

THÈSE

en vue de l'obtention du: **DOCTORAT**

Structure de Recherche : Intelligent Processing and Security of Systems (I.P.S.S)

Discipline : Informatique

Spécialité : Intelligence artificielle et Cybersécurité

Présentée et Soutenue le: 30/06/2025

par :

EL ASRY CHADIA

Détection Avancée d'Intrusions par Machine Learning: Synergie entre Apprentissage Séquentiel, Sémantique et Représentation en Graphe

Devant le JURY :

Fouzia OMARY	PES	Université Mohammed V, Faculté des sciences, Rabat	Présidente
Mouad BEN MAMOUN	PES	Université Mohammed V, Faculté des sciences, Rabat	Rapporteur/Examineur
Abdelkrim HAQIQ	PES	Université Hassan 1 ^{er} , Faculté des Sciences et Techniques, Settat	Rapporteur/Examineur
Mahmoud NASSAR	PES	Université Mohammed V, Ecole Nationale Supérieure d'Informatique et d'Analyse des Systèmes, Rabat	Rapporteur/Examineur
Salima CHANTIT	MCH	Université Hassan II de Casablanca, Faculté des Sciences et Techniques, Mohammedia	Examinatrice
Samira DOUZI	MCH	Université Mohammed V, Faculté de Médecine et de Pharmacie, Rabat	Co-Directrice de thèse
Bouabid EL OUAHIDI	PES	Université Mohammed V, Faculté des sciences, Rabat	Directeur de thèse

Année Universitaire : 2024 - 25

Dédicace

À la mémoire de mes parents, de ma grand-mère et de mon grand-père, qui ont quitté ce monde mais jamais mon cœur. Qu'Allah leur accorde sa miséricorde infinie. Leur amour, leurs prières silencieuses et leur sagesse continuent de guider mes pas dans l'ombre comme dans la lumière. Ce travail est aussi le fruit de leurs sacrifices et de leurs valeurs.

À mon mari, mon roc dans la tempête et mon compagnon de chaque instant. Ta présence, ta patience, ton soutien sans faille et ton amour constant ont été les fondations invisibles mais solides de cette thèse. Ce que j'ai accompli, je l'ai fait avec toi, grâce à toi.

À mes filles, Jannat, Jinane et Jihane, mes étoiles, mes raisons, mes horizons. Vous êtes l'élan qui m'a portée quand la fatigue menaçait, et la lumière qui m'a guidée quand le doute s'installait. Que ces pages vous apprennent que le savoir est un acte d'amour et un acte de foi. Rêvez haut, travaillez fort, et ne craignez jamais de viser l'impossible.

À ma sœur et à mon frère, piliers de mon existence, merci pour votre amour indéfectible, votre écoute et votre force tranquille. Vous avez été un refuge, un repère. Ce parcours vous doit plus que vous ne l'imaginez.

À Hajar, ma sœur d'âme, ma confidente, mon pilier. Nos vies se sont liées par un lien du cœur aussi fort que les liens du sang. Ton amour m'a portée, ta fidélité m'a guidée, et ta tendresse a bercé chaque étape de cette aventure.

À ma famille élargie et à mes amis sincères, dont la chaleur et les encouragements m'ont enveloppée dans les moments de doute comme dans les moments de joie. Votre présence a été un phare. Cette réussite est aussi la vôtre.

Enfin, à tous ceux et celles qui croient en la puissance du savoir, en la noblesse de l'effort, en la beauté des idées partagées. Que cette thèse, modeste dans sa forme mais grande par l'intention, contribue à nourrir cette quête infinie. Que jamais ne s'éteigne en nous la soif d'apprendre, de comprendre, de transmettre.

Remerciements

La réalisation de ce travail n'aurait pu aboutir sans le soutien précieux de nombreuses personnes, dont les contributions, tant scientifiques qu'humaines, ont été déterminantes. Je leur exprime ma profonde gratitude pour l'influence positive qu'elles ont exercée tout au long de ce parcours de recherche.

Je voudrais tout d'abord exprimer ma plus sincère gratitude envers mon directeur de thèse, Monsieur **Bouabid EL OUAHIDI**, Professeur d'Enseignement Supérieur à la Faculté des Sciences de Rabat, pour avoir accepté de diriger cette thèse. Je lui suis particulièrement reconnaissante pour la confiance qu'il m'a témoignée, la qualité de son encadrement, la pertinence de ses conseils et le temps qu'il a généreusement consacré à ce travail. Son soutien constant, son exigence scientifique et ses encouragements ont été déterminants dans l'aboutissement de cette recherche. Travailler sous sa direction a été pour moi un véritable privilège.

Je tiens à remercier tout particulièrement ma co-directrice de thèse, Madame **Samira DOUZI**, Professeure Habilitée à la Faculté de Médecine et de Pharmacie de Rabat, pour son encadrement attentif et son engagement sans faille. Son soutien constant, ses conseils avisés et son enthousiasme pour la recherche scientifique ont grandement contribué à la réussite de ce travail. Cette thèse restera dans ma mémoire comme un moment de stimulation scientifique et de bonne humeur grâce à elle.

Ma profonde gratitude s'adresse à la Présidente du jury, Madame **Fouzia OMARY**, Professeure d'Enseignement Supérieur à la Faculté des Sciences de Rabat, pour l'honneur qu'elle m'a fait en acceptant de présider ce jury. Je la remercie vivement pour sa confiance, ses encouragements et la bienveillance qu'elle m'a témoignée tout au long de cette étape.

Mes sincères remerciements à Monsieur **Mouad BEN MAMOUN**, Professeure d'Enseignement Supérieur à la Faculté des Sciences de Rabat, pour le temps engagé dans l'évaluation de mon travail, pour ses lectures attentives et ses précieux compléments. Vous m'avez fait honneur en faisant partie de mon jury.

Mes sincères remerciements vont à Monsieur **Abdelkrim HAQIQ**, Professeur d'Enseignement Supérieur à la Faculté des Sciences et Techniques de Settat. Son examen approfondi et ses commentaires réfléchis m'ont apporté de nouvelles perspectives et ont joué un rôle déterminant dans l'affinement de ma recherche.

Je tiens à exprimer ma profonde gratitude à Monsieur **Mahmoud NASSAR**, Professeur d'Enseignement Supérieur à l'École Nationale Supérieure d'Informatique et d'Analyse des Systèmes à Rabat. Ses évaluations éclairées et ses retours constructifs sur le manuscrit de ma thèse ont considérablement contribué à l'amélioration de la qualité de mon travail.

Je remercie également Madame **Salima CHANTIT**, Professeure Habilitée à la Faculté des Sciences et Techniques de Mohammedia, pour son implication, ses nombreux conseils et remarques, ainsi que pour les compléments éclairés qu'elle a bien voulu m'apporter. Vous m'avez fait honneur en faisant partie de mon jury.

Enfin, j'exprime également toute ma gratitude à mes collègues de la structure de recherche **Intelligent Processing and Security of Systems (I.P.S.S)** pour leur amitié et leur soutien, et pour avoir créé un environnement de travail cordial.

Résumé

Les systèmes de détection d'intrusion (IDS) occupent une place centrale dans la cybersécurité moderne. Leur mission consiste à surveiller et analyser les événements d'un système informatique ou d'un réseau afin d'identifier toute activité malveillante ou non autorisée. Bien que renforcés par l'apprentissage automatique, ils font face à des défis majeurs notamment la contextualisation des attaques, la détection des attaques rares, le traitement en temps réel et la réduction des faux positifs et faux négatifs.

Dans cette thèse, nous développons plusieurs approches innovantes basées sur l'apprentissage automatique pour renforcer la détection des intrusions réseau. Nous commençons par exploiter les séquences d'événements à l'aide d'un réseau de neurones récurrent de type LSTM, afin de modéliser les comportements malveillants. Pour améliorer la qualité des données en entrée, l'algorithme SHAP est utilisé pour identifier les caractéristiques les plus pertinentes, permettant ainsi de réduire le bruit, d'optimiser l'apprentissage et d'accroître la capacité du modèle à détecter les anomalies.

Dans une seconde contribution, nous adaptons le modèle ParagraphVector–Distributed Memory (PV-DM) pour générer des représentations denses (embeddings) résumant les séquences d'événements. Les dimensions d'embedding les plus pertinentes sont sélectionnées à l'aide de SHAP, puis utilisées pour entraîner un classifieur XGBoost. Cette combinaison assure un encodage contextuel riche, tout en maîtrisant la complexité du modèle et en assurant une classification efficace.

Afin de traiter la détection des classes rares sans recourir aux méthodes classiques de rééchantillonnage, nous proposons une approche originale reposant sur la subdivision des données selon la fréquence des attaques. Nous identifions ensuite les caractéristiques propres à chaque type d'attaque grâce à l'algorithme ShapRFECV, avant d'entraîner XGBoost sur ces sous-ensembles optimisés. Cette méthode améliore la précision de détection tout en conservant l'intégrité des données.

Enfin, nous introduisons une approche basée sur les graphes pour exploiter les relations structurelles entre événements réseau. Les données sont d'abord regroupées en clusters via l'algorithme K-Means, puis connectées à leurs plus proches voisins par KNN pour former les arêtes d'un graphe. Un réseau de neurones convolutionnel pour graphes (GCN) est ensuite entraîné pour classifier les événements en comportements normaux ou intrusifs, en exploitant la topologie du réseau et les corrélations entre entités.

Ces contributions forment une chaîne cohérente, allant de l'analyse temporelle à la représentation topologique, aboutissant à des IDS plus robustes, intelligents et adaptatifs face à l'évolution des menaces.

Mots-clés : Intelligence artificielle, Système de détection d'intrusion, Deep Learning , PV-DM, sélection de caractéristiques, déséquilibre des classes, Graphes.

Abstract

Intrusion Detection Systems (IDS) play a central role in modern cybersecurity. Their mission is to monitor and analyze the events of a computer system or network to identify any malicious or unauthorized activity. Increasingly, these systems rely on machine learning algorithms capable of processing data flows in real time. However, several challenges remain, including attack contextualization, the detection of rare threats, real-time processing, and the reduction of false positives and false negatives.

In this thesis, we develop several innovative machine learning-based approaches to enhance network intrusion detection. We begin by modeling sequences of network events using a Long Short-Term Memory (LSTM) recurrent neural network to capture malicious behavior. To improve the quality of input data, the SHAP algorithm is used to identify the most relevant features, helping to reduce noise, optimize learning, and enhance anomaly detection.

In a second contribution, we adapt the ParagraphVector–Distributed Memory (PV-DM) model to generate dense vector representations (embeddings) of event sequences. The most significant embedding dimensions are selected using SHAP and used to train an XGBoost classifier. This combination provides rich contextual encoding while controlling model complexity and ensuring efficient classification.

To address the detection of rare classes without relying on traditional resampling techniques, we propose an original approach based on data subdivision according to attack frequency. The specific features of each attack type are identified using the ShapRFECV algorithm, and XGBoost is then trained on these optimized subsets. This method improves detection accuracy while preserving data integrity.

Finally, we introduce a graph-based approach to capture structural relationships between network events. Data is first clustered using the K-Means algorithm, then connected to their nearest neighbors via KNN to form graph edges. A Graph Convolutional Network (GCN) is trained to classify events as normal or intrusive by leveraging the network topology and inter-entity correlations.

These contributions form a coherent and complementary processing pipeline, ranging from temporal analysis to topological representation, ultimately leading to more robust, intelligent, and adaptive IDS capable of keeping pace with evolving threats.

Keywords : Artificial Intelligence, Intrusion Detection System, Deep Learning, PV-DM, Feature Selection, Class Imbalance, Graphs.

