REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
جامعة أبي بكر بلقايد – تلمسان –
Université Aboubakr Belkaïd – Tlemcen –
Faculté de TECHNOLOGIE

MEMOIRE

Présenté pour l'obtention du **diplôme** de **MASTER**

En : Télécommunications

Spécialité : Réseaux et Télécommunications

Par : IKNI Afaf Fatima Zohra et SEKKAK Ibrahim

Sujet

Détection d'applications Android malveillantes à l'aide de réseaux de neurones profonds

Soutenu publiquement, le 13/06/2024, devant le jury composé de :

M. FEHAM Mohammed	Professeur	Université de Tlemcen	Président
Mme. BENAISSA Amal	MAB	Université de Tlemcen	Examinateur
Mme. TALEB Sarra	MAB	Université de Tlemcen	Encadrant

2023/2024

Remerciements

Nous tenons à la fin de ce travail à remercier avant tout notre Dieu, le Tout-Puissant, d'avoir insufflé en nous le courage, la patience et la santé nécessaires à la réalisation de ce modeste travail.

Nous exprimons notre profonde gratitude à l'ensemble des enseignants du département de Télécommunications pour les précieux cours dispensés.

Nous tenons à exprimer notre plus profonde gratitude au Président, M. FEHAM Mohammed, pour son dévouement et son excellence dans l'enseignement tout au long des années précédentes. Sa pédagogie exceptionnelle et sa passion pour le savoir nous ont profondément marqués et inspirés. Nous lui témoignons notre plus grande appréciation et notre profond respect.

Nous tenons également à remercier Mme BENAISSA Amal pour avoir accepté d'examiner ce travail et Mme TALEB Sarra pour sa patience, sa disponibilité et surtout ses judicieux conseils, qui ont contribué à enrichir notre réflexion.

Dédicace

À toutes nos compagnes :

Après vous avoir remerciées pour votre aide et votre soutien moral lors de la réalisation de nos travaux de fin d'études, et en souvenir des moments agréables que nous avons passés ensemble, nous tenons à vous dire que notre amitié perdurera pour toujours.

À nos mères et à toute notre famille :

Nous ne saurions oublier de vous remercier pour votre soutien moral, vos encouragements et votre patience durant les étapes de réalisation de ce travail.

À nos chers parents :

Que nulle dédicace ne puisse exprimer ce que nous vous devons, pour tous vos efforts et vos sacrifices durant cette vie. Merci pour votre bienveillance, votre amour, votre tendresse, votre soutien et vos prières tout au long de nos études. Que ce travail soit le témoignage de notre profond amour et de notre grande reconnaissance.

"Que Dieu vous garde".

RÉSUMÉ

Ces dernières années, Android a gagné en popularité en tant que système d'exploitation mobile le plus populaire à travers le monde, ce qui en fait une cible importante pour les cybercriminels. Ainsi, la question de la sécurité de cette plateforme revêt une grande importance. Malgré tout ce que ce système propose en termes de mécanismes solides pour protéger sa plateforme, il présente cependant plusieurs vulnérabilités et failles.

La prolifération des applications Android a conduit à une augmentation des malwares, posant des risques significatifs pour la sécurité des utilisateurs et de leurs données. Les techniques traditionnelles de détection, basées sur des signatures et des heuristiques, peinent à suivre l'évolution rapide et la complexité croissante des malwares. Les réseaux de neurones profonds (Deep learning) offrent une solution prometteuse pour améliorer la détection des applications malveillantes grâce à leur capacité à analyser des schémas complexes et variés.

Mots clés : Malware, sécurité, Application Android, Deep Learning, détection.

ABSTRACT

In recent years, Android has gained popularity as the most popular mobile operating system worldwide, making it an important target for cybercriminals. Thus, the issue of the security of this platform is of great importance. En despite everything this system offers in terms of robust mechanisms to protect its platform, it has several vulnerabilities and weaknesses.

The proliferation of Android apps has led to an increase in malware, posing significant risks to the security of users and their data. Traditional signature and heuristic-based detection techniques struggle to keep up with the rapid evolution and increasing complexity of malware. Deep Learning provides a promising solution to improve detection of malicious applications through their ability to analyse complex and varied patterns. .

Keywords: Malware, Security, Android Application, Deep Learning, Detection.

ملخص

في السنوات الأخيرة، اكتسبت أندرويد شعبية كإستراتيجية التشغيل المحمولة الأكثر شعبية في جميع أنحاء العالم، مما يجعلها هدفاً هاماً على الرغم من كل ما يوفره هذا النظام من آليات قوية لحماية منصة الشركة، إلا للشركاء المحليين. فإن مسألة أمن هذه المنصة مهمة للغاية أنه يحتوي على العديد من الضعف.

مما يشكل مخاطر كبيرة على أمن المستخدمين والبيانات. تستطيع وقد أدى انتشار تطبيقات اندرويد إلى زيادة عدد البرامج الضارة، التقنيات التقليدية التي تستند إلى تسجيلات وتقارير الذكاء الاصطناعي أن تتبع التطور السريع والتكلفة المتزايدة للبرمجيات الخبيثة. توفر .الشبكات العصبية العميقة حلًا ملموسًا لتعزيز اكتشاف التطبيقات الخبيثة من خلال القدرة على تحليل النماذج المعقدة المختلفة

كلمات البحث: البرامج الضارة للأمان، تطبيق أندرويد، التعلم العميق، الكشف.

TABLE DE MATIÈRES

TABLE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ABRÉVIATIONS :

ANN : Artificial neural networks

API : interface de programmation d'application

CNN : Convolutional neural networks

DL : Deep Learning

DNN: Deep Neural Network

DT : Decision Tree

GPU : Graphics Processing Unit

GUI: Graphical User Interface

IRM : Imagerie par résonance magnétique JIT : Just In Time

Ml : Machine Learning

RAM: Random Access Memory

RNN: Recurrent neural networks

SDK: Software Development Kit

SVM : Machine à vecteurs de support

INTRODUCTION GÉNÉRALE

L'ère numérique actuelle est marquée par une utilisation massive et croissante des smartphones, parmi lesquels les appareils Android dominent largement le marché. Cette prolifération des dispositifs Android s'accompagne d'une augmentation exponentielle des applications disponibles, facilitant la vie quotidienne des utilisateurs en leur offrant une multitude de services et de fonctionnalités. Cependant, cette expansion rapide et diversifiée des applications Android a également ouvert la porte à une augmentation significative des risques de sécurité, notamment sous forme de logiciels malveillants, communément appelés malwares.

Les malwares représentent une menace sérieuse pour la confidentialité, l'intégrité et la disponibilité des données des utilisateurs. Ils peuvent voler des informations sensibles, endommager des systèmes, et même utiliser les ressources des appareils pour des activités malveillantes telles que l'espionnage ou l'extorsion. Face à l'ingéniosité et à la sophistication croissante des auteurs de malwares, les méthodes de détection traditionnelles, basées sur des signatures statiques et des heuristiques simples, se révèlent de moins en moins efficaces. Ces approches peinent à suivre le rythme des nouvelles variantes de malwares et des techniques d'offuscations utilisées pour échapper à la détection.

C'est dans ce contexte que les techniques d'apprentissage automatique, et plus spécifiquement les réseaux de neurones profonds (ou Deep Learning), émergent comme une solution prometteuse pour améliorer la détection des applications Android malveillantes. Les réseaux de neurones profonds sont capables d'apprendre des représentations complexes et d'analyser de grandes quantités de données pour identifier des schémas subtils et des comportements anormaux, ce qui les rend particulièrement adaptés à la tâche de détection de malwares.

L'objectif de ce mémoire est d'explorer l'application des réseaux de neurones profonds à la détection des applications Android malveillantes. Nous examinerons les différentes étapes nécessaires pour construire un système de détection basé sur le Deep Learning, allant de la collecte et du prétraitement des données à l'entraînement, la validation et le déploiement des modèles. En évaluant les performances de ces modèles, nous chercherons à démontrer leur efficacité et leur capacité à surmonter les limitations des méthodes traditionnelles.

Ce travail est structuré comme suit : nous commencerons par une revue de la littérature existante sur la détection de malwares et les techniques de Deep Learning. Ensuite, nous détaillerons la méthodologie adoptée pour notre étude, en décrivant les données utilisées, les processus de prétraitement, et les architectures de réseaux de neurones explorées. Nous présenterons ensuite les résultats expérimentaux obtenus et discuterons de leur implication en termes de sécurité mobile. Enfin, nous conclurons par une réflexion sur les perspectives futures et les améliorations possibles dans ce domaine.

En somme, ce mémoire vise à apporter une contribution significative à la lutte contre les malwares Android en démontrant le potentiel des réseaux de neurones profonds pour fournir une détection plus précise, efficace et adaptable aux nouvelles menaces.

Dans ce contexte et dans le cadre de notre projet de fin d’étude, nous avons organisé ce manuscrit de la façon suivante :

Dans le premier chapitre, nous aborderons le contexte d'étude en présentant les systèmes d'exploitation Android les plus couramment utilisés. Par la suite, nous examinons de manière approfondie la comparaison des systèmes d'exploitation mobiles, ainsi que l'architecture logicielle Android. Finalement, nous allons exposer la conclusion concernant la technologie mobile.

Le deuxième chapitre abordera la manière dont les menaces de sécurité et les logiciels malveillants sur Android sont présentées. Par la suite, nous examinerons les programmes malveillants en commençant par définir les applications Android malveillantes et les méthodes de détection de ces dernières. Enfin, nous terminerons avec les mesures préventives mises en place afin de diminuer les risques d'attaques contre ce système.

Dans le troisième chapitre, nous allons d'abord définir l'apprentissage automatique, ses diverses catégories, ainsi que les modèles les plus couramment employés. Par la suite, nous aborderons l'apprentissage profond en débutent par ces définitions, son histoire, ces divers modèles, ces indicateurs d'évaluation et ces fonctions d'activation.

Le quatrième chapitre présent notre contribution, où nous exposons notre architecture pour détecter les logiciels malveillants Android, ainsi nous exposons les données utilisées, les résultats expérimentaux et les outils fournis pour mettre en œuvre notre contribution proposée, suivis des interfaces de notre application pour donner le meilleur résultat.

CHAPITRE 1 :

LE SYSTEME D'EXPLOITATION ANDROID

1.1 Introduction :

L'Android est le système d'exploitation le plus fréquemment utilisé dans le domaine des Portables [1]. Cependant, étant donné que plus de 20 % des téléphones Android ont été vendus en 2018, le parc de téléphones Android n'est pas à jour dans son ensemble. Les téléphones portables sont toujours sous la version 2015 (Marshmallow) [2]. Les malwares ont donc des privilèges : environ 8400 nouveaux malwares sont recensés chaque jour [3].

La détection d'applications Android malveillantes est d'une importance capitale pour assurer la sécurité des utilisateurs de smartphones. Les applications malveillantes peuvent causer des dommages importants, tels que la collecte de données personnelles, l'installation de logiciels malveillants et la violation de la vie privée. Afin de lutter contre ces menaces, l'utilisation de réseaux de neurones profonds s'est révélée être une approche prometteuse.

Un utilisateur lambda a tendance à accorder automatiquement les autorisations demandées lors de l'installation d'une application, surtout lorsqu'il doit choisir entre autoriser toutes les permissions pour installer l'application ou ne pas installer du tout. [4]

Dans cette section, nous exposons les études les plus significatives portant sur la détection de logiciels malveillants sur les appareils Android, en explorant diverses techniques utilisées à cet effet.

1.2 Système d'exploitation Android :

Nous mettons l'accent sur la plateforme que nous avons choisie pour construire notre application. Nous avons choisi Android pour les raisons suivantes :

- Android est une plateforme ouverte, puissante, contemporaine et sécurisée. Les développeurs peuvent intégrer, développer et modifier des composants existants en ouvrant le code source et les API. Les applications peuvent être personnalisées par les utilisateurs en fonction de leurs besoins.
- Android utilise le noyau Linux, les grands magasins, la gestion des processus, le modèle de sécurité, la bibliothèque de soutien partagée, etc. sont quelques-uns des avantages.
- Le SDK Android fournit une API complète pour la création d'applications Android. Android Develop Challenge offre aux développeurs la possibilité de gagner beaucoup d'argent, car Google offrira 10 millions de dollars de prix inconditionnels pour les applications sur la plateforme Android. [5]

1.3 Comparaison des systèmes d'exploitation mobile :

1. Le tableau ci-dessous offre une analyse de la comparaison en ce qui concerne la configuration, le déploiement et le constructeur.

	Ios	**Black Berry**	**Windows Phone**	**Android**
Langage de programmation	Objective-C Intégré a Xcode	Java Gratuit	C, C++ Gratuit	Java Gratuit
Disponibilité de l'environnement De développement	Xcode	JDE	Visual Studio, eMbeddeb VC++	Eclipse Netbeans, Android Studio
Multiplateforme de de déploiement	IPhone, iPod touch, iPad	BlackBerry Seulement	Windows Mobile Windows CE	Android seulement
Cout d'outils de développement	Gratuit	Gratuit	Gratuit	Gratuit
Open source	Non	Oui	Non	Oui
Constructeur	Apple	RIM	Microsoft	Google

Table 1:Une comparaison entre les systèmes d'exploitation mobile

2. La gamme Windows de Microsoft, créée en 1985, équipe en 2008 près de 90 % des ordinateurs personnels, ce qui la place en position de monopole, notamment auprès du grand public. Pour la première fois depuis 15 ans, ses parts de marché ont chuté en dessous de 90 % en 2008. [7] Ensuite, suite à la forte croissance du marché des smartphones et à la prise de retard de Microsoft sur ce

marché, ses parts de marché sur les appareils personnels ont augmenté de 95 % en 2005 à 20 % en 2013 [8].

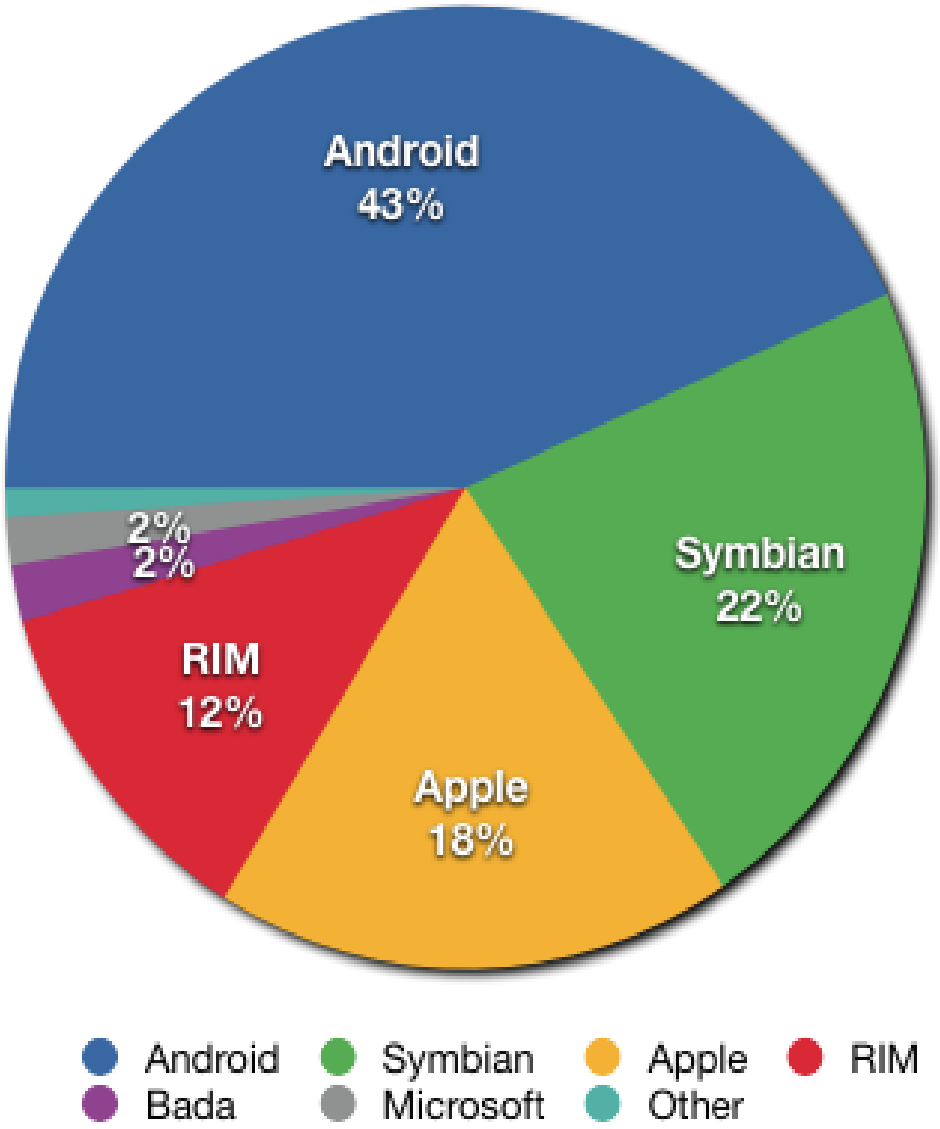


Figure 1:Parts des systèmes d'exploitation des smartphones en 2011 dans le monde[9].

3. Dans cette section, nous présentons les systèmes d'exploitation pour smartphones entre 2009 et 2016.

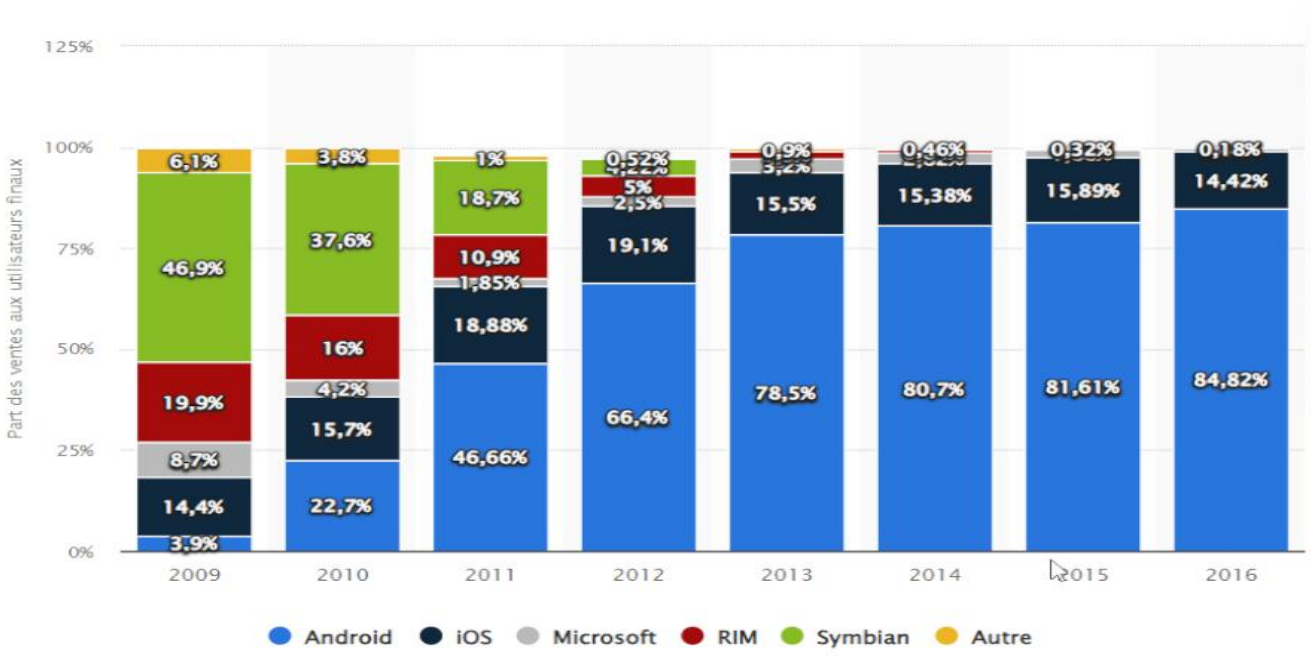


Figure 2: Part de marché mondiale détenue par les systèmes d'exploitation pour Smartphones entre 2009 et 2016 [10].

