost, SVM et NB***) et d'apprentissage profond (***ANN, LSTM-CNN et AdaBoost***)

2.1. Collection de données

Pour créer un modèle de Machine Learning, le premier prérequis est d'avoir un dataset annoté. Sans données, il est impossible de créer ou de mesurer les performances d'un modèle d'apprentissage alors pour cela, dans un premier temps nous nous sommes concentrées sur l'étude des datasets suivants afin de comprendre comment ils sont constitués ? quelles sont les données qu'ils contiennent ? Comment ces dernières peuvent être exploitées :

- ***dataset_sdn* :** contient 104345 lignes et 23 colonnes, Il existe une variable cible appelée label qui contient seulement 1(malveillant) et 0(bénin). La tâche principale est de classer si le trafic est normal ou non en utilisant des algorithmes classiques d'apprentissage automatique [17].
- ***UNSW-NB15* :** est un ensemble de données de trafic réseau qui comprend des données de trafic normal et d'attaque. Il contient plus de 2 millions d'enregistrements et couvre 49 protocoles réseau différents [18].
- **KDD :** il a été divisé en deux ensembles de données différents : un ensemble de Train appelé "KDD_Train" qui possède une taille de 125972*42 et un ensemble de test appelé " KDD_Test " avec une taille de 22544*42.
- **Dataset-Unicauca-Version2-87Atts :** Ce jeu de données contient 87 fonctionnalités. Chaque instance contient les informations d'un flux IP généré par un périphérique réseau, c'est-à-dire les adresses IP source et destination, les ports, les heures intervalles [19].
- ***100_each :*** Cette dataset a été préalablement divisé en deux ensembles de données différents : un ensemble de train appelé "***100_each_train***" avec une taille de 1000*300 et qui est utilisé pour entraîner le modèle sur une partie des données et un ensemble de test appelé "***100_each_test***" avec la taille de 1000*300 et qui est utilisé pour évaluer les performances de ce modèle sur l'autre partie des données [20].

Nous avons d'abord opté pour ce dernier à cause de sa richesse mais en raison desa petite taille, nous avons essayé de faire une augmentation des données pour faire croître les données à une taille satisfaisante pour les algorithmes souhaités. Malheureusement, cette procédure a conduit à des classifications erronées à cause de la génération aléatoires et non toujours cohérentes des nouvelles données.

Par ailleurs, notre souci majeur dans le choix du dataset approprié était d'avoir les caractéristiques permettant d'identifier le type d'un appareil IOT connecté.

Nos recherches nous ont enfin conduits à dataset public **« Aalto_IoTDevID »[2].** Ce jeu de données se compose de deux fichiers csv **« Aalto_train_IoTDevID »** et **« Aalto_test_IoTDevID ».** Il contient 96 caractéristiques parmi lesquelles : la taille des paquets, les ports, les protocoles tcp_ack, et surtout un label indiquant le type de l'appareil ce qui en faisait le dataset idéal pour nous. A figure qui suit résume la liste des appareils du dataset avec la technologie qu'ils utile

Identifier	Device Model	WiFi	ZigBee	Ethernet	Z-Wave	Other
Aria	Fitbit Aria WiFi-enabled scale	●	○	○	○	○
HomeMaticPlug	Homematic pluggable switch HMIP-PS	○	○	○	○	●
Withings	Withings Wireless Scale WS-30	●	○	○	○	○
MAXGateway	MAX! Cube LAN Gateway for MAX! Home automation sensors	○	○	●	○	●
HueBridge	Philips Hue Bridge model 3241312018	○	●	●	○	○
HueSwitch	Philips Hue Light Switch PTM 215Z	○	●	○	○	○
EdnetGateway	Ednet.living Starter kit power Gateway	●	○	○	○	●
EdnetCam	Ednet Wireless indoor IP camera Cube	●	○	●	○	○
EdimaxCam	Edimax IC-3115W Smart HD WiFi Network Camera	●	○	●	○	○
Lightify	Osram Lightify Gateway	●	●	○	○	○
WeMoInsightSwitch	WeMo Insight Switch model F7C029de	●	○	○	○	○
WeMoLink	WeMo Link Lighting Bridge model F7C031vf	●	●	○	○	○
WeMoSwitch	WeMo Switch model F7C027de	●	○	○	○	○
D-LinkHomeHub	D-Link Connected Home Hub DCH-G020	●	○	●	●	○
D-LinkDoorSensor	D-Link Door & Window sensor	○	○	○	●	○
D-LinkDayCam	D-Link WiFi Day Camera DCS-930L	●	○	●	○	○
D-LinkCam	D-Link HD IP Camera DCH-935L	●	○	○	○	○
D-LinkSwitch	D-Link Smart plug DSP-W215	●	○	○	○	○
D-LinkWaterSensor	D-Link Water sensor DCH-S160	●	○	○	○	○
D-LinkSiren	D-Link Siren DCH-S220	●	○	○	○	○
D-LinkSensor	D-Link WiFi Motion sensor DCH-S150	●	○	○	○	○
TP-LinkPlugHS110	TP-Link WiFi Smart plug HS110	●	○	○	○	○
TP-LinkPlugHS100	TP-Link WiFi Smart plug HS100	●	○	○	○	○
EdimaxPlug1101W	Edimax SP-1101W Smart Plug Switch	●	○	○	○	○
EdimaxPlug2101W	Edimax SP-2101W Smart Plug Switch	●	○	○	○	○
SmarterCoffee	Smarter SmarterCoffee coffee machine SMC10-EU	●	○	○	○	○
iKettle2	Smarter iKettle 2.0 water kettle SMK20-EU	●	○	○	○	○

Figure 8 : liste des appareils avec les technologies utilisés. [21]

Avant de pouvoir tester les différents classifieurs, il faut d'abord effectuer une étape très importante qui est le ***prétraitement*** des données. Durant cette étape, les données passent par une série de tâches parmi lesquelles :

Nettoyage des données : Nous avons nettoyé les données en supprimant les lignes possédant des valeurs nulles.

Le tableau ci-dessous, présente le nombre d'occurrences de « entrainement » et d' « apprentissage » pour chaque appareil :

[2] https://github.com/kahramankostas/IoTDevIDv2

Type d'appareil	Occurrence train	Occurrence test	Type d'appareil	Occurrence train	Occurrence test
HueSwitch	14649	3799	**EdimaxPlug2101W**	764	246
HueBridge	11043	2893	**D-LinkDayCam**	849	214
D-LinkHomeHub	6899	1696	**EdimaxCam**	666	165
WeMoLink	5265	1360	**TP-LinkPlugHS100**	524	143
D-LinkSensor	5204	1345	**EdnetGateway**	546	137
D-LinkSwitch	5199	1320	**Withings**	555	133
D-LinkCam	4994	1250	**TP-LinkPlugHS110**	512	124
D-LinkWaterSensor	5192	1243	**MAXGateway**	451	116
D-LinkSiren	4949	1237	**HomeMaticPlug**	507	104
WeMoInsightSwitch	4735	1227	**Aria**	351	90
WeMoSwitch	3667	810	**EdnetCam**	330	78
Lightify	3375	774	**SmarterCoffee**	121	28
D-LinkDoorSensor	1534	358	**IKettle2**	117	28
EdimaxPlug1101W	896	264			

Tableau 4 : Nombre d'occurrences d'entrainement et de tests pour chaque appareil.

Nous avons testé six algorithmes de classification par apprentissage supervisé (***KNN, DT, RF, XGboost, SVM et NB***) puis trois algorithmes d'apprentissage profond (***ANN, LSTM-CNN et AdaBoost*).** Puis, nous les avons évalué grâce aux mesures de performance comme : **F1-*score*, *rappel*, *précision, accuracy, classification report*** et ***matrice de confusion*** pour savoir lequel seraitmeilleur.

2.2. Algorithmes testés

2.2.1. K- plus proches voisins

KNN (pour K Nearest Neighbors en Anglais) est un type d'algorithme d'apprentissage automatique supervisé qui peut être utilisé à la fois pour les problèmes de classification et de régression [22].

Avec le principe sous-jacent « *dis-moi qui sont tes amis, je te dirai qui tu es* », Il classe les nouvelles instances en utilisant les informations fournies par les ***k*** plus proches voisins afin que la classe attribuée soit la plus courante parmi les autres (vote majoritaire). Il stocke toutes les données d'entraînement à l'aide de la fonction de distance. La plus utilisée étant la distance euclidienne [23] et qui est définie parla fonction suivante [24]

$$D\left((x1,\ldots,xp),(u1,..,up)\right) = \sqrt{((x1-u1)^2+\cdots(xp-up)^2)}$$

Paramètre	Explication
n_neighbors	Le nombre des voisins.
Algorithm	Algorithme utilisé pour calculer les plus proches voisins, {'auto', 'ball_tree', 'kd_tree', 'brute'} default='auto'
Weights	Fonction de pondération utilisée dans la prédiction. Valeurs possibles : {'uniform', 'distance'}, default='uniform'

Tableau 5 : Les paramètres de l'algorithme KNN.

Un exemple de classification KNN est illustré par la **figure 9** où le nouveau point étoile appartient soit à la première classe carrée ou bien la deuxième classe hexagone.

- Si K =3 le nouveau point est classé en deuxième classe parce qu'il y a deux hexagones et un seul carré parmi les trois plus proches exemples à l'intérieur du cercle
- Si K=5 il est classé dans la première classe

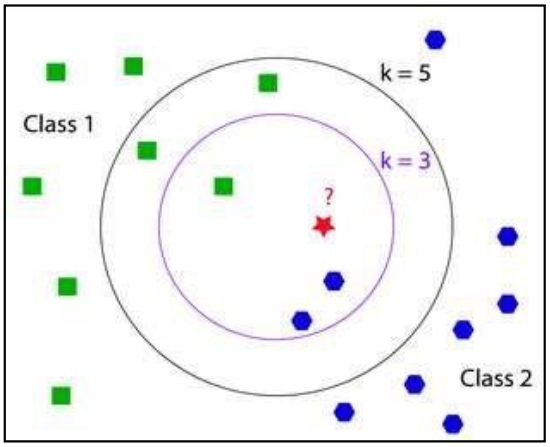

Figure 9 : Exemple de classification KNN.

2.2.2. Réseau de Neurones Artificiels

Les Algorithmes à Réseaux de Neurones Artificiels **(ANN)** sont des algorithmes basés sur la fonction cérébrale et sont utilisés pour modéliser des modèles complexes et prévoir des problèmes. Le réseau de neurones artificiels (ANN) est une méthode d'apprentissage en profondeur issue du concept de réseaux de neurones biologiques du cerveau humain.

Le développement de l'ANN est le résultat d'une tentative de reproduire le fonctionnement du cerveau humain. Le fonctionnement des ANN est extrêmement similaire à celui des réseaux de neurones biologiques, bien qu'ils ne soient pas identiques. L'algorithme ANN n'accepte que des données numériques et structurées.