Table des matières

Dédicace	i
Remerciements	ii
Résumé	iii
Abstract	iv
Table des matières	v
Table des figures	vi
Liste des tableaux	ix
Introduction générale	1
1 Contexte	1
2 Types d'attaques	3
3 Les techniques de sécurité	4
4 Système de détection d'intrusion (IDS)	6
4.1 Définition	6
4.2 Classification des IDS en fonction de la source de données	6
4.3 Classification des IDS selon les techniques de détection	7
5 Motivation	8
6 Objectif	9
7 Contributions	9
1 Utilisation des données historiques dans la détection des intrusions	15
1 Introduction	16
2 Concepts de base	17
2.1 Classification en cybersécurité	17
2.2 Les réseaux de neurones	18
2.3 Les réseaux de neurones artificiels	18
2.4 Les Réseaux de Neurones Récurrents	20
2.5 Les Réseaux LSTM	21
2.6 Les réseaux de neurones récurrents LSTM Bidirectionnelle	23
2.7 L'ingénierie des features	23
3 Approche proposée	26
3.1 Description du dataset	27
3.2 Pré-traitement des données	28
3.3 La sélection des caractéristiques	29
3.4 Modèles proposés	32
4 Résultats et discussions	35
5 Conclusion	37
2 Apprentissage Contextuel et Sémantique des attaques	40
1 Introduction	41
2 Travaux connexes	42
3 Concepts de base	46
3.1 Modèle shallow learning	46
3.2 Word Embedding	46

3.3	Word2vec	47
3.4	Apprentissage Superficiel : Paragraph Vector-Distributed Memory	47
3.5	Information Mutuelle (MI)	49
3.6	Approche proposée	50
3.7	Résultats expérimentaux et discussion	53
4	Conclusion	66
3	Une approche efficace pour la détection des attaques rares	69
1	Introduction	70
2	Travaux connexes	71
3	Le problème du déséquilibre des classes	75
3.1	Définition	75
3.2	Problème lié au déséquilibre de classe	75
3.3	Approches proposées pour les données déséquilibrées	76
3.4	Concepts de base	80
4	Approche proposée	81
4.1	Dataset utilisé	81
4.2	Mise en oeuvre de l'approche	83
5	Conclusion et perspectives	95
4	Détection d'attaques basée sur la représentation des données en graphe	97
1	Introduction	98
2	Synthèse des limites et contributions	99
3	Concept de base	101
3.1	Les réseaux neuronaux graphiques (GNN)	101
3.2	K-means clustering (K-means)	104
3.3	Recherche des plus proches voisins avec KNN via FAISS	105
4	Approche proposée	107
4.1	Déscription du dataset	108
4.2	Prétraitement des données	109
4.3	Construction du graphe pour l'apprentissage GCN	110
5	Résultats expérimentaux et discussion	112
6	Évaluation sur le dataset CICIOT2023	115
7	Comparaison des performances avec les approches existantes	116
8	Conclusion	117
9	Perspectives de recherche	118
	Conclusion générale	120
	Bibliographie	I

Table des figures

1	Coût estimé de la cybercriminalité dans le monde de 2018 à 2029 (en billions de dollars américains) [5]	1
2	Dépenses annuelles en milliard de dollars américains [6]	2
3	Exemple d'attaque MiTM	4
4	fonctionnement du pare-feu [20]	5
5	Zone Tampon Entre Internet et le Réseau Interne [23]	5
6	Les types des systèmes de détection d'intrusion	6
7	Utilisation des réseaux LSTM pour la classification des attaques	10
8	Génération d'une représentation vectorielle du trafic réseau à l'aide du modèle PV-DM pour la classification des attaques	11
9	Méthode pour traiter le déséquilibre de classe	12
10	Utilisation d'un Graph Convolutional Network (GCN) pour la classification des attaques	13
1.1	classification binaire et multiclasse [56]	18
1.2	Réseaux de neurones humain	18
1.3	Architecture d'un réseau de neurones artificiels	19
1.4	Structure d'un réseau de neurones	19
1.5	Structure d'un réseau de neurones récurrent déroulé dans le temps, où une copie du modèle est générée pour chaque étape temporelle	20
1.6	Cellule LSTM.	22
1.7	L'architecture du modèle proposé pour la détection d'intrusion	27
1.8	Les nombres d'enregistrements d'attaques et d'enregistrements normaux dans le dataset NSL-KDD	28
1.9	La valeur SHAP indique l'importance des fonctionnalités dans NSL-KDD	30
1.10	Score d'Anova f-test	31
1.11	Validation croisée k_{fold}	34
1.12	Matrices de confusion des modèles	36
2.1	word embedding	47
2.2	Modèle de PV-DM pour l'apprentissage des vecteurs de paragraphes [108]	48
2.3	L'architecture du modèle proposé pour la détection d'intrusion	51
2.4	Enregistrements d'attaques et normaux dans l'ensemble de données NSL-KDD	51
2.5	Enregistrements d'attaques et normaux dans l'ensemble de données UNSW-NB15	52
2.6	L'impact de toutes les caractéristiques sur les différentes catégories d'attaques et les comportements normaux pour l'ensemble de données UNSW-NB15	54
2.7	L'impact de toutes les caractéristiques sur les différentes catégories d'attaques et les comportements normaux pour l'ensemble de données NSL-KDD	54
2.8	Scores d'information mutuelle des caractéristiques de l'ensemble de données NSL-KDD	55
2.9	Scores d'information mutuelle des caractéristiques de l'ensemble de données UNSW-NB15	55
2.10	Importance des caractéristiques dans l'ensemble de données NSL-KDD en utilisant SHAP	56
2.11	Importance des caractéristiques dans l'ensemble de données UNSW-NB15 en utilisant SHAP	56
2.12	Impact des caractéristiques sélectionnées par MI sur les différentes catégories d'attaques et le comportement normal pour l'ensemble de données NSL-KDD	57

2.13	Impact des caractéristiques sélectionnées par MI sur les différentes catégories d'attaques et le comportement normal pour l'ensemble de données UNSW-NB15 . . .	58
2.14	Impact des caractéristiques sélectionnées par les valeurs SHAP sur les différentes catégories d'attaques et le comportement normal pour l'ensemble de données NSL-KDD	58
2.15	Impact des caractéristiques sélectionnées par les valeurs SHAP sur les différentes catégories d'attaques et le comportement normal pour l'ensemble de données UNSW-NB15	58
2.16	Performances de classification sur les ensembles NSL-KDD	60
2.17	Performances de classification sur les ensembles UNSW15.	61
2.18	Courbe ROC du classificateur XGBOOST.	62
2.19	Performances de classification du modèle LSTM sur les ensembles réduits de NSL-KDD.	63
2.20	Performances de classification du modèle LSTM sur les ensembles réduits de UNSW-NB15.	64
3.1	Distribution extrêmement déséquilibrée des classes	75
3.2	Méthode d'Oversampling et d'undersampling	76
3.3	Tomek links	78
3.4	Aperçu de la technique Boosting	79
3.5	Aperçu de la technique Bagging	80
3.6	Distribution des données d'entraînement et de test par type de connexion à partir du jeu de données UNSW-NB15.	82
3.7	L'approche proposée.	83
3.8	L'approche proposée.	85
3.9	La matrice de confusion utilisant toutes les fonctionnalités	90
3.10	La matrice de confusion du modèle proposé	91
4.1	Principaux types de structures de graphes	102
4.2	Schéma fonctionnel d'un réseau neuronal graphique (GNN)	102
4.3	Les tâches courantes sur les graphes incluent la classification de nœuds, la prédiction de liens et la classification du graphe	103
4.4	Exemple de résultats de clustering	104
4.5	Schéma du modèle proposé	107
4.6	Courbe de perte pendant l'entraînement	113
4.7	Courbe de perte pendant l'entraînement	113
4.8	Projection bidimensionnelle des embeddings des nœuds, montrant la séparation entre les différentes catégories de trafic.	115

Liste des tableaux

1.1	Classes d’attaques et leurs types dans le dataset NSL-KDD	27
1.2	Les attributs sélectionnés par la méthode Boruta	30
1.3	Les caractéristiques sélectionnées par différentes méthodes	31
1.4	Tableau des paramètres des architectures LSTM, Stacked-LSTM et BiLSTM . . .	33
1.5	Résultats expérimentaux	36
1.6	Comparaison des performances	37
2.1	Aperçu des différents modèles bien établis dans les systèmes de détection d’intru- sion (IDS)	45
2.2	Les différentes catégories d’attaques dans les deux ensembles de données	52
2.3	Caractéristiques des ensembles de données réduits	57
2.4	Tableau des paramètres du modèle PV-DM (ParagraphVector-Distributed Memory)	59
2.5	Taille de l’ensemble d’entraînement et de l’ensemble de test pour chaque ensemble	60
2.6	Paramètres du modèle LSTM	63
2.7	Classification performance of our method with 6 and 4 attributes on NSL-KDD .	65
2.8	Performances de classification de notre méthode avec 6 et 4 caractéristiques sur l’ensemble de données UNSW-NB15	65
2.9	Comparaison entre différents modèles et notre approche sur l’ensemble de don- nées NSL-KDD	66
2.10	Comparaison entre différents modèles et notre approche sur l’ensemble de don- nées UNSW15	66
3.1	Travaux connexes et de leurs limites.	74
3.2	Distribution des données d’entraînement et de test par type de connexion à partir du jeu de données UNSW-NB15	81
3.3	Partitionnement de l’ensemble de données en sous-ensembles	84
3.4	Caractéristiques sélectionnées pour chaque sous-ensemble	86
3.5	Fréquence d’occurrence des caractéristiques	87
3.6	Caractéristiques finales sélectionnées	87
3.7	Résultats expérimentaux	88
3.8	Les mesures de performance des deux modèles dans différentes catégories. . . .	89
3.9	Évaluation des performances du modèle proposé sur différentes classes dans les jeux de données CICIDS2019 et CICIOT2023	92
3.10	Analyse comparative des résultats avec les modèles existants	94
4.1	Description et nombres des instances par catégorie d’attaque	108
4.2	Configuration des hyperparamètres du modèle GCN	112
4.3	Résultats globaux de performance du modèle GCN	113
4.4	Performances du modèle par classe	114
4.5	Performance du modèle GCN sur le dataset CICIOT2023	116
4.6	Comparaison des performances des différentes techniques de classification	116

Liste des abréviations

***ANN** : Artificial Neural Network*
***ANOVA** : Analyse de la Variance*
***AUC** : Area under the Curve*
***BiLSTM** : Les réseaux de neurones récurrents Bidirectionnels*
***CIC** : Institut Canadien pour la Cybersécurité*
***CNN** : Réseau de neurones convolutionnel*
***D3M** : Drive-by-Download Malware Dataset*
***DMZ** : Demilitarized Zone*
***DoS** : Denial of Service*
***Faiss** : Facebook AI Similarity Search*
***GCN** : Graph Convolutional Networks*
***GNN** : Graph Neural Network*
***GRU** : Unité récurrente à portes*
***HIDS** : Système de détection d'intrusion basé sur l'hôte*
***IA** : Intelligence Artificielle*
***IDS** : Systèmes de détection d'intrusion*
***Iot** : Internet of Things*
***Kmeans** : K-means clustering*
***KNN** : K-Nearest Neighbors*
***LGBM** : Light Gradient Boosting Machine*
***LSTM** : Les réseaux à mémoire court-terme et long-terme (Long Short-Term Memory)*
***MCC** : Matthews Correlation Coefficient*
***MitM** : Attaque Man-in-the-Middle*
***MTA** : Malicious Traffic Analysis*
***Nadam** : Nesterov-accelerated Adaptive Moment Estimation*
***NIDS** : Système de détection d'intrusion basé sur le réseau*
***PV-DM** : Paragraph Vector - Distributed Memory*
***R2L** : Attaque distant vers locale*
***T-SNE** : t-Distributed Stochastic Neighbor Embedding*
***VPN** : Virtual Private Network*
***Wi-Fi** : Wireless Fidelity*
***XGBoost** : eXtreme Gradient Boosting*

Liste des publications

Revue internationale :

- * **Titre** : A robust intrusion detection system based on a shallow learning model and feature extraction techniques.
Auteurs : Chadia El Asry, Ibtissam benchaji, Samira Douzi, Bouabid El Ouahidi.
Journal : Plos One
<https://doi.org/10.1371/journal.pone.0295801>
- * **Titre** : Enhancing cybersecurity : A high-performance intrusion detection approach through boosting minority class recognition.
Auteurs : Chadia El Asry, Ibtissam benchaji, Samira Douzi, Bouabid El Ouahidi.
Journal : Plos One
<https://doi.org/10.1371/journal.pone.0317346>
- * **Titre** : Effective Approaches for Intrusion Detection Systems in the Face of Low-Frequency Attacks.
Auteurs : Chadia El Asry, Ibtissam benchaji, Samira Douzi, Bouabid El Ouahidi.
Journal : Journal of Advances in Information Technology
doi : 10.12720/jait.15.9.1070-1078
- * **Titre** : A Graph Neural Network-Based Framework for Intrusion Detection in IoT using KMeans and FAISS-Accelerated KNN.
Auteurs : Chadia El Asry, Fouad Yakoubii, Ibtissam benchaji, Samira Douzi, Bouabid El Ouahidi.
Journal : Plos One (soumis)
- * **Titre** : Explanatory predictive model for COVID-19 severity risk employing machine learning, shapley addition, and LIME.
Auteurs : Mariam Laati, Samira Douzi, Hind Ezzine, Chadia El Asry, Abdellah Naya, Abdelaziz Bouklouze, Younes Zaid & Mariam Naciri.
Journal : Scientific Reports
doi : 10.1038/s41598-023-31542-7

Articles acceptés pour publication et présentation orale dans des conférences internationales :

- * **Titre** : A Deep Learning Model for Intrusion Detection with Imbalanced Dataset.
Auteurs : Chadia El Asry, Samira Douzi, Bouabid El Ouahidi.
Présentée à : 'The International Conference on Intelligent System and Smart Technologies' Settati, Morocco.
Publié dans : la collection lecture notes in networks and systems ((LNNS, volume 826)) Advances in Intelligent System and Smart Technologies. I2ST 2023. Lecture Notes in Networks and Systems, vol 826. Springer, Cham. <https://doi.org/10.1007/978-3-031-47672->

3_26.

- * **Titre** : Intrusion Detection System, a New Approach to R2L and U2R Attack Classification.
Auteurs : Chadia El Asry, Samira Douzi, Bouabid El Ouahidi.
Présentée à : 'The 5th International Conference on Artificial Intelligence and Smart Environment(ICAISE'23)' Errachidia, Maroc.
Publié dans la collection lecture notes in networks and systems (LNNS, volume 837)
Dans l'ouvrage Artificial Intelligence, Data Science and Applications (pp.297-304).vol 837. Springer, Cham. https://doi.org/10.1007/978-3-031-48465-0_39.

- * **Titre** : An intrusion detection system using paragraph vector-distributed memory approach.
Auteurs : Chadia El Asry, Ibtissam benchaji, Samira Douzi, Bouabid El Ouahidi.
Présentée à : International Conference on Artificial Intelligence in Cybersecurity and Sustainability (AiSec'2024) Essaouira, Morocco.
Publié dans l'ouvrage : Recent Advances in Internet of Things Security (pp.105-114)
DOI :10.1201/9781003587552-12.

- * **Titre** : Toward A Novel IDS Approach based on Paragraph Vector-Distributed Memory.
Auteurs : Chadia El Asry, Ibtissam benchaji, Samira Douzi, Bouabid El Ouahidi.
Présentée à : International Conference on Circuit, Systems and Communication (ICCSC 2024) Fez, Maroc.