La représentation graphique illustre la part de marché des systèmes d'exploitation pour smartphones en termes d'expéditions d'unités entre 2009 et 2016. Il est évident qu'Android occupe une position dominante sur le marché international. Le système d'exploitation Android devrait avoir environ 86% des ventes mondiales de smartphones en 2024, ce qui en fait le leader du marché.

1.3.1 Naissance d'Android :

Android a été créé en 2003 par une société américaine appelée Android, mais Google l'a acheté deux ans plus tard (2005). L'objectif principal était de créer un système d'exploitation simple pour l'utilisateur. Chaque fabricant avait auparavant son propre système d'étiquetage. En conséquence, il était impossible de créer une application compatible pour tous les appareils, ni de concevoir des bibliothèques de développement dont les secrets de la marque n'ont pas été révélés car ils avaient été fournis et verrouillés. [5]

1.3.2 Les versions d'Android :

S'il n'avait pas changé en six ans, le système de Google n'aurait pas eu autant de succès. Nous voyons ici la puissance d'un tel système d'exploitation, qui peut s'adapter aux besoins des utilisateurs avec chaque version principale et être enrichi de nouveautés. [5]

Nom de code	Version	Les fonctions importantes qui ajouter	Logo
Apple pie	1.0	Seulement ou presque connu des développeurs car c'est la version du SDK qui a été déployée avant la sortie du premier téléphone Android.	
Bananas split	1.1	• Beta, version sur le premier téléphone, qui est HTC G1 / Dream	
Cupcake	1.5	• Nouvelles fonctionnalités et mises à jour GUI	
Donut	1.6	• Nouvelles fonctionnalités et mises à jour GUI	
Eclair	2.0	• Nouvelles fonctionnalités et mises à jour GUI	
Froyo	2.2	• Vitesse améliorée, nouvelles fonctionnalités et mises à jour GUI	
Gingerbread	2.3	• La dernière version dédiée uniquement aux Smartphones. Cette version est parfois utilisée sur les petites tablettes	
Marshmallow	6.0	• Supports USB de type C. • Support pour l'authentification des empreintes digitales. • Meilleure autonomie de l'accumulateur en mode sommeil profond. • Panneau de commande pour contrôler les autorisations d'utilisation.	
Nougat	7.0	• Meilleur support multitâche. • Comptage multiple. • Améliorations des systèmes (par le biais de la double partition du système). • Capacité de code améliorée et performance avec un nouveau compilateur JIT.	
Nougat	7.1	• Éclairage en mode nuit. • Gestionnaire de mémoire amélioré. • Meilleure gestion de l'écran et du toucher. • Option pour activer les gestes sur le capteur d'empreintes digitales. • Mises à jour du système "transparentes".	

Oreo	8.0	• Mode Picture in Picture. • Gestion multitâche et démarrage rapide. • Gestion des notifications. • Les API d'auto complétion.	
Oreo	8.1	• Ajout de l'affichage du pourcentage de batterie d'un appareil Bluetooth dans les paramètres (et paramètres rapides) du Bluetooth. • Ajout de l'affichage des vitesses de transmission des réseaux Wifi disponibles. • Ajout de l'API « Neural Networks » (NNAPI) afin de permettre à des opérations hors-ligne de machine learning d'utiliser l'accélération matérielle d'un appareil • Amélioration des performances pour les appareils à faible RAM.	
Pie	9.0	• Nouvelle navigation système à gestes, elle introduit des déférents systèmes intelligents comme la gestion de luminosité, économiseur de la batterie.	
Pas de Nom	10	• Le système d'autorisations est mise à jour. • Offre un bon contrôle sur les autorisations des applications sur vos données.	
Pas de Nom	11	• - Google se localise sur la confidentialité, surtout sur la mise à jour des autorisations sur un accès limité sur l'appareille photo, le microphone et l'emplacement de l'appareil, seules les applications légitimes selon Google peuvent accéder aux emplacements des appareilles d'utilisateurs.	

Table 2: Les versions Android. [5] Ce tableau présente tous les versions Android à partir de 2011 à 2020.

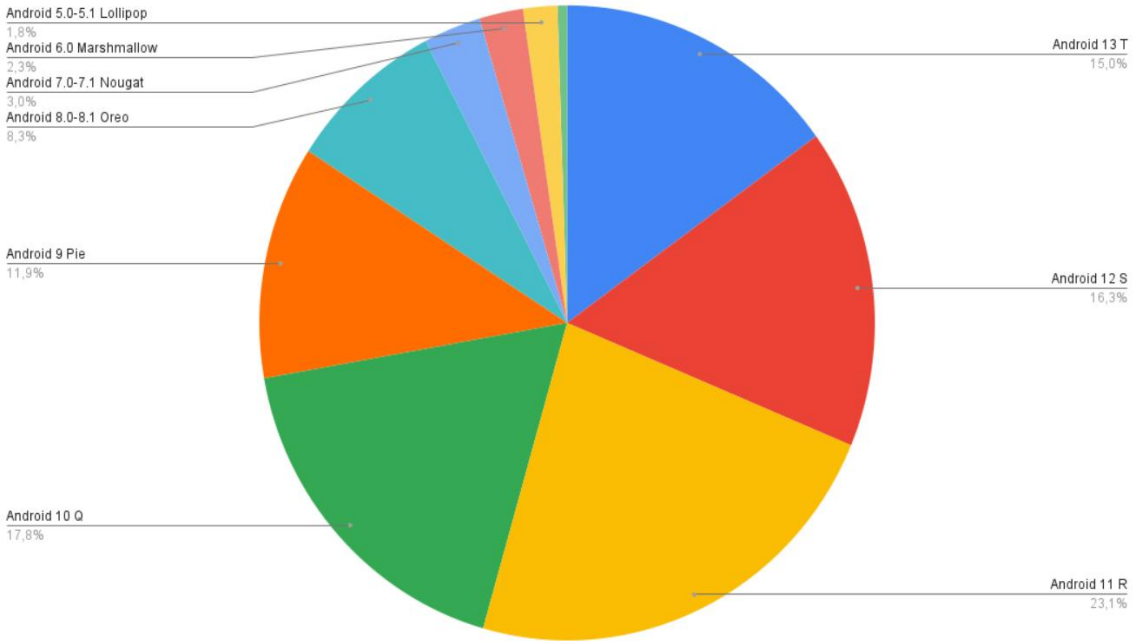


Figure 3: Fragmentation d'android en mai 2023 [11].

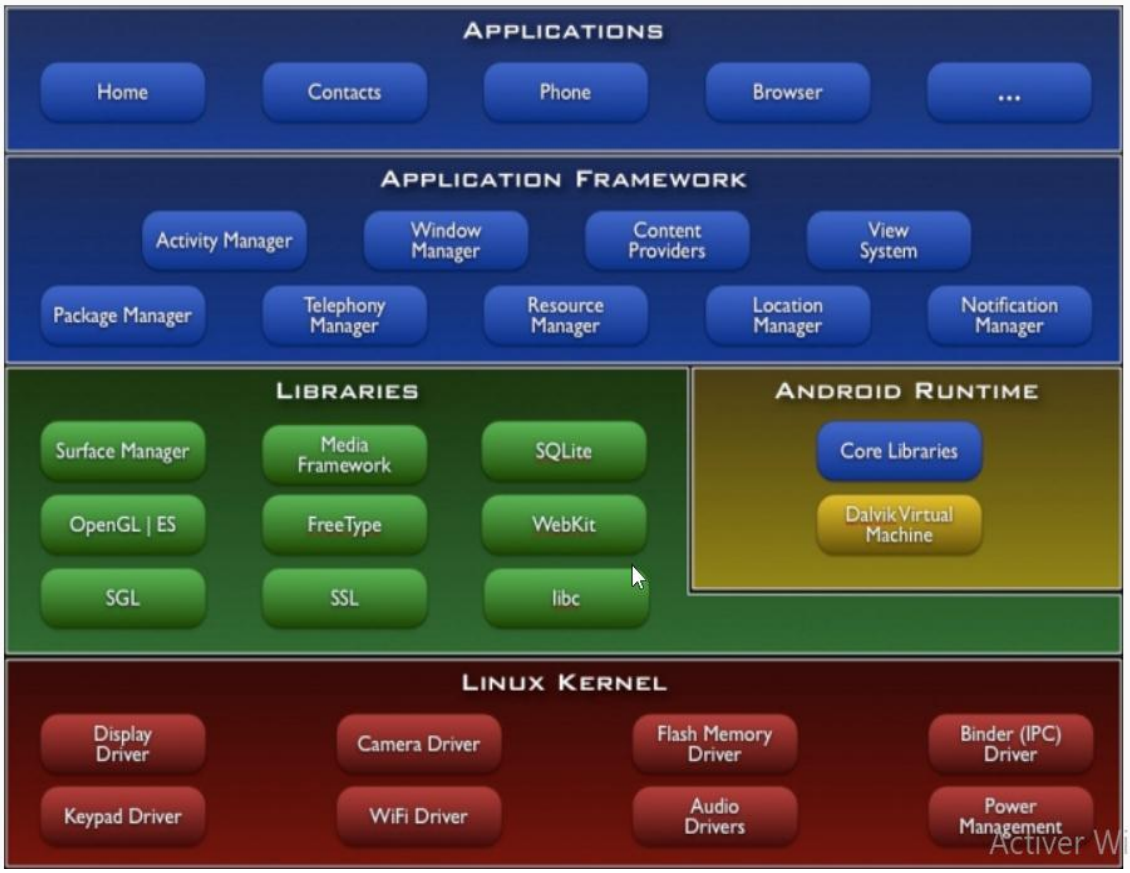


Figure 4: Architecture du logiciel Android [12].

Android est un système d'exploitation en temps réel avec différents niveaux, y compris les applications, le cadre d'application, les bibliothèques, le noyau Linux et le gestionnaire de l'exécution

Android. Les applications comprennent toutes les applications téléphoniques comme la navigation, les contacts, les appareils photo et les calendriers. Le cadre de l'application comprend des classes et des services pour accéder aux données, tels que la gestion des stocks, la gestion de ressources et la gestion d'alarmes. Ces niveaux sont cruciaux pour une exécution efficace de l'application. [12]

En français, le niveau "Libraries" correspond au niveau de la bibliothèque, qui correspond au niveau le plus bas du logiciel. Il comprend plusieurs bibliothèques en C/C, langues, plateformes web, bibliothèques multimédias (image, audio, vidéo), etc. [12]. Le "Linux Kernel", également connu sous le nom de "couche de base de Linux", joue un rôle essentiel dans l'architecture Android, car il offre des services essentiels, ces services incluent la connexion avec le matériel et logiciel. Gestion de la sécurité, de l'énergie, du stockage... etc. Aussi, en français, la couche "Moteur d'exécution Android" comprend la Dalvik Virtual Machine (DVM) ainsi que les comptoirs principaux. [12]

1.4 Conclusion :

Android reste un pilier central de l'innovation technologique mobile, avec un impact majeur sur la manière dont les gens interagissent avec la technologie au quotidien. Son modèle open-source a permis une adoption massive et une diversité d'appareils, tout en posant des défis que Google continue d'aborder pour améliorer l'expérience utilisateur.

Android est non seulement un système d'exploitation, mais aussi une force motrice dans l'évolution des technologies mobiles et une plateforme incontournable pour les développeurs et les utilisateurs du monde entier.

CHAPITRE 2 :

MENACES DE SÉCURITÉ ET MALWARES

ANDROID

2.1 : Introduction :

Une application Android malveillante est une application informatique conçue pour s'exécuter sur des appareils Android, mais créée dans le but de causer des préjudices, compromettre la sécurité des données de l'utilisateur ou effectuer des actions nuisibles sans le consentement explicite de l'utilisateur. Ces applications peuvent être diffusées via divers canaux, notamment des boutiques d'applications tierces non autorisées ou des liens de téléchargement sur des sites Web peu fiables, [13]. Il existe principalement deux méthodes de détection : l'analyse statique et l'analyse dynamique. Chacune des deux suit un principe de détection unique.

L'analyse, qu'elle soit statique ou dynamique, vise à créer des signatures et des profils comportementaux pour identifier un logiciel malveillant, c'est-à-dire les caractéristiques partagées par les échantillons du malware lors de leur exécution. Les méthodes d'apprentissage sont aujourd'hui employées pour classifier et repérer des comportements anormaux.

L'analyse d'une application est effectuée dans le but d'obtenir des informations pertinentes. Comme son comportement, les ressources qu'elle doit utiliser, et tout ce qui est attendu pour exécuter une partie de son code. [14]

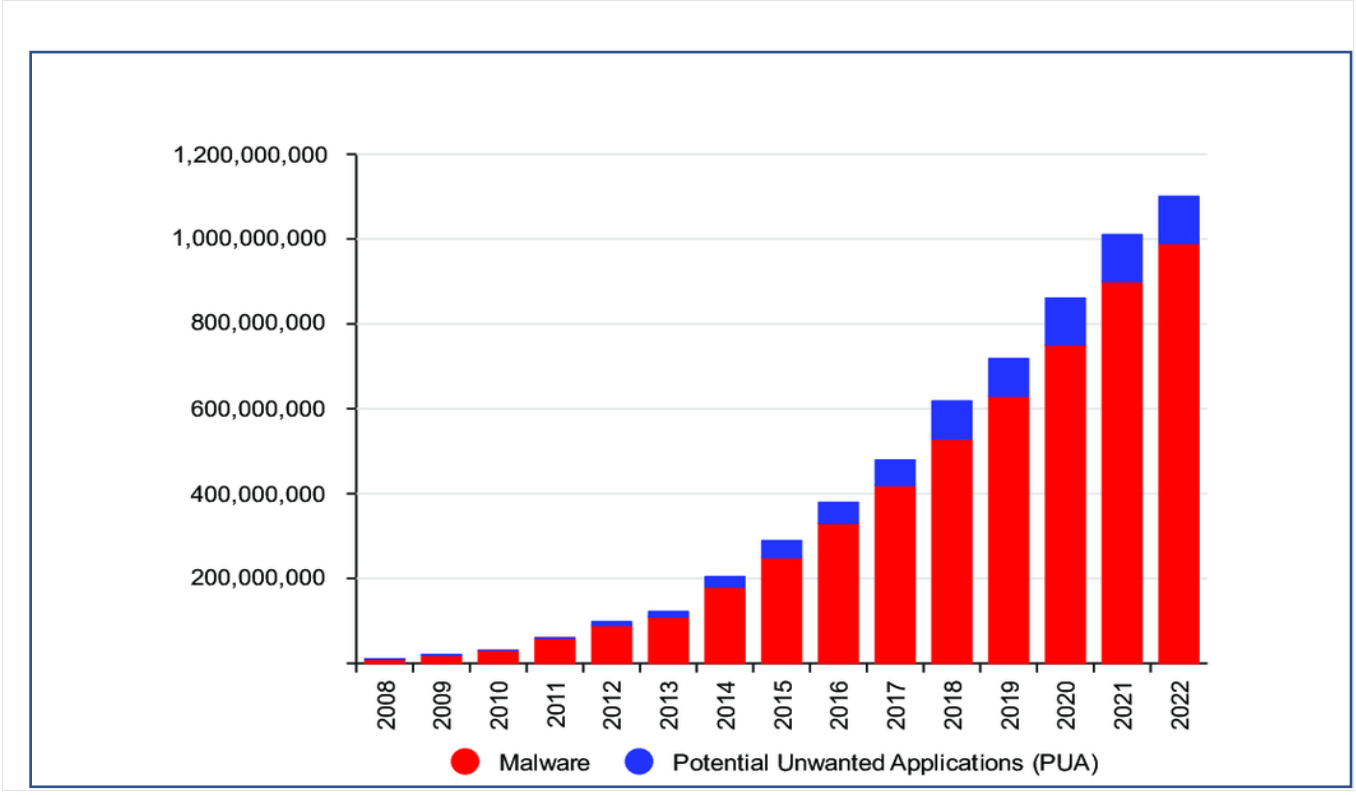


Figure 5:Nombre total de logiciels malveillants et de PUA de 2008 à 2022[15].

2.2 Techniques d'analyse des malwares :

Plusieurs chercheurs ont récemment examiné les méthodes et les techniques de détection de logiciels malveillants pour les applications Android. Naway et LI [16] ont présenté une étude de la dynamique, statique, et l'analyse hybride qui utilise des techniques d'apprentissage profond pour détecter le logiciel malveillant Android. Ils ont fourni un détail des méthodes et ont expliqué leurs forces et faiblesses. Une autre enquête sur la détection de logiciels malveillants Android a été menée par Alzahrani et Alghazzawi [17]. Ils ont étudié huit articles de recherche qui se concentraient sur l'apprentissage profond pour la détection de Ransomware Android et l'apprentissage profond pour la détection de logiciels malveillants Android. Rubiya et Radhamani [18] ont toutefois analysé neuf méthodes de détection de logiciels malveillants Android, en décrivant leurs avantages et inconvénients.

Yan et Zheng dans leur article [19] ont examiné la détection de logiciels malveillants mobiles dynamiques. Ils ont analysé, synthétisé et comparé des études antérieures sur la détection de logiciels malveillants dans les smartphones. Dans une autre enquête, Arshad et al. [20], ont analysé les techniques statiques et dynamiques pour la détection et la protection contre les logiciels malveillants Android. Les techniques analysées sont classées selon le mécanisme de détection utilisé.

Base de données	**dynamique**	**statique**	**Hybride**	**Nombre l'article étudié**	**Années**
Naway and LI	X	X	X	25	2014–2018
Alzahrani et Al-ghazzawi	X	X		8	2014–2019
Rubiya et Radhamani	X	X	-	9	2009-2014
Yan et Zheng	X	-	-	29	2011–2017
Arshad et al	X	X	-	20	2009-2013

Table 3: Comparaison des techniques d'analyses des malwares

Comme on peut voir dans le tableau 3, l'enquête se concentre sur les méthodes de détection de logiciels malveillants Android statiques, dynamiques et hybrides. Il suit l'évolution de la détection de logiciels malveillants au cours d'une période de onze ans (entre 2009 et 2020).