Dans l'algorithme ANN les paramètres à fixer sont :

Paramètre	**Explication**
Random_state	Entier, instance de RandomState, par défaut=Aucun. Détermine la génération de nombres aléatoires pour les pondérations et l'initialisation du biais.
Solver	{'lbfgs', 'sgd', 'adam'}, default='adam' Le solveur pour l'optimisation du poids.
Alpha	Flottant, par défaut=0.0001 Force du terme de régularisation L2. Le terme de régularisation L2 est divisé par la taille de l'échantillon lorsqu'il est ajouté à la perte.
Hidden_layer_size	Default= (100,) Le ième élément représente le nombre de neurones dans la ième couche cachée.

Tableau 6 : Les paramètres de l'algorithme ANN..

2.2.3. Arbres de décisions

Les DT (pour Decision Trees en Anglais) sont une technique d'apprentissage supervisé qui peut être utilisée pour les problèmes de classification et de régression.

Le DT applique une idée rigoureuse pour résoudre le problème de classification en posant une série de questions soigneusement conçues sur les attributs du test. Chaque fois qu'il reçoit une réponse, une autre question est posée jusqu'à ce qu'une conclusion soit atteinte concernant l'étiquette de la classe du dossier. Un arbre de décision permet de créer un modèle d'apprentissage qui peut être utilisé pour prédire la classe ou la valeur de la variable cible en apprenant des règles de décision simples déduites de données antérieures (données d'entraînement). Dans les arbres de décisions, pour prédire une classe, la méthode commence de parcourir l'arbre de la racine. Ensuite elle compare les valeurs de l'attribut racine avec l'attribut de l'enregistrement. Sur la base de la comparaison, nous suivons la branche correspondant à cette valeur et passons au nœud suivant [25]

Dans l'algorithme DT, les paramètres à fixer sont :

Paramètre	Explication
Criterion	La fonction (« gini » ou « entropy ») à prendre en compte pour calculer l'incertitude sur la règle de discrimination utilisée.
Splitter	***(« best » ou « random »)*** – la stratégie utilisée pour choisir le feature sur lequel créer une règle de discrimination. La valeur par défaut est « best ».
Max_features	***(Nombre entier)*** Le nombre de feature à prendre en compte lors de la recherche du meilleur split (de la meilleure règle de discrimination).
Random_state	***(Nombre entier)*** – permet de contrôler l'aléatoire dans l'algorithme. Si aucun chiffre n'est indiqué, l'algorithme lui en donne un au hasard à chaque nouvel entraînement.
Max_depth	**(Nombre entier)** – la profondeur maximale de l'arbre.

Tableau 7 : Les paramètres de l'algorithme Arbre de décisions.

2.2.4. Modèle LSTM-CNN

Il s'agit d'un réseau de neurones convolutifs dont la sortie finale est envoyée directement dans un LSTM qui va alors réaliser sa prédiction.

L'idée derrière est de se dire que le CNN est un algorithme très efficace pour analyser des images, donc on va l'utiliser pour traiter la séquence vidéo et la rendre plus simple et explicite pour le LSTM. Lors de la phase d'entraînement, le CNN et le LSTM s'adapteront aux données pour que le premier comprenne quoi extraire et que le second saisisse ce qu'il doit analyser/mémoriser d'une séquence à l'autre [26].

L'architecture CNN LSTM implique l'utilisation de couches de réseau neuronal convolutif (CNN) pour l'extraction de caractéristiques sur les données d'entrée combinées avec des LSTM pour prendre en charge la prédiction de séquence. Les LSTM CNN ont été développés pour les problèmes de prédiction de séries chronologiques visuelles et l'application de la génération de descriptions textuelles à partir de séquences d'images (par exemple, des vidéos). Plus précisément, les problèmes de [27]:

- **Reconnaissance d'activité :** génération d'une description textuelle d'une activité démontrée dans une séquence d'images.
- **Description de l'image :** génération d'une description textuelle d'une seule image.
- **Description vidéo :** génération d'une description textuelle d'une séquence d'images.

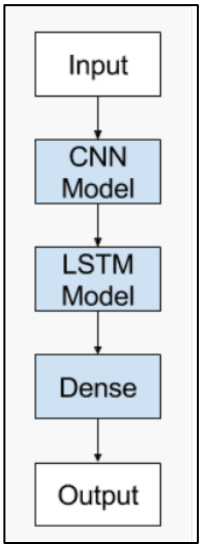

Figure 10 : Architecture de LSTM-CNN. [27]

Dans l'algorithme LSTM_CNN, les paramètres à fixer sont :

Paramètre	Explication
batch_size	Combien d'échantillons dans chaque lot pendant la formation et les tests
Timesteps	Signifie combien de valeurs existent dans une séquence.
Filtres	Entier, la dimensionnalité de l'espace de sortie (c'est-à-dire le nombre de filtres de sortie dans la convolution).
kernel_size	Un entier ou tuple/liste de 2 entiers, spécifiant la hauteur et la largeur de la fenêtre de convolution 2D.
POOL_SIZE	Le nombre de tampons réservés en mémoire pour conserver les données reçues de l'hôte jusqu'à ce qu'elles puissent être traitées. Chaque mémoire tampon est de la taille de la PIU.

Tableau 8 : Les paramètres de l'algorithme LSTM-CNN.

2.2.5. Extreme Gratient boosting

Le classificateur **XGBoost** est un algorithme d'apprentissage automatique appliqué aux données structurées et tabulaires. Il implémente des arbres de décision boostés par gradient conçus pour la vitesse et les performances, le principe est de combiner les résultats d'un ensemble de modèles dans le but de fournir une meilleure prédiction. De plus, il s'agit d'un algorithme d'amplification de gradient extrême, ce qui signifie qu'il s'agit d'un algorithme d'apprentissage automatique avec un large éventail d'applications. Les applications incluent la résolution de problèmes concernant la classification [2 8] .

L'algorithme XGBoost travaille de manière séquentielle, il améliore le boosting de gradient en matière d'échelle et de vitesse de calcul de différentes façons.

Dans l'algorithme XGBoost, les paramètres à fixer sont :

Paramètre	Explication
Booster	[Défaut= gbtree] Quel booster utiliser. Peut-être gbtree, gblinear ou dart ; gbtree et dart utilisent des modèles basés sur des arbres tandis que gblinear utilise des fonctions linéaires.
Verbosity	[Défaut=1] Verbosité des messages d'impression. Les valeurs valides sont 0 (silencieux), 1 (avertissement), 2 (info), 3 (débogage).
validate_parameters	[Valeur par défaut à false, sauf pour les interfaces Python, R et CLI] Lorsqu'il est défini sur True, XGBoost effectuera la validation des paramètres d'entrée pour vérifier si un paramètre est utilisé ou non.

Nthread	Nombre de threads parallèles utilisés pour exécuter XGBoost.
disable_default_eval_metric	Indicateur pour désactiver la métrique par défaut. Défini sur 1 ou vrai pour désactiver [défaut = faux]

Tableau 9 : Les paramètres de l'algorithme XGBoost.

2.2.6. Algorithme Forêts Aléatoires

Le modèle **RF** (pour Random Forest en Anlgais) est un algorithme d'apprentissage automatique couramment utilisé, qui permet d'assembler les analyses de différents arbres de décision pour atteindre un résultat unique.

L'estimation finale consiste à choisir la catégorie de réponse la plus fréquente. Plutôt qu'utiliser tous les résultats obtenus, on procède à une sélection en recherchant la prévision qui revient le plus souvent [29].

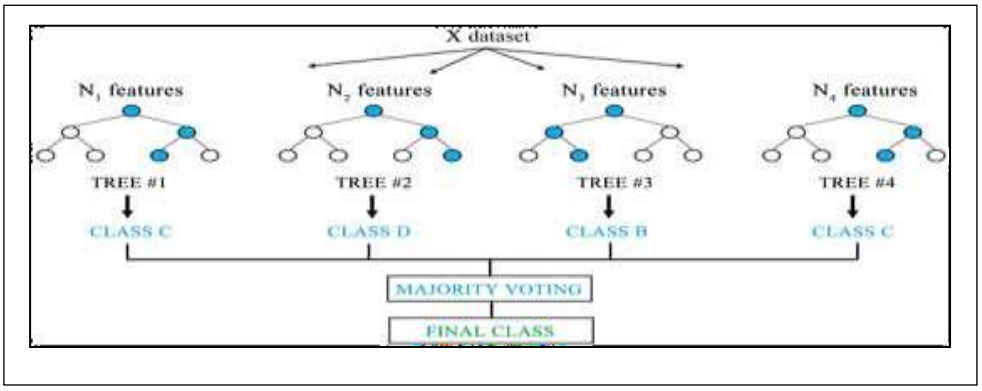

Figure 11 : Classification Random Forest.

Dans l'algorithme RF, les paramètres à fixer sont :

Paramètre	Explication
n_estimators	Entier, par défaut=100. Le nombre d'arbres dans la forêt.
Criterion	{"gini", "entropy", "log_loss"}, default=" gini" La fonction pour mesurer la qualité d'une division.
Max_depth	Entier, défaut=Aucun. La profondeur maximale de l'arbre.
Min_samples_split	int ou float, par défaut=2. Le nombre minimum d'échantillons requis pour diviser un nœud interne

Tableau 10 : Les paramètres de l'algorithme Forêts Aléatoires.

2.2.7. Machine à vecteurs de support

L'algorithme **SVM** (Pour Support Vecotr Machine en Anglais) est un type d'algorithme d'apprentissage automatique qui est principalement utilisé pour résoudre des tâches de classification supervisée. Il consiste en la séparation linéaire des données projetées dans un espace en utilisant un hyperplan optimal. Cette séparation est effectuée en maximisant la marge entre les données de chaque classe : l'hyperplan optimal doit être le plus éloigné des données des différentes classes tout en les séparant.

Pour avoir des résultats pertinents il faut que les données soient linéairement séparables dans le plan de projection. Si ce n'est pas le cas, l'utilisation de différents noyaux va permettre de modifier l'espace et donc la répartition des données dans cet espace. On peut par exemple voir la projection de données 2D en 3D, permettant de rendre certaines données linéairement séparables.