Introduction générale

1 Contexte

Les intrusions sont définies généralement comme des actes malicieux commis intentionnellement par des individus ou des groupes organisés qui cherchent à compromettre l'intégrité des systèmes d'information, voler des données sensibles ou perturber le fonctionnement normal des services basés sur les Technologies de l'Information et de la Communication. En effet, ces attaques désirent occasionnellement déstabiliser l'activité des entités ou des infrastructures en ciblant leurs points faibles. Ce paradigme ne se rattache pas exclusivement à l'époque numérique moderne, mais un phénomène qui a accompagné les systèmes informatiques depuis leur apparition [1] [2].

Par ailleurs, cette problématique continue de s'accroître et le nombre de cyberattaques ayant réussies à compromettre les systèmes d'information demeure très significatif, malgré le développement et la sophistication des technologies de sécurité informatique, telles que les nouvelles générations de pare-feux (NGFW) et les solutions EDR (Endpoint Detection and Response).

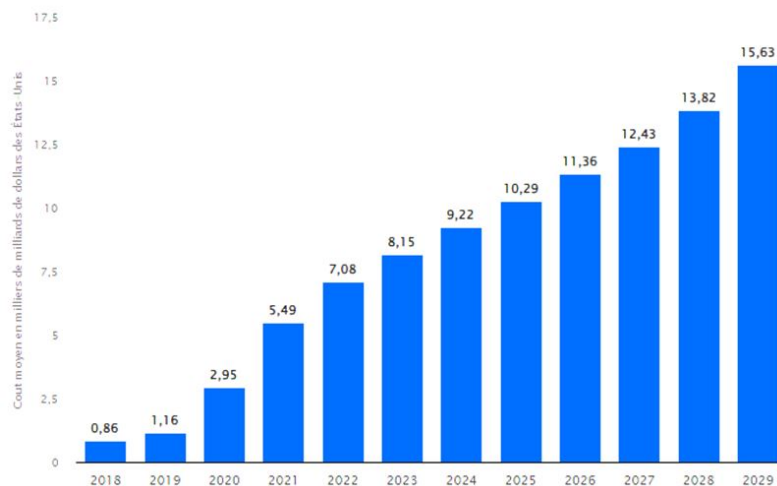


FIGURE 1 – *Coût estimé de la cybercriminalité dans le monde de 2018 à 2029 (en milliards de dollars américains) [5]*

En effet, selon le rapport annuel de Cybersecurity Ventures 2024, les pertes financières mondiales causées par les cyberattaques ont atteint 8 trillions de dollars en 2023 et pourrait atteindre 10.5 trillions de dollars d'ici 2025 [3]. Ce chiffre inclut divers coûts tels que la destruction de données, le vol d'argent, la perte de productivité, le vol de propriété intellectuelle, le vol de données personnelles et financières, le détournement de fonds, la fraude, les perturbations des activités commerciales, les enquêtes médico-légales, la restauration et la suppression de données et de systèmes piratés, ainsi que l'atteinte à la réputation [4]. La Figure 1 illustre le coût estimé de la cybercriminalité dans le monde de 2018 à 2029, en milliers de milliards de dollars des États-Unis [5].

Il est important de noter que les pertes réelles pourraient être plus élevées, car de nombreuses organisations choisissent de ne pas divulguer publiquement les incidents de sécurité pour préserver leur réputation [3].

À cet égard, les dépenses consacrées aux produits et services liés à la cybersécurité connaîtront certainement une croissance significative afin de protéger les entreprises en pleine numérisation, les équipements de l'Internet des objets (IoT) ainsi que les consommateurs. Cette affirmation est soutenue par la société de recherche et de conseil spécialisée dans la cybersécurité « Cybersecurity Ventu », qui déclare que les investissements mondiaux en cybersécurité devraient dépasser 1,75 billion de dollars américains sur la période 2021-2025 (voir figure 2).

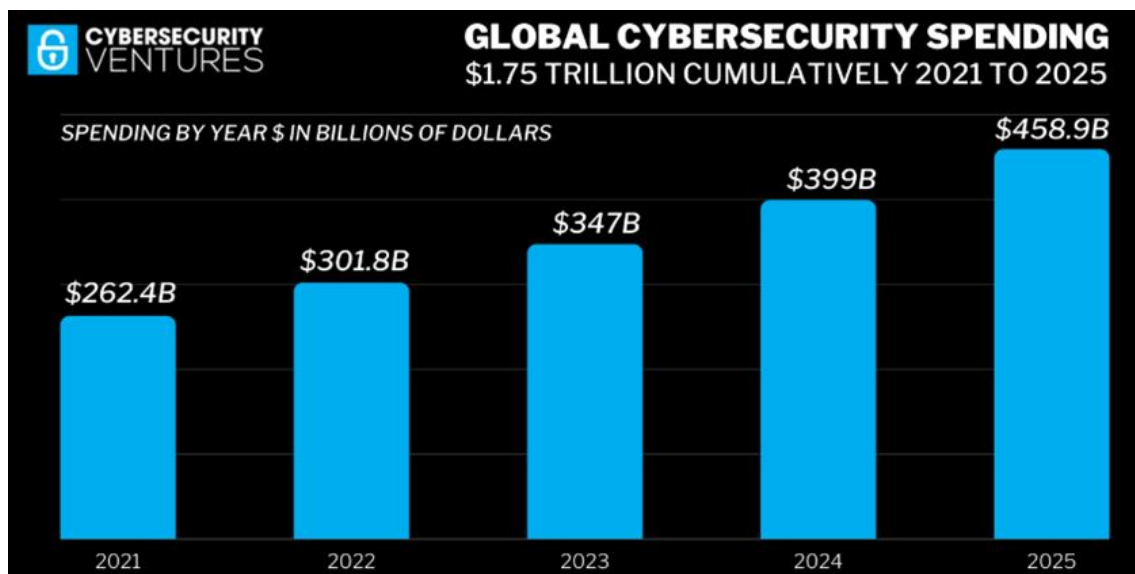


FIGURE 2 – Dépenses annuelles en milliard de dollars américains [6]

Dans le contexte marocain, un rapport de Kaspersky révèle qu'au premier semestre 2024, plus de 40 % des ordinateurs des systèmes de contrôle industriels (ICS) ont subi des cyberattaques [7]. En outre, l'Association Marocaine des Utilisateurs des Systèmes d'Information (AUSIM) a publié en 2024, dans le cadre d'une étude réalisée en collaboration avec PwC (PricewaterhouseCoopers), que 84 % des organisations ont signalé une augmentation du volume des attaques, dont les rançongiciels représentent la menace principale. De plus, 61 % des entreprises ont signalé des incidents liés à la fuite de données, tandis que 45 % ont rapporté des compromissions de comptes. Concernant les investissements en cybersécurité, 78 % des organisations consacrent jusqu'à 25% de leur budget technologique à la sécurité des systèmes d'information [8].

Dans cette optique, la cybersécurité a pris un rôle stratégique qui ne cessera de s'élargir en parallèle avec la hausse continue non seulement du nombre et de la gravité des menaces numériques, mais également des types des attaques auxquelles les organisations et les individus doivent faire face.

Ainsi, il s'avère crucial de comprendre les différentes formes que peuvent prendre les attaques, afin d'adopter des stratégies de défense adaptées.

2 Types d'attaques

Une attaque est littéralement définie comme toute action visant à cibler un ordinateur ou tout élément d'un système d'information, afin de détruire, modifier, exposer, voler des données, obtenir un accès non autorisé, exploiter ou perturber un réseau. Avec la digitalisation croissante des entreprises, ces attaques se multiplient et sont devenues de plus en plus populaires ces dernières années.

Il existe plusieurs types d'attaques, certaines étant particulièrement redoutées en raison de leur impact et des dommages qu'elles peuvent causer. Parmi les plus courantes, celles qui ciblent les entreprises et les individus incluent [9] :

- **Une attaque par déni de service (Denial of service (DoS))** : est une menace majeure pour les réseaux informatiques qui consiste à saturer les ressources d'un système ou d'un réseau pour empêcher ses utilisateurs légitimes d'accéder à un service. Ces attaques peuvent bloquer complètement le service ou ralentir considérablement son fonctionnement [10]. Elles sont utilisées pour diverses motivations, comme nuire à un concurrent commercial, extorquer de l'argent à une entreprise en menaçant son fonctionnement, ou exprimer une opposition politique. Un exemple célèbre d'attaque par déni de service (DoS) est l'attaque GitHub de 2018. Cette attaque utilisait un type spécifique de DoS appelé amplification DNS. Les attaquants ont exploité des serveurs DNS mal configurés pour envoyer des quantités massives de trafic vers GitHub, culminant à 1,35 téraoctets par seconde. Cela a temporairement mis le site hors ligne, bloquant l'accès aux utilisateurs légitimes [11].
- **Attaque de logiciels malveillants** : également appelé malware, est un programme conçu pour endommager, perturber ou infiltrer un ordinateur, un serveur, un réseau informatique ou une infrastructure, souvent à l'insu des utilisateurs. Le malware est généralement classé en plusieurs catégories, selon la manière dont il est introduit dans le système cible et le type de violation qu'il cherche à provoquer. La classification traditionnelle a été introduite par Peter Denning à la fin des années 1980 [12] [13]. Il existe différents types de malwares, tels que les virus, qui se propagent en infectant d'autres fichiers et perturbent le fonctionnement des systèmes ; les chevaux de Troie, qui prennent l'apparence de programmes légitimes mais exécutent des actions malveillantes une fois installés ; et les rançongiciels, qui bloquent l'accès aux données ou aux systèmes en échange d'une rançon. Pour se protéger efficacement contre ces menaces, il est essentiel d'adopter plusieurs bonnes pratiques, comme l'installation d'un logiciel antivirus régulièrement mis à jour, l'évitement des téléchargements et ouvertures de fichiers provenant de sources non vérifiées, ainsi que la prudence lors de la navigation sur des sites web inconnus. L'utilisation d'un pare-feu correctement configuré et la mise à jour régulière des systèmes d'exploitation et des logiciels permettent également de bénéficier des derniers correctifs de sécurité et de renforcer les défenses contre les cyberattaques [14].
- **Attaque par injection SQL** : est une technique d'attaque qui consiste à insérer du code malveillant dans une requête SQL afin de manipuler les données d'une base de données. Cette méthode permet à un attaquant d'accéder, modifier ou supprimer des informations sensibles, voire de contourner les mécanismes d'authentification [15]. Un exemple marquant est celui de l'entreprise asiatique VTech, spécialisée dans la vente de jouets connectés. Une personne malveillante a exploité une vulnérabilité liée à l'injection SQL pour accéder à la base de données de l'entreprise et voler les informations de 4,8 millions de clients. L'attaque est passée inaperçue pour VTech, qui n'en a été informée qu'après que le site américain Motherboard ait été contacté par le pirate lui-même [15].
- **Attaque de l'homme du milieu (MitM)** : Une attaque Man-in-the-Middle (MitM) survient lorsqu'un attaquant s'interpose discrètement entre deux parties en communication (figure 3). Il peut ainsi intercepter, écouter, voire modifier les données échangées sans que les vic-

times ne s'en aperçoivent. Cette intrusion permet à l'attaquant d'accéder à des informations sensibles ou de manipuler la communication à des fins malveillantes [16].

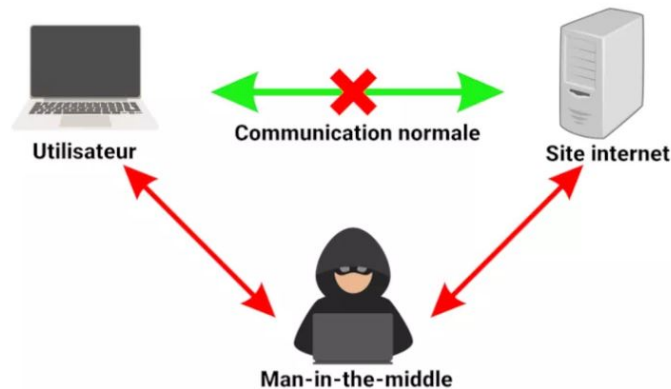


FIGURE 3 – Exemple d'attaque MiTM

- **Attaque distant vers locale (R2L)** : se produit lorsqu'un attaquant exploite des vulnérabilités dans un ordinateur ou un logiciel de sécurité réseau pour obtenir un accès non autorisé à une machine ou à un système. En envoyant des paquets à une machine distante via Internet, l'attaquant cherche à obtenir des privilèges similaires à ceux d'un utilisateur local. L'objectif principal de cette attaque est d'explorer ou de voler illégalement des données, d'introduire des logiciels malveillants (comme des virus) ou de causer des dommages à la victime [17].

En conséquence, la multiplicité et la diversité des types d'attaques a stimulé l'innovation et le déploiement de plusieurs techniques et moyens de se permettant d'atténuer, de bloquer ou d'éviter les conséquences des attaques cybernétiques.

3 Les techniques de sécurité

En vue de renforcer la sécurité, les administrateurs s'appuient sur diverses solutions, telles que l'utilisation de réseaux privés virtuels (VPN) pour chiffrer le trafic Internet, les antivirus, les pare-feux, les zones démilitarisées (DMZ) et les systèmes de détection d'intrusion, afin de surveiller et de protéger contre les accès non autorisés. L'objectif est d'assurer une protection optimale du réseau informatique. Nous détaillerons chacune de ces méthodes dans la suite.