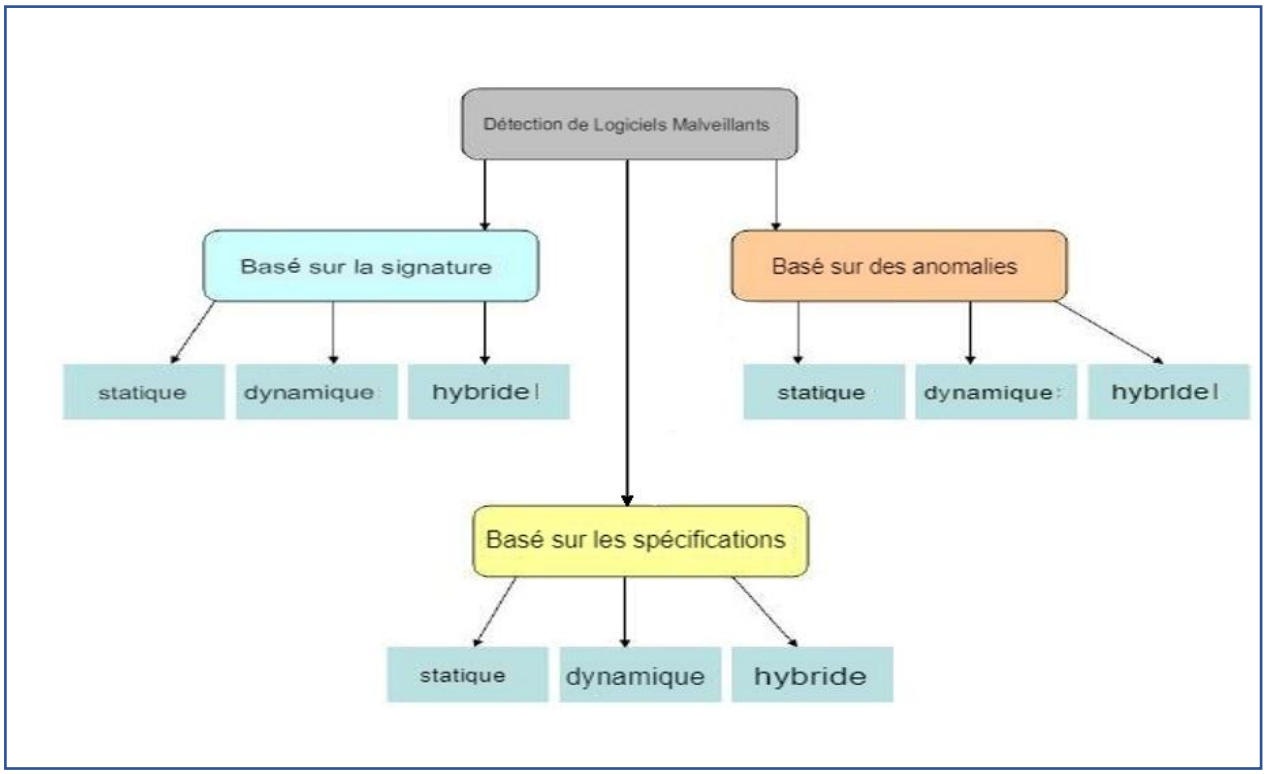


Figure 6: Classification des techniques de détection des logiciels malveillants [21].

L'étape d'analyse revêt une importance capitale dans la détection des logiciels malveillants. Pour ce faire, il est nécessaire de comprendre la fonctionnalité de ces programmes ainsi que les motivations qui ont présidé à leur développement. Cela facilitera leur compréhension par les développeurs lors de l'élaboration d'un plan de défense contre ces malwares. Les techniques d'analyse des logiciels malveillants peuvent être classées en trois catégories distinctes :

2.2.1 Analyse statique :

L'analyse statique se concentre sur l'analyse des informations statiques, telles que les fichiers exécutables et les codes sources, obtenues sans exécuter l'application. L'avantage de l'analyse statique est que la vitesse de détection est rapide et que le surcoût de calcul est relativement faible. L'inconvénient est que les malwares utilisant la technologie d'obscurcissement ne peuvent pas être correctement analysés par une analyse statique. [22]

Bien qu'il soit impossible de détecter toutes les erreurs dans tous les programmes, ce n'est pas une raison d'abandonner les méthodes d'analyse statistique. Au lieu de cela, utilisez des méthodes qui fonctionnent assez bien sur la plupart des programmes réels [23].

Les techniques appliquées doivent être fiables, mais pas nécessairement optimales. Il est possible que certaines erreurs restent dans le système. Les outils d'analyse statistique ne peuvent pas déterminer exactement s'il y a ou non une erreur. Il y a donc toujours un risque que les outils signalent

des incidents qui n'existent pas ou qu'ils ne trouvent pas d'erreurs dans le code. Il existe deux grandes catégories d'analyses statiques : la vérification de modèle et l'interprétation abstraite.

2.2.2 Analyse dynamique :

L'analyse dynamique est une technique de surveillance d'une application en un environnement sandbox ou sur un appareil réel, permettant un examen approfondi de l'échantillon de logiciels malveillants sans endommager le système. Il recueille et analyse les données sur le comportement de l'application, en identifiant les comportements malveillants non détectés par des méthodes d'analyse statistique. Cependant, il a des temps d'analyse et de détection longs et des coûts de calcul élevés. [24]

La recherche suggère un moyen de collecter et de diviser un grand nombre d'échantillons de mauvaises pratiques. Tout échantillon de malware est analysé de manière dynamique pour identifier ses traces et détecter son comportement. Le cluster algorithme utilise une méthode probabiliste et atteint une grande précision de 98 % [25]. Ils ont présenté un moyen d'effectuer une analyse dynamique des applications Android pour les classer. Ils ont créé un système de capture d'appels, système qui collecte et extrait les traces d'appels du système pour les applications lors des interactions d'exécution. Après cela, les données d'appel système recueillies sont analysées pour identifier et classer le comportement des applications Android. Pour évaluer leur comportement, ils ont utilisé une base de données contenant cinquante applications Android mauvaises provenant du Google Play Store officiel et cinquante applications mauvaises provenant du projet Android Malware Genome. Ils ont atteint un niveau de précision de 85 % en utilisant un arbre de décision et un niveau de précision de 88 % en utilisant un algorithme de forêt aléatoire dans leurs expériences. [26]

L'analyse dynamique consiste à exécuter et à surveiller les actions exécutées par une application. Au contraire de l'analyse statique et de l'étude du code-source de l'application, l'analyse dynamique consiste à étudier le comportement de l'application : appel de fonctions, chaines stockées en mémoire, trafic généré, etc.

On utilise l'analyse dynamique pour l'étude de malwares (dans ce cas-là, le recours à un environnement sécurisé), et quand il n'est pas possible d'accéder au code-source de l'application (légalité du reverse engineering) [27].

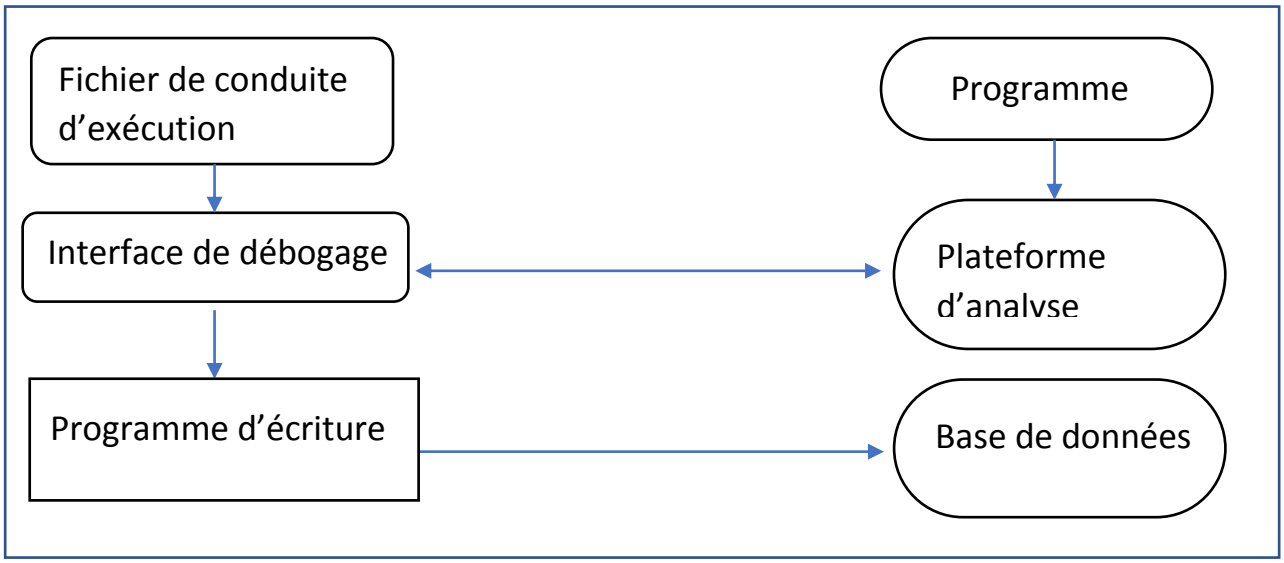


Figure 7:Fonctionnement de la plate-forme d'analyse dynamique proposé par [28].

2.2.3 Analyse hybride :

L'analyse hybride est une combinaison d'analyse statique et dynamique. Cette méthode analyse l'application pour déterminer ses caractéristiques statiques et dynamiques. Cela augmente la précision de la reconnaissance. Son avantage est qu'elle est plus complète en analyse applicative et qu'elle allie les avantages des deux méthodes d'analyse. Cependant, les inconvénients sont que l'analyse et la détection prennent beaucoup de temps, que l'analyse nécessite beaucoup de ressources système et que la surcharge de calcul est énorme. [29]

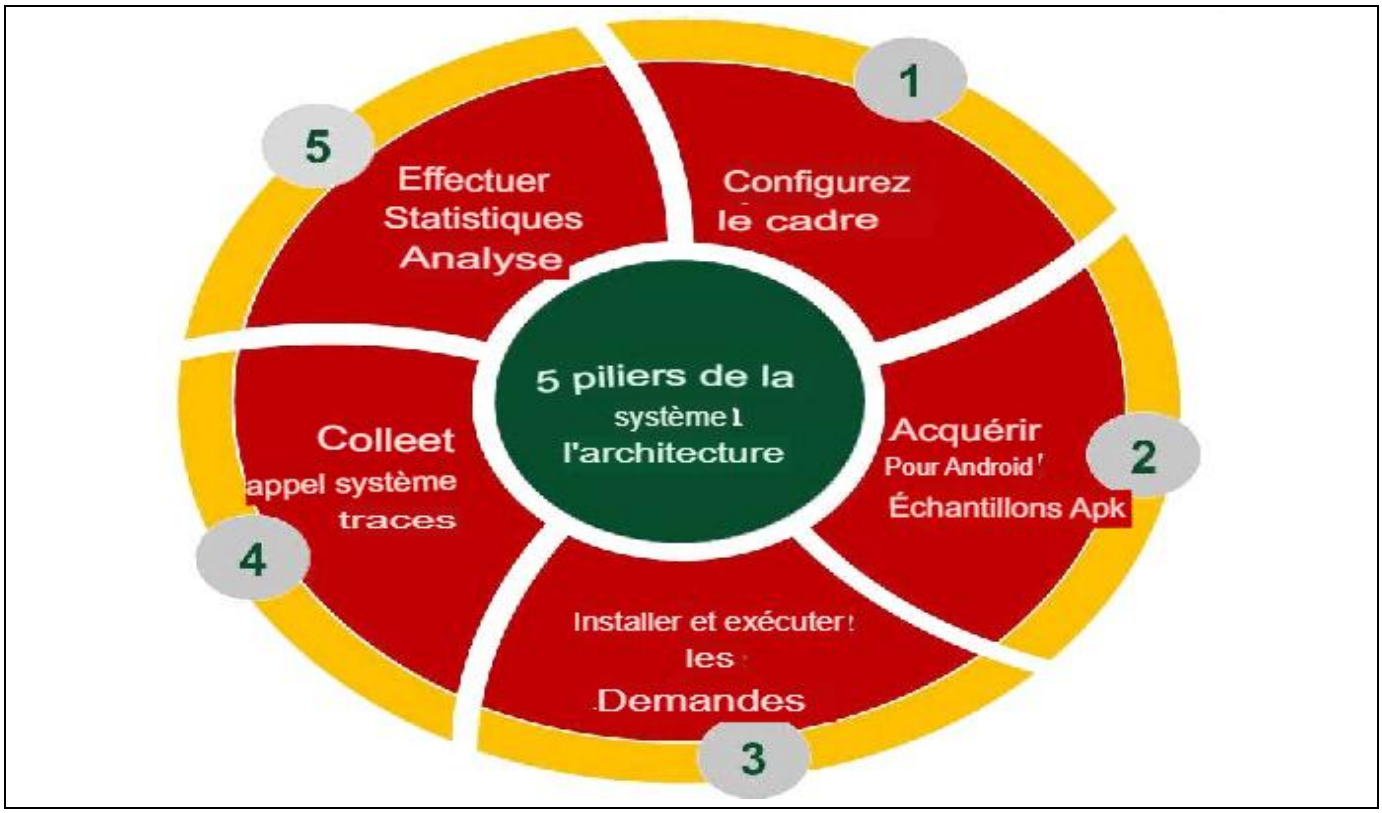


Figure 8: L'architecture du système proposé dans [30].

Lutter efficacement contre l'obscurcissement ou le cryptage du code. La méthode d'analyse hybride est toujours considérée comme la méthode la plus approfondie et la plus complète, malgré le coût élevé et la complexité de mise en œuvre qui limitent son déploiement [31].

2.3 Classification des malwares :

Les malwares, ou logiciels malveillants, sont des programmes informatiques conçus dans le but de nuire à un système informatique, d'endommager des données, de voler des informations confidentielles ou de compromettre la sécurité d'un réseau. Ils sont généralement classés en fonction de leur fonctionnalité et de leur mode d'action. Voici quelques-unes des principales catégories de malwares :

2.3.1 Virus informatique :

Les virus sont des programmes malveillants qui s'attachent à des fichiers exécutables ou à des secteurs de démarrage de périphériques de stockage. Ils se propagent en infectant d'autres fichiers lors de leur exécution.

2.3.2 Ver informatique (computer Worms) :

Un ver informatique désigne un programme malveillant qui se propage à travers un réseau. La technologie informatique, telle que l'Internet, vise à infecter un maximum de systèmes. Une fois qu'il a été exécuté, il peut se reproduire. À l'inverse du virus, les vers informatiques ont la capacité d'exploiter les informations. Les réseaux sont configurés de manière erronée. Le ver se propage sans avoir besoin de se connecter à d'autres programmes d'exécution. Il tire parti des vulnérabilités de sécurité afin de se propager d'un ordinateur à un second. [32] Deux types de vers informatiques se distinguent par la façon dont ils se propagent, La première approche repose sur l'envoi d'un courrier électronique infecté, où le virus se propage en utilisant le carnet d'adresses d'un client de la messagerie électronique. Pour le deuxième, les vers sur les serveurs vulnérables.

2.3.3 Cheval de Troie (Trojan Horse) :

Les Trojans sont des programmes malveillants qui se font passer pour des logiciels légitimes pour inciter les utilisateurs à les installer. Ils peuvent ouvrir des portes dérobées, voler des données, ouvrir des backdoors, etc.

Les chevaux de Troie peuvent permettre au hacker de réaliser diverses tâches à distance. La machine infectée peut être infectée par différentes actions, comme la suppression des fichiers, le téléchargement de fichiers, les modifications des paramètres du registre, l'arrêt de certaines applications comme les anti-virus, etc.

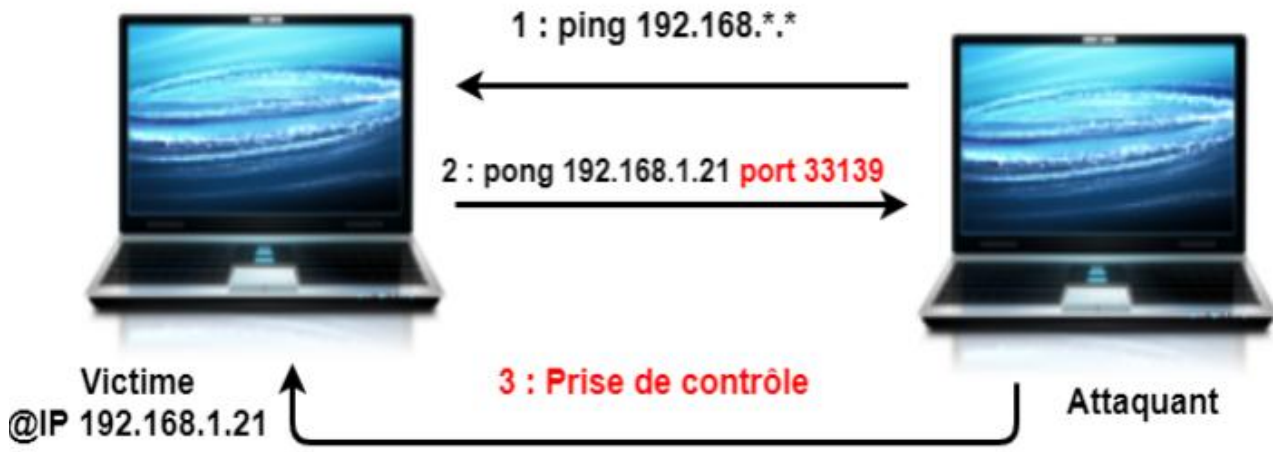


Figure 9: Le fonctionnement du cheval de Troie.

2.3.4 Porte dérobée (backdoor) :

L'utilisateur légitime n'a pas connaissance de la porte dérobée. Elle tire son origine d'un cheval de Troie dissimulé dans un programme ou un service connecté. Fréquemment mise en place par un développeur de logiciel ou un prestataire de service. Une fois que la porte dérobée a réussi à pénétrer le système, elle va surveiller un utilisateur, gérer ses fichiers, installer des logiciels supplémentaires ou des logiciels malveillants et surveiller l'ensemble du système de l'ordinateur. Bien que les vulnérabilités de sécurité du système soient résolues, elles ne seront pas détectées.

2.3.5 Les keyloggers :

Un Keylogger a pour fonction de recueillir les caractères enregistrés sur un clavier d'une machine. En général, les keyloggers sont conçus pour démarrer le système et enregistrent tout caractère saisi, que ce soit dans les courriers, les messages, les documents, voire même les noms d'utilisateurs et les mots-passes. En général, cela est réalisé en utilisant une fonction Hook [33]. Il est possible que l'administrateur d'une entreprise installe un keylogger pour superviser les activités des employés, tout comme un hacker peut l'installer pour obtenir des informations sur ses victimes.

La figure 9 présente la manière dont un Keylogger fonctionne.

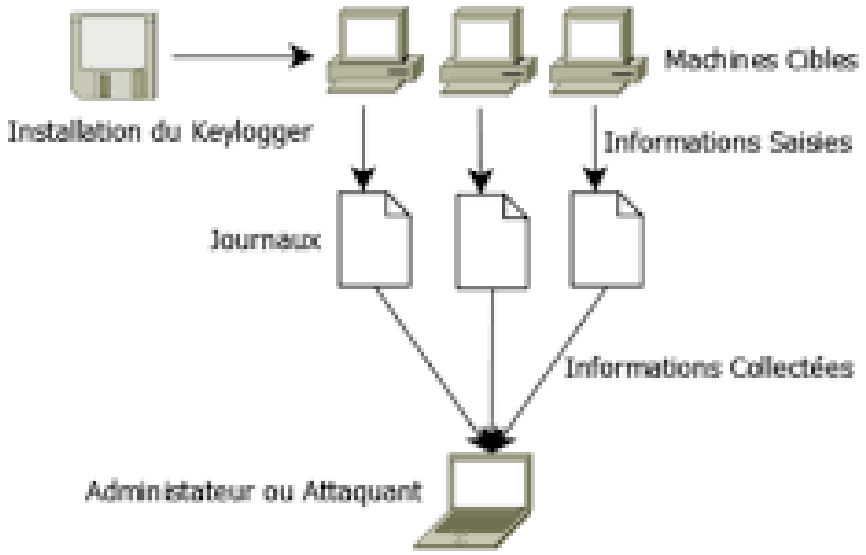


Figure 10: Fonctionnement de Keylogger **[33].**

2.3.6 Logiciel de sécurité falsifié (rogueware) :

On peut également le désigner sous le terme de "rogue". Ce genre de logiciel se présente comme un outil de sécurité, prétendant être un antivirus ou un antispyware, mais en réalité, il s'agit d'un faux logiciel ou d'un malware vendu par des développeurs de logiciels malveillants ou de fausses entreprises. Ces programmes commencent par persuader les utilisateurs que leur machine est infectée par des malwares, mais une fois achetés, ils se révèlent être eux-mêmes des logiciels malveillants.