Cette méthode est très utilisée en classification, étant assez rapide à apprendre. De plus, elle est très performante lorsque l'on possède peu de données d'apprentissage [30]

Voici un schéma représentatif (**figure 12**) :

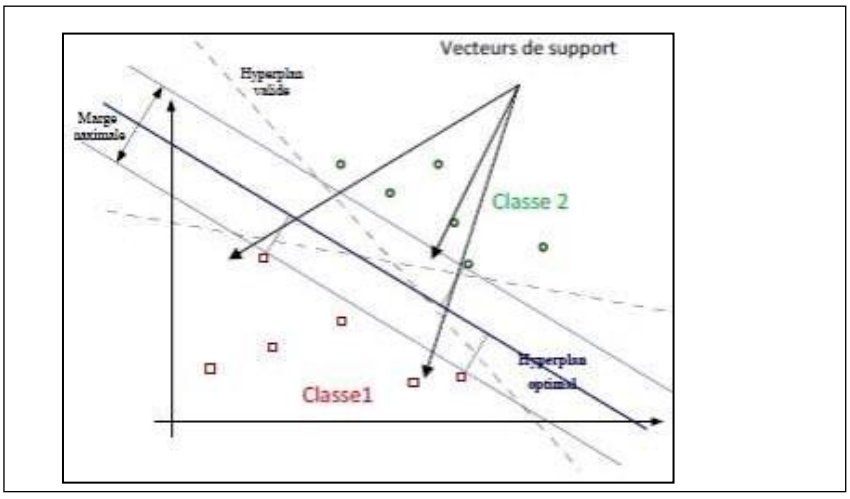

Figure 12 : les vecteurs à support.

Dans l'algorithme SVM, les paramètres à fixer sont :

Paramètre	Explication
decision_function_shape	L'option permet de transformer de manière monotone les résultats des classificateurs " one-versus-one " en une fonction de décision " one-vs-rest " de forme (n_samples, n_classes).
C	float, par défaut=1.0. Paramètre de régularisation. La force de la régularisation est inversement proportionnelle à C. Doit être strictement positive. La pénalité est une pénalité de l2 au carré.
Kernel	{'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'} ou callable, default='rbf'. Spécifie le type de noyau à utiliser dans l'algorithme.
Degré	Entier, défaut=3. Degré de la fonction noyau polynomiale ('poly'). Doit être non négatif. Ignoré par tous les autres noyaux.
Gamma	{'scale', 'auto'} ou float, default='scale' Coefficient de noyau pour 'rbf', 'poly' et 'sigmoïde'.

Tableau 11 : Les paramètres de l'algorithme SVM.

2.2.8. Naïve Bayes

Le classificateur Naïve Bayes (NB) est un algorithme d'apprentissage automatique supervisé, qui est utilisé pour les tâches de classification, [31] , la structure de naïve bayes a prouvé expérimentalement qu'elle était capable de donner de bons résultats, l'hypomémoire de base de ce modèle est de supposer que toutes les observations étaient indépendantes les unes des autres conditionnellement a la variable classe , ce qui revient à une simplification de la loi jointe suivante [25]

$$P(A|B) = \frac{P(A).P(B|A)}{P(B)}$$

Où :

A **et** B **:** Événements et $(B) > 0$

$(A|B)$ **:** Probabilité que l'événement A se produise sachant que l'événement B s'estproduit.

P(B|A): Probabilité que l'événement **B** se produise sachant que l'événement **A** s'est produit.

(A) **et** (B) **:** Probabilités des événements A et B indépendamment les uns des autres.

Dans l'algorithme NB, les paramètres à fixer sont :

Paramètre	Explication
Priors	Type tableau de forme (n_classes,), défaut=Aucun
var_smoothing	float, défaut=1e-9. Portion de la plus grande variance de toutes les caractéristiques qui est ajoutée aux variances pour la stabilité du calcul.

Tableau 12 : Les paramètres de l'algorithme Naïve Bayes.

2.2.9. Algorithme ADaBoost

Ada-boost ou Adaptive Boosting est l'un des classificateurs de renforcement d'ensemble proposés par Yoav Freund et Robert Schapire en 1996. Il combine plusieurs classificateurs pour augmenter la précision des classificateurs. AdaBoost est une méthode d'ensemble itérative. Le classificateur AdaBoost crée un classificateur puissant en combinant plusieurs classificateurs peu performants afin d'obtenir un classificateur puissant de haute précision. Le concept de base derrière Adaboost est de définir les poids des classificateurs et de former l'échantillon de données à chaque itération de manière à garantir des prédictions précises d'observations inhabituelles. Tout algorithme d'apprentissage automatique peut être utilisé comme classificateur de base s'il accepte des pondérations sur l'ensemble d'apprentissage. Adaboost doit remplir deux conditions [32]:

- Le classificateur doit être entraîné de manière interactive sur divers exemples d'entraînement pondérés.
- À chaque itération, il essaie de fournir un excellent ajustement pour ces exemples en minimisant l'erreur d'apprentissage.

Les paramètres les plus importants sont :

Paramètre	Explication
base_estimator	Il s'agit d'un apprenant faible utilisé pour entraîner le modèle. Il utilise DecisionTreeClassifier comme apprenant faible par défaut à des fins de formation. Vous pouvez également spécifier différents algorithmes d'apprentissage automatique.
n_estimators	Nombre d'apprenants faibles à former de manière itérative.
learning_rate	Il contribue au poids des apprenants faibles. Il utilise 1 comme valeur par défaut.

Tableau 13 : Paramètres de l'algorithme ADaBoost.

2.3. Mesures de performances

Une fois qu'un modèle est formé, il doit être testé pour vérifier ses performances.

Le but de l'évaluation est de quantifier la performance du modèle et de la comparer à d'autres, Ainsi pour évaluer les performances des modèles ML/DL, différentes mesures statistiques sont utilisées [23].

- **TP** : signifie « Vrai positif » qui indique (que le nombre d'exemples positifs classifiés avec exactitude [33] (prédit positif et la valeur réelle est positive).
- **FP** : indique une valeur fausse positive, c'est-à-dire le nombre d'exemples négatifs réels classés comme positifs [33](prédit positif, mais la valeur réelle est négative).
- **TN** : signifie « Vrai négatif », qui indique le nombre d'exemples négatifs classifiés avec exactitude [33] (prédit être négatif et la valeur réelle est négative).
- **FN** : signifie une valeur fausse négative, c'est-à-dire le nombre d'exemples positifs réels classés comme négatifs [33] (prédit comme étant négatif, mais la valeur réelle est positive).

➢ **Rappel :** (« recall » en anglais), est le taux de vrais positifs, c'est à dire la proportion de positifs que l'on a correctement identifiés.

$$Rappel = \frac{TP}{TP+FN}$$

➢ **Précision :(** «precision» en anglais) la proportion de prédictions correctes parmi les points que l'on a prédits positifs.

$$Précision = \frac{TP}{TP+FP}$$

➢ **F1-score :** F1- score ou F-mesure est la moyenne harmonique de précision et de rappel qui est utilisée pour intégrer ces deux métriques en une seule métrique. Ainsi, si sa valeur est élevée, les performances de classification sont meilleures.

$$F1 - score = \frac{2 \times Précision \times Recall}{Précision + Recall}$$

➢ **Matrice de confusion :** Une matrice de confusion est un résumé des résultats de prédiction pour un problème de classification. Les prédictions

correctes et incorrectes sont mises en évidence et regroupées par classe. Par conséquent, le résultat est comparé à la valeur réelle.

Voici un exemple de matrice de confusion :

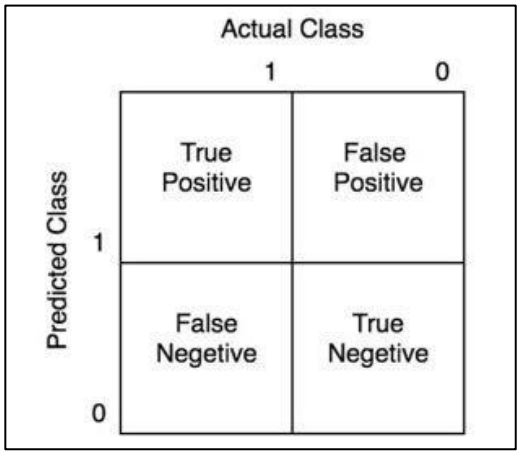

Figure 13 : La matrice de confusion.

- **Classification report :** C'est l'une des mesures d'évaluation des performances d'un modèle d'apprentissage automatique basé sur la classification. Il affiche la précision, le rappel, le score F1 et le support d'un modèle. Il fournit une meilleure compréhension de la performance globale du modèle formé.

Voici un exemple de classification report :

	precision	recall	f1-score	support
0	0.94	0.98	0.96	57475072
1	0.78	0.53	0.63	8060928
accuracy			0.92	65536000
macro avg	0.86	0.75	0.79	65536000
weighted avg	0.92	0.92	0.92	65536000

Figure 14 : Exemple d'un classification report.

3. Conception Plateforme finale

Après avoir évaluer les différents algorithmes dont nous présentons les résultats dans le prochain chapitre, nous avons conçu une petite application basée sur une architecture de traitement de données distribuée pour recueillir et traiter les données du trafic sur les réseaux IoT. L'architecture sera conçue pour être scalable, extensible et évolutive pour s'adapter aux futurs besoins de l'analyse de données.

Le but principal de l'application final et de pouvoir visualiser les résultats des tests des différents algorithmes sur notre dataset retenu

Pour le moment, notre application permet à notre acteur principal « Administrateur », outre le choix d'un algorithme et la visualisation des résultats de classification à travers ce dernier, elle affiche les statistiques de classification grâce à des charts.

Ci-dessous, nous présentons un petit diagramme de cas d'utilisation illustrant toutes les fonctionnalités que notre utilisateur pourra faire, suivi par le diagramme de séquence pour détailler les différentes fonctionnalités.