- **Antivirus** : est un programme conçu pour détecter, bloquer et éliminer les virus présents sur un ordinateur. L'élimination d'un virus, appelée éradication, peut se faire de différentes manières. Elle peut consister à nettoyer le fichier infecté en supprimant uniquement le code malveillant, à le supprimer complètement ou encore à le mettre en quarantaine. Cette dernière méthode permet de déplacer le fichier infecté vers un emplacement sécurisé où il ne peut être exécuté, empêchant ainsi toute propagation du malware [18]. Les antivirus reposent souvent sur des techniques de détection basées sur les signatures et partagent plusieurs similitudes avec les systèmes de détection d'intrusion (IDS).
- **Les Pare-feu** : est un système, matériel ou logiciel, conçu pour filtrer le trafic réseau en analysant les paquets entrants et sortants. Il autorise ou bloque leur passage en fonction d'un ensemble de règles prédéfinies, appelées listes de contrôle d'accès (ACL) (figure 4). En plus de réguler le flux des données, le pare-feu enregistre les tentatives d'intrusion dans un journal destiné aux administrateurs du réseau. Il joue également un rôle essentiel dans le

contrôle de l'accès aux applications et la prévention des usages malveillants ou détournés [19].

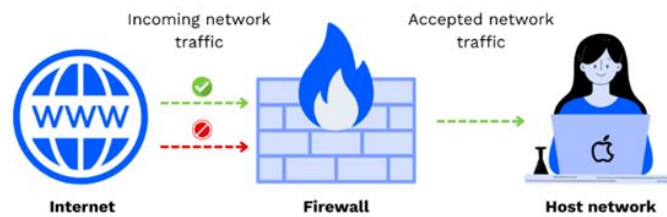


FIGURE 4 – fonctionnement du pare-feu [20]

- **VPN (Virtual Private Network)** : est un réseau privé virtuel qui permet à des utilisateurs ou des systèmes distants de communiquer de manière sécurisée via une infrastructure publique, comme Internet. Il fonctionne en créant un tunnel chiffré entre les points de connexion, empêchant ainsi l'interception ou l'altération des données. L'accès au VPN est restreint à une communauté définie d'utilisateurs, et il utilise des protocoles de tunnelisation pour garantir la confidentialité et l'intégrité des échanges [21].
- **DMZ (Demilitarized Zone)** : est une zone intermédiaire de sécurité dans un réseau informatique, placée entre le réseau interne d'une organisation et Internet. Elle sert de tampon pour héberger des serveurs accessibles depuis l'extérieur (comme les serveurs web, DNS, mail ou FTP) tout en protégeant le réseau interne contre les menaces potentielles (figure 5). En isolant ces services dans une zone distincte derrière un pare-feu, la DMZ ajoute une couche de sécurité supplémentaire, empêchant les connexions directes entre Internet et le réseau local de l'organisation [22].

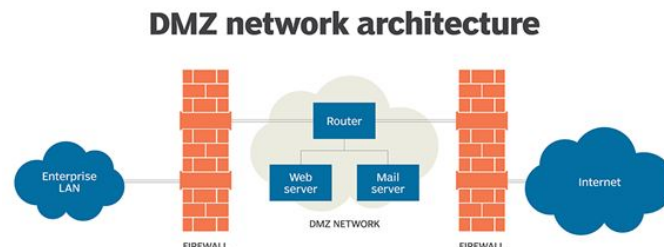


FIGURE 5 – Zone Tampon Entre Internet et le Réseau Interne [23]

- **Systèmes de détection d'intrusion (IDS)** : sont parmi les outils les plus efficaces pour surveiller en continu l'activité d'un réseau ou d'un système informatique afin de détecter les comportements suspects et les tentatives d'accès non autorisées. [24].
Par ailleurs, la combinaison de ces techniques de sécurité ainsi que la prise de certaines mesures comme les mises à jour régulières des logiciels, amélioreraient certainement la capacité des organisations à détecter les cyberattaques et à réagir rapidement pour en limiter l'impact.

Alors, parmi les différentes techniques de sécurité évoquées précédemment pour lutter contre les menaces cybernétiques, les systèmes de détection d'intrusion se distinguent par leur approche innovante et leurs résultats prometteurs. C'est précisément sur cette solution que nous allons maintenant nous pencher, afin d'en explorer les mécanismes, les avantages et les défis qu'elle soulève.

4 Système de détection d'intrusion (IDS)

4.1 Définition

Les Systèmes de Détection d'Intrusion (IDS) sont des solutions de sécurité informatique conçues pour surveiller le trafic réseau et les activités des systèmes afin de détecter toute tentative d'intrusion ou comportement anormal. Apparues dans les années 1980 [25] avec des projets de recherche comme STAT [26], EMERALD [27], Bro [28] et Snort [29], ces technologies ont évolué, donnant naissance à des solutions commerciales dans les années 1990, telles que Cisco Secure IDS [30] et ISS RealSecure [31].

Lorsqu'une activité suspecte est détectée, un IDS peut réagir de différentes manières. Il peut alerter l'administrateur système, enregistrer les événements pour analyse, bloquer le trafic malveillant ou prendre d'autres mesures de protection adaptées. Certains systèmes IDS disposent même d'une capacité de réponse automatique, leur permettant de neutraliser certaines attaques en temps réel, par exemple en bloquant le trafic provenant d'une source suspecte ou en désactivant un service vulnérable.

4.2 Classification des IDS en fonction de la source de données

Il existe trois principaux types d'IDS (figure 6) qui sont :

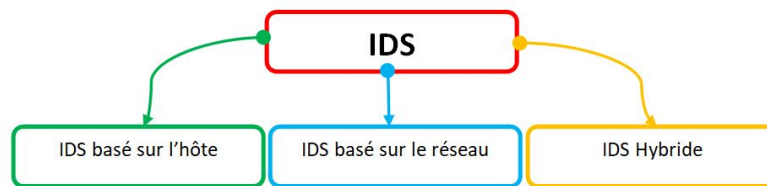


FIGURE 6 – Les types des systèmes de détection d'intrusion

- **Système de détection d'intrusion basé sur l'hôte (HIDS)** : est un type de système de détection d'intrusion conçu pour surveiller l'activité d'un ordinateur ou d'un appareil spécifique afin de repérer les accès non autorisés et les comportements malveillants [32]. Installés directement sur les hôtes, ils analysent les journaux système, les appels système, les fichiers DLL et les clés de registre pour identifier toute activité suspecte [33]. En fonction des méthodes de détection employées (basées sur les signatures ou basées sur les anomalies), ils signalent les menaces aux administrateurs ou bloquent les processus malveillants afin de renforcer la sécurité du système.
- **Système de détection d'intrusion basé sur le réseau (NIDS)** : est un type de système de détection d'intrusion chargé de surveiller le trafic réseau afin d'identifier d'éventuelles attaques ou activités suspectes [34]. Généralement déployés sur des périphériques stratégiques comme les routeurs, les commutateurs ou aux points d'étranglement du réseau [35], ils analysent les paquets en examinant divers éléments tels que les en-têtes, le contenu, la taille, les numéros de port et les connexions établies [36]. À l'aide de capteurs, ils détectent les paquets douteux et, selon la méthode de détection utilisée, ils peuvent soit signaler les menaces aux administrateurs, soit bloquer ou abandonner les paquets malveillants.
- **Système de détection d'intrusion hybride (NIDS+HIDS)** : combinent les fonctionnalités des IDS basés sur le réseau (NIDS) et des IDS basés sur l'hôte (HIDS) pour offrir une détection d'intrusion plus complète. Ils surveillent à la fois le trafic réseau et l'activité des hôtes afin d'identifier les menaces de manière plus efficace.

4.3 Classification des IDS selon les techniques de détection

La détection d'intrusion repose sur l'hypothèse que les comportements d'une activité intrusive diffèrent, dans une certaine mesure, de ceux des activités normales, ce qui les rend détectables [37]. De nombreuses approches de détection ont été proposées dans la littérature, parmi lesquelles figurent certaines des techniques les plus couramment utilisées pour les IDS :

- **Détection basée sur des règles** : repose sur l'utilisation d'un ensemble de règles prédéfinies pour identifier les activités suspectes sur un réseau ou un système. Ces règles peuvent être simples, comme le blocage du trafic provenant de sources malveillantes connues, ou plus complexes, en s'appuyant sur des modèles de comportement et d'autres facteurs. Cette approche est particulièrement efficace pour détecter les menaces connues, car elle applique des critères précis pour reconnaître les attaques déjà répertoriées. Toutefois, elle présente une limitation majeure : son incapacité à identifier les attaques nouvelles ou inconnues, qui n'ont pas encore été intégrées dans la base de règles, réduisant ainsi son efficacité face aux menaces émergentes [38].
- **Détection basée sur les signatures** : repose sur la comparaison du trafic réseau ou de l'activité du système avec une base de données de signatures d'attaques connues. Cette approche permet à l'IDS d'identifier toute activité correspondant à un modèle d'attaque préalablement répertorié et de déclencher une alerte en conséquence. Bien qu'efficace pour repérer les menaces connues, elle présente une limite majeure : son incapacité à détecter les attaques nouvelles ou encore inconnues [39].
- **Détection basée sur les anomalies** : également appelés systèmes de détection d'intrusion basés sur les profils, analysent les activités des réseaux et des hôtes pour établir des modèles de comportement normal. Toute activité s'écartant significativement de ces modèles est considérée comme une intrusion [40], déclenchant ainsi une alerte aux administrateurs système. Ce système est particulièrement efficace pour détecter les attaques inconnues et zero-day mais si la ligne de base définissant le comportement normal n'est pas établie avec précision, l'IDS risque de générer un nombre excessif de faux positifs [41].
- **Détection basée sur l'analyse de protocole avec état** : repose sur le comportement d'un protocole, cette approche utilise des modèles standards établis par des fournisseurs, définissant la manière dont un protocole doit ou ne doit pas être utilisé [42]. Cette méthode ne vise pas à détecter une attaque spécifique et déjà identifiée, mais plutôt à repérer une anomalie dans le déroulement d'un protocole ou dans les champs d'en-tête. Par exemple, elle est capable de repérer une attaque par déni de service (DoS) en détectant une adresse IP unique qui génère simultanément plusieurs requêtes de connexion TCP sur une courte période [50]. Cependant, elle nécessite une analyse approfondie des données et peut être plus gourmande en ressources, ce qui peut poser des défis en termes de performances et d'efficacité dans des environnements à forte charge [43].
- **Détection basée sur l'analyse statistique** : consiste à analyser le trafic réseau ou l'activité du système afin d'identifier des irrégularités ou des anomalies pouvant indiquer une activité malveillante. Cela peut englober des méthodes comme le l'analyse de corrélation, le regroupement ou d'autre méthode statistique. Cette approche est particulièrement efficace pour détecter des menaces inconnues. Cependant, elle peut être gourmande en ressources, notamment en termes de puissance de calcul et de stockage, et nécessite parfois une analyse approfondie des données pour garantir son efficacité [42].
- **Détection basée sur l'intelligence artificielle (IA)** : La détection basée sur l'IA implique l'emploi d'algorithmes d'apprentissage automatique pour examiner de manière automatique le trafic réseau ou l'activité du système et repérer les modèles ou comportements suspects. Cela peut permettre à l'IDS d'identifier des menaces jusque-là inconnues ou émergentes qui pourraient ne pas être identifiables via d'autres méthodes. La détection fondée sur l'IA est

performante pour identifier diverses menaces, toutefois elle peut exiger une consommation importante de ressources et requiert souvent des données et une formation approfondie pour être efficace [44].

Parmi les différentes méthodes de détection d'intrusion évoquées précédemment, cette thèse se distingue par l'adoption de la détection basée sur l'intelligence artificielle, exploitant les capacités d'apprentissage automatique pour identifier de manière proactive les intrusions dans les systèmes.

5 Motivation

Bien que les systèmes de détection d'intrusion (IDS) basé sur l'IA soient considérés comme une avancée marquante dans le domaine de la sécurité informatique, plusieurs défis entravent encore son développement et son application optimale. Parmi ces défis, on peut citer :

- **Gestion des classes déséquilibrées** : Dans le contexte des systèmes de détection d'intrusions, un défi majeur réside dans le déséquilibre des classes entre les intrusions et les événements légitimes. Les intrusions représentent généralement une fraction très faible du trafic réseau global, souvent inférieure à 1% des événements détectés par un IDS. La majorité des connexions sont légitimes, ce qui crée un déséquilibre de classe : une faible proportion d'intrusions parmi une masse d'événements normaux. Ce déséquilibre entre la classe majoritaire et la classe minoritaire peut entraîner une mauvaise performance des modèles de classification, privilégiant les classes dominantes (majoritaires) au détriment des classes moins représentées (minoritaires), et affectant ainsi la détection d'attaque [45].
- **Modèles dynamiques d'attaques** : Un autre défi majeur pour les systèmes de détection d'intrusions réside dans l'évolution rapide des techniques d'attaque. Les cybercriminels adaptent constamment leurs méthodes pour contourner les systèmes de détection. Les IDS basés sur des modèles supervisés, qui sont formés sur des données historiques d'attaques passées, peuvent avoir des difficultés à détecter de nouvelles attaques qui n'ont pas été observées auparavant. Ce phénomène, souvent désigné sous le nom de nouveaux types d'attaques ou apprentissage de nouvelles classes, met en évidence la nécessité pour un système de détection d'être capable de s'adapter et d'apprendre à partir de nouveaux comportements d'intrusion, même sans disposer de données d'entraînement spécifiques. En conséquence, un IDS doit être capable de détecter de nouvelles menaces tout en restant efficace face aux attaques déjà connues.
- **Détection en temps réel** : Les systèmes de détection d'intrusions doivent être capables de fonctionner en temps réel, car les attaques doivent être détectées et contrées immédiatement pour éviter des dommages graves. Cependant, cette détection en temps réel pose plusieurs défis techniques : (i) La vitesse des flux de données générés par les utilisateurs légitimes et les attaquants rend leur traitement en temps réel complexe, (ii) L'analyse des données pour détecter des intrusions est coûteuse en ressources informatiques, (iii) L'algorithme de détection doit être à la fois performant et capable de traiter des volumes de données massifs avec une faible latence. Pour y parvenir, les IDS doivent gérer efficacement l'analyse de flux en continu, tout en garantissant une détection rapide des comportements malveillants.
- **Mesure de la performance** : est un défi supplémentaire dans la détection des intrusions, qui est causé par la distribution déséquilibrée des données. Le taux d'exactitude (observations correctement classées) est couramment utilisé pour un problème de classification standard. Pourtant, cette mesure n'est pas toujours adaptée à la détection d'intrusion. Par exemple, si les données comportent seulement 1% d'observation d'attaques, un taux d'exactitude

inférieur à 99% est inacceptable. La cause est simple : le système peut classer tous les observations comme normal et donner un taux élevé de précision. Alors, il est nécessaire de considérer des métriques de performance sensibles aux coûts, prenant en compte les observations mal classées, sans provoquer une multitude de fausses alertes qui consommeraient de temps et de ressources considérables à enquêter. Ces statistiques, qu'il est recommandé de prendre en compte dans l'analyse de la détection d'intrusion, sont notamment le score F1, la sensibilité, le coefficient de corrélation de Matthew et la courbe de Precision-Recall (AUC).