2.3.7 Rançongiciel (ransomware) :

Définir tous les types de logiciels malveillants utilisés dans le but de demander une somme d'argent en échange de la restitution d'une information ou d'une fonctionnalité dérobée. Ce genre de logiciel malveillant a pour objectif de chiffrer toutes les informations de la machine et de demander à une victime de transférer de l'argent afin d'obtenir la clé de chiffrement.

2.3.8 Rootkit :

Les Rootkit, à l'instar d'un fantôme, sont des programmes malveillants, autonomes et extrêmement stricts. Ils utilisent cette technique pour éliminer les traces de l'attaquant sur la machine infectée. Leur présence est dissimulée dans le registre du système. L'objectif est d'accéder à un ordinateur le plus rapidement possible [34]. Un rootkit a pour but d'espionner et d'accéder aux données stockées ou en transit sur la machine cible. En utilisant ces rootkits, le pirate a également la capacité d'intercepter le trafic Internet, de lire les e-mails et de suivre les entrées de clavier de la machine infectée.

2.3.9 Logiciel espion (Spyware) :

Ce type de logiciel se trouve souvent intégré à des logiciels gratuits, des pages web et des sites proposant des contenus illégaux. Les logiciels espions utilisent diverses méthodes pour espionner et se propager. Parmi celles-ci, on peut citer : l'affichage d'offres publicitaires incitant au téléchargement d'autres logiciels infectés, la capture de mots de passe en enregistrant les touches tapées sur le clavier, la surveillance des programmes exécutés à des heures spécifiques, ainsi que la surveillance des sites d'internet visités. [35] Cette classification n'est pas exhaustive, et de nouveaux types de malwares peuvent émerger à mesurer que les techniques d'attaque évoluent et que de nouvelles vulnérabilités sont découvertes.

Malwares/propriété	virus	Trojan	Worn	Spyware	Adware	Rootkit
Auto réplication	+	-	+	-	-	+
Espionnage	+	+	+	+	-	+
Composants cachés	-	-	+	-	-	+
Déni de service	+	+	+	-	-	+
Auto-propagation	+	+	+	-	-	+
Furtivité	-	+	+	+	-	-
Exécutable	-	-	+	-	-	+
Insérer une charge Dans la cible	+	-	+	-	+	+
Publicité	-	-	-	-	+	-

Table 4: Comparaison de fonctionnement entre quelques types des malwares

+ : à cette propriété - : n'as pas cette propriété **[36].**

2.4 Menaces de sécurité des applications mobiles :

La principale source de menace en matière de sécurité des applications mobiles est la propagation de code malveillant qui exploite les vulnérabilités du système d'exploitation et des applications en utilisant des méthodes d'ingénierie sociale, comme le courrier indésirable et les messages SMS, ce qui peut entraîner la fuite de données personnelles ou des bénéfices financiers [37].

La Menace	Explication
Malware	Menaces installées sur le terminal pour comportement malveillant.
Spam	Menaces utilisées pour distribuer des publicités et des logiciels malveillants qui peuvent être envoyés à un nombre indéterminé de personnes.
Application vulnérabilités	Menaces qui exécutent des actions malveillantes telles que l'élévation des privilèges en utilisant la vulnérabilité de l'application développée.
Informations personnelles extrusion	Menace de fuite d'informations personnelles en raison de la négligence de l'utilisateur lors du développement applications installées.
Contournement d'authentification	Menaces qui contournent ou volent l'authentification au hasard pour les applications qui nécessitent une authentification.
DoS	Menaces qui rendent le service fourni par l'application inutilisable

Table 5: Menaces à la sécurité dans les applications mobiles [37].

2.3 Les logiciels malveillants Android :

De différentes manières, les applications malveillantes constituent un véritable problème qui impacte le système d'exploitation Android. Dans cette partie, nous exposons le cycle de vie des logiciels malveillants, sa famille et les méthodes de détection.

2.3.1 Cycle de vie des logiciels malveillants :

a) Création :

Lors de cette étape, le programmeur élabore et met en place l'intégralité du code malveillant qui sera intégré dans le logiciel malveillant.

b) Gestation :

Étape au cours de laquelle l'application malveillante s'infiltre et s'installe dans le système qu'elle souhaite infecter. Il reste inactif tout au long de cette étape. C'est pourquoi sa présence reste totalement inconnue pour l'utilisateur.

c) Reproduction ou infection :

Le logiciel malveillant se reproduit de manière significative avant de se manifester à cette étape. Le programme malveillant vise à prendre le contrôle à distance des appareils et à accéder aux informations personnelles. Sur Android, les logiciels malveillants se propagent par le biais du partage

de fichiers ou des méthodes d'ingénierie sociale. Il utilise des moyens de communication tels que SMS, Bluetooth et Wifi et se transforme souvent en une application ordinaire.

d) Activation :

Certains programmes malveillants mettent en marche leur processus de destruction lorsque certaines conditions sont remplies (par exemple, lorsque le compte à rebours interne atteint). L'activation peut aussi se faire à distance. Cette étape vise à acquérir progressivement toutes les ressources des appareils.

e) Découverte :

Un comportement étrange est observé par l'utilisateur, qui soupçonne la présence d'une application malveillante. Cet étrange comportement peut entraîner des performances diminuées, des changements en cours dans la page d'accueil du navigateur Web ou l'absence de certaines fonctionnalités du système. La plupart du temps, les antivirus permettent à l'utilisateur de repérer les actions malveillantes en envoyant des alertes au propriétaire de l'appareil.

f) Assimilation :

La base de données virale des antivirus est mise à jour après la découverte de nouveaux logiciels malveillants. Une solution ou un antidote est également suggérée pour éradiquer cette menace, si cela est possible.

g) Élimination :

C'est la phase où l'antivirus détecte le logiciel malveillant et encourage l'utilisateur à le désinstaller.

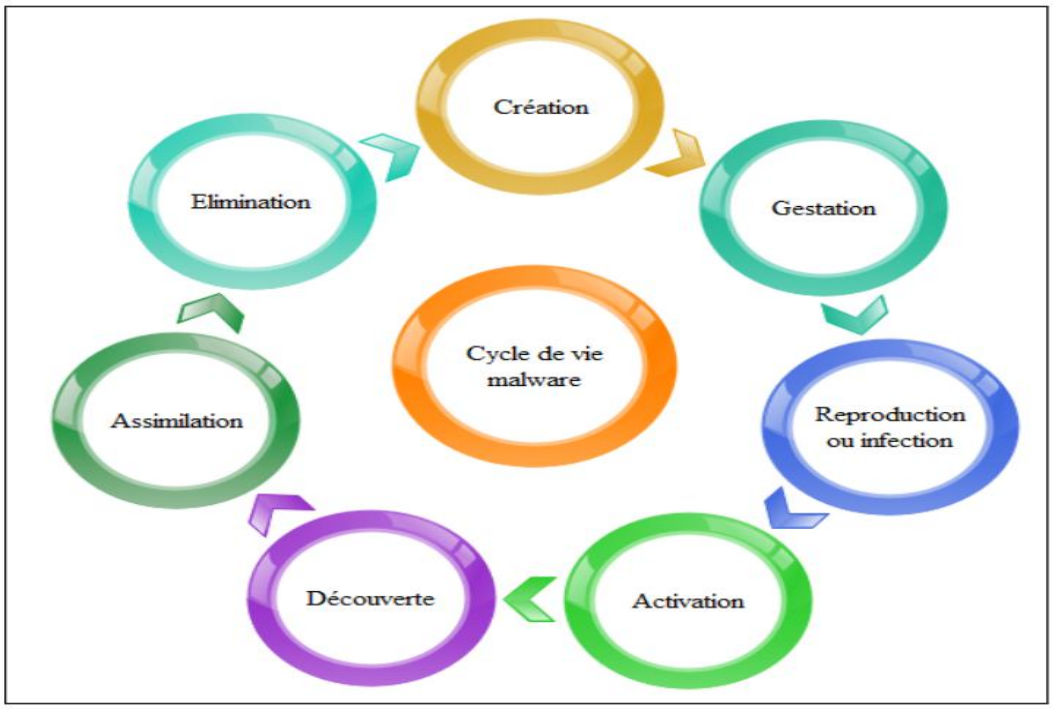


Figure 11: Cycle de vie des logiciels malveillants

2.4 Contre-mesures :

Les contre-mesures contre les logiciels malveillants sont des actions et des stratégies mises en place pour prévenir, détecter, contrer et neutraliser les menaces posées par les malwares. Voici quelques-unes des contre-mesures les plus couramment utilisées :

2.4.1 Antivirus :

Les antivirus sont des logiciels de sécurité largement utilisés sur les appareils mobiles, bénéficiant de la popularité qu'ils ont acquise sur les ordinateurs. Des noms bien connus tels qu'Avast, AVG et F-Secure sont des exemples d'antivirus réputés pour Android. Cependant, avec l'évolution rapide des applications malveillantes, ces antivirus font face à de nouveaux défis. Leur efficacité sur les appareils mobiles est étroitement liée à leurs capacités de détection, similaire à ce qui est observé sur les plates-formes de bureau. [38]

2.4.2 Pare-feu :

Il est peut-être moins important d'utiliser un pare-feu sur les appareils mobiles Android qu'un PC, mais il peut être bénéfique pour gérer l'accès Internet afin d'améliorer la sécurité, l'optimisation des données et des performances. [39]

2.4.3 Système de détection d’intrusions :

Un système d'intrusion de détection (IDS) est un dispositif qui permet de repérer les tentatives de compromettre un système. Il est possible que cela empêche de telles tentatives. Ainsi, le système est désigné sous le nom de système de prévention des intrusions. Les systèmes de détection des intrusions utilisés sur les appareils mobiles Android reposent sur les mêmes principes que ceux utilisés dans d'autres systèmes (ordinateurs personnels, réseaux informatiques). Malgré les grandes différences entre les systèmes en termes de type et d'architecture, les bases de la protection contre les attaques demeurent les mêmes. Grâce à cela, les techniques peuvent être adoptées et utilisées dans la zone de sécurité Android [40].

2.4.4 Analyse des logiciels malveillants :

L'étude du fonctionnement des logiciels malveillants se concentre sur l'analyse des conséquences potentielles de l'infection d'un logiciel malveillant particulier. Il convient de noter que les logiciels malveillants peuvent présenter diverses caractéristiques.

Les attaquants ont développé chaque fonctionnalité malveillante afin d'accéder au système à travers diverses sources et d'infecter sans le consentement de l'utilisateur [41].

1.5 Conclusion :

Dans ce chapitre, nous avons examiné les méthodes de détection de logiciels malveillants pour Android, en se concentrant sur les avantages et les inconvénients de chacun. Malgré le fait qu'un grand nombre de solutions ont été proposées, plusieurs défis restent à résoudre, en particulier en raison de la nature en évolution rapide du logiciel malveillant. Nous citons les difficultés liées à l'obscurcissement

du code, l'indisponibilité du code source et le problème émergent de la collusion de logiciels malveillants comme des problèmes qui nécessitent une attention particulière dans un avenir proche.
À la lumière de notre analyse, il apparait clairement que le développement d'outils de détection de logiciels malveillants plus efficaces est une nécessité.

Le deep learning offre une approche plus dynamique et adaptative en utilisant des réseaux de neurones artificiels pour analyser de grandes quantités de données et apprendre des modèles et des comportements malveillants.

Chapitre3 :

TECHNIQUE D'APPRENTISSAGE PROFOND POUR AMELIORER LA SECURITE

3.1. Introduction

À partir des années 2000 et surtout après 2010 les réseaux de neurones font des progrès fulgurants grâce à l'apprentissage profond. Yann Le Cun démontre l'efficacité des couches de convolution pour la reconnaissance des chiffres. On réalise et entraîne alors des réseaux ayant de plus en plus de couches grâce à des progrès matériels (par exemple le calcul sur les processeurs graphiques GPU) mais surtout grâce aux couches de convolution qui extraient des caractéristiques abstraites des images [42]. L'apprentissage profond, également connu sous le nom de deep learning, est une méthode utilisant l'intelligence artificielle et l'apprentissage automatique pour faciliter la collecte, l'analyse et l'interprétation de grandes quantités de données par les data scientists. Cette approche, également appelée apprentissage neuronal profond ou réseau neuronal profond, permet aux ordinateurs d'apprendre en observant, imitant ainsi le processus d'acquisition de connaissances humaines.

Le cerveau humain est constitué d'une grande quantité de neurones connectés les uns aux autres, qui servent de messagers lors du traitement des informations. Ces neurones communiquent entre eux en utilisant à la fois des impulsions électriques et des signaux chimiques, permettant ainsi la transmission d'informations à travers les différentes régions du cerveau.

L'apprentissage profond améliore aussi la prise de décision en identifiant des tendances subtiles dans de vastes ensembles de données, ouvrant ainsi de nouvelles possibilités dans des domaines allant de la finance à la médecine. De plus, il favorise la personnalisation des expériences utilisateur en comprenant les comportements individuels. Ces réseaux de neurones sont capables de détecter des schémas et des relations complexes dans les données, ce qui leur permet de résoudre des problèmes difficiles et de prendre des décisions autonomes. L'apprentissage profond est largement utilisé dans des domaines tels que la vision par ordinateur, la reconnaissance vocale et la traduction automatique la prédiction des marchés financiers, et bien d'autres.

L'avènement de l'apprentissage automatique ouvre de nouvelles portes aux entreprises, leur offrant la possibilité de résoudre des défis complexes, d'accroître leur efficacité opérationnelle et de stimuler l'innovation.

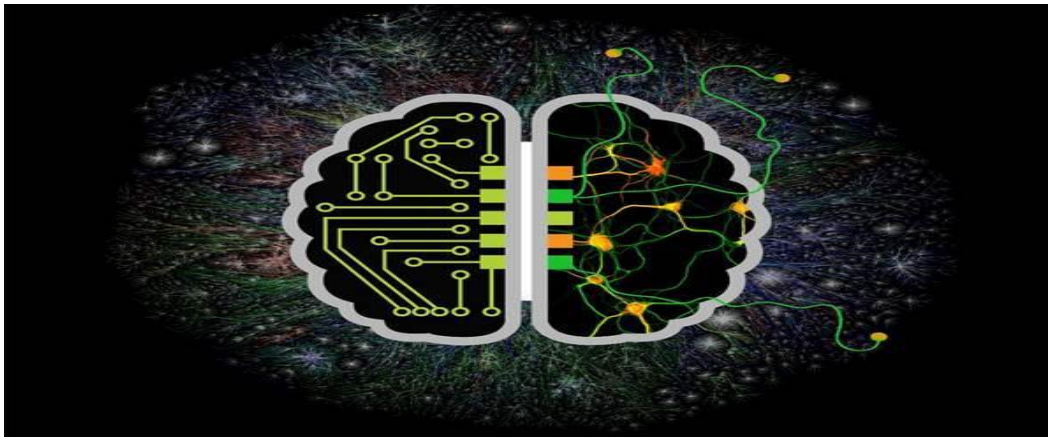


Figure 12: Concept de micro puce en forme de cerveau de l'intelligence artificielle

3.2. Apprentissage automatique (ML) :

3.2.1 Définition

L'apprentissage automatique vu comme un domaine multidisciplinaire qui combine les statistiques et l'informatique. Plusieurs définitions sont données dans ce contexte : L'apprentissage automatique a été inventé par Arthur Samuel, dès 1952 qui a créé le premier programme permettant de jouer et d'apprendre le jeu de dames [43]. L'apprentissage automatique répond à la question de savoir comment construire des ordinateurs qui améliorent automatiquement par l'expérience. Il s'agit aujourd'hui de l'un des domaines techniques à la croissance la plus rapide, situé à l'intersection de l'informatique et des statistiques, et au cœur de l'intelligence artificiel et science des données [44]. L'apprentissage automatique (ML) est utilisé pour aider les machines à gérer efficacement les données. Lorsque les informations extraites des données ne sont pas facilement interprétables, l'apprentissage automatique est appliqué. Avec la disponibilité croissante de vastes ensembles de données, la demande d'apprentissage automatique est en hausse, et de nombreuses industries l'utilisent pour extraire des données pertinentes.

Le but de la machine learning est d'apprendre à partir des données. De nombreuses études ont été réalisées sur la façon de fabriquer des machines apprendre par eux-mêmes sans être explicitement programmé. De nombreux mathématiciens et programmeurs appliquent plusieurs approches pour trouver la solution à ce problème [45]. Le processus d'apprentissage est défini comme une combinaison de trois activités [46] :

A— Représentation :
Le modèle doit être représenté dans un langage formel que l'ordinateur peut le gérer.

B— Evaluation :
Une fonction (fonction objective ou de notation) est nécessaire pour distinguer entre les bons modèles et les mauvais.

C— Optimisation :
Une technique d'optimisation est fondamentale pour rechercher parmi les modèles le plus performant.

3.2.2 Différents types d'apprentissage automatique :

3.2.2.1 Apprentissage supervisé :

L'apprentissage supervisé consiste à apprendre une cartographie entre un ensemble de variables d'entrée X et une variable de sortie Y et à appliquer cette cartographie pour prédire les sorties pour les données invisibles. L'apprentissage supervisé est la méthodologie la plus importante en machine learning et il a également une importance centrale dans le traitement des données multimédias [47].

3.2.2.2 Apprentissage non-supervisé :

L'approche d'apprentissage non supervisé consiste à reconnaître des modèles existants non identifiés à partir des données. Afin d'en déduire des règles. Cette technique est appropriée dans une situation où les catégories des données sont inconnues. Ici, les données d'entraînement ne sont pas

étiquetées. L'apprentissage non supervisé est considéré comme une statistique approche basée sur l'apprentissage et fait ainsi référence au problème de trouver une structure cachée dans des données non étiquetées [48].

3.2.2.3 Apprentissage semi-supervisé :

Ces algorithmes utilisent à leur avantage à la fois l'apprentissage supervisé et l'apprentissage non supervisé pour offrir une approche combinée puissante.
L'utilisation de l'apprentissage semi-supervisé permet de combiner l'apprentissage non supervisé de manière efficace. Parfois, certaines observations sont étiquetées, mais la plupart ne le sont pas en raison du coût élevé de l'étiquetage et du manque d'experts qualifiés.
Dans de telles circonstances, les algorithmes semi-supervisés sont les plus appropriés pour construire des modèles. L'apprentissage semi-supervisé peut-être appliqué à des problèmes tels que la classification, la régression et la prédiction.

3.2.2.4 Apprentissage par renforcement :

L'apprentissage par renforcement est considéré comme un type d'apprentissage intermédiaire car l'algorithme ne reçoit qu'une réponse indiquant si le résultat est correct ou non. L'algorithme doit explorer et exclure diverses possibilités pour obtenir le résultat correct. Cela est considéré comme un apprentissage avec un critique car l'algorithme ne propose aucune sorte de suggestions ou de solutions au problème [48].

3.2.3 Modèles d'apprentissage automatique :

Nous vous proposons ci-dessous un aperçu des modèles de classification les plus populaires utilisés pour détecter les malwares Android :

3.2.3.1 Réseau de neurones artificiels (ANN) :

L'idée de conception d'un ANN est d'imiter le fonctionnement du cerveau humain [47]. Un réseau de neurones artificiels (ANN) est constitué d'un grand nombre d'unités et a la capacité théorique d'approximer des fonctions arbitraires, ce qui lui confère une grande adaptabilité, notamment pour les fonctions non linéaires. La formation d'un ANN est longue en raison de la complexité de sa structure. Il est important de noter que les modèles ANN sont entraînés à l'aide de l'algorithme de rétro-propagation, qui n'est pas adapté à la formation de réseaux profonds. Ainsi, un ANN est considéré comme un modèle peu profond et se distingue des modèles d'apprentissage profond.