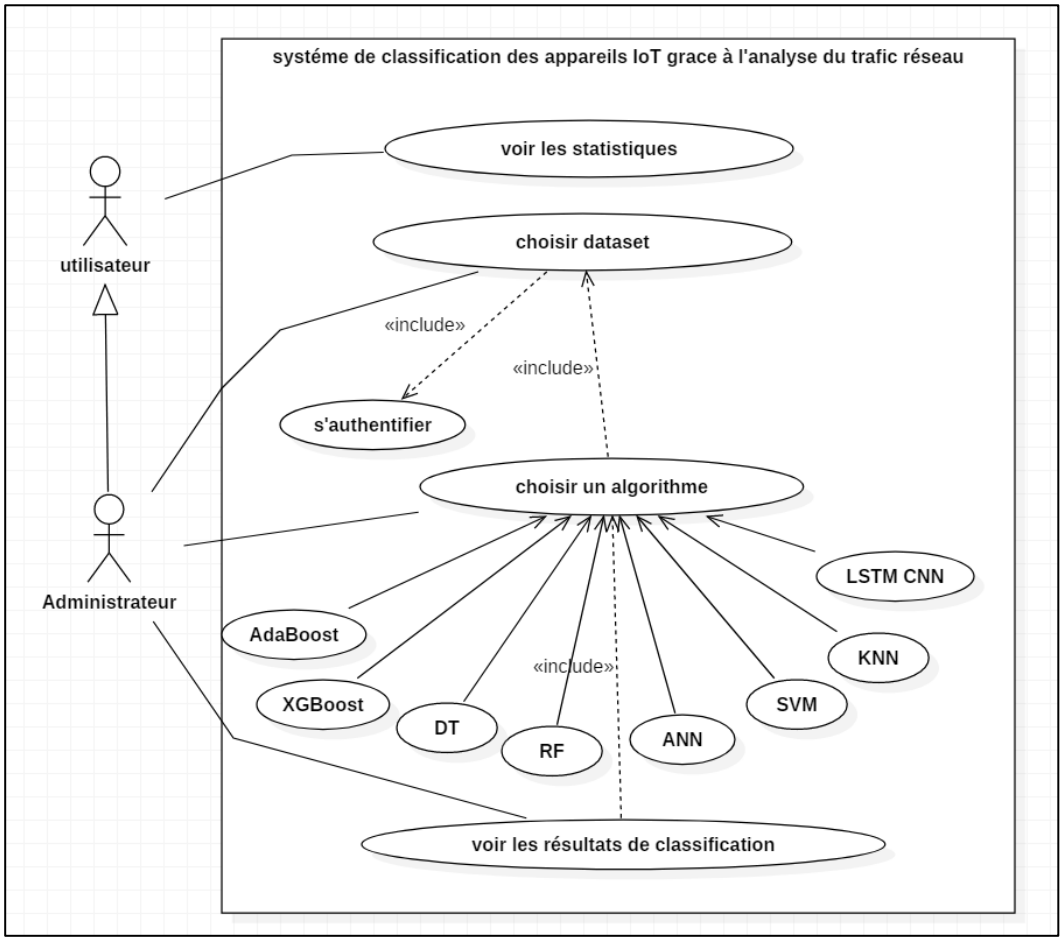

Figure 15 : Le diagramme de cas utilisation global.

Description textuelle de cas d'utilisation **:**

Élément	**Description**
Cas d'utilisation	**Voir les statistiques :** lorsqu'un utilisateur du système accède à l'application, il peut voir les histogrammes des mesures de performances et le tableau de comparaison des algorithmes **S'authentifier :** lorsqu'un administrateur du système veut accéder à l'application, il doit se connecter s'il a un compte. Pour se faire, il doit saisir son émail et son mot de passe, ensuite le système vérifie s'ils sont corrects ou pas afin d'autoriser ou bien refuser l'accès sinon s'il n'a pas de comptes, il doit en créer un en saisissant son nom, son émail et son mot de passe dans le formulaire approprié. **Choisir le dataset :** l'administrateur doit choisir le dataset **Choisir un algorithme :** l'administrateur doit choisir un algorithme pour faire la classification. Les algorithmes disponibles sont: K-Nearest-Neaghbors, eXtreme Gratient Boosting , Random Forest , Decision Tree
Acteur principal	Administrateur
Acteurs secondaires	Utilisateurs

Tableau 14 : Description du diagramme cas utilisation.

Le diagramme de séquences ci-dessous décrit l'interaction entre les utilisateurs et le système.

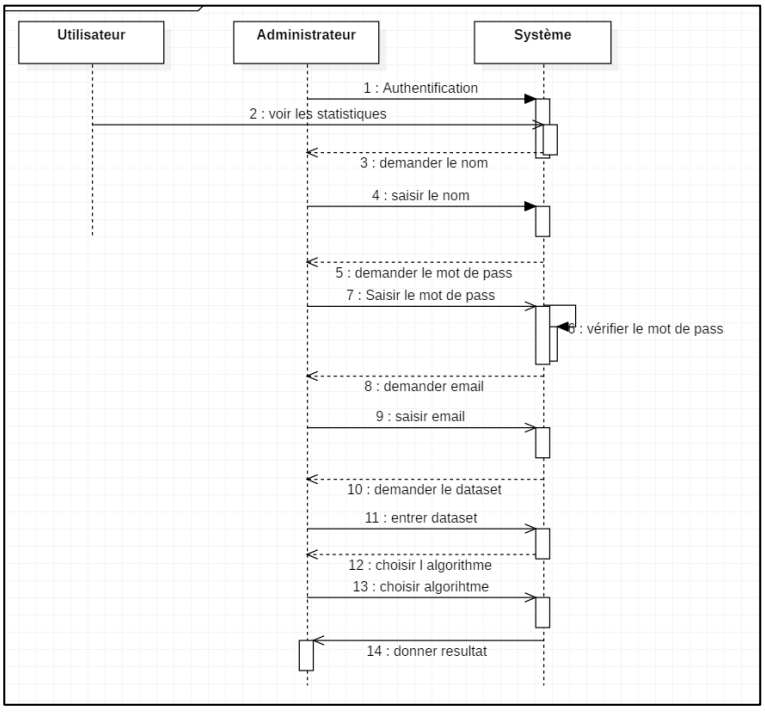

Figure 16 : Le diagramme de séquences.

4. Conclusion

Dans ce chapitre, nous avons présenté l'architecture globale de notre système. Le choix et description de notre dataset, aussi l'apprentissage et les algorithmes qui on a testé avec les mesures de performances retenues pour nos tests. Nous avons clôturé le chapitre en décrivant sommairement notre plateforme finale qui illustre l'objectif de notre travail. Dans le chapitre suivant, nous allons présenter les résultats que nous avons obtenu, ainsi que nos choix d'implémentation et quelques captures de notre application.

Chapitre III : Tests et Implémentation de la solution

4.1. Introduction

Dans ce chapitre, nous présentons les résultats obtenus pour les algorithmes testés avec notre jeu de données. Ainsi, la section 2 présente l'environnement logiciel et matériel de notre travail. Puis la section 3 parle sur l'implémentation et évaluation (chargement des données, évaluation et comparaison avec les travaux récents). Enfin, nous présentons quelques interfaces de notre application web pour la classification automatique des appareils de l'IoT.

4.2. Environnements de travail

Dans cette partie nous présentons les outils utilisés pour la mise en place de ce projet.

2.1. Environnement matériel

Le tableau ci-dessous présente les caractéristiques des machines utilisés pour notre travail :

	Machine 1	**Machine 2**
Modèle	Dell Latitude E7270	Lenovo 80QR
CPU	Intel(R) Core (TM) i5-6300U CPU 2.40GHz 2.50 GHz	Intel Celeron N2840 2.16 GHz
RAM	8 GO	4 GO
Stockage	256 GO	116 GO
Système d'exploitation	Windows 10 64 bits	Windows 10 64 Bits

Tableau 15 : Les caractéristiques des machines utilisées.

2.2. Environnement logiciel

Les outils logiciels utilisés pour notre travail sont résumés dans ce qui suit :

- Le jeu de données « **Aalto_train_IoTDevID »** et **« Aalto_test_IoTDevID ».**
- Nous avons formé nos classificateurs dans ***Jupyter***.
- ***KNN, ANN, DT, RF, XGboost, SVM, NB et LSTM-CNN*** sont utilisés comme des classifieurs dans notre projet.

Outils	Descriptions
Python 3.11	Utilisé comme langage de programmation
Pandas[3]	Une bibliothèque open source sous licence BSD fournissant des structures de données et des outils d'analyse de données hautes performances et faciles à utiliser pour le langage de programmation Python.
Scikit learn[4]	Une bibliothèque d'apprentissage automatique open source qui prend en charge l'apprentissage supervisé et non supervisé.
Numpy[5]	Le package fondamental pour le calcul scientifique en Python. Il s'agit d'une bibliothèque Python qui fournit un objet tableau multidimensionnel.
Matplolib[6]	Une bibliothèque complète pour créer des visualisations statiques, animées et interactives en Python.
Keras[7]	Bibliothèque pratique pour construire les algorithmes d'apprentissage profond
Flask[8]	Flask Python est un framework Web léger et adapté. Open source, il se différencie par sa légèreté permettant de disposer d'une solide base de développement tout en conservant la flexibilité et la lisibilité du langage de programmation Python. Facile à prendre en main, il optimise le processus de développement. Il accompagne ainsi les entreprises dans leurs besoins en applications web de petite et moyenne envergure.

Tableau 16 : Les outils logiciel utilisés.

4.3. Implémentation et résultats

Dans cette section, nous présentons le détail technique des différentes étapes du processus global introduit dans le chapitre précédent :

[3] https://pandas.pydata.org/
[4] https://scikit-learn.org
[5] https://numpy.org/
[6] https://matplotlib.org/
[7] https://keras.io/
[8] https://webojob.ch/technologies/python/flask

3.1. Chargement et visualisation des données

Notre dataset contient plus que 81000 instances pour l'apprentissage (train) et plus que 21000 instances pour les tetes (test). Nous commençons par charger le jeu de données (figure 17) et le visualisons (figure 18) :

```
Entrée [2]: train = pd.read_csv("Aalto_train_IoTDevID.csv")
            test = pd.read_csv("Aalto_test_IoTDevID.csv")
```

Figure 17 : Chargement des données.

```
train.head()
```

Out[2]:

	pck_size	Ether_type	LLC_dsap	LLC_ssap	LLC_ctrl	EAPOL_version	EAPOL_type	EAPOL_len	IP_version	IP_ihl	...	sport	dport	TCP_sport	TCP_dport
0	117	34958	0	0	0	1	3	117	0	0	...	0	0	0	0
1	95	34958	0	0	0	1	3	95	0	0	...	0	0	0	0
2	328	2048	0	0	0	0	0	0	4	5	...	68	67	0	0
3	328	2048	0	0	0	0	0	0	4	5	...	68	67	0	0
4	60	2048	0	0	0	0	0	0	4	5	...	59904	53	0	0

5 rows × 96 columns

Figure 18 : Visualisation des données.

Nous comptons par la suite le nombre total de chaque appareil dans train et test :

```
Entrée [28]: train['Label'].value_counts()

Out[28]: HueSwitch            14649
         HueBridge            11043
         D-LinkHomeHub         6899
         WeMoLink              5265
         D-LinkSensor          5204
         D-LinkSwitch          5199
         D-LinkWaterSensor     5192
         D-LinkCam             4994
         D-LinkSiren           4949
         WeMoInsightSwitch     4735
         WeMoSwitch            3667
         Lightify              3375
         D-LinkDoorSensor      1534
         EdimaxPlug1101W        896
         D-LinkDayCam           849
         EdimaxPlug2101W        764
         EdimaxCam              666
         Withings               555
         EdnetGateway           546
         TP-LinkPlugHS100       524
         TP-LinkPlugHS110       512
         HomeMaticPlug          507
         MAXGateway             451
         Aria                   351
         EdnetCam               330
         SmarterCoffee          121
         IKettle2               117
         Name: Label, dtype: int64
```

Figure 19 ; Nombre total des appareils dans train.

```
Entrée [27]: test['Label'].value_counts()

Out[27]: HueSwitch            3799
         HueBridge            2893
         D-LinkHomeHub        1696
         WeMoLink             1360
         D-LinkSensor         1345
         D-LinkSwitch         1320
         D-LinkCam            1250
         D-LinkWaterSensor    1243
         D-LinkSiren          1237
         WeMoInsightSwitch    1227
         WeMoSwitch            810
         Lightify              774
         D-LinkDoorSensor      358
         EdimaxPlug1101W       264
         EdimaxPlug2101W       246
         D-LinkDayCam          214
         EdimaxCam             165
         TP-LinkPlugHS100      143
         EdnetGateway          137
         Withings              133
         TP-LinkPlugHS110      124
         MAXGateway            116
         HomeMaticPlug         104
         Aria                   90
         EdnetCam               78
         SmarterCoffee          28
         IKettle2               28
         Name: Label, dtype: int64
```

Figure 20 : Nombre total des appareils dans test.