Les défis décrits ci-dessus motivent ce projet de recherche, étant donné que le problème se situe au cœur des préoccupations actuelles dans le domaine de la cybersécurité, et que les défis précédemment cités doivent être relevés, ce qui pourrait être très utile pour renforcer la sécurité des systèmes informatiques et réduire des risques pour les infrastructures critiques.

6 Objectif

Dans le domaine de la détection d'intrusion, l'efficacité est un facteur clé pour assurer une protection optimale des systèmes informatiques. Elle permet d'améliorer la capacité à identifier et bloquer les intrusions malveillantes tout en minimisant les alertes inutiles. Une amélioration même minime du taux de détection peut avoir un impact significatif sur la sécurité des infrastructures.

L'objectif de cette étude est de concevoir et développer des méthodes basées sur l'apprentissage automatique, capables de réduire les faux positifs et les faux négatifs, afin de détecter les menaces avec précision tout en préservant la disponibilité et la performance des systèmes protégés

7 Contributions

Cette section présente les différentes contributions de la thèse, réparties en quatre chapitres, en plus du chapitre introductif qui décrit le contexte des attaques informatiques et précise la problématique de recherche.

- **Chapitre 1 : Utilisation des données historiques dans la détection des intrusions [46]**

Dans ce chapitre, nous proposons un système de détection d'intrusion basé sur la modélisation séquentielle des données, en utilisant les réseaux de neurones récurrents. En effet, des schémas d'attaques similaires peuvent, dans certains contextes, être considérés comme des comportements normaux du système, tandis que dans d'autres, ils peuvent représenter des menaces évidentes. Une attaque ne se définit donc pas uniquement par ses caractéristiques intrinsèques, mais également par le contexte spécifique dans lequel elle se produit.

Pour construire un tel contexte résumant l'historique des comportements d'attaque, nous utilisons les réseaux à mémoire court-terme et long-terme LSTM (Long Short-Term Memory) comme classifieur dynamique, afin de capturer la dépendance séquentielle entre des événements d'attaque successifs. L'objectif est de permettre au classifieur d'identifier plus efficacement des actions anormales au sein d'un système.

la robustesse de ce modèle est construite en combinant la force de trois méthodes de sélection de caractéristiques : Boruta, SHAP et ANOVA pour extraire les informations les plus

significatives, les réseaux LSTM sont utilisés pour incorporer les séquences d'attaques afin d'améliorer la classification du modèle. Les résultats obtenus suggèrent que la modélisation basée sur des réseaux de neurones récurrents constitue une approche prometteuse pour caractériser les séquences d'attaques et améliorer l'efficacité des mécanismes de détection d'intrusions (figure 7).

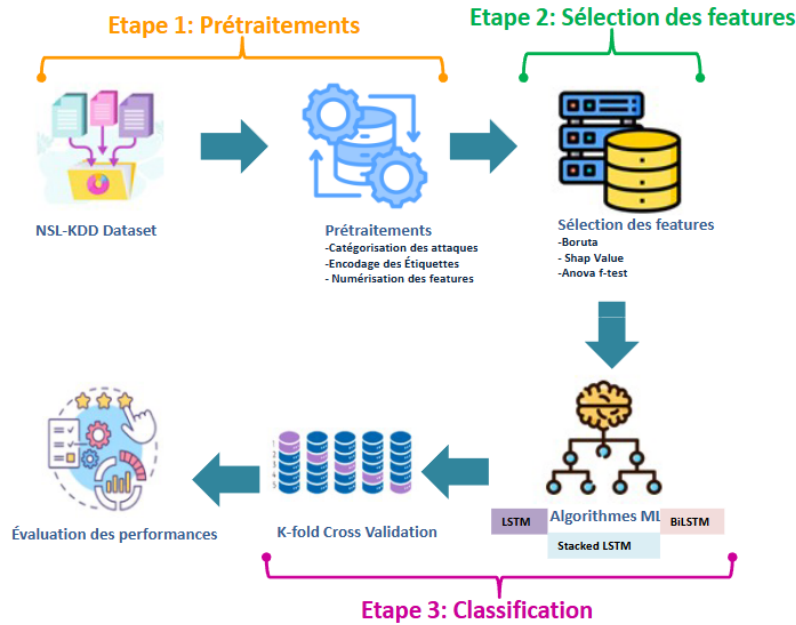


FIGURE 7 – Utilisation des réseaux LSTM pour la classification des attaques

• Chapitre 2 : Apprentissage Contextuel et Sémantique des attaques [47]

Dans le travail précédent, Nous avons modélisé les comportements d'attaque en utilisant Dans le travail précédent, nous avons modélisé les comportements d'attaque à l'aide de réseaux de neurones récurrents. Les expériences ont montré que la prise en compte de l'historique des attaques améliore significativement les performances de détection. Cependant, ces modèles nécessitent une puissance de calcul élevée en raison de leur complexité, notamment liée aux mémoires à court et à long terme, qui stockent des informations sur différentes périodes. Bien que cette approche soit efficace, elle peut devenir coûteuse en ressources informatiques.

Les réseaux LSTM sont performants pour capturer les relations temporelles entre les événements, mais leur architecture limite leur capacité à conserver toutes les informations pertinentes lorsque les séquences deviennent très longues, entraînant ainsi une perte d'information. De plus, leur capacité de généralisation peut être restreinte face aux variations complexes des modèles d'attaque. Pour pallier ces limites, nous explorons une approche alternative basée sur le modèle PV-DM (ParagraphVector-Distributed Memory) (figure 8), qui génère une représentation vectorielle des événements réseau en tenant compte de la séquence et du contexte global des intrusions. Ce modèle permet d'apprendre une représentation dense et fixe des séquences, indépendamment de leur longueur, tout en capturant efficacement les relations sémantiques et structurelles des attaques. L'objectif est de produire des vecteurs de caractéristiques intégrant à la fois les informations locales des événements et le contexte global du flux de données. Ces vecteurs sont ensuite utilisés pour entraîner un modèle de détection d'intrusions.

Les résultats expérimentaux montrent que l'approche PV-DM offre des performances comparables à celles du modèle LSTM, tout en étant plus simple et moins coûteuse en calcul, ce qui la rend particulièrement adaptée aux systèmes de détection d'intrusions en temps réel.

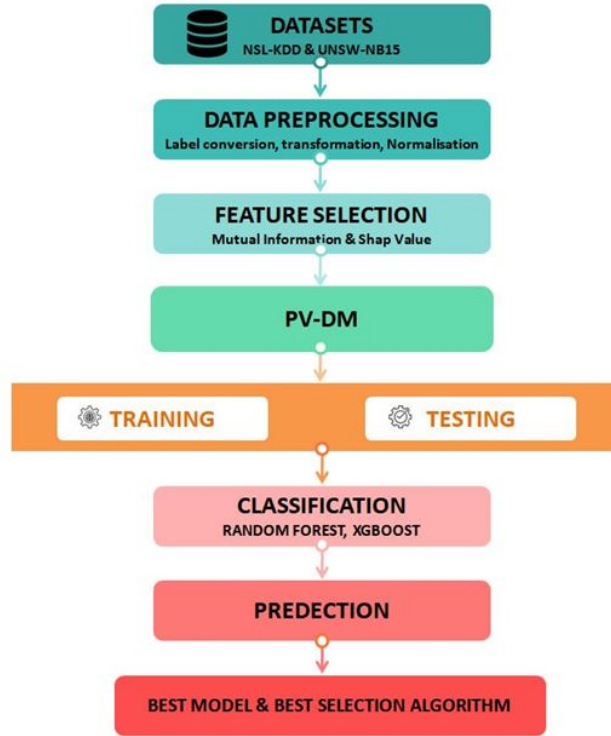


FIGURE 8 – Génération d'une représentation vectorielle du trafic réseau à l'aide du modèle PV-DM pour la classification des attaques

• Chapitre 3 : Une approche efficace pour la détection des attaques rares [48] [49]

Les travaux précédents ont démontré des performances satisfaisantes en matière de classification des attaques, notamment grâce à l'utilisation des réseaux LSTM et PVDM. Ces modèles se sont révélés efficaces pour capturer les relations temporelles entre les événements et détecter divers types d'intrusions. Cependant, un problème persistant demeure : la difficulté à reconnaître les classes minoritaires. En effet, les modèles traditionnels ont tendance à privilégier les classes majoritaires, ce qui entraîne une détection moins efficace des attaques rares, souvent les plus critiques en cybersécurité.

Dans ce contexte, notre approche, présentée dans le chapitre 3, vise à améliorer la reconnaissance des classes minoritaires sans recourir aux techniques classiques de rééchantillonnage comme le suréchantillonnage ou le sous-échantillonnage. Plutôt que de modifier artificiellement la distribution des données, nous nous concentrons sur l'extraction des caractéristiques discriminantes propres à chaque type d'attaque. Pour cela, nous appliquons une stratégie en plusieurs étapes (figure 9) : d'abord, nous segmentons les données en fonction de la fréquence des attaques ; ensuite, nous utilisons l'algorithme ShapRFECV pour identifier les caractéristiques les plus pertinentes pour chaque catégorie. Nous analysons ensuite la fréquence d'apparition de ces caractéristiques et priorisons celles présentes dans au moins trois sous-groupes distincts. Enfin, nous exploitons XGBoost comme classificateur, garantissant une distinction efficace entre le trafic normal et anormal.

Les résultats expérimentaux démontrent que cette méthode améliore considérablement la détection des classes minoritaires, tout en préservant l'intégrité des données et en maintenant de bonnes performances globales. En évitant les biais liés au rééchantillonnage, notre approche assure une classification plus équilibrée des attaques, rendant le système de détection plus robuste et efficace face aux intrusions rares mais critiques

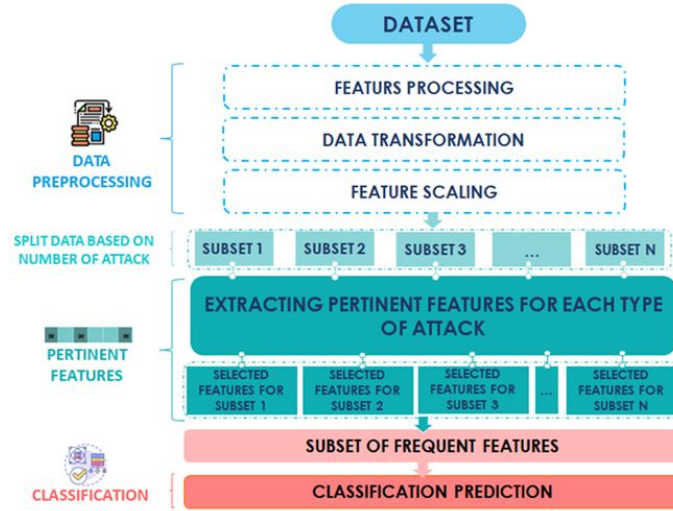


FIGURE 9 – Méthode pour traiter le déséquilibre de classe

• Chapitre 4 : Détection d’attaques basée sur la représentation des données en graphe

Bien que notre approche précédente ait permis d’améliorer la détection des classes minoritaires en mettant l’accent sur les caractéristiques discriminantes des attaques, elle reste confrontée à certaines limites inhérentes aux algorithmes classiques de classification. En effet, les modèles basés sur l’apprentissage supervisé, y compris ceux utilisant des architectures profondes comme les LSTM, dépendent fortement des données historiques et peinent à s’adapter aux nouvelles attaques inconnues. De plus, ils restent sensibles aux déséquilibres de classes et ne prennent pas toujours en compte les relations complexes entre les événements, ce qui peut limiter leur capacité à identifier des schémas d’intrusion plus subtils.

Dans le cadre du chapitre 4 de cette thèse, nous abordons ces limites en proposant une approche (figure 10) alternative reposant sur une représentation des données sous forme de graphes. Cette méthodologie permet de capturer efficacement les relations complexes entre les événements réseau, un aspect particulièrement pertinent dans le contexte des données IoT. Les dispositifs IoT génèrent un grand volume de données interconnectées, où la prise en compte des relations entre les événements est essentielle pour identifier des comportements anormaux.