3.2.3.2 Machine à vecteurs de support (SVM) :

L'algorithme SVM utilise des techniques non paramétriques dérivées de la théorie de l'apprentissage statistique. SVM utilise souvent les fonctions du noyau pour projeter l'espace multidimensionnel des données sous forme de points, puis trouve la meilleure classification de l'hyperplan, étant finalement classé selon ce plan [49].

3.2.3.3 K-means :

K-means (partitioning clustering) est une méthode de clustering utilisée pour classer les objets en k groupes (clusters). Cette méthode doit spécifier le nombre initial de k groupes supervisés. Chaque

groupe est normalement centré autour de sa moyenne, et l'algorithme itératif est utilisé pour minimiser la distance entre chaque observation et sa moyenne correspondante [50].

3.2.3.4 Arbre de décision (DT) :

L'arbre de décision est un graphique pour représenter les choix et leurs résultats sous la forme d'un arbre. Les nœuds du graphique représentent un événement ou un choix et les bords du graphique représentent les règles ou conditions de décision. Chaque arbre est constitué de nœuds et de branches. Chaque nœud représente des attributs dans un groupe qui doit être classé et chaque branche représente une valeur que le nœud peut prendre [45].

3.2.3.5 Clustering :

L'analyse groupée, également connue sous le nom de clustering, est une technique d'apprentissage automatique non supervisée permettant d'identifier et de regrouper des points de données associés dans de grands ensembles de données sans se soucier du résultat spécifique. Il regroupe une collection d'objets de telle manière que les objets de la même catégorie, appelés cluster, soient en quelque sorte plus similaires les uns aux autres que les objets des autres groupes [51].

3.2.3.6 Classifieur naïf de Bayes (NB) :

Naïf bayes est un modèle de classification obtenu en appliquant une méthode relativement simple à un ensemble de données de formation. Un classificateur NB calcule la probabilité qu'une instance donnée (exemple) appartienne à une certaine classe. Cela fait l'hypothèse simplificatrice que les fonctionnalités constituant l'instance sont conditionnellement indépendantes étant donné la classe [52].

3.2.3.7 Comparaison de différents modèles d'apprentissage automatique :

Algorithmes	Avantages	Inconvénient
ANN	Capable de traiter des données non linéaires ; Forte capacité d'adaptation ;	Convient au sur-apprentissage ; A tendance à rester coincé dans un optimum local ; La formation des modèles prend du temps ;
SVM	Apprenez des informations utiles à partir d'un petit train ; Forte capacité de génération	Ne fonctionne pas bien sur les méga données ou les tâches de classification multiples ; Sensible aux paramètres des fonctions du noyau ;
Naïve Bayes	Robuste au bruit ; Capable d'apprendre progressivement	Ne donne pas de bons résultats sur les données liées aux attributs ;
Arbre de décision	Sélectionnez automatiquement les fonctionnalités ; Interprétation forte ;	Tendances des résultats de la classification à la classe majoritaire ; Ignorer la corrélation des données ;

K-means	Simple, peut être formé rapidement ; Grande évolutivité ; Peut s'adapter aux méga données ;	Ne fonctionne pas bien sur des données non convexes ; Sensible à l'initialisation ; Sensible au paramètre K ;

Table 6: Les avantages et les inconvénients des algorithmes d'apprentissage automatique [53].

3.3 Apprentissage profond (DL) :

L'apprentissage profond (ou « deep learning ») est un algorithme qui tente d'utiliser l'abstraction de haut niveau des données en utilisant plusieurs couches de traitement constituées de structures complexes ou de multiples transformations non linéaires.

L'apprentissage profond est un algorithme basé sur la caractérisation des données d'apprentissage. L'essence de l'apprentissage profond est un réseau neuronal multicouche qui utilise l'entrée de la couche précédente comme sortie de la couche suivante pour apprendre des caractéristiques de données très abstraites [54].

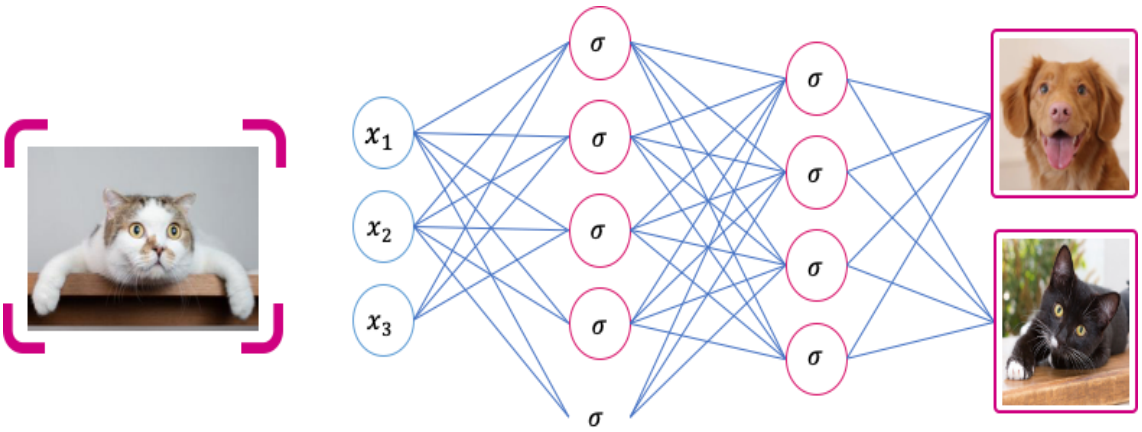


Figure 13: Classification d'une image

3.3.1 Fonctionnement :

Les réseaux neuronaux profonds sont constitués de multiples couches de nœuds interconnectés, chacune tirant parti de la couche précédente pour affiner et optimiser les prédictions ou catégorisations. Cette progression des calculs à travers le réseau est appelée avant propagation. Les couches d'entrée (Input Layer) et de sortie (Output Layer) d'un réseau neuronal profond sont appelées « couches visibles ».
La couche d'entrée est celle où le modèle d'apprentissage profond prend en compte les données à traiter, tandis que la couche de sortie est celle où la prédiction ou la classification finale est réalisée. Chaque paire de couches adjacentes est interconnectée par des poids (Weights).

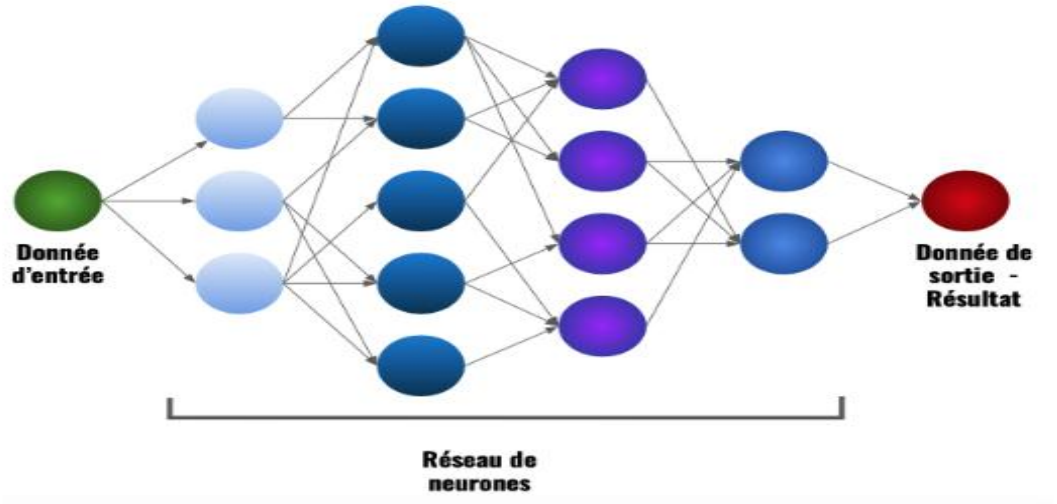


Figure 14: L'architecture d'un modèle Deep Learning

3.3.2 Classification des méthodes Deep Learning :

La classification joue un rôle important dans toutes les sciences et techniques qui font appel à la statistique multidimensionnelle, et ces méthodes peuvent être classées en trois modèles, en fonction de leur formation et leur intention d'utilisation :

3.3.2.1 Apprentissage supervisé :

En apprentissage supervisé, on dispose d'un ensemble de données étiquetées, où chaque exemple a été associé à une classe par un enseignant ou un expert. Cet ensemble d'exemples forme la base d'apprentissage. Les méthodes d'apprentissage supervisé visent à construire des fonctions de classification à partir de cette base d'apprentissage. Ces fonctions permettent, à partir de la description d'un objet, de reconnaître un attribut spécifique.

3.3.2.2 Apprentissage non supervisé :

L'apprentissage non supervisé implique l'entraînement de modèles sans étiquetage préalable des données, les algorithmes regroupant les données en fonction de leur similitude sans intervention humaine.

3.3.2.3 Apprentissage par renforcement :

Une méthode de plus en plus utilisée. Elle consiste à laisser les ordinateurs apprendre de leurs expériences grâce à un système de récompense ou de pénalité. Il pourrait même s'agir de la clé permettant l'avènement d'une intelligence artificielle générale comparable à celle de l'humain… [55]

3.3.3 Quelques méthodes d'apprentissage profond :

Les réseaux neuronaux profonds se caractérisent par un agencement de neurones disposés en plusieurs couches interconnectées. Ce qui les distingue principalement, c'est leur architecture spécifique, déterminée par l'organisation et le fonctionnement des neurones au sein du réseau. Parmi les nombreuses implémentations de modèles d'apprentissage profond, on peut citer :

3.3.3.1 Deep Neural Network (DNN) :

Un réseau de neurones profond est composé d'une couche d'entrée, d'une couche de sortie et d'au moins une couche intermédiaire. La profondeur d'un réseau est déterminée par le nombre de ces couches, et plus ce nombre est élevé, plus le réseau est considéré comme profond. Chaque couche effectue des opérations de tri et de catégorisation spécifiques dans le cadre d'un processus appelé "hiérarchie de caractéristiques".

Le principe fondamental de la perception a été établi par Rosenblatt en 1958. La perception opère en calculant une sortie unique à partir de plusieurs entrées de valeurs réelles (xi). Elle réalise cela en combinant linéairement les entrées avec leurs poids respectifs (w), puis en appliquant une fonction d'activation non linéaire à cette combinaison. Mathématiquement, cette opération peut être représentée comme suit :

$$Y = \delta\left(\sum_{n=1}^{n} W_i x_i + b\right) = \delta\left(W^T X + b\right)$$

W : le vecteur des poids.

X : le vecteur des entrées.

b : le biais.

δ : la fonction d'activation.

Un réseau classique de perception multi-couches (MLP) comprend un ensemble de nœuds sources qui constituent la couche d'entrée, une ou plusieurs couches cachées de nœuds de calcul et une couche de sortie de nœuds. Le signal d'entrée se répand à travers le réseau couche par couche.
DNN sont souvent employés dans les situations d'apprentissage supervisé. L'apprentissage implique l'ajustement de tous les poids et les biais à leurs valeurs optimales.

3.3.3.2 Convolutional neural networks (CNNs) :

Les réseaux de neurones convolutifs (CNN) démontrent des performances de pointe dans les tâches de classification d'images multi-classes à grande échelle [56].

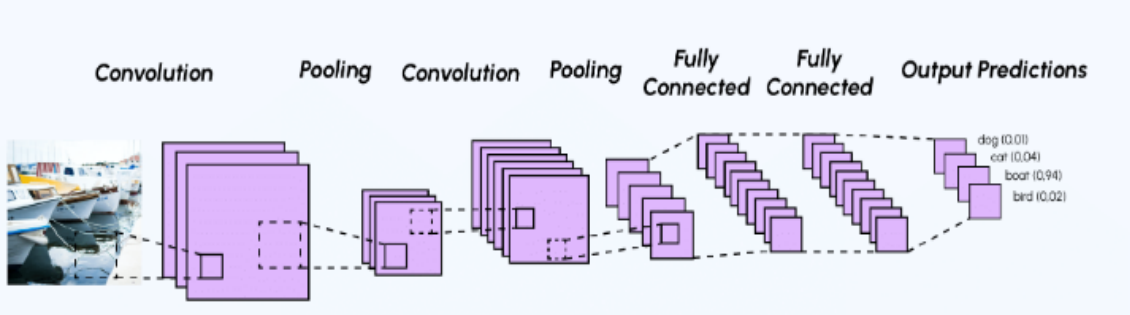


Figure 15 : Exemple d'architecture d'un CNN

3.3.3.3 Recurrent neural networks (RNNs) :

Les réseaux de neurones récurrents (RNN) sont adaptés au traitement de données séquentielles. Un RNN traite une séquence de données en traitant chaque élément de la séquence un à la fois. Un réseau RNN n'a qu'un seul réseau caché **[**57**]**.

Les réseaux de neurones récurrents sont très flexibles et ont donc été utilisés pour résoudre divers problèmes tels que la modélisation du langage, la reconnaissance vocale, traduction automatique et sous-titrage d'images **[**58**]**.

Les RNNs ont la capacité de traiter une séquence de données en entrée et de générer une séquence de données en sortie. Leur particularité réside dans leur capacité à conserver en mémoire toutes les informations calculées précédemment. De plus, ils partagent les mêmes paramètres pour chaque entrée, ce qui permet de réduire la complexité des paramètres par rapport à d'autres types de réseaux neuronaux.

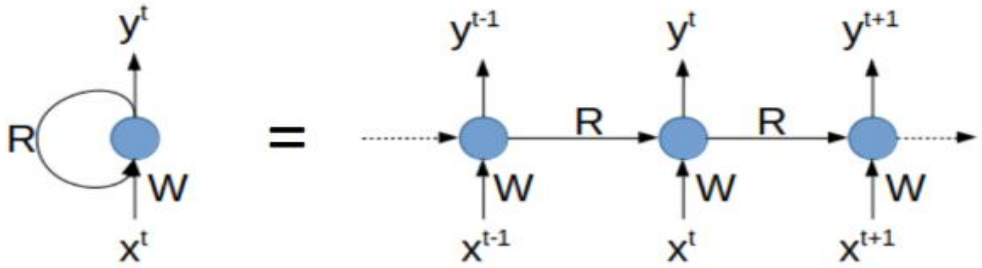


Figure 16: L'architecture d'un modèle RNN

3.3.4 Applications d'apprentissage profond :

Au cours des dernières années, l'apprentissage profond a été appliqué avec succès pour résoudre un large éventail de problèmes dans une variété de domaines d'application : Les robots, les entreprises, la cyber sécurité, les assistants virtuels, la reconnaissance d'images, les soins de santé et bien d'autres encore en sont des exemples. Ils impliquent également l'analyse des sentiments et le traitement du langage naturel [59].

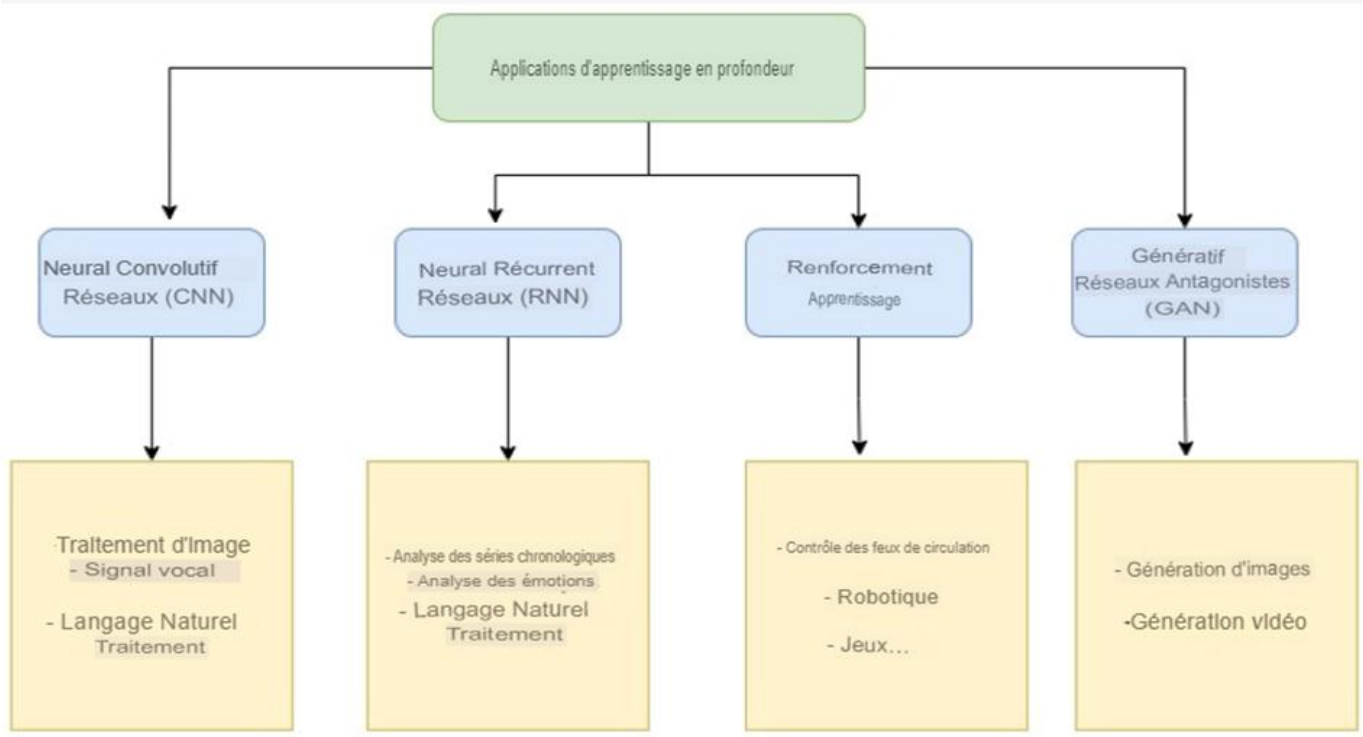


Figure 17: Domaines d'application potentiels d'apprentissage profond dans le monde réel [60].

3.3.4.1 Reconnaissance d'objets dans les images :

L'apprentissage profond a permis des avancées significatives dans des applications telles que la détection d'objets, la reconnaissance faciale, la classification d'images et la segmentation sémantique. Ces modèles peuvent être entraînés sur de grands ensembles de données annotées pour améliorer leur précision et leur capacité à généraliser de nouvelles images.

3.3.4.2 Biométrie

Dans le domaine de la sécurité, en particulier pour le contrôle d'accès, l'apprentissage profond est utilisé en conjonction avec les caractéristiques biométriques. La création et la mise au point des dispositifs de reconnaissance faciale ont été accélérées par l'utilisation du DL. Cette entreprise estime qu'en neuf mois, les capacités d'identification d'une personne à une autre de ses produits pourront être accrues. Le développement de ce moteur aurait pu prendre cinq fois plus de temps sans la mise en œuvre de DL. Il a permis d'accélérer la fabrication et l'entrée sur le marché des nouvelles machines. Les algorithmes d'apprentissage profond s'améliorent au fur et à mesure qu'ils acquièrent de l'expérience. Les prochaines années devraient être remarquables, car la technologie continue à progresser [61].

3.3.4.3 Traitement du langage naturel :

L'application de l'apprentissage profond dans le traitement du langage naturel considérablement améliore la capacité des machines à comprendre et à générer le langage humain, ouvrant la voie à de nombreuses applications innovantes dans ce domaine.