Nous supprimons les valeurs manquantes :

```
Entrée [34]: #suppression des lignes de valeurs null
             train.dropna(axis=0, inplace=True)
             test.dropna(axis=0, inplace=True)
```

Figure 21 : Suppression des valeurs nulles.

3.2. Evaluation

Après le test des 9 algorithmes de classification, nous évaluons les différentes mesures de performance à travers : l'accuracy, le rappel, la précision et le F1_score.

Tous les résultats trouvés sont résumés dans le tableau suivant :

Algorithme	Score	Accuracy	F1_score	Rappel	Précision
KNN	0.92	0.92	0.89	0.88	0.91
Arbre de décision	0.99	0.99	0.98	0.97	0.99
Forêts aléatoires	0.99	0.99	0.98	0.98	0.98
ANN	0.35	0.35	0.26	0.28	0.32
SVM	0.23	0.23	0.07	0.10	0.07
Naïve Bayes	0.21	0.21	0.05	0.12	0.08
XGBoost	0.99	0.99	0.98	0.98	0.98
LSTM-CNN	0.36	0.84	0.33	0.50	0.25
AdaBoost	0.18	0.18	0.10	0.14	0.14

Tableau 17 : Les résultats obtenus par les algorithmes testés.

➢ **Score :**

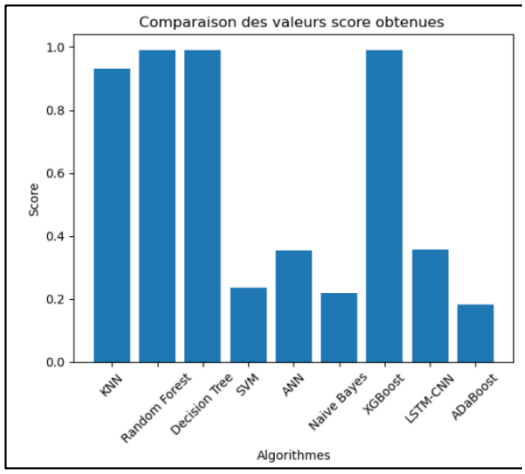

Figure 22 : Comparaison des valeurs Score obtenues par les algorithmes.

➢ **Accuracy :**

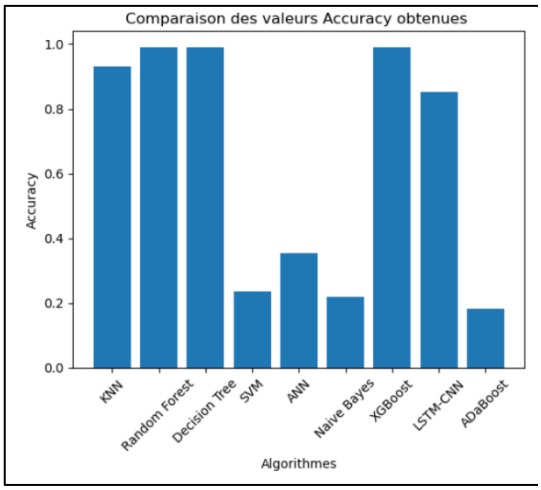

Figure 23 : Comparaison des valeurs d'Accuracy obtenues par les algorithmes.

➢ **F1_score :**

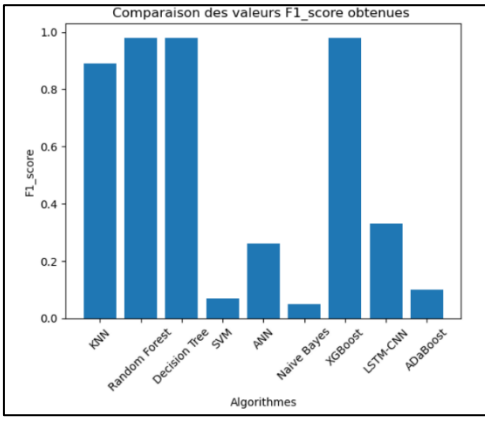

Figure 24 : Comparaison des valeurs F1_scores obtenues par les algorithmes.

➢ **Rappel :**

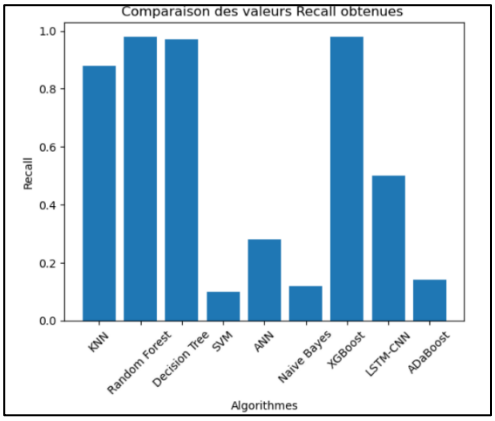

Figure 25 : Comparaison des valeurs de Rappel obtenues par les algorithmes.

➢ **Précision :**

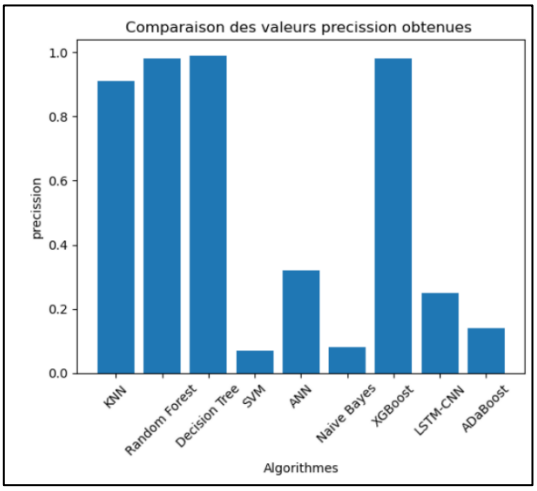

Figure 26 : Comparaison des valeurs de précision obtenues par les algorithmes.

Nous concluons que les algorithmes ***KNN***, ***Arbre de Décision***, ***Forêt Aléatoire*** et ***XGBoost*** donnent les meilleurs résultats.

Exemple de matrice de confusion et classification report d'algorithme arbre de décision :

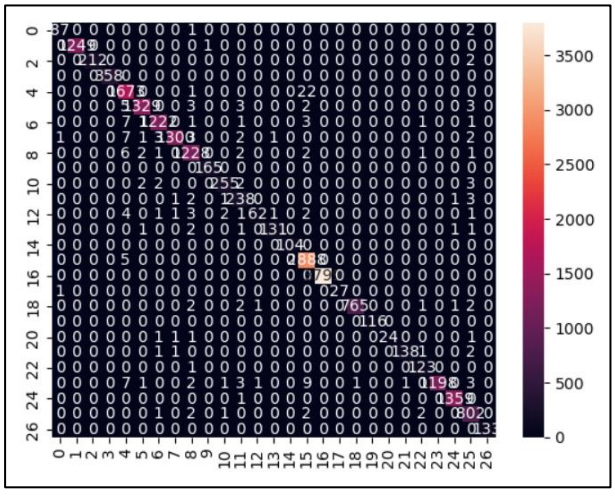

Figure 27 : Matrice de confusion des Arbres de Décision.

	precision	recall	f1-score	support
Aria	0.98	0.97	0.97	90
D-LinkCam	1.00	1.00	1.00	1250
D-LinkDayCam	1.00	0.99	1.00	214
D-LinkDoorSensor	1.00	1.00	1.00	358
D-LinkHomeHub	0.98	0.99	0.98	1696
D-LinkSensor	0.99	0.99	0.99	1345
D-LinkSiren	0.99	0.99	0.99	1237
D-LinkSwitch	1.00	0.98	0.99	1320
D-LinkWaterSensor	0.98	0.99	0.98	1243
EdimaxCam	0.99	1.00	1.00	165
EdimaxPlug1101W	0.98	0.97	0.97	264
EdimaxPlug2101W	0.93	0.97	0.95	246
EdnetCam	0.97	0.79	0.87	78
EdnetGateway	0.98	0.96	0.97	137
HomeMaticPlug	1.00	1.00	1.00	104
HueBridge	0.99	1.00	0.99	2893
HueSwitch	1.00	1.00	1.00	3799
IKettle2	1.00	0.96	0.98	28
Lightify	1.00	0.99	0.99	774
MAXGateway	1.00	1.00	1.00	116
SmarterCoffee	1.00	0.86	0.92	28
TP-LinkPlugHS100	0.99	0.97	0.98	143
TP-LinkPlugHS110	0.95	0.99	0.97	124
WeMoInsightSwitch	1.00	0.98	0.99	1227
WeMoLink	1.00	1.00	1.00	1360
WeMoSwitch	0.97	0.99	0.98	810
Withings	1.00	1.00	1.00	133
accuracy			0.99	21182
macro avg	0.99	0.97	0.98	21182
weighted avg	0.99	0.99	0.99	21182

Figure 28 : Classification report des Arbres de Décision.

3.3. Discussion

D'après notre traitement des algorithmes et résultats obtenus nous remarquons que les résultats étaient divers indépendamment du fait qu'ils s'agissent d'algorithme d'apprentissage automatique ou bien d'apprentissage profond.

Il est intéressant de noter que les algorithmes *KNN, Random Forest, Decision Tree et XGBoos*t ont donné des performances supérieures à 92% en termes de score, d'accuracy, de f1_score, de rappel et de précision. Cela peut être dû à la capacité de ces algorithmes à gérer des ensembles de données complexes et à effectuer l'opération de classification multi-classes.

D'autre part, les algorithmes *SVM, ANN, Naïve Bayes, Adaboost et le modèle LSTM-CNN* ont donné de mauvaises performances. Cela peut être dû à plusieurs raisons, telles que la complexité de l'ensemble de données, le choix des paramètres d'entraînement, ou une conception inappropriée de l'algorithme. Par ailleurs, il est à noter que la nature des algorithmes peut ne pas convenir pour notre problème de classification ou l'équilibrage de nos données (les différentes étiquettes).