Pour cela, nous utilisons l’algorithme K-Means afin de regrouper les événements similaires, segmentant ainsi les flux réseau en sous-groupes homogènes. Chaque sous-groupe est ensuite structuré en un graphe de proximité à l’aide de KNN et FAISS, où chaque nœud représente un événement ou une transaction et les arêtes indiquent les relations ou similarités entre ces événements. Enfin, nous exploitons les Graph Convolutional Networks (GCN) pour apprendre des représentations locales et globales des événements et de leurs connexions, permettant ainsi la classification des attaques.

Cette approche est particulièrement adaptée aux réseaux IoT, où les interactions entre appareils peuvent être complexes et difficiles à détecter avec des méthodes traditionnelles. Les résultats expérimentaux montrent que notre architecture améliore significativement la détection des intrusions, en identifiant plus efficacement les attaques subtiles et les comportements anormaux, grâce à la capacité des GCN à modéliser les relations topologiques entre les événements du réseau.

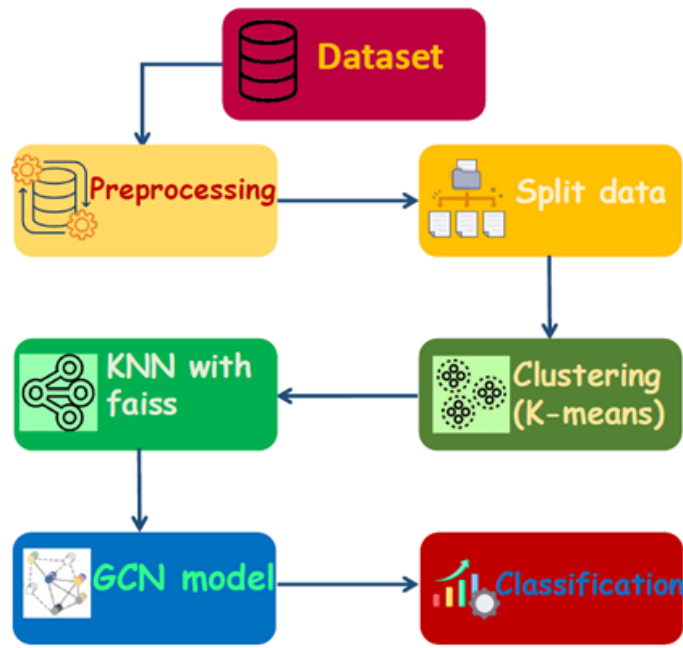


FIGURE 10 – Utilisation d'un Graph Convolutional Network (GCN) pour la classification des attaques

Enfin, nous présentons une conclusion sur le travail accompli dans cette thèse et proposons des perspectives pour de futures recherches et développements.

Le lecteur est informé que chaque chapitre de ce document a été rédigé de manière autonome. Chaque section intègre les concepts fondamentaux, la problématique abordée, les approches et modèles proposés, ainsi que les implémentations réalisées. Cette structuration vise à faciliter la lecture en rendant chaque chapitre compréhensible indépendamment des autres.

Utilisation des données historiques dans la détection des intrusions

La détection des intrusions dans les systèmes informatiques constitue un enjeu de taille en matière de cybersécurité, du fait de la sophistication croissante des attaques et de la mutation constante des tactiques des assaillants. L'un des problèmes essentiels réside dans le fait que les caractéristiques décrivant une connexion réseau ne tiennent pas compte fréquemment de l'historique des événements, alors que ce dernier est crucial pour repérer des schémas d'attaques récurrents ou progressifs. En outre, les modèles de classification classiques deviennent rapidement obsolètes face aux nouvelles tactiques d'intrusion s'ils ne sont pas régulièrement mis à jour, ce qui peut compromettre l'efficacité des systèmes de détection. Ce chapitre aborde ces défis en utilisant des méthodes d'apprentissage automatique spécifiquement conçues pour l'analyse des séquences d'événements. Plus précisément, nous faisons usage de réseaux de neurones récurrents LSTM, qui sont en mesure de saisir les dépendances temporelles et de représenter les liens entre les différentes attaques successives. Cette méthode permet d'approfondir la compréhension des mécanismes des intrusions et d'en améliorer la détection. De plus, afin d'améliorer l'efficacité du modèle et d'en augmenter l'interprétabilité, nous incorporons les valeurs SHAP (SHapley Additive exPlanations). Cette méthode permet de déterminer les caractéristiques les plus déterminantes dans le processus de prise de décision du modèle, ce qui facilite l'explication des prédictions et l'optimisation du modèle en se basant sur les attributs les plus pertinents. En associant la capacité des réseaux de neurones récurrents LSTM à saisir les dépendances temporelles et l'efficacité des valeurs SHAP pour accroître la compréhension et l'optimisation du modèle, notre méthode représente une solution prometteuse pour améliorer la détection des intrusions. Les résultats obtenus valident la pertinence de ce modèle, lequel pourrait être étendu à d'autres tâches supervisées faisant intervenir des données séquentielles.

1 Introduction

La fréquence des cyberattaques augmente au fur et à mesure que la technologie des réseaux évolue et que les services Internet se généralisent. Une généralisation qui favorise un accroissement substantiel du volume de données de trafic généré par le nombre croissant d'individus et d'appareils connecté à Internet.

A cet effet, les systèmes de détection d'intrusion basés sur les algorithmes de classification utilisés pour contrer les intrusions, doivent être en mesure de gérer efficacement une grande quantité de données et d'en extraire que les informations pertinentes et utiles pour la détection des intrusions.

Par ailleurs, la détection automatique des attaques, à l'aide des algorithmes de classification, est un défi à relever dans le domaine de machine Learning. Pour cela différentes modèles ont été proposées dans la littérature pour le résoudre [51] [52] [53] [54] [55]. Cependant, plusieurs facteurs rendent cette tâche complexe et doivent être considérées simultanément lors de la conception d'un système de détection d'intrusion. En premier lieu, les stratégies et les techniques des attaquants peuvent changer au fil du temps, ce qui rend une fonction de détection acquise par un classificateur dépassée si elle n'est pas fréquemment mise à jour. En second lieu, le système doit être apte de répondre dans des délais très brefs pour être efficace lors de scénarios réels. Troisièmement, les systèmes ne tiennent pas compte des données historiques des événements qui peuvent en principe constituer une référence très utile pour renforcer leurs capacités pour détecter des schémas d'attaques pertinentes.

En effet, une attaque n'est pas uniquement une caractéristique de l'activité réseau elle-même, mais dépend aussi du contexte spécifique dans lequel elle s'est produite, c'est-à-dire l'utilisateur et le système cible. Par conséquent, des événements similaires peuvent à la fois représenter une activité tout à fait légitime dans le cadre de certains utilisateurs ou des anomalies dans le contexte d'autres utilisateurs.

Pour établir un tel contexte qui prend en compte le comportement des utilisateurs dans la détection d'intrusion, il est essentiel de résumer l'historique des activités effectuées par les utilisateurs, afin de modéliser les corrélations séquentielles entre les événements réseau successifs. L'objectif est de restreindre l'intervention de l'expert dans l'élaboration manuelle des stratégies d'agrégation d'événements ainsi que de permettre à un système de détection d'intrusion de mieux repérer des comportements anormaux parmi les actions habituelles d'un utilisateur.

De ce fait, nous proposons, dans ce chapitre, une nouvelle approche qui s'appuie sur les informations historiques des événements pour construire automatiquement un contexte d'activité réseau par le biais des réseaux de neurones récurrents LSTM. Cette approche permet d'analyser des événements complexes et de détecter des menaces potentielles ou explicites d'intrusions sur les nouvelles connexions réseau. Les résultats obtenus démontrent que notre approche est efficace, précise et améliore la performance du modèle de détection d'intrusions.

Nos contributions dans ce chapitre sont les suivantes :

1. Optimiser le processus d'apprentissage des classifieurs en utilisant des algorithmes de sélection de caractéristiques tels que Anova (Analysis of Variance), Boruta et Shap Values (SHapley Additive exPlanations).

2. Concevoir un système de détection d'intrusions capable de construire un contexte d'activité réseau en utilisant LSTM, BiLSTM et Stacked LSTM comme classifieur séquentiel.

3. Tester les performances de notre modèle proposé sur l'ensemble de données de référence NSL-KDD (Network Security Lab-KDD dataset) qui présente un large éventail de scénarios d'attaque, et comparer les résultats obtenus avec les travaux antérieurs.

L'organisation de ce chapitre se présente comme suit. La section 2, nous exposons les notions fondamentales employées dans cette recherche. La section 3, nous présentons les détails de notre approche suggérée et examinons les résultats que nous avons obtenus. Pour finir, la section 4 clôture le document tout en suggérant des pistes pour de prochaines études.

2 Concepts de base

Cette section se concentre sur des notions plus spécialisées en cybersécurité, telles que la classification des attaques, les réseaux LSTM et les méthodes de sélection de caractéristiques. Ces concepts jouent un rôle clé dans la compréhension des choix méthodologiques réalisés et des améliorations apportées aux systèmes de détection d'intrusion.

2.1 Classification en cybersécurité

La classification est une tâche fréquente en apprentissage automatique et consiste à utiliser un modèle entraîné pour affecter une nouvelle entrée à l'une des diverses classes ou catégories prédéfinies. Dans le domaine de la sécurité informatique, la classification est utilisée pour détecter et classer les différentes attaques et activités malveillantes des réseaux ou des systèmes informatiques, et les catégories correspondent aux types d'attaques ou d'activités malveillantes. Les modèles de classification sont entraînés sur des ensembles de données annotés, qui sont des données où chaque entrée est associée à la catégorie appropriée.

Dans le contexte de la cybersécurité, les modèles de classification sont utilisés pour identifier et catégoriser les attaques en fonction des attributs extraits des données du réseau ou du système. Les attaques sont considérées comme un problème de classification car elles présentent généralement des schémas distinctifs qui peuvent être reconnus et différenciés par les modèles d'apprentissage automatique. Cette technique permet ainsi de détecter les attaques et de prendre des mesures appropriées pour y remédier.

Il existe diverses catégories de classification en apprentissage automatique, et le type de classification opté dépend du problème spécifique et des propriétés des données. En cybersécurité, on distingue fréquemment deux formes de classification : binaire et multiclasse (figure 1.1).

- **Classification binaire** : est employée lorsque l'objectif est de prédire l'une des deux classes ou catégories possibles. Dans le domaine de la cybersécurité, la classification binaire est fréquemment employée pour déterminer si une activité est malveillante ou non. Par exemple, un modèle de classification binaire peut être entraîné pour détecter si un fichier est malveillant ou non, ou si un comportement de réseau est une intrusion ou une activité normale.
- **Classification multiclasse** : cette technique est employée lorsque le but est de prédire l'une des multiples classes ou catégories. En cybersécurité, la classification multiclasse est utilisée pour déterminer le type particulier d'une attaque ou d'une activité malveillante parmi diverses catégories possibles. Par exemple, un modèle de classification multiclasse peut être entraîné pour catégoriser les attaques réseau en différentes classes telles que les attaques de DoS, les attaques R2L ou les attaques par injection de code.

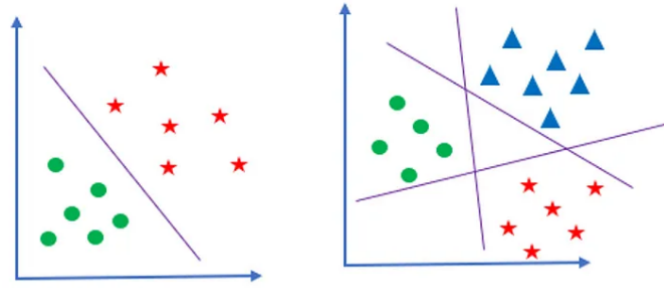


FIGURE 1.1 – *classification binaire et multiclass* [56]

2.2 Les réseaux de neurones

Les êtres humains possèdent une capacité incroyable à apprendre de nouvelles tâches simplement en observant des exemples, sans être précisément programmés pour cette tâche. Dans le cerveau humain ce genre d'apprentissage est accordé à l'interaction entre des millions de neurones. Chaque neurone possède un corps cellulaire ainsi que de nombreux dendrites et axones ramifiés, qui servent à véhiculer les signaux entre neurones. Lorsqu'un neurone reçoit un influx via ses dendrites, Il l'intègre, et si l'intensité de cette entrée atteint un seuil critique, il génère à son tour un potentiel d'action qu'il propage le long de son axone vers les neurones voisins. C'est par ce mécanisme de transmission synaptique que le cerveau traite et conserve l'information.

Le réseau neuronal humain constitue un système incroyablement complexe et hautement interconnecté, composé de milliards de neurones et d'autant de connexions. Il constitue le socle de nos pensées, de nos actions et de nos expériences, nous permettant de percevoir, d'apprendre et de nous adapter à notre environnement.

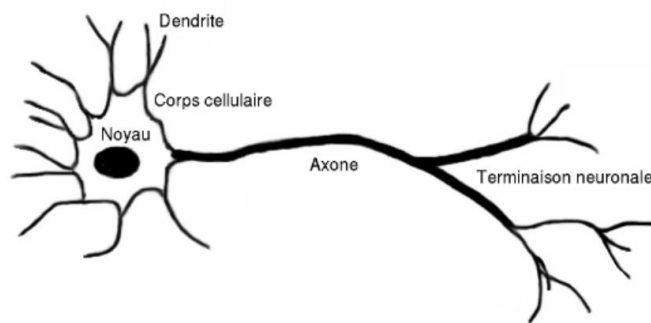


FIGURE 1.2 – *Réseaux de neurones humain*

2.3 Les réseaux de neurones artificiels

Un réseau de neurones artificiels (ANN) est un modèle informatique inspiré du fonctionnement et de la structure des neurones présents dans le cerveau humain. Il s'agit d'un type d'algorithme d'apprentissage automatique conçu pour simuler le traitement de l'information tel qu'il s'opère dans le cerveau humain [57].