3.3.4.4 Mobile :

Les smartphones et les vêtements équipés de capteurs révolutionnent un certain nombre d'applications mobiles, notamment la surveillance de la santé [62]. L'apprentissage profond est considéré comme un élément crucial pour comprendre cette nouvelle forme de données.
Néanmoins, seules quelques publications récentes dans le domaine de la détection des soins de santé ont utilisé des modèles profonds, principalement en raison de restrictions matérielles.

En réalité, l'exécution d'une architecture profonde efficace et fiable sur un appareil mobile pour analyser des données de capteurs bruyantes et complexes est un effort difficile qui risque d'épuiser les ressources de l'appareil [63].

3.3.4.5 Imagerie clinique :

L'apprentissage profond a ouvert de nouvelles perspectives passionnantes dans le domaine de la médecine, en particulier en ce qui concerne l'analyse d'images médicales telles que les IRM cérébrales. Grâce à des algorithmes d'apprentissage profond, il est possible de détecter des motifs subtils dans les images médicales qui pourraient échapper à l'œil humain, ce qui peut être crucial pour le diagnostic précoce de maladies comme la maladie d'Alzheimer. Ces avancées technologiques offrent de nouvelles opportunités pour améliorer les soins de santé et la précision des diagnostics tels que : détection et diagnostic des maladies graves, découverte et fabrication de médicaments, tenue des dossiers de santé, essais cliniques et recherche.

3.3.5 Apprentissage profond pour la cyber sécurité :

Pour répondre aux défis du cyber sécurité, diverses approches interdisciplinaires telles que l'apprentissage automatique, l'exploration de données, les statistiques et d'autres techniques sont utilisées en exploitant les vastes ensembles de données provenant des infrastructures réseau, des systèmes d'exploitation et des systèmes d'information.

L'apprentissage profond, une composante de l'apprentissage automatique [64], pourrait être employé pour repérer les logiciels malveillants en se basant sur leur signature, il est devenu un outil crucial en cyber sécurité :

3.3.5.1 Identification des Menaces :

L'apprentissage profond permet une analyse avancée des schémas de cyber menaces, facilitant ainsi la détection rapide et précise des activités suspectes.

3.3.5.2 Surveillance en Temps Réel :

Grâce à des modèles sophistiqués, les systèmes alimentés par l'apprentissage profond peuvent surveiller en temps réel les réseaux et les comportements pour détecter les anomalies.

3.3.5.3 Adaptation Continue :

La capacité des algorithmes d'apprentissage profond à apprendre à partir de nouvelles données leur permet de s'adapter aux nouvelles menaces et de renforcer la résilience des systèmes.

3.4 Comparaison entre la machine learning et le Deep Learning :

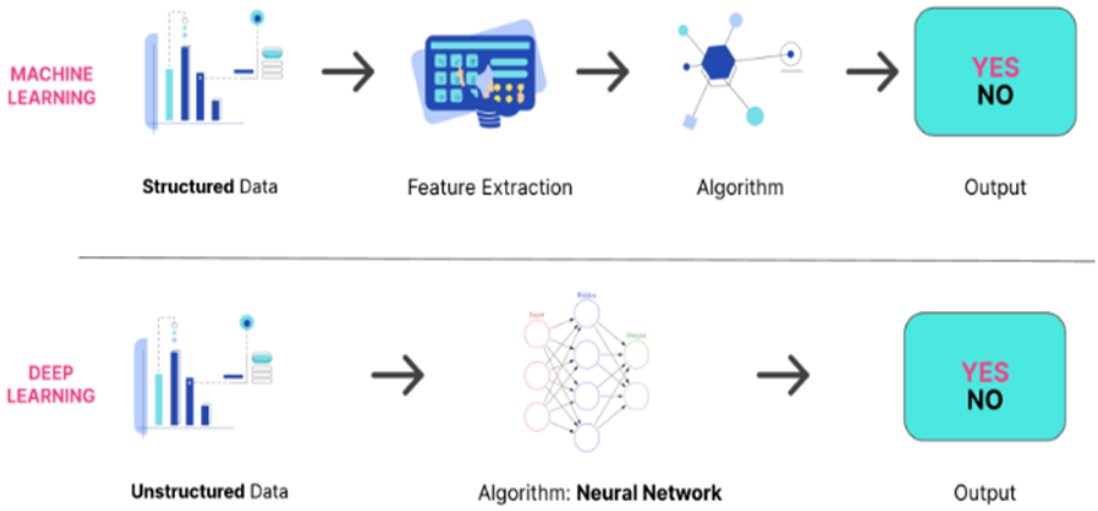


Figure 18: Comparaison entre Machine learning et deep learning

La machine Learning et le deep Learning sont deux sous-domaines de l'intelligence artificielle (IA) qui visent à permettre aux machines d'apprendre à partir de données. Voici une comparaison entre les deux :

3.4.1 Matériel :

Les programmes d'apprentissage automatique sont généralement moins complexes que les algorithmes de deep learning et peuvent souvent être exécutés sur des ordinateurs conventionnels. En revanche, les systèmes de deep learning requièrent des ressources matérielles et informatiques beaucoup plus puissantes.

3.4.2 Intervention humaine :

L'apprentissage automatique implique généralement une supervision humaine continue pour obtenir des résultats, tandis que le deep Learning, bien que plus complexe à configurer initialement, nécessite ensuite une intervention humaine minimale.

3.4.3 Approche :

L'apprentissage automatique utilise généralement des données structurées et des algorithmes classiques tels que la régression linéaire, tandis que le deep Learning repose sur des réseaux neuronaux conçus pour traiter de grandes quantités de données non structurées.

3.4.4 Temps :

Les systèmes d'apprentissage automatique sont souvent rapidement, opérationnels mais peuvent avoir des limites de performance. En revanche, les systèmes d'apprentissage profond exigent plus de temps pour être mis en place mais peuvent générer des résultats plus puissants instantanément.

3.4.5 Applications :

L'apprentissage automatique est couramment appliqué dans divers domaines tels que la messagerie électronique, finance et la médecine. Le deep Learning, quant à lui, propulse des technologies plus avancées et autonomes comme les véhicules autonomes et les robots chirurgicaux.

3.5 Conclusion :

Le deep Learning représente une avancée significative dans le domaine de l'intelligence artificielle, permettant aux machines d'apprendre des représentations de données complexes de manière hiérarchique et automatique.

Grâce à ses capacités à traiter des volumes massifs de données non structurées, le deep Learning a ouvert la voie à des applications innovantes dans divers domaines tels que la vision par ordinateur, le traitement du langage naturel, la reconnaissance vocale, la santé, finance, et bien d'autres encore.

Dans le domaine de la cybersécurité, notamment pour la détection de malwares dans les applications Android, le deep learning joue également un rôle crucial. En utilisant des techniques avancées d'apprentissage automatique, les modèles de deep learning peuvent analyser le comportement des applications et détecter des activités suspectes, contribuant ainsi à renforcer la sécurité des utilisateurs. La mise en œuvre d'un modèle basé sur le deep learning pour la détection des malwares dans les applications Android représente une avancée significative dans la cybersécurité.

CHAPITRE 4 :

IMPLÉMENTATION ET EXPÉRIMENTATION

4.1 Introduction :

À mesure que la diversité des logiciels malveillants augmente, les scanners antivirus traditionnels deviennent insuffisants pour fournir une protection adéquate, laissant ainsi des millions de systèmes vulnérables aux attaques. La protection des systèmes informatiques contre ces menaces est donc une priorité majeure en cybersécurité pour les particuliers et les entreprises, car une seule attaque peut entraîner une compromission des données et des pertes significatives. Par conséquent, l'utilisation de techniques basées sur l'apprentissage automatique s'avère nécessaire.

L'objectif de ce projet est de développer une preuve de concept pour la classification des logiciels malveillants en utilisant des classificateurs robustes. Parmi toutes les caractéristiques disponibles, les plus pertinentes sont déterminées et extraites. L'algorithme offrant la meilleure précision sera capable de distinguer les fichiers malveillants avec un taux d'erreur minimal.

L'évaluation des performances des différentes approches de détection se fait à travers plusieurs mesures telles que la précision, le rappel, le score F1, le taux de détection et le taux de fausses alarmes. Ces évaluations visent à assurer une analyse complète des algorithmes de deep learning appliqués, contribuant ainsi à renforcer la sécurité des systèmes informatiques face aux menaces croissantes des logiciels malveillants.

4.2 Environnement d'exécution :

Pour l'étude de jeux de données, commencer par créer un compte sur kaggle.com et compléter votre profil avec vos informations personnelles et professionnelles. Explorez ensuite la vaste collection de jeux de données disponibles en accédant à la section "Datasets". Utilisez la barre de recherche ou parcourez les catégories pour trouver des jeux de données qui correspondent à vos intérêts.

Une fois un jeu de données sélectionné, vous pouvez soit le télécharger sur votre machine pour une analyse locale, soit utiliser les notebooks intégrés de Kaggle, appelés « kernels ». Les kernels permettent d'exécuter du code Python directement sur la plateforme, facilitant l'exploration et l'analyse des données sans avoir à quitter Kaggle. Commencez par charger les données dans un kernel, puis effectuez une analyse exploratoire pour comprendre sa structure et identifier les tendances et anomalies. Utilisez des bibliothèques de visualisation comme Matplotlib, Seaborn ou Plotly pour créer des graphiques qui aident à interpréter les données.

Kaggle encourage le partage et la collaboration, donc une fois votre analyse terminée, partager votre kernel avec la communauté. Cela permet à d'autres utilisateurs de voir votre travail, de commenter et de collaborer avec vous. Engagez-vous dans les forums de discussion pour poser des questions et partager des idées, et consulter les kernels tutoriels partagés par d'autres utilisateurs pour apprendre de nouvelles techniques.

En raison de la faible efficacité de nos ordinateurs, nous avons utilisé kaggel pour développer notre modèle.

4.2.1 Kaggel :

Kaggle est une plateforme en ligne populaire dédiée à la science des données, à l'apprentissage automatique et à l'analyse de données. Fondée en 2010, Kaggle offre une variété de fonctionnalités et de ressources pour les data scientists, les chercheurs et les passionnés de données.

Nous écrivons notre code en utilisant : le langage de programmation python3 sous kaggel notebook et en utilisant les bibliothèques suivantes :

1_ Os :

Le système d'exploitation est l'un des modules utilitaires de Python standard. Ce module offre une solution portable pour l'utilisation des fonctionnalités qui dépendent du système d'exploitation disponible. Le module [os] comprend de multiples fonctionnalités permettant d'interagir avec le système de fichiers.

2_Pandas :

Pandas est un logiciel développé dans le langage de programmation Python qui permet de manipuler et d'analyser des données. Plus précisément, il offre des méthodes et des structures de données pour gérer des tableaux numériques et des séries chronologiques.

3_ Matplotlib :

Permet de créer des graphiques statiques, des histogrammes, des graphiques en barres, des graphiques en boîte, etc.

4_Seaborn :

Bibliothèque de visualisation de données basée sur Matplotlib, offrant une interface de haut niveau pour dessiner des graphiques statistiques attrayants et informatifs.

5_Numpy :

Il s'agit d'une bibliothèque de calcul scientifique qui supporte des tableaux et des matrices multidimensionnelles de grande taille, avec une vaste gamme de fonctions mathématiques de premier ordre pour manipuler ces tableaux**.**

6_Scikit-learn :

La bibliothèque Scikit-learn en Python offre une variété d'algorithmes d'apprentissage non supervisés et supervisés. Il utilise différentes technologies telles que NumPy, pandas et Matplotlib. Les caractéristiques proposées par scikit-learn comprennent :

— Régression, y compris régression linéaire et logistique.
— Classification, y compris K-voisins les plus proches.
— Clustering, y compris K-Means et K-Means ++.
— Sélection du modèle.
— Prétraitement, y compris la normalisation Min-Max.

7_TensorFlow :

Framework d'apprentissage automatique développé par Google pour la création et le déploiement de modèles d'apprentissage automatique et d'apprentissage profond.

8_ LabelEncoder :

Cet encodeur est inclus dans la bibliothèque SciKit-learn et sert à convertir du texte ou des données catégorielles en données numériques auxquelles le modèle s'attend et avec lesquelles il fonctionne de manière optimale**.**

9_Keras :

Interface haut niveau pour (TensorFlow et Theano) permettant de créer et d'expérimenter rapidement avec des réseaux de neurones.

10_Csv :

Il s'agit d'une bibliothèque qui met en place des classes permettant de lire et d'écrire des données Excel au format CSV.

4.3 Dataset :

Le jeu de données **Android Malware Dataset (CIC-AndMal2017)**, développé par le Canadian Institute for Cybersecurity en 2017, est une ressource précieuse pour l'étude et l'analyse des malwares sur les appareils Android. Ce dataset comprend des informations détaillées sur diverses applications Android, incluant à la fois des applications bénignes et des malwares, classées en différentes familles. Les données sont organisées en plusieurs fichiers CSV, chacun contenant des types spécifiques de données telles que les permissions demandées (**permissions.csv**), les appels API effectués (**api_calls.csv**), les caractéristiques statiques et dynamiques (**features.csv**), les étiquettes de classification (**classes.csv**), les services utilisés (**services.csv**), et les intents déclarés (**intents.csv**). Ces fichiers fournissent une vue complète et détaillée des comportements et des caractéristiques des applications, permettant ainsi une analyse approfondie et la mise en œuvre de techniques d'apprentissage automatique pour la détection des malwares. Le CIC-AndMal2017 est largement utilisé par les chercheurs pour développer et évaluer des modèles de sécurité mobile, contribuant ainsi à l'amélioration des mesures de protection contre les menaces de logiciels malveillants sur les plateformes Android.

CIC-AndMal2017 collecte plus de 10 854 échantillons (4 354 logiciels malveillants et 6 500 bénins) provenant de plusieurs sources ; collecte plus de six mille applications inoffensives du marché « Google play » publiées en 2015, 2016 et 2017 ;

Les malwares de logiciels malveillants dans l'ensemble de données CICAndMal2017 sont classés en Six catégories :

- ✓ Benign.
- ✓ Adware.
- ✓ Ransomware.
- ✓ Scareware.
- ✓ SMS Malware.
- ✓ Malware.

4.4 Préparation des données :

Pour un bon apprentissage en deep learning, la quantité de données est cruciale (plus de données de qualité équivaut à une meilleure précision et de meilleurs résultats).

Dans notre étude, nous devons passer par deux grandes étapes pour résoudre les problèmes de déséquilibre des classes de données et des étiquettes variées :

4.4.1 Réduction des données :

Les données présentés dans la data-set « CIC-AndMal2017 » sont considérablement importantes et requièrent beaucoup d'espaces et de temps. C'est pourquoi nous avons été contraints de réduire ces données.
En conclusion, nous disposons de deux fichiers CSV : un pour l'entraînement et un pour le test.

Label	
BENIGN	**500131**
SCAREWARE	**10117**
ADWARE	**10000**
RANSOMWARE	**10000**
SMSMALWARE	**9664**
MALWARE	**88**

Name : count, dtype : int64

Table 7: Répartition des données de notre Dataset (multi-classification)

4.4.2 Pré-traitements des données :

Cette étape revêt une grande importance dans la construction d'un modèle hautement précis. Le prétraitement de l'ensemble des données est effectué avant leur application au réseau neuronal profond. Parmi les étapes de ce processus figurent :

On remplace chaque étiquette de chaîne de caractères par une valeur numérique :

- **'BENIGN'** est remplacé par **0**
- **'ADWARE'** est remplacé par **1**
- **'RANSOMWARE'** est remplacé par **2**
- **'SCAREWARE'** est remplacé par **3**
- **'SMSMALWARE'** est remplacé par **4**
- **'MALWARE'** est remplacé par **5**

La plupart des modèles de machine learning ne peuvent pas traiter directement les étiquettes sous forme de chaînes de caractères et nécessitent des entrées numériques.
Nous avons éliminé toutes les lignes redondantes qui correspondent aux instances des classes.

Nous avons éliminé aussi toutes les colonnes qui étaient vides, ce qui signifie que leurs valeurs égales à zéro.

Ensuite, nous avons examiné toutes les données contenant des valeurs NAN (NOT A NUMBER) ou INF (INFINITE VALUE), en utilisant la classe de scikit-learn.

Nous avons également éliminé les colonnes ne contenant pas de valeurs numériques, telles que la date, l'adresse IP, l'heure, etc.

Après, nous avons standardisé l'ensemble des données d'entrée. Cette étape vise à améliorer le taux d'apprentissage et à réduire les erreurs.

Enfin, nous avons séparé toutes les données en deux phases : les données d'entraînement et les données de test. La répartition est de 80 % pour l'entraînement et de 20 % pour le test.

Source IP	579999	non-null	Object
Source Port	579999	non-null	float64
Destination Port	579999	non-null	float64
Protocol	579999	non-null	float64
Timestamp	579999	non-null	Object
Flow Duration	579999	non-null	float64
Total Fwd Packets	579999	non-null	float64
Total Backward Packets	579999	non-null	float64
Total Length of Bwd Packets	579999	non-null	float64

Table 8: Répartition des données de notre Dataset (multi-classification)

4.5 Implémentation et Résultats :

Dans le cadre de cette étude, l'objectif est de réduire les taux d'erreur et de fausses alarmes au minimum, tout en maximisant les niveaux de précision possibles. Pour y parvenir, plusieurs tests ont été réalisés afin de déterminer les paramètres optimaux du modèle.

Ces tests ont porté sur des variables structurant le réseau (nombre de neurones, nombre de couches, fonction d'activation, etc.), la taille des lots d'échantillons (Batch Size), le nombre d'itérations, et d'autres aspects similaires.

Une fois le modèle de qualité, affichant un taux d'erreur minimal et une précision maximale, a été mis au point, nous l'avons testé sur un sous-ensemble spécifique. Les résultats de ces tests sont illustrés dans les figures 20.a et 20.b.