Ainsi, nous avons retenus les quatre algorithmes ayant donné de bons résultats pour les implémenter dans notre application finale.

3.4. Comparaison avec les travaux récents

Le tableau suivant compare les résultats trouvés dans les travaux récents et notre résultat après le test des algorithmes :

Travail	Algorithmes	Performances (Accuracy)	Nos performances (Accuracy)
Travail 1 [9]	DT, KNN, ADaBoost	**99%**	**99%, 91%, 18%**
Travail 2 [8]	SVM, RF, LSTM-CNN	**74.8%**	**23%, 99%, 84%**
Travail 3 [11]	ANN, GNB	**99.98%**	**35%, 21%**
Travail 4 [7]	RF, DT	**99.9%**	**99%, 99%**
Travail 5 [12]	XGBoost, RF	**99.81%**	**99%, 99%**

Tableau 18 : Comparaison des travaux récents avec nos résultats.

4.4. Interfaces d'application

Il s'agit d'une application web qui propose plusieurs fonctionnalités.

- Tout d'abord, il y a une page d'accueil qui présente l'application et ses différentes fonctionnalités. Les utilisateurs ont accès à une page de statistiques qui leur permet de visualiser les données collectées par l'application. Cette page permet également aux utilisateurs de suivre leur propre progression.
- La page "a propos" quant à elle, présente des informations sur l'application, son fonctionnement et son équipe de développeurs.
- Ensuite, les utilisateurs peuvent s'inscrire ou se connecter grâce à une page de login.
- Enfin, la page de classification est l'une des fonctionnalités les plus importantes de l'application. Elle permet aux utilisateurs de choisir un jeu de données, un algorithme de classification et de visualiser les résultats de cette classification. Les utilisateurs peuvent ainsi explorer et analyser les données de manière interactive.

La figure qui suit représente le schéma d'accessibilité de notre application :

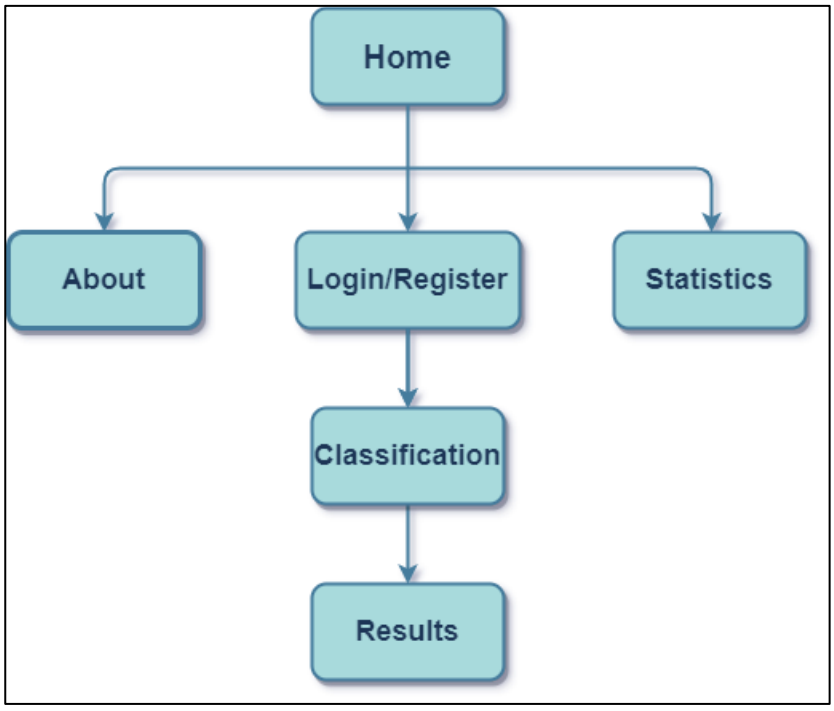

Figure 29 : Schéma d'accessibilité d'application.

4.1. Accueil

La page d'accueil de l'application comprend des informations sur les fonctionnalités principales et des boutons pour se connecter ou s'inscrire.

Figure 30 : Page accueil de l'application.

4.2. A propos

La page "à propos" de l'application contient des informations sur l'application.

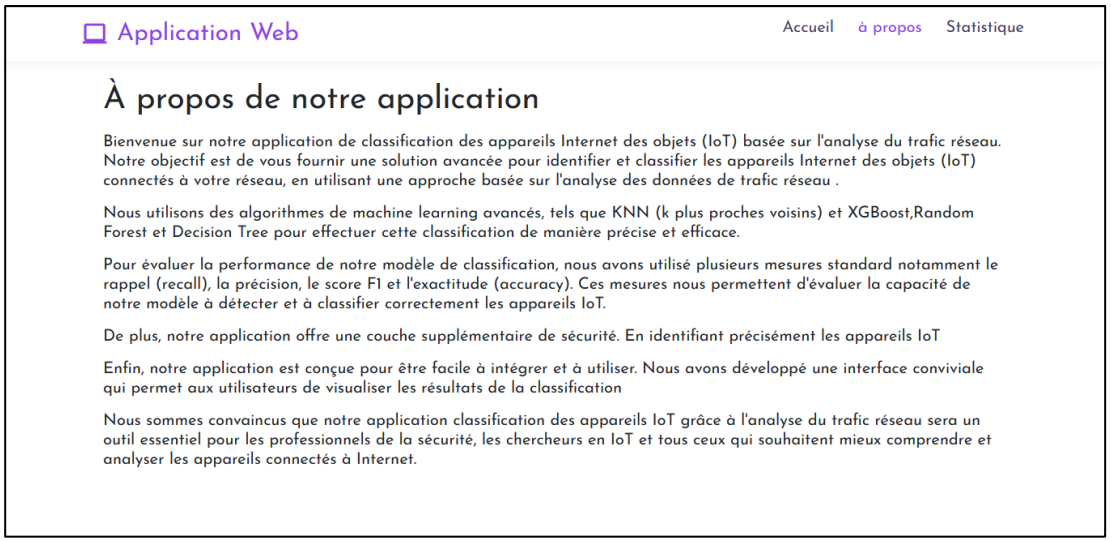

Figure 31 : Page à propos d'application.

4.3. Statistiques

La page statistiques affiche des graphiques et des chiffres pour fournir des informations sur l'utilisation de l'application.

Figure 32 : Page statistique d'application.

4.4. Connexion/ inscription

Les pages de connexion et inscription permettent aux utilisateurs de se connecter à leur compte existant ou de créer un nouveau compte.

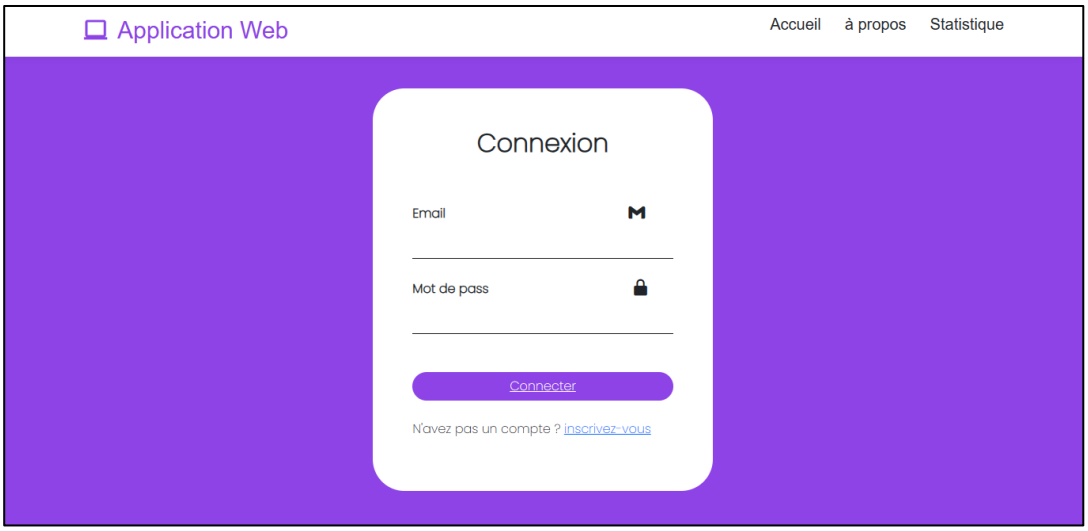

Figure 33 : Page de connexion.

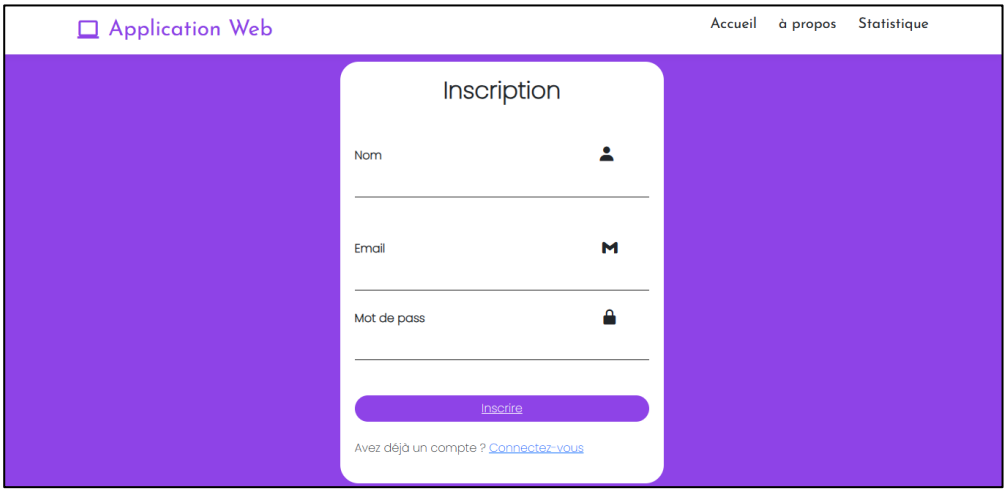

Figure 34 : Page d'inscription.

4.5. Classification

La page de classification permet aux utilisateurs de classifier les appareils selon un jeu de données et algorithme choisis.