Un réseau de neurones artificiels se compose de plusieurs couches de nœuds interconnectés, chacun capable de détecter et d'apprendre des motifs complexes à partir des données d'entrée.

Ces nœuds sont organisés en couches successives : la couche d'entrée reçoit les données brutes, les couches cachées réalisent des transformations progressives, et la couche de sortie fournit le résultat final. Chaque connexion entre nœuds porte un poids, ajusté pendant la phase d'apprentissage en fonction des données d'entrée et des sorties attendues, ce qui permet au réseau de modéliser précisément la relation entre entrée et sortie. La figure 1.3 illustre la configuration d'un réseau de neurones en mettant en évidence les diverses couches qui le composent.

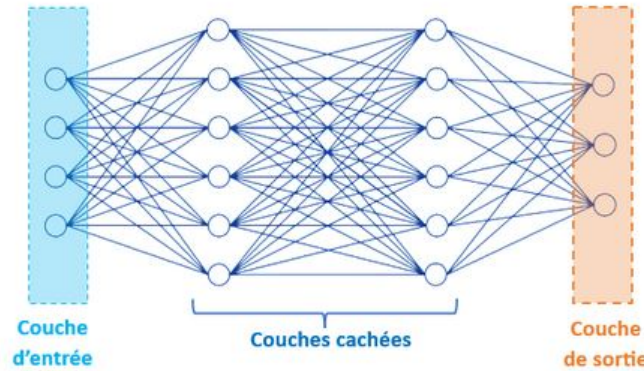


FIGURE 1.3 – Architecture d'un réseau de neurones artificiels

Le réseau de neurones se forme en produisant une prédiction pour chaque ensemble d'entrées et en ajustant les poids dans chaque nœud jusqu'à atteindre la prédiction correcte. Ce processus est effectué de manière itérative jusqu'à l'obtention d'une performance satisfaisante du modèle. Dans ce réseau, chaque nœud du réseau (perceptron), apprend en obtenant des entrées et en attribuant des poids à chacune de ces entrées. Par la suite, une fonction d'activation est mise en œuvre et le résultat est transmis au perceptron suivant. Le processus d'apprentissage pour chaque perceptron est illustré dans la figure 1.4.

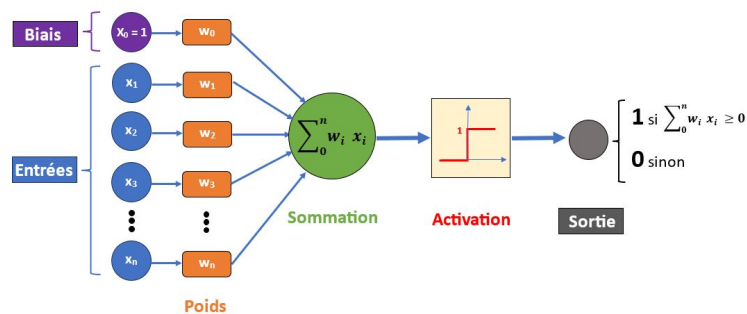


FIGURE 1.4 – Structure d'un réseau de neurones

Au départ, les poids sont initialisés de façon aléatoire, ce qui engendre des prédictions imprécises. Le réseau apprend ensuite par itérations : on lui présente des exemples dont la sortie est connue, on compare la prédiction à la valeur réelle, on calcule l'erreur et on ajuste les poids pour la réduire. Ce mécanisme, appelé rétropropagation (back-propagation), est crucial pour optimiser le modèle. Au fur et à mesure que l'entraînement se fait, le réseau affine ses prédictions sur les données d'entraînement et devient capable, une fois entraîné, de généraliser ses connaissances pour estimer des sorties sur de nouvelles données inconnues.

Lors de la conception d'un réseau de neurones, il faut prendre en compte divers paramètres tels que le taux d'apprentissage (learning rate), le type de fonction d'activation et l'optimiseur.

Le paramètre de taux d'apprentissage est utilisé pour contrôler la vitesse avec laquelle les poids du réseau sont ajustés. Durant le processus de rétropropagation, les poids sont rapidement modifiés en fonction de l'erreur déterminée. Lorsque la variation de poids est considérable, cela peut entraîner des bonds plus significatifs et conduire à une incapacité à identifier les minima globaux. Il est donc essentiel de maintenir un taux d'apprentissage adéquat.

La fonction d'activation est utilisée pour introduire la non-linéarité dans le réseau, ce qui permet au réseau d'apprendre et de représenter des relations plus complexes entre les données d'entrée et de sortie. Sans fonctions d'activation, un réseau de neurones serait simplement un modèle de régression linéaire, qui n'est pas capable d'apprendre des modèles complexes dans les données. Il existe plusieurs fonctions d'activation couramment utilisées en apprentissage automatique telles que Sigmoid, Tanh, softmax, ReLU, etc.

Les optimiseurs sont des algorithmes utilisés pour ajuster les paramètres d'un modèle afin de minimiser l'erreur du modèle. Ils sont utilisés pour trouver les valeurs des paramètres du modèle qui conduisent aux meilleures performances.

2.4 Les Réseaux de Neurones Récurrents

Les réseaux de neurones récurrents (RNN) sont particulièrement adaptés aux problèmes impliquant des données séquentielles ou temporelles, ce qui les rend utiles pour la détection d'intrusion lorsque l'analyse des séquences de paquets réseau est nécessaire [58] [59].

Les RNN distinguent des réseaux de neurones à propagation avant (feed-forward) par leur capacité à traiter les dépendances temporelles et à prendre en compte l'information contextuelle. Les réseaux de neurones récurrents maintiennent un état interne qui leur permet de se souvenir des informations précédentes et de les utiliser pour traiter les données en cours. Cette propriété les rend particulièrement adaptés aux tâches où la prise en compte de l'ordre des données est cruciale, comme dans le cas de la détection d'intrusion basée sur l'analyse des séquences de paquets réseau.

Dans un RNN, les neurones sont connectés non seulement aux couches précédentes et suivantes, mais aussi à eux-mêmes, créant ainsi des boucles de rétroaction. Cette connexion récurrente permet de capturer les dépendances temporelles entre les éléments d'une séquence (voir figure 1.5).

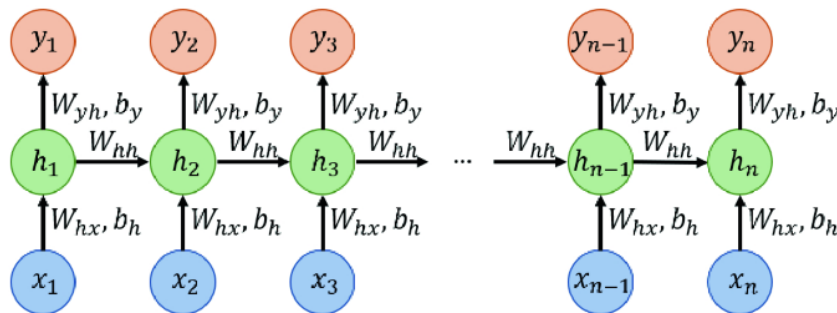


FIGURE 1.5 – Structure d'un réseau de neurones récurrent déroulé dans le temps, où une copie du modèle est générée pour chaque étape temporelle

Mathématiquement, un RNN simple peut être décrit par les équations suivantes :

$$h_t = f(W_{hh}h_{t-1} + W_{hx}x_t + b_h) \quad (1.1)$$

$$y_t = W_{hy}h_t + b_y \quad (1.2)$$

Avec x_t est l'entrée à l'instant t , h_t est l'état caché à l'instant t , y_t est la sortie à l'instant t , W_{xh} , W_{hh} et W_{hy} sont les matrices de poids, et b_h et b_y sont les vecteurs de biais. La fonction f est une fonction d'activation non-linéaire, telle que la tangente hyperbolique.

Les RNN peuvent être déroulés dans le temps, de sorte que chaque état caché est une fonction des entrées précédentes temporelles sur une fenêtre de temps variable. Cependant, les RNN classiques rencontrent des difficultés pour capturer les dépendances à long terme en raison de problèmes de disparition et d'explosion du gradient lors de l'apprentissage [60].

La mise en place de cellules de mémoire permet de surmonter le problème de la disparition et de l'explosion du gradient. La structure de réseau qui comprend des cellules de mémoire est appelée réseau de mémoire à long terme (LSTM) [61]. Les LSTM se sont révélés capables d'apprendre des dépendances à long terme dans des séquences avec un grand succès pour des tâches séquentielles telles que la reconnaissance vocale et la traduction automatique [62] [63].

2.5 Les Réseaux LSTM

Les réseaux de neurones récurrents à mémoire court-terme et long terme (LSTM) représentent une variante des réseaux de neurones récurrents RNN qui ont été proposés par Hochreiter et Schmidhuber en 1997 [64]. Les LSTM sont particulièrement élaborés pour traiter les problèmes de dépendances à long terme présentes dans les séquences de données, qui sont souvent difficiles à appréhender pour les RNN conventionnels en raison du problème du gradient qui s'évanouit ou explose [65]. Ils sont spécialement conçus pour résoudre des problèmes dans des domaines tels que la traduction automatique et la reconnaissance vocale.

2.5.1 Structure des réseaux LSTM

Une cellule LSTM se compose de trois portes (gates) et d'une cellule mémoire. Les portes sont des unités de réseaux de neurones qui contrôlent le flux de l'information à travers la cellule LSTM. Les trois portes fondamentales dans une cellule LSTM sont :

- La porte d'entrée (input gate).
- La porte d'oubli (forget gate).
- La porte de sortie (output gate).

En plus de ses trois portes, chaque cellule LSTM contient une cellule mémoire chargée de stocker son état interne (voir figure 1.6). Cette mémoire permet aux cellules LSTM de capturer et de maintenir informations pertinentes sur de longues séquences de données.

2.5.2 Fonctionnement des réseaux LSTM

Pour saisir le fonctionnement des réseaux LSTM, examinons à chaque instant t le cheminement de l'information à travers les portes et la cellule mémoire. En notant x_t l'entrée au temps t , h_{t-1} l'état caché précédent et c_{t-1} l'état mémoire antérieur, les opérations réalisées sont les suivantes :

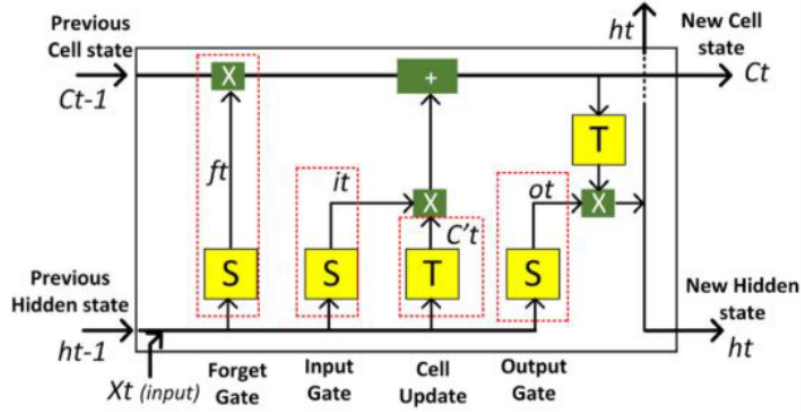


FIGURE 1.6 – Cellule LSTM.

1. Calcul de la porte d'oubli f_t :

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (1.3)$$

Avec : σ : fonction sigmoïde et W_f : matrice des poids associé à la porte d'oubli.

Cette porte utilise une fonction sigmoïde pour déterminer quelle informations de la cellule mémoire doivent être conservées ou oubliées à chaque étape temporelle. Elle prend en entrée l'état caché précédent h_{t-1} ainsi que l'entrée x_t (à l'étape temporelle t) et délivre un nombre entre 0 et 1 pour chaque élément du vecteur mémoire c_{t-1} .

2. Calcul de la porte d'entrée i_t :

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (1.4)$$

Avec : W_i : matrice des poids associé à la porte d'entrée et b_i : biaie associé à la porte d'entrée.

La porte d'entrée est responsable de déterminer si de nouvelles valeurs obtenues à partir des données entrantes peuvent être autorisées à passer. Cette décision est prise à l'aide de deux fonctions : "sigmoid" et "tanh". La fonction "sigmoid" attribue une valeur de zéro pour restreindre la modification, tandis qu'une valeur de 1 permet la modification. D'autre part, la fonction "tanh" génère un nouveau vecteur de valeurs qui sera ajouté à l'état de la cellule.

Ensuite, le calcul du candidat pour l'état de la cellule mémoire c_t .

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (1.5)$$

Avec : W_c est la matrice des poids associé à la cellule memoire et b_c est le biais associé à la cellule mémoire.

3. Mise à jour de le cellule mémoire c_t : une fois les sorties des portes calculées, nous pouvons utilise ces sorties pour gérer la cellule mémoire courante.

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (1.6)$$

Avec $\odot$ désigne la multiplication élément par élément.

4. Calcule de la porte de sortie o_t :

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (1.7)$$

Avec W_o est la matrice des poids associé à la porte de sortie et b_o est le biais associé à la porte de sortie.

Cette porte de sortie est chargée de déterminer les informations spécifiques à extraire de l'état de la cellule. Avant d'utiliser la fonction "sigmoid", la sortie est d'abord soumise à la fonction "tanh" afin de garantir que les valeurs se situent dans l'intervalle de -1 à +1.