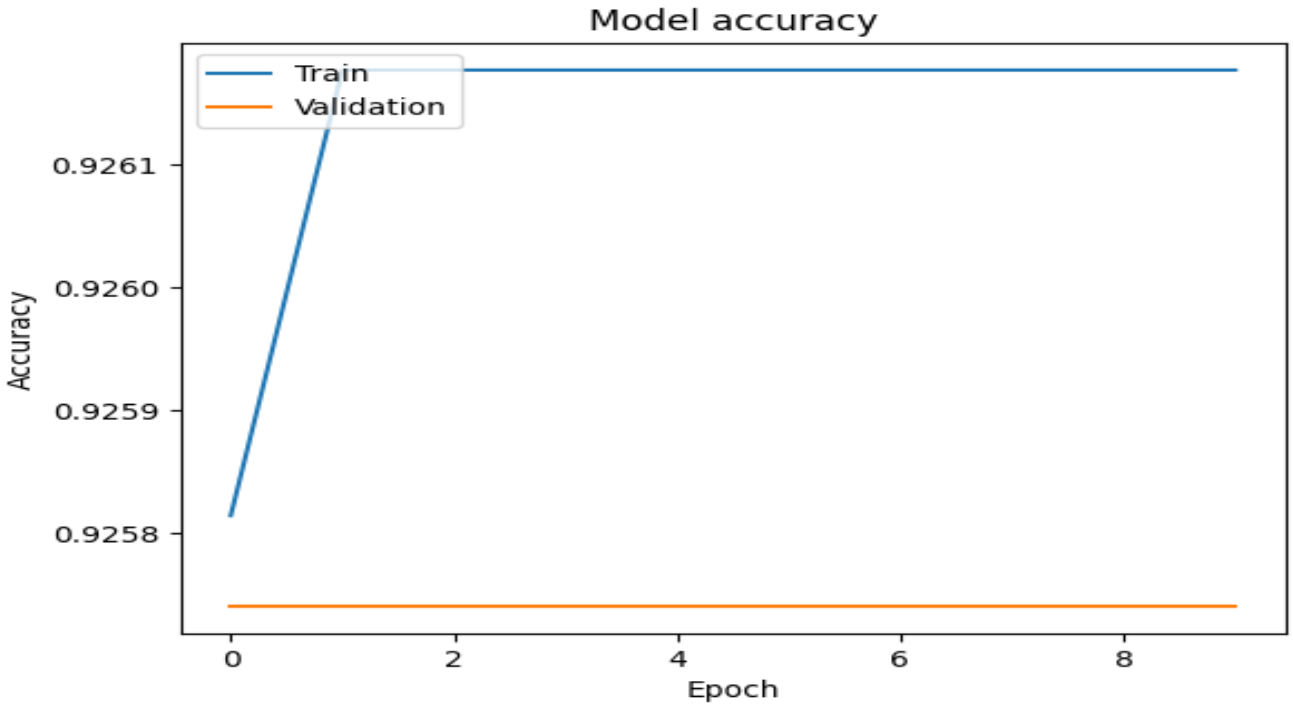


Figure 19-a: Visualisation des performances du modèle-Entraînement et validation Accuracy

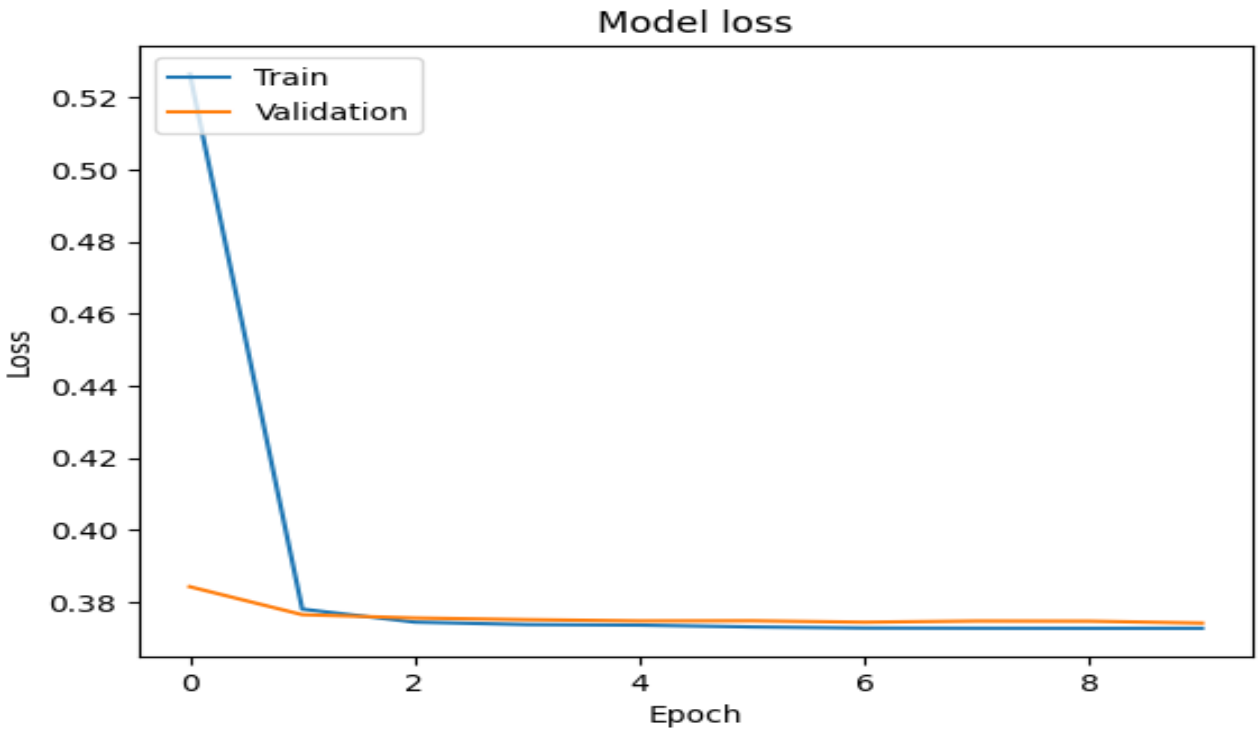
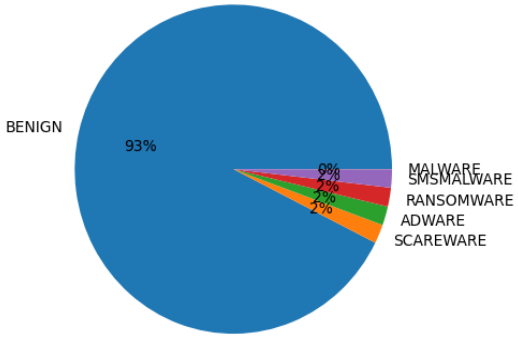


Figure 20-b : Visualisation des performances du modèle- Entraînement et validation Loss

La figure 20-a présente l'évolution de la précision (accuracy) du modèle sur les ensembles d'entraînement et de validation. On observe une amélioration progressive de la précision au cours de l'entraînement, atteignant environ 92.57% pour l'ensemble d'entraînement et pour l'ensemble de validation.

La figure 20-b illustre la perte (loss) du modèle, une mesure de l'erreur commise par celui-ci. On remarque une réduction continue de la perte au fur et à mesure de l'entraînement, passant d'environ 0.38 pour l'ensemble d'entraînement et pour l'ensemble de validation.

Comme illustre la figure 21, l'apprentissage profond s'avère être une approche efficace pour la caractérisation des logiciels malveillants, et son efficacité s'accroît particulièrement lorsque le volume de données d'entraînement augmente.



	precision	recall	f1-score	support
0	0.93	0.22	0.36	9998
1	0.02	0.03	0.02	185
2	0.02	0.16	0.03	208
3	0.02	0.29	0.03	202
4	0.02	0.11	0.04	205
5	0.00	0.00	0.00	2
accuracy			0.21	10800
macro avg	0.17	0.13	0.08	10800
weighted avg	0.86	0.21	0.33	10800

Figure 20: L'évaluation et les résultats du modèle

4.6 Mesures d'évaluation d'un modèle :

❖ Précision (Pr) :

Parmi tous les exemples prédits comme malwares, le pourcentage des malwares identifiés comme des malwares TP est donné par :

$$\text{Précision} = \frac{TP_{malware}}{TP_{malware} * FP_{BENIGN}}$$

❖ Recall (Rc) :

Le taux de malwares identifiés comme TP parmi tous les malwares dans l'ensemble des données est de :

$$\text{Recall} = \frac{TP_{malware}}{TP_{malware} * FN_{malwre}}$$

❖ **F1-score (F1)** :

La moyenne harmonique pondérée de précision et de rappel (Recall), est donné par :

$$\text{F1-Score} = \frac{2 * (Pr * Rc)}{Pr + Rc}$$

❖ **True Positive Rate (TPR) :**

Le nombre de logiciels malveillants identifiés comme malveillants multiplié par le nombre de logiciels malveillants TP dans l'ensemble des données est donné par :

$$\text{TPR} = \frac{\sum TP_{malware}}{\sum malwares}$$

❖ **False Positive Rate (FPR) :**

On peut calculer le nombre de cas bénins incorrectement classés comme des Malwares en divisant le nombre total de cas bénins dans l'ensemble des données par :

$$\text{FPR} = \frac{\sum FP_{BENIGN}}{\sum Total\ BENIGN}$$

	Precision	**Recall**	**F1-score**	**Support**
BENIGN	0.93	0.22	0.36	9998
ADWARE	0.02	0.03	0.02	185
RANSOMWARE	0.02	0.16	0.03	208
SMSMALWARE	0.02	0.29	0.03	202
SCAREWARE	0.02	0.11	0.04	205
MALWARE	0.00	0.00	0.00	2

Tableau 9 : Le rapport de la multi-classification

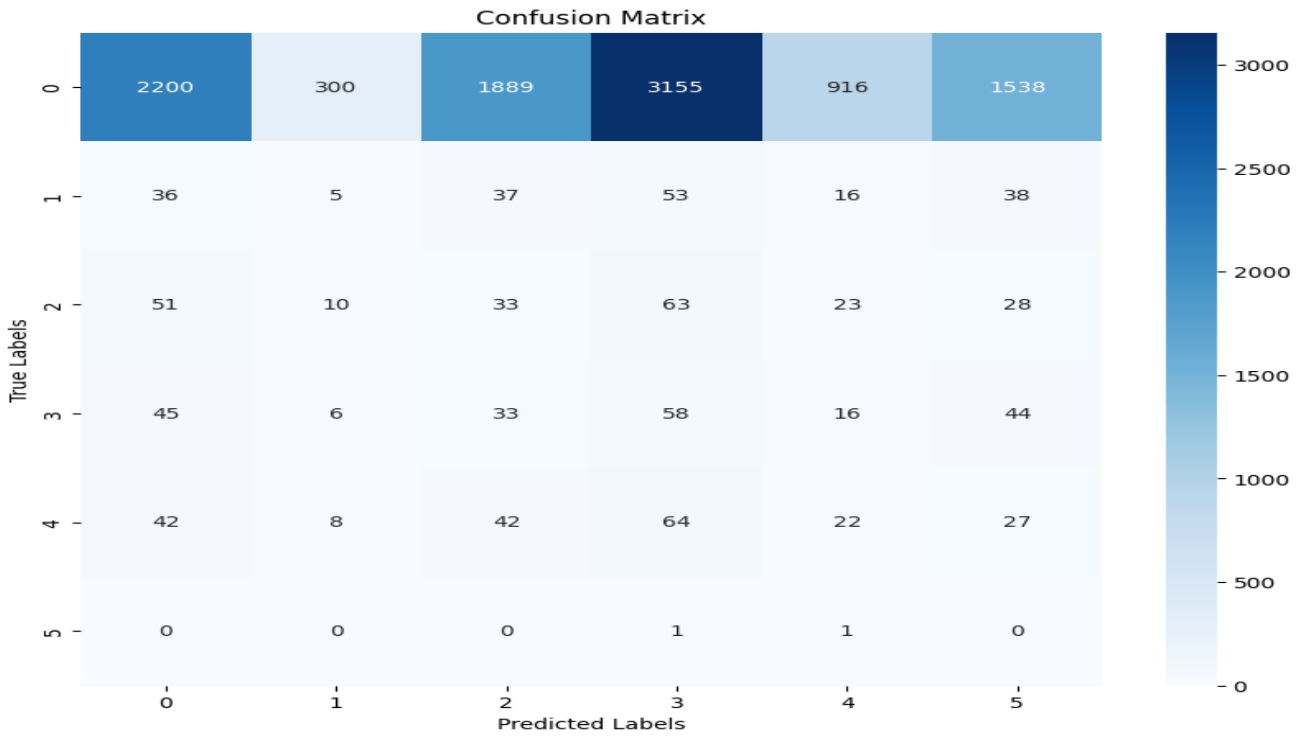


Figure 21 : Matrice de confusion de notre modèle

Une matrice de confusion (figure 22) est un outil utilisé en apprentissage profond et en statistique pour évaluer les performances d'un modèle de classification. Les cellules en haut, représentent les instances correctement classées, tandis que les autres cellules mettent en évidence les erreurs de classification. La matrice est organisée de telle sorte que chaque ligne représente la véritable classe (True Labels) tandis que chaque colonne représente la classe prédite par le modèle (Predicted Labels).

Classe BENIGN (0):

- Le modèle a correctement prédit la classe BENIGN pour 2200 échantillons.
- Il a confondu cette classe avec les classes SCAREWARE (300 échantillons), ADWARE (1889 échantillons), RANSOMWARE (3155 échantillons), SMSMALWARE (916 échantillons) et MALWARE (1538échantillons).
- La classe BENIGN semble être celle qui est la plus souvent confondue avec les autres classes par le modèle.

Classe ADWARE (1):

- Le modèle a correctement prédit la classe ADWARE pour 5 échantillons.
- Il a confondu cette classe avec les classes BENIGN (36 échantillons), RANSOMWARE (37échantillons), SMSMALWARE (53 échantillons) et MALWARE (38 échantillons).
- La classe ADWARE est également sujette à des confusions avec d'autres classes.

Classe RANSOMWARE (2):

- Le modèle a correctement prédit la classe RANSOMWARE pour 33 échantillons.

- Il a confondu cette classe avec les classes BENIGN (51 échantillons), ADWARE (10échantillons), SMSMALWARE (63échantillons) et MALWARE (28 échantillons).
- La classe RANSOMWARE semble également être sujette à des confusions, en particulier avec la classe MALWARE.

Classe SCAREWARE (3):

- Le modèle a correctement prédit la classe SCAREWARE pour 58 échantillons.
- Il a confondu cette classe avec les classes BENIGN (45 échantillons), ADWARE (6 échantillons), RANSOMWARE (56 échantillons), SMSMALWARE (16 échantillons) et MALWARE (44 échantillons).
- La classe SCAREWARE montre également des confusions avec plusieurs autres classes.

Classe SMSMALWARE (4):

- Le modèle a correctement prédit la classe SMSMALWARE pour 22 échantillons.
- Il a confondu cette classe avec les classes BENIGN (42 échantillons), ADWARE (8 échantillons), RANSOMWARE (42 échantillons), SCAREWARE (22 échantillons) et MALWARE (27 échantillons).
- La classe SMSMALWARE présente également des confusions importantes avec les autres classes, en particulier avec la classe MALWARE.

Classe MALWARE (5):

- Le modèle a correctement prédit la classe MALWARE pour 0 échantillon.
- Il a confondu cette classe avec les classes BENIGN (0 échantillon), ADWARE (0 échantillon), SCAREWARE (1 échantillon) et RANSOMWARE (1 échantillon).
- La classe MALWARE est la moins bien prédite, avec un nombre très élevé d'échantillons confondus avec d'autres classes.

*** Comparaison des algorithmes :**

La figure 23 présente un histogramme de comparaison des précisions de divers modèles de machine learning. On y observe huit modèles : la régression logistique, l'arbre de décision, la forêt aléatoire, le gradient boosting, XGBoost, les k-plus proches voisins (K-Nearest Neighbors), le réseau de neurones et le LSTM (Long Short-Term Memory).

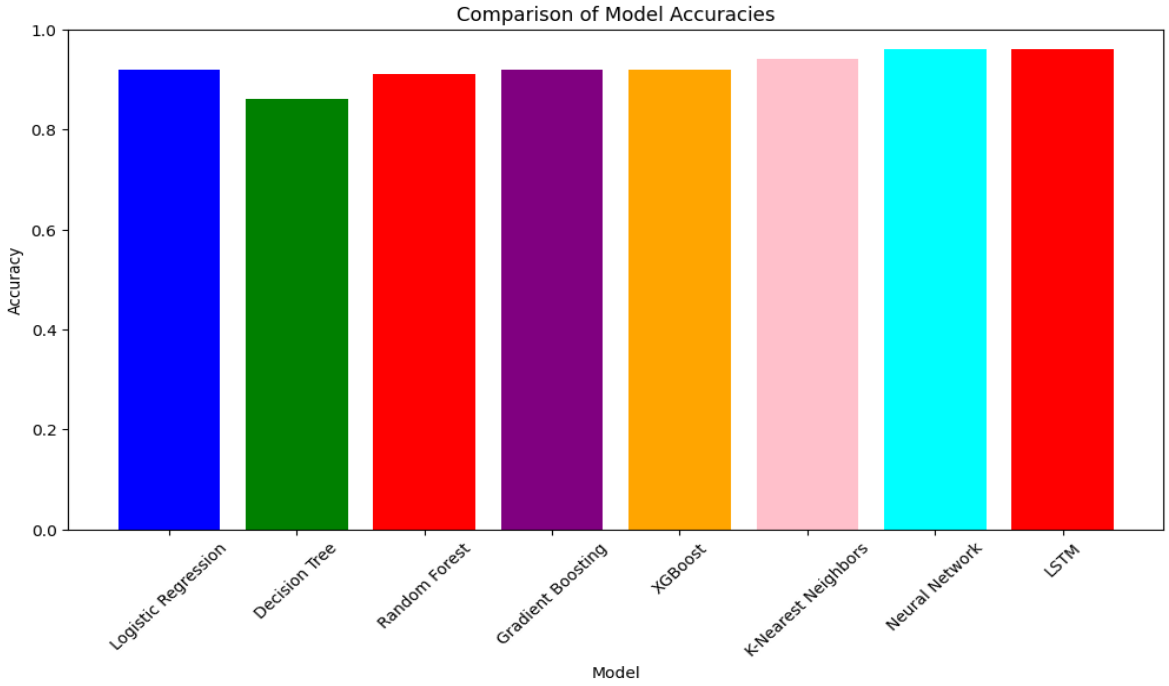


Figure 22: Histogramme de comparaison des algorithmes

Tous les modèles affichent des précisions relativement élevées, proches de 1.0 (ou 100%), ce qui indique une performance globalement bonne pour la tâche spécifique de classification ou de prédiction à laquelle ils sont appliqués. Parmi les modèles, le réseau de neurones et le LSTM semblent avoir une légère avance en termes de précision, suivis de près par la régression logistique et le gradient boosting. Les différences de précision entre les modèles sont minimes, mais ces petites variations peuvent être significatives selon les critères et les besoins de l'application.

Cette figure suggère que, bien que chaque modèle ait ses avantages et inconvénients propres, les réseaux de neurones et les LSTM peuvent offrir une performance légèrement supérieure dans ce contexte particulier. Cependant, il est également essentiel de considérer d'autres facteurs tels que le temps de formation, la complexité du modèle, et les ressources nécessaires pour décider du modèle le plus approprié pour une application donnée.

Le tableau ci-dessous présente une comparaison des performances de différents algorithmes d'apprentissage automatique et profond en termes de précision

des algorithmes	Logistic Regression	Decision Tree	Random Forest	Gradient Boosting	XGBoost	KNN	NN	LSTM
Accuracy	92.4%	86.7%	91.2%	92.4%	92.5%	92.2%	96%	96.2%

Table 10 : Comparaison des algorithmes

En comparaison avec d'autres modèles comme Decision Tree (précision de 0,867), regression logistique (92.4%), LSTM (96.2%), Random Forest Classifier (91.2%) et K-NN (92.2%), les modèles DNN, KNN, et LSTM montrent une supériorité significative en termes de précision.

Ces modèles, surtout DNN et RandomForest, sont particulièrement efficaces pour capturer les variations des données et fournir des prédictions précises avec un taux d'erreurs minimal.