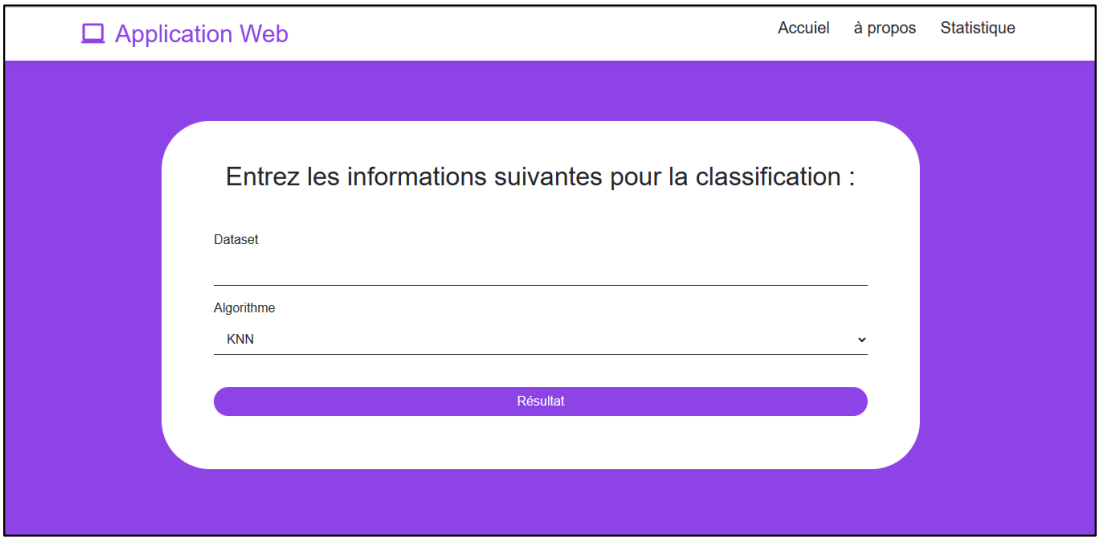

Figure 35 : Page de la classification des appareils.

4.5. Affichage du traitement

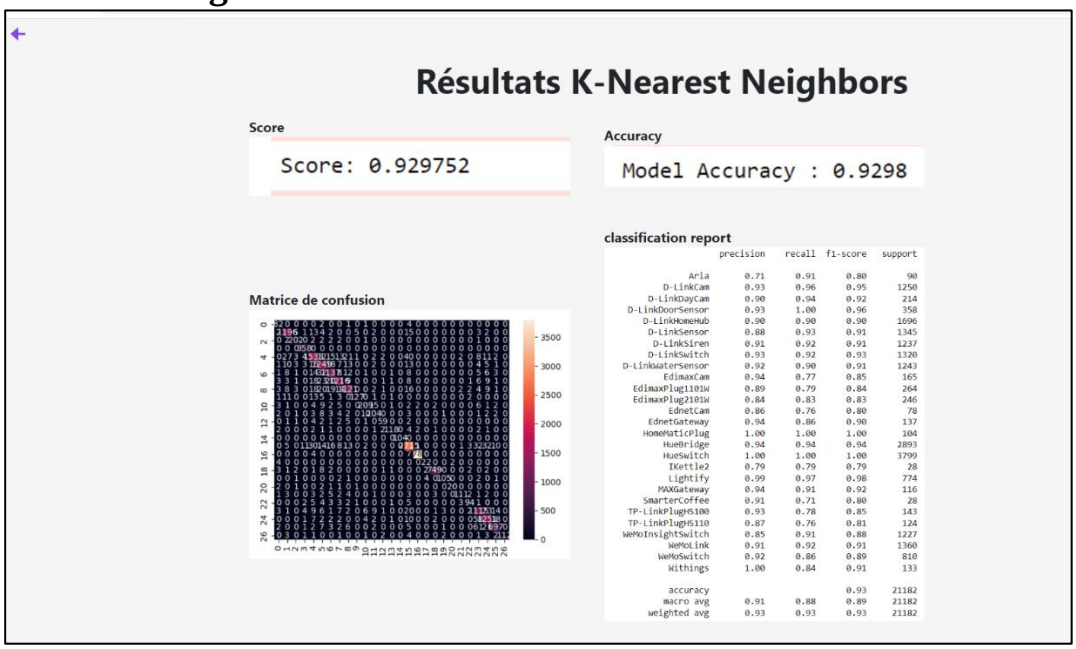

Figure 36 ; Affichage du traitement KNN.

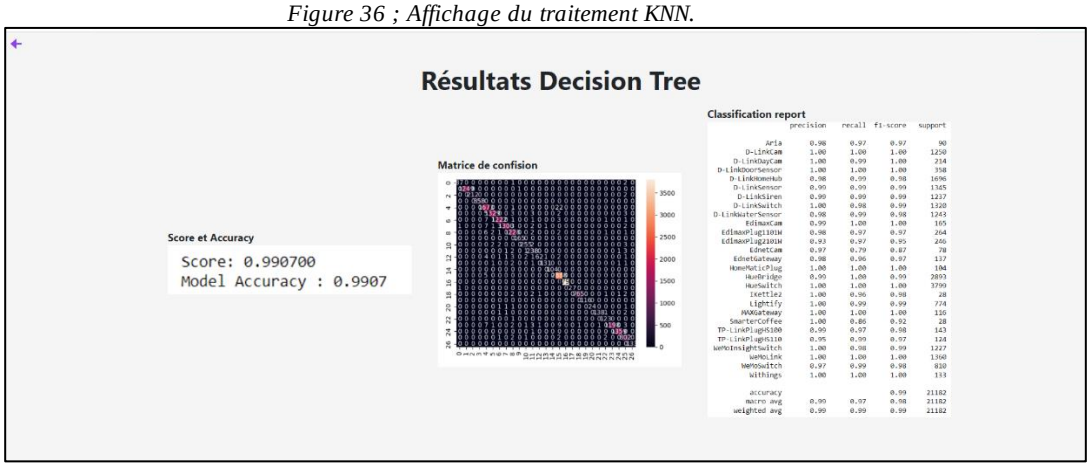

Figure 37 : Affichage du traitement Decision Tree.

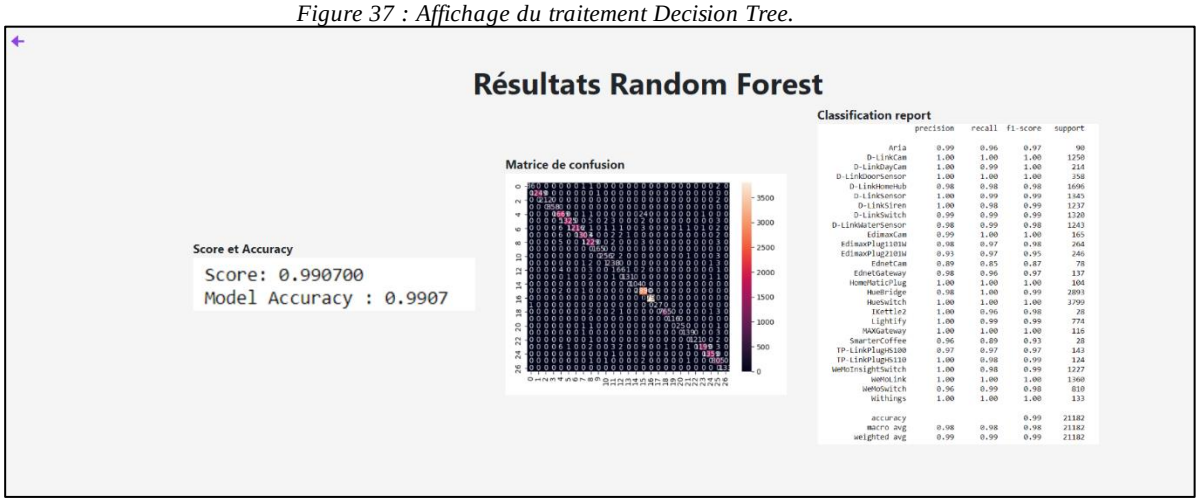

Figure 38 : Affichage du traitement Random Forest.

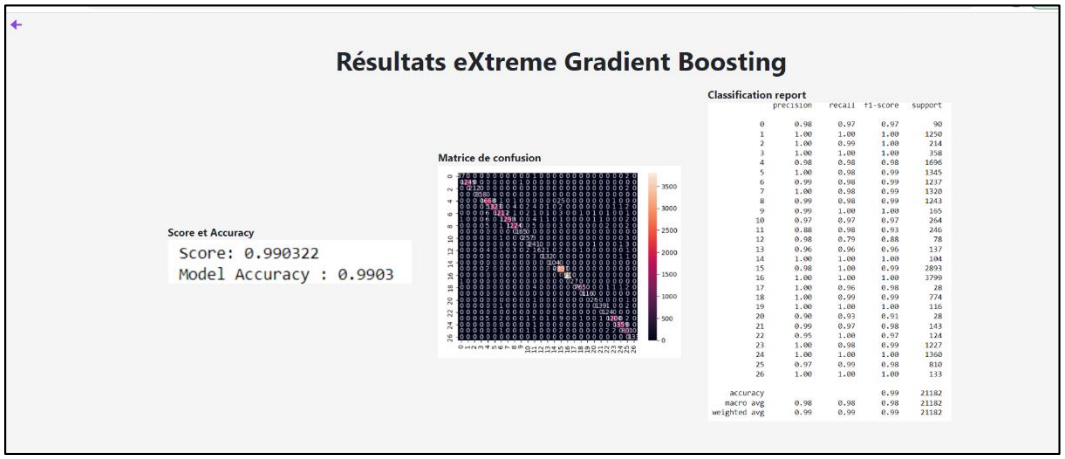

Figure 39 : Affichage du traitement XGBoost.

4.6. Conclusion

Dans ce dernier chapitre, nous avons présenté les résultats obtenus après les tests des algorithmes (*KNN, forêt aléatoire, Arbre de décision, SVM, Naïve Bayes, ANN, XGboost, LSTM-CNN et Adaboost*). Nous avons obtenu une accuracy supérieure à 90% pour les algorithmes ***KNN, RT, DT*** et ***XGBoost*** pour classifier les appareils IoT grâce à l'analyse de trafic réseau.

Nous avons enfin, présenter les interfaces de notre application web pour la classification automatique des équipements IoT.

Conclusion générale

1. Conclusion

L'Internet des objets (IoT) décrit le réseau de terminaux physiques, les « objets », qui intègrent des capteurs, des softwares et d'autres technologies en vue de se connecter à d'autres terminaux et systèmes sur Internet et d'échanger des données avec eux. Ces terminaux peuvent aussi bien être de simples appareils domestiques que des outils industriels d'une grande complexité.

Le but principal de notre étude est de trouver une solution pour la reconnaissance des diapositifs IoT à travers l'Analyse du trafic réseau. Nous avons travaillé sur la classification de 27 appareils d'internet des objets, pour cela nous avons testé 9 algorithmes (KNN, ANN, RF, DT, XGBoost, SVM, Naïve bayes, ADaBoost et le modèle LSTM-CNN)

Les résultats obtenus montrent que *KNN, Decision Tree, Random Forest et XGBoost* donnent les meilleurs performances (dépassant 92% de score, accuracy …etc.), alors nous nous sommes plus particulièrement intéressées à eux pour notre système.

Pour finir, à l'aide des résultats obtenus, nous avons développé une application web pour simuler le travail de classification des appareils IoT.