5. Calcule de l'état caché h_t :

$$h_t = o_t \odot \tanh(c_t) \quad (1.8)$$

Ces calculs offrent aux LSTM la capacité de gérer efficacement les dépendances à long terme au sein des séquences de données. Les LSTM, grâce à la porte d'oubli et à la porte d'entrée, ont la capacité de décider quelles information conserver ou écarter à chaque étape temporelle. La cellule mémoire stocke l'état interne de la cellule, permettant ainsi de retenir des données sur de longues périodes. Enfin, la porte de sortie contrôle l'information qui est transmise au reste du réseau [66].

2.6 Les réseaux de neurones récurrents LSTM Bidirectionnelle

Depuis leur introduction, plusieurs variantes des LSTM ont été développées afin d'améliorer leur efficacité et leurs performances. Parmi les plus populaires, on mentionne le BiLSTM, que nous avons également utilisé dans notre recherche pour le comparer au LSTM.

LSTM Bidirectionnel constitué de deux LSTM dont les sorties sont empilées ensemble. L'une traite la séquence dans le sens avant et l'autre dans le sens inverse [67], techniquement, BiLSTM applique deux unités LSTM distinctes, une pour la direction avant et une pour la direction arrière. Deux états cachés h_t vers l'avant et h_t vers l'arrière de ces unités LSTM sont concaténés en un état caché final.

$$H_t^{bilstm} = h_t^{forward} \oplus h_t^{backward} \quad (1.9)$$

Avec $\oplus$ est l'opérateur de concaténation.

2.7 L'ingénierie des features

L'ingénierie des fonctionnalités est étape importante du traitement des données, consiste à transformer des données brutes en caractéristiques exploitables pour les algorithmes d'apprentissage automatique. Ce processus implique généralement la sélection, l'extraction et la transformation des attributs les plus significatifs afin d'optimiser la précision et l'efficacité des modèles [74].

Il existe plusieurs types d'ingénierie des features, parmi lesquels :

- Sélection : consiste à choisir un sous-ensemble de caractéristiques significatives à partir d'un ensemble de données [68]. Cela peut être effectué manuellement en se basant sur la connaissance du domaine ou le jugement d'un expert, ou de façon automatique grâce à des algorithmes tels que la sélection de caractéristiques ou la réduction de la dimensionnalité ;
- Création : cela implique de créer de nouvelles caractéristiques à partir des données d'origine, telles que l'extraction de features ou la combinaison (ex : somme, ratios), afin de les rendre plus significatives ou prédictives ;
- Transformation : consiste à transformer les caractéristiques d'origine, comme la normalisation ou la mise à l'échelle, pour les rendre plus ajustée à une application dans le modèle.

Dans notre étude, les méthodes de sélection de caractéristiques Anova, Boruta et shap Values seront utilisées pour améliorer la qualité des modèles en identifiant les fonctionnalités les plus importantes et en optimisant la performance du modèle.

2.7.1 Shap Values

Les valeurs Shap (Shapley Additive exPlanations) sont une technique pour interpréter les prédictions des modèles d'apprentissage automatique. Elaboré par Lundberg et Lee dans leur publication [69]. Les valeurs Shap s'appuient sur la théorie des jeux coopératifs offrant une évaluation de l'importance de chaque attribut pour une prédiction donnée [70].

La formule de base pour calculer les valeurs Shap pour un jeu coopératif est la suivante :

$$\phi_i(v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N|-|S|-1)!}{|N|!} [v(S \cup \{i\}) - v(S)] \quad (1.10)$$

Avec N représente l'ensemble des joueurs (dans notre contexte les caractéristiques). S constitue un sous-ensemble de joueurs, v représente une fonction spécifique qui assigne une valeur à chaque sous-ensemble de joueurs, et $\phi_i(v)$ est la valeur Shap attribuée à la caractéristique i . Cette expression détermine la contribution marginale moyenne de la caractéristique i à toutes les combinaisons de caractéristiques possible.

Dans le domaine de l'apprentissage automatique, les valeurs Shap offrent une mesure de l'importance de chaque caractéristique pour une prédiction donnée. En attribuant un score Shap à chaque attribut pour chaque prédiction, on peut comprendre la manière dont chaque attribut contribue, de façon positive ou négative, à la prédiction. Cela peut être bénéfique pour le choix des caractéristiques les plus importantes pour la prédiction du modèle [71].

De plus, les valeurs Shap possèdent une caractéristique d'additivité, indiquant que la somme des valeurs Shap pour tous les attributs correspond à la différence entre la prédiction du modèle pour une instance particulière et la prédiction moyenne du modèle pour l'ensemble des instances. Ceci permet de répartir la contribution de la prédiction en contributions attribuables à chaque caractéristique.

2.7.2 Anova f-test (Analyse de la variance)

L'ANOVA (Analyse de la Variance) est une approche de sélection de caractéristiques statistiques basée sur le score f_1 , permettant de comparer les moyennes de deux ou plusieurs groupes distincts afin de vérifier si elles sont statistiquement différentes les uns des autres. La valeur F de l'ANOVA représente la relation entre chaque caractéristique et le vecteur cible. Les scores de la valeur F analysent si les moyennes de chaque groupe sont considérablement différentes lorsque l'on classe les caractéristiques numériques en fonction du vecteur cible [76].

2.7.3 Boruta

La technique de sélection des caractéristiques Boruta est une méthode robuste basée sur l'utilisation d'un classificateur de forêt aléatoire [72]. Elle opère en comparant la pertinence des variables réelles avec celle des attributs « fantômes » (shadow attributes) pour identifier celles qui sont véritablement significatives pour la prédiction. Cet algorithme suit un processus structuré d'étapes, qui sont minutieusement décrites ci-dessous [73] [75] :

- La matrice des caractéristiques est élargie en intégrant des attributs fantômes (shadow attributes), qui représentent des duplications des attributs initiaux, mais avec leurs valeurs disposées de façon aléatoire. Si le nombre d'attributs initiaux est moins de 5, au moins cinq attributs fantômes seront générés. Soit $M = x_1, x_2, \dots, x_p$ la matrice des caractéristiques originales et $MS = x_1^{shadow}, x_2^{shadow}, \dots, x_p^{shadow}$ la matrice de caractéristiques fantômes ;
- Les valeurs de chaque attribut fantôme x_i^{shadow} sont mélangées de façon aléatoire pour rompre toute corrélation avec la variable cible ;
- Un modèle de forêt aléatoire est formé en utilisant toutes les caractéristiques réelles et fantômes, c'est-à-dire $M \cup MS$;
- L'importance des attributs est mesurée en fonction de la perte d'impureté (telle que l'impureté de Gini ou l'entropie), et les Z-scores des réductions d'impuretés sont déterminés pour chaque attribut. Ces scores indiquent l'importance de chaque variable dans le processus de classification.

$$Z_i = \frac{I(x_i) - \mu_{shadow}}{\epsilon_{shadow}} \quad (1.11)$$

Avec : $I(x_i)$ est l'importance du caractéristique x_i μ_{shadow} est la moyenne des importances des caractéristiques fantôme ϵ_{shadow} est l'écart type des importances des attributs fantômes.

- Le maximum des Z-scores parmi les caractéristiques fantômes est identifié :

$$MZSA = \text{Max} \{Z_i^{shadow} | x_i^{shadow} \in X_{shadow}\} \quad (1.12)$$

- Un « hit » est attribué à chaque caractéristique réelle dont la signification (Z-score) dépasse le MZSA. Si $Z(x_i) > MZSA$, alors l'attribut x_i est « important ».

- Pour les caractéristiques dont le Z-score est inférieur au MZSA, une analyse statistique (telle qu'un test t bilatéral ou un test de Wilcoxon) est réalisée afin de vérifier si leur importance diffère significativement de celle des attributs fantômes.

$$H_0 : \mu_{x_i} = \mu_{shadow} , H_1 : \mu_{x_i} \neq \mu_{shadow}$$

Si la p-value du test est inférieure à un seuil prédéfini (par exemple, $\alpha = 0.03$), donc l'attribut diffère est considéré significativement différent des attributs fantômes.

- Les caractéristiques dont l'importance est nettement inférieure à celle des attributs fantômes sont jugés insignifiantes et retirés de la matrice des caractéristiques.
Si $Z(x_i) < MZSA$, alors x_i est supprimé.
- Dans la matrice finale des caractéristiques, les attributs dont l'importance dépasse nettement celle des attributs fantômes sont jugés importants et retenus.
Si $Z(x_i) > MZSA$, alors x_i est retenu comme important.

Ce processus est répété plusieurs fois (sur plusieurs itérations ou jusqu'à l'obtention de la convergence). À chaque itération, les caractéristiques considérées comme non importants sont éliminées et la procédure se poursuit jusqu'à ce que toutes les caractéristiques soient identifiées comme importants ou non importants.

3 Approche proposée

Dans cette section, nous décrivons le modèle, basé sur l'architecture LSTM, que nous proposons pour la détection des attaques. Notre architecture utilise en premier lieu des techniques de pré-traitement des données qui est une étape cruciale pour obtenir des résultats fiables et améliorer les performances des algorithmes d'apprentissage automatique. Ensuite, la sélection des caractéristiques qui consiste à réduire le nombre des attributs d'entrée avant de les introduire dans le modèle. Enfin, les modèles à apprentissage séquentiel LSTM, BiLSTM et Stacked LSTM, sont utilisés comme classificateurs responsables de l'identification dynamique de la dépendance séquentielle entre les événements réseaux consécutives. L'architecture du système proposé est illustrée dans la figure 1.7 :

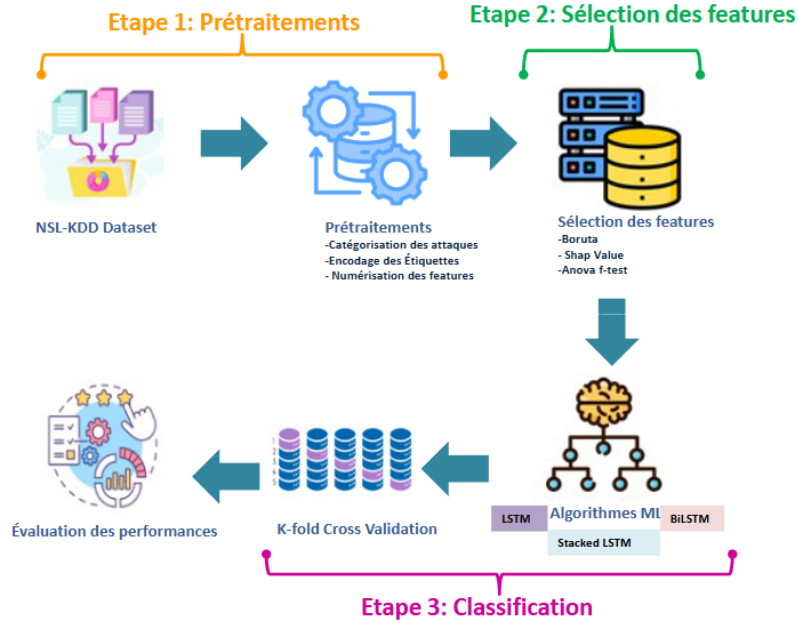


FIGURE 1.7 – L'architecture du modèle proposé pour la détection d'intrusion

Les modèles à apprentissage séquentiel, tels que LSTM, BiLSTM et Stacked LSTM, sont utilisés comme classificateurs.

3.1 Description du dataset

Le dataset NSL-KDD est une version améliorée et contemporaine de l'ensemble de données KDD-Cup 99 [77]. Dans le domaine de la détection d'intrusion, ce dataset fait partie des ensembles de données les plus utilisés, notamment parce qu'il ne contient ni enregistrements redondants ni doublons [78]. De plus, l'analyse menée par [79] démontre que le dataset NSL-KDD est particulièrement adapté pour la comparaison de différents modèles de détection d'intrusion. L'ensemble de données NSL-KDD original comprenait un total de 1 074 992 enregistrements, dont 812 814 étaient classés comme normaux et 262 178 comme des enregistrements d'attaques.

Par ailleurs, NSL-KDD comprend un total de 43 caractéristiques, réparties en quatre groupes distincts :

- Catégorielles : caractéristiques 2, 3, 4 et 42,
- Binaires : caractéristiques 7, 12, 14, 20, 21 et 22,
- Discrètes : caractéristiques 8, 9, 15, et de 23 à 41, ainsi que 43,
- Continues : caractéristiques 1, 5, 6, 10, 11, 13, 16, 17, 18 et 19.

TABLE 1.1 – Classes d'attaques et leurs types dans le dataset NSL-KDD

Classe d'Attaque	Types d'Attaques
DoS	Back, Neptune, Land, Pod, Smurf, Udpstorm, Teardrop, Apache2, Worm, Processtable
Probe	Nmap, Ipsweep, Portsweep, Saint, Mscan, Satan
R2L	Ftp_write, Warezmaster, Httpunnel, Guess_Password, Phf, Warez-client, Snmpguess, Multihop, Xlock, Spy, Xsnoop, Sendmail, Snmp-getattack, Imap, Named
U2R	Rootkit, Loadmodule, Xterm, Sqlattack, Perl, PS, Buffer_overflow

Le dataset NSL-KDD regroupe plusieurs types d'attaques, chacun subdivisé en différentes sous-classes (voir tableau 1.1). Les quatre principales classes d'attaques sont les suivantes :

- DoS : Les attaques de type DoS (Denial of Service) sont conçues pour épuiser les ressources du système cible, dans le but de mettre hors service une machine ou un réseau et de le rendre inaccessible à ses utilisateurs légitimes.
- Probe : Les attaques de sondage sont conçues pour recueillir