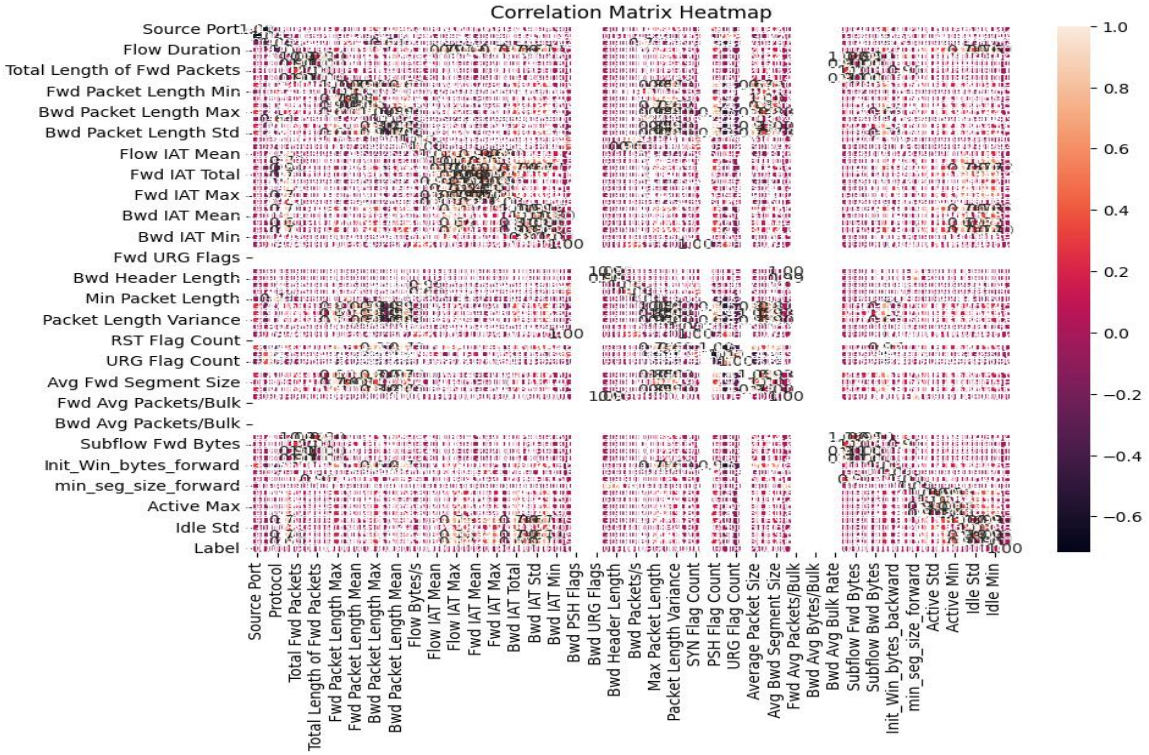


Figure 23: Matrix de correlation « heatmap »

Pour détecter efficacement les malwares, nous choisissons une fonctionnalité qui classe ou étiquette directement les événements réseau comme des attaques. La figure 24 présente est une matrice de corrélation affichée sous forme de heatmap, qui permet de visualiser les relations linéaires entre différentes variables de dataset. Chaque cellule de la matrice montre le coefficient de corrélation entre une paire de variables, avec des valeurs allant de -1 à 1. Les couleurs varient pour représenter ces coefficients : des teintes claires indiquent des corrélations fortes (positives ou négatives), tandis que des teintes foncées représentent des corrélations faibles ou nulles. Cette visualisation est cruciale pour identifier les relations significatives entre les variables, permettant ainsi d'améliorer la sélection des caractéristiques et de détecter d'éventuels problèmes de multicolinéarité. Par exemple, des variables comme "Fwd Packet Length Max" et "Fwd Packet Length Mean" pourraient montrer une corrélation forte, suggérant qu'elles apportent des informations redondantes et qu'une d'elles pourrait être supprimée pour simplifier le modèle sans perte notable d'information. De plus, la compréhension des corrélations aide à interpréter les modèles prédictifs et à identifier les caractéristiques ayant un impact significatif sur la cible, en l'occurrence, les différentes catégories de logiciels malveillants et le trafic bénin. Cette matrice de corrélation est donc un outil essentiel pour l'analyse exploratoire des données, facilitant la compréhension des interactions complexes entre les différentes variables du dataset.

4.7 Conclusion :

Parmi les modèles testés, celui basé sur l'ensemble de données "CIC-AndMal2017 Dataset" s'est distingué par sa performance exceptionnelle, affichant une précision de détection des malwares de 92,57%.

Plus d'un million de données issues du "CIC-AndMal2017 Dataset" ont été organisées en deux fichiers CSV : "Train.csv" (80%) et "Test.csv" (20%). Avant d'entraîner le modèle, un prétraitement minutieux a été effectué, supprimant les colonnes vides, les valeurs manquantes (NAN et INF) et les colonnes de type objet.

Les résultats obtenus sont très encourageants, avec une précision de détection de 92,57% et une perte de 0,38%, démontrant le potentiel prometteur de ce modèle pour la détection des malwares.

Conclusion générale :

Ce mémoire a exploré en profondeur l'application des réseaux de neurones profonds pour la détection des applications Android malveillantes, un domaine crucial face à la prolifération des menaces de sécurité mobile. À travers une revue exhaustive de la littérature et une série d'expérimentations rigoureuses, nous avons démontré que le Deep Learning offre des capacités avancées pour identifier des malwares qui échappent aux méthodes de détection traditionnelles.

Les résultats obtenus confirment que les réseaux de neurones profonds peuvent analyser des quantités massives de données et reconnaître des schémas complexes et subtils associés aux comportements malveillants. En exploitant des architectures telles que les CNN (Convolutional Neural Networks) et les RNN (Recurrent Neural Networks), nous avons pu développer des modèles capables de différencier efficacement les applications malveillantes avec une précision et une robustesse accrues.

L'approche basée sur le Deep Learning surpasse les méthodes traditionnelles en termes de détection et de capacité à généraliser face à des nouvelles menaces. Les modèles de Deep Learning peuvent s'adapter et évoluer en permanence, répondant ainsi à l'évolution rapide et constante des techniques de malwares. Cette adaptabilité est essentielle dans le contexte dynamique des menaces et de sécurité mobile.

Cependant, malgré les avancées significatives, plusieurs défis subsistent. La complexité et les ressources computationnelles nécessaires pour entraîner et déployer des modèles de Deep Learning restent des obstacles majeurs. De plus, la collecte et le prétraitement des données, étape cruciale pour le succès des modèles, nécessitent des efforts continus pour garantir des ensembles de données diversifiés et représentatifs.

En conclusion, l'intégration des réseaux de neurones profonds dans les systèmes de sécurité mobile représente une avancée prometteuse pour la détection des malwares Android. Cette technologie offre une solution puissante et évolutive, capable de renforcer la protection des utilisateurs contre les menaces sophistiquées. Pour l'avenir, il est essentiel de continuer à améliorer les techniques de Deep Learning, d'optimiser les modèles pour une utilisation efficace sur les appareils mobiles, et de maintenir une vigilance constante face à l'évolution des cybermenaces.

BIBLIOGRAPHIE

[1] Statista, “Global market share held by smartphone operating systems from 2009 to 2016,” 2018. [Online]. Available : https://www.statista.com/statistics/263453/global-market-share-held-by-smartphone-operating-systems/

[2] Android, “Dashboards,” Janvier 2018. [Online]. Available: https://developer.android.com/about/dashboards/index.html

[3] C. Lueg, “8,400 new android malware samples every day,” Avril 2017. [Online]. Available: https://www.gdatasoftware.com/blog/2017/04/29712-8-400-new-android-malware-samples-every-day.

[4] Wang, Z., Cai, J., Cheng, S., and Li, W. (2016). Droiddeeplearner: Identifying android malware using deep learning. In 2016 IEEE 37th Sarnoff Symposium, pages 160–165. IEEE

[5] S.K. Justin. A machine learning approach to android malware detection. European Intelligence and Security Informatics Conference IEEE, 2012.

[6] https://www.koreascience.or.kr/article/JAK0202034465346164.pdf, avril,22,2022.

[7] « Windows market share drop to 15 years low [archive] », TG Daily (30 septembre 2013).

[8] Sebastian Anthony, « Microsoft's share of the consumer market has dropped from 95% to 20% in 8 years [archive] », ExtremeTech (2013)

[9] Frandroid En mai 2023, Android 11 R est la version la plus utilisée de l’OS mobile de Google.

[10] Commercialization Challenges of Mobile Software Development in an Exponentially Fragmenting Ecosystem 2017, Stephen King, 6 déc. 2016,

[11] Quelle version d’Android est installée sur votre smartphone? Omar Belkaab, Frandroid avec Android Studio.

[12] S.K. Justin. A machine learning approach to android malware detection. European Intelligence and Security Informatics Conference IEEE, 2012.

[13] Symantec Corporation. (s.d.). "Malicious Android Applications and Their Exploitation." https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/malicious_androi d_apps_and_their_exploitation.pdf

[14] Andriatsimandefitra Ratsisahanana, R. (2014). Caractérisation et détection de malware Android basées sur les flux d'information (Doctoral dissertation, Supélec.

[15] Malware. Available online: https://www.av-test.org/en/statistics/malware/ (accessed on 16 October 2022)

[16] Naway, A. et Y. Li. 2018. « A review on the use of deep learning in android malware detection », arXiv preprint arXiv :1812.10360

[17] Alzahrani, N. et D. Alghazzawi. 2019. « A review on android ransomware detection using deep learning techniques ». In Proceedings of the 11th International Conference on Management of Digital EcoSystems, p. 330–335.

[18] Sweetlin, R. et A.S.Radhamani. 2016. « Survey on detection of malware in android », International Journal of Latest Trends in Engineering and Technology, vol. 11, p. 44–47.

[19] Yan, P. et Z. Yan. 2018. « A survey on dynamic mobile malware detection », Software Quality Journal, vol. 26, no. 3, p. 891–919.

[20] Arshad, S., M. A. Shah, A. Khan, et M. Ahmed. 2016. « Android malware detection & protection: a survey », International Journal of Advanced Computer Science and Applications, vol. 7, no. 2, p. 463–475.

[21] Z. Yuan, Y. Lu and Y. Xue, "Droiddetector: android malware characterization and detection using deep learning," in Tsinghua Science and Technology, vol. 21, no. 1, pp. 114-123, Feb.2016. doi: 10.1109/TST.2016.7399288

[22] Feng, Y., Anand, S., Dillig, I., and Aiken, A. (2014). Apposcopy: Semantics-based detection of android malware through static analysis. In Proceedings of the 22nd ACM SIGSOFT- international symposium on foundations of software engineering, pages 576–587.

[23] KP, S., & Alazab, M. (2020). A Comprehensive Tutorial and Survey of Applications of Deep Learning for Cyber Security

[24] Wong, M. Y. and Lie, D. (2016). Intellidroid: A targeted input generator for the dynamic analysis of android malware. In NDSS, volume 16, pages 21–24

[25] Bayer, U., Comparetti, P. M., Hlauschek, C., Kruegel, C., and Kirda, E. (2009). Scalable, behavior-based malware clustering. In NDSS, volume 9, pages 8–11. Citeseer.

[26] Bhatia, T. and Kaushal, R. (2017). Malware detection in android based on dynamic analysis. In 2017 International Conference on Cyber Security And Protection Of Digital Services (Cyber Security), pages 1–6. IEEE

[27] Cédric BERTRAND. Les malwares sous android, Panorama des malwares sous Android, Analyser une application Android, juin 2012

[28] Ferlin, A., Bon, P., Wiels, V., & Collart-Dutilleul, S. (2015, June). Vérification parallélisée de propriétés temporelles sur des traces d'exécution, par analyse dynamique formelle. In Approches Formelles dans l'Assistance au Développement Logiciel.

[29] Wong, M. Y. and Lie, D. (2016). Intellidroid: A targeted input generator for the dynamic analysis of android malware. In NDSS, volume 16, pages 21–24

[30] Bhatia, T. and Kaushal, R. (2017). Malware detection in android based on dynamic analysis.
In 2017 International Conference on Cyber Security And Protection Of Digital Services (Cyber Security), pages 1–6. IEEE

[31] Vidas, T., Tan, J., Nahata, J., Tan, C. L., Christin, N., and Tague, P. (2014). A5: Automated analysis of adversarial android applications. In Proceedings of the 4th ACM Workshop on Security and Privacy in Smartphones & Mobile Devices, pages 39–50.

[32] Auray, N. (2015). L'invisible et le clandestin. Une ethnographie du virus informatique Storm. Terrain. Anthropologie & sciences humaines, (64), page 32-49

[33] Erbschloe, M. (2004). Trojans, worms, and spyware: a computer security professional's guide to malicious code. Butterworth-Heinemann

[34] Chiang, K., & Lloyd, L. (2007). A Case Study of the Rustock Rootkit and Spam Bot. HotBots.

[35] TRUDEL, P., ABRAN, F., & DUPUIS, G. (2007). Analyse du cadre réglementaire québécois et étranger à l'égard du pourriel, de l'hameçonnage et des logiciels espions

[36] A Flow Based Approach to Detect Advanced Persistent Threats in Communication Systems - Scientific Figure on ResearchGate. Available from:
https://www.researchgate.net/figure/Malware-types-and-their-properties_tbl1_327813496

[37] https://www.koreascience.or.kr/article/JAK0202034465346164.pdf, 2022.

[38] Franklin Tchakounte. A Malware Detection System for Android. Journal of Chemical Information and Modeling, 53(9) :1689–1699, 2015.

[39] Meilleures applications de pare-feu android pour une meilleure sécurité internet sur les téléphones. https://www.digitalprivatevault.com/blogs/best-android-firewall-apps-for-better-internet-security-on-phones, 2019.

[40] Martin Borek, Gideon Creech, and Unsw Canberra. Intrusion Detection System for Android:
Linux kernel system calls analysis. 2017

[41] What is malware analysis? defining and outlining the process of malware analysis.
https://enterprise.comodo.com/blog/what-is-malware-analysis/, 2018.

[42] Arnaud Bodin & François Recher « Deep Math Mathématiques (Simples)Des Réseaux De Neurones ». Février 2023, p 275

[43] Aurélien Géron. Machine Learning avec Scikit-Learn.page 256, 2017.

[44] Jordan, Michael I., and Tom M. Mitchell. "Machine learning: Trends, perspectives, and prospects." Science 349.6245 (2015): 255-260.

[45] Mahesh, Batta. "Machine learning algorithms-a review." International Journal of Science and Research (IJSR). [Internet] 9.1 (2020): 381-386.

[46] Pedro Domingos. A few useful things to know about machine learning. Communications of the ACM, 55(10) :78–87, 2012.

[47] Cunningham, P., Cord, M., & Delany, S. J. (2008). Supervised learning. In Machine learning techniques for multimedia: case studies on organization and retrieval (pp. 21-49). Berlin, Heidelberg: Springer Berlin Heidelberg.

[48] Alzubi, Jafar, Anand Nayyar, and Akshi Kumar. "Machine learning from theory to algorithms: an overview." Journal of physics: conference series. Vol. 1142. IOP Publishing, 2018.
[49] Liu, Zelin, et al. "Application of machine-learning methods in forest ecology: recent progress and future challenges." Environmental Reviews 26.4 (2018): 339-350.

[50] Nedyalkova, Miroslava, Sergio Madurga, and Vasil Simeonov. "Combinatorial k-means clustering as a machine learning tool applied to diabetes mellitus type 2." International journal of environmental research and public health 18.4 (2021): 1919.

[51] Han J, Pei J, Kamber M. Data mining: concepts and techniques. Amsterdam: Elsevier; 2011.

[52] Yousef, Malik, et al. "Naïve Bayes for microRNA target predictions—machine learning for microRNA targets." Bioinformatics 23.22 (2007): 2987-2992.

[53] Hongyu Liu and Bo Lang. Machine learning and deep learning methods for intrusion detection systems: A survey. Applied Sciences (Switzerland), 9(20), 2019.

[54] Zhiying Hao MATEC Web of Conferences 277(C):02035 "Deep learning review and discussion of its future development", 2018, p 1.2

[55] Bastien L."Reinforcement Learning:qu’est-ce que l’apprentissage par renforcement?",30 juin 2021

[56] Bhiksha Raj, Dejing Dou, Feng Luo, M. Arif Wani "Deep Learning Applications, Volume 3",12 novembre 2021, p 193

[57] John D. Kelleher "Deep Learning",10 septembre 2019, p 170

[58] Frank Millstein "Deep Learning with Keras Beginner’s Guide To Deep Learning With Keras",7 juillet 2020, p 118.119

[59] Ahmad, J.; Farman, H.; Jan, Z. Deep learning methods and applications. In Deep Learning: Convergence to Big Data Analytics; SpringerBriefs in Computer Science; Springer: Singapore, 2019; pp. 31–42.

[60] Mohammad Mustafa Taye"Understanding of Machine Learning with Deep Learning:Architectures, Workflow, Applcations and Future Directions",25 April 2023,p 17

[61] Vetrekar, N.; Raja, K.B.; Ramachandra, R.; Gad, R.; Busch, C.; Ramachandra, R.; Gad, R.; Busch, C. Multi-spectral imaging for robust ocular biometrics. In Proceedings of the 2018 International Conference on Biometrics, ICB 2018, Gold Coast, QLD,Australia, 20–23 February 2018; pp. 195–201.

[62] Han, J.; Zhang, Z.; Mascolo, C.; Andre, E.; Tao, J.; Zhao, Z.; Schuller, B.W. Deep Learning for Mobile Mental Health: Challenges and recent advances. IEEE Signal Process. Mag. 2021, 38, pp. 96–105.

[63] Zou, J.; Zhang, Q. UbiEi-Edge: Human Big Data Decoding Using Deep Learning on Mobile Edge. In Proceedings of the roceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS, Guadalajara, Mexico, 1–5 November 2021; pp. 1861–1864.

[64] Guo Y, Liu Y, Oerlemans A, et al. Deep learning for visual understanding: A review[J]. Neurocomputing, 2016, 187(C):pp 27-48.

RÉSUMÉ

Ces dernières années, Android a gagné en popularité en tant que système d'exploitation mobile le plus populaire à travers le monde, ce qui en fait une cible importante pour les cybercriminels. Ainsi, la question de la sécurité de cette plateforme revêt une grande importance. En malgré tout ce que ce système propose en termes de mécanismes solides pour protéger sa plateforme, il présente cependant plusieurs vulnérabilités et failles.

La prolifération des applications Android a conduit à une augmentation des malwares, posant des risques significatifs pour la sécurité des utilisateurs et de leurs données. Les techniques traditionnelles de détection, basées sur des signatures et des heuristiques, peinent à suivre l'évolution rapide et la complexité croissante des malwares. Les réseaux de neurones profonds (Deep learning) offrent une solution prometteuse pour améliorer la détection des applications malveillantes grace à leur capacité à analyser des schémas complexes et variés.

Mots clés : Malware, sécurité, Application Android, Deep Learning, détection.

ABSTRACT

Recent years, Android has gained popularity as the most popular mobile operating system worldwide, making it an important target for cybercriminals. Thus, the issue of the security of this platform is of great importance. En despite everything this system offers in terms of robust mechanisms to protect its platform, it has several vulnerabilities and weaknesses.

The proliferation of Android apps has led to an increase in malware, posing significant risks to the security of users and their data. Traditional signature and heuristic-based detection techniques struggle to keep up with the rapid evolution and increasing complexity of malware. Deep Learning provides a promising solution to improve detection of malicious applications through their ability to analyse complex and varied patterns.

Keywords: Malware, Security, Android Application, Deep Learning, Detection.

ملخص

في السنوات الأخيرة، اكتسبت أندرويد شعبية كإستراتيجية التشغيل المحمولة الأكثر شعبية في جميع أنحاء العالم، مما يجعلها هدفاً هاماً للشركاء المحليين .فإن مسألة أمن هذه المنصة مهمة للغايةعلى الرغم من كل ما يوفره هذا النظام من آليات قوية لحماية منصة الشركة، إلا أنه يحتوي على العديد من الضعف.

و قد أدى انتشار تطبيقات اندرويد إلى زيادة عدد البرامج الضارة،مما يشكل مخاطر كبيرة على أمن المستخدمين والبيانات .تستطيع التقنيات التقليدية التي تستند إلى تسجيلات وتقارير الذكاء الاصطناعي أن تتبع التطور السريع والتكلفة المتزايدة للبرمجيات الخبيثة .توفر الشبكات العصبية العميقة حلًا ملموسًا لتعزيز اكتشاف التطبيقات الخبيثة من خلال القدرة على تحليل النماذج المعقدة المختلفة .

الكلمات المفتاحية : البرامج الضارة للأمان، تطبيق أندرويد، التعلم العميق، الكشف