2. Perspectives

Arrivées au terme de notre projet, nous pouvons dire que l'expérience vécu était enrichissante et nous a permis de capitaliser les connaissances acquises le long de notre parcours Informatique. Surtout, nous avons eu l'occasion d'apprendre et de nous familiariser avec des concepts tendance comme l'apprentissage automatique ou le deep learning.

Parmi les fonctionnalités que nous aurions aimé implémenter ou qui serait intéressant d'avoir dans le futur, nous citons :

- La collecte de données à partir d'un réseau réel ou simulé pour faire des tests plus concrets.

- Le développement d'un tableau de bord indiquant les périodes d'activité et d'inactivité des différents appareils détectés.
- Allier l'application à une application de sécurité afin e détecter lesquels parmi les appareils connectés à notre réseau sont menaçants et lesquels sont sans danger.
- Profiter des techniques domotiques pour ajouter les fonctionnalités de gestion et de maintenance des appareils.
- Mais surtout améliorer la classification en testant d'autres algorithmes automatiques, personnalisés ou bien de transfert learning.

- La création de dataset par fusion avec des éléments issus d'autres dataset.
- La manipulation des différents dispositifs IoT (ajout, modification suppression)
- Le tracking (suivi) ou le monitoring d'un appareil particulier ou bien du réseau entier dans une période donnée ou en temps réel

Bibliographie

[1] K. Kahraman, J. Mike et A. Michael, «IoTDevID: A Behavior-Based Device Identification Method for the IoT,» *IEEE Internet of Things Journal,* n° %122290499, pp. 23741 - 23749, 2022.

[2] B. Lei, Y. Lina, S. Salil, W. Xianzhi et Y. Zheng, «Automatic Device Classification from Network,» chez *Local Computer Networks (LCN)*, Chicago, IL, USA, 2018.

[3] T. S. Z. JUBA, «Analyse et classification du trafic réseau,» Tizi-Ouzou, 2018.

[4] «Network Traffic Analysis ou NTA, de quoi parle-t-on ?,» 28 Juin 2022. [En ligne]. Available: https://tehtris.com/fr/blog/network-traffic-analysis-nta-de-quoi-parle-t-on. [Accès le Avril 2023].

[5] A. L. rosa, «Analyse du trafic réseau : paquets et flux,,» 20 Octobre 2020. [En ligne]. Available: https://pandorafms.com/blog/fr/analyse-du-trafic-reseau/.. [Accès le Avril 2023].

[6] P. Kassaa, «L'internet des Objets (IoT) et les réseaux haut et bas débit, est-ce un outil essentiel pour une ville intelligente et moins Polluée ?,» Ecole Nationale des Ponts et Chaussées,, 2018-2019.

[7] F. MERABET, «Solutions de Sécurité pour l'Internet des Objets dans le Cadre de,» Tizi Ouzou, 2021.

[8] M. RYMA et W. MAZARI, «Introduction à l'internet de l'objet et réalisation,» Bejai, 2016.

[9] R. MEKRIOU et W. MAZARI, *Introduction à l'internet de l'objet et réalisation,* Bejaïa, Université A/Mira de Bejaïa, 2016.

[10] A. A. Amira Bouguerra, «Prédiction du trafic internet dans l'université de M'Sila,» M'Sila, 2021/2022.

[11] «Conseil de l'Europe,» [En ligne]. Available: https://www.coe.int/fr/web/artificial-intelligence/glossary. [Accès le Avril 2023].

[12] N. Belaidi, «L'apprentissage supervisé : définition et exemples,» [En ligne]. Available: https://blent.ai/blog/a/apprentissage-supervise-definition. [Accès le Avril 2023].

[13] R. S. Mustafizur, B. Gregory, Z. Zonghua et D. Hervé, «IoT Devices Recognition Through Network Traffic Analysis,» chez *IEEE International Conference on Big Data*, Seattle, United States, 2018.

[14] A. M. D. R. Adriano, H. A. D. D. M. Pedro, R. S. Jefferson, G. C. Renan et S. M. Rodrigo, «Traffic Classification of Home Network Devices using Supervised,» chez *Conference on Agents and Artificial Intelligence*, Uberlandia, Brazil, 2022.

[15] B. Avewe et G. Bamba, «A Group-Based IoT Devices Classification Through Network Traffic Analysis Based on Machine Learning Approach,» chez *e-Infrastructure and e-Services for Developing Countries*, Ebène City, Mauritius, 2020.

[16] M. Yair, B. Michael, S. Asaf, D. G. Juan, O. Martin, O. T. Nils et E. Yuval, «ProfilIoT: a machine learning approach for IoT device identification based on network traffic analysis,» chez *Proceedings of the Symposium on Applied Computing*, New York, NY, United States, 2017.

[17] A. KAZIN, «DDoS SDN dataset.,» [En ligne]. Available: https://www.kaggle.com/datasets/aikenkazin/ddos-sdn-dataset.. [Accès le Mars 2023].

[18] S. |. L. D'HOOGE, «UNSW-NB15,,» [En ligne]. Available: https://www.kaggle.com/datasets/dhoogla/unswnb15.. [Accès le Mars 2023].

[19] J. S. ROJAS, «IP Network Traffic Flows Labeled with 75 Apps,,» [En ligne]. Available: https://www.kaggle.com/datasets/jsrojas/ip-network-traffic-flows-labeled-with-87-apps.. [Accès le Mars 2023].

[20] D. E. I. Mosseri, «IoT-device-type-identification,,» [En ligne]. Available: https://github.com/Mosseridan/IoT-device-type-identification/tree/3de6669457d806ea7ca6ad9172e38a8c9d9ff059.. [Accès le Avril 2023].

[21] S. A.-R. M. S. N. A. H. I. e. T. S. M. Markus, « Sasu, IOT SENTINEL: Automated Device-Type Identification for Security Enforcement in IoT,,» chez *IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, Atlanta, GA, USA,.

[22] S. A. WALLELIGN, «An Intelligent System for Coffee Grading and Disease Identification,,» Plouzané, 04 Février 2020.

[23] O. Aouedi, *Machine Learning-Enabled Network Traffic Analysis*, NANTES UNIVERSITÉ, 2022.

[24] S. R. M. Madina, «conception et mise en oeuvre d'un systeme de classification du trafic réseau crype,,» Blida, 2019/2022..

[25] M. B. A. M. E. Lazrag, «Extraction des caractéristiques discriminantes basées sur LBP pour la classification des textures,,» Tiaret, 2021.

[26] L. R, «MiniQ : Qu'est-ce qu'une IA hybride CNN-LSTM ?,,» 22 Décembre 2019. [En ligne]. Available: https://penseeartificielle.fr/mini-question-ia-hybride-cnn-lstm/#:~:text=LSTM%20%3A%20un%20r%C3%A9seau%20avec%20de%20la%20m%C3%A9moire&text=Un%20%C2%AB%20long. [Accès le Mai 2023].

[27] J. Brownlee, «CNN Long Short-Term Memory Networks,» 21 Aout 2017. [En ligne]. Available: https://machinelearningmastery.com/cnn-long-short-term-memory-networks/.

[28] «Algorithme XGBoost, Présentation et fonctionnement,.,» [En ligne]. Available: https://www.jedha.co/formation-ia/algorithme-xgboost. [Accès le Mai 2023].

[29] A. Crochet-Damais, «Random forest (ou forêt aléatoire) : définition et cas d'usage,,» 30 Mai 2022. [En ligne]. Available: https://www.journaldunet.fr/web-tech/guide-de-l-intelligence-artificielle/1501905-random-forest-ou-foret-aleatoire/.. [Accès le Mai 2023].

[30] M. Macary, «informatique,» 2022.

[31] «What are Naïve Bayes classifiers?,» [En ligne]. Available: https://www.ibm.com/topics/naive-bayes.. [Accès le Avril 2023].

[32] «AdaBoost Classifier in Python,,» Novembre 2018. [En ligne]. Available: https://www.datacamp.com/tutorial/adaboost-classifier-python. [Accès le Mai 2023].

[33] .. F. A. B. Ajay Kulkarni, «Confusion Matrice,» 2020. [En ligne].

[34] [En ligne]. Available: https://github.com/kahramankostas/IoTDevIDv2. [Accès le Mai 2023].

[35] «AdaBoost Classifier in Python,» Novembre 2018. [En ligne]. Available: https://www.datacamp.com/tutorial/adaboost-classifier-python. [Accès le Mai 2023].

[36] «Algorithme XGBoost, Présentation et fonctionnement,» Mai 2023. [En ligne].

Available: https://www.jedha.co/formation-ia/algorithme-xgboost.

[37] «Getting Started,» aveil 2023. [En ligne]. Available: https://scikit-learn.org.

[38] M. Markus, S. Ahmad-Reza, M. Samuel, N. Asokan, H. Ibbad et T. Sasu, «IOT SENTINEL: Automated Device-Type Identification for Security Enforcement in IoT,» chez *IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, Atlanta, GA, USA, 2017.

[39] D. E. Idan Mosseri, «IoT-device-type-identification,» [En ligne]. Available: https://github.com/Mosseridan/IoT-device-type-identification/tree/3de6669457d806ea7ca6ad9172e38a8c9d9ff059.

[40] P. Kassaa, *L'internet des Objets (IoT) et les réseaux haut et bas débit, est-ce un outil essentiel pour une ville intelligente et moins Polluée ?,* Ecole Nationale des Ponts et Chaussées, 2018-2019.

[41] «Le framework Flask,» [En ligne]. Available: https://webojob.ch/technologies/python/flask. [Accès le Mai 2023].

[42] «Matplotlip,» Avril 2023. [En ligne]. Available: https://matplotlib.org/.

[43] L. R, «MiniQ : Qu'est-ce qu'une IA hybride CNN-LSTM ?,» 22 décembre 2019. [En ligne]. Available: https://penseeartificielle.fr/mini-question-ia-hybride-cnn-lstm/#:~:text=LSTM%20%3A%20un%20r%C3%A9seau%20avec%20de%20la%20m%C3%A9moire&text=Un%20%C2%AB%20long%20short%2Dterm%20memory,de%20neurones%20r%C3%A9currents%20(RNN)..

[44] «NumPy documentation,» Avril 2023. [En ligne]. Available: https://numpy.org/doc/stable/.

[45] «pandas documentation,» Avril 2023. [En ligne]. Available: https://pandas.pydata.org/docs/.

[46] A. L. rosa, «Analyse du trafic réseau : paquets et flux,» 20 Octobre 2022. [En ligne]. Available: https://pandorafms.com/blog/fr/analyse-du-trafic-reseau/. [Accès le Mars 2023].

[47] «Network Traffic Analysis ou NTA, de quoi parle-t-on ?,» 28 juin 2022. [En ligne]. Available: https://tehtris.com/fr/blog/network-traffic-analysis-nta-de-quoi-parle-t-on. [Accès le Avril 2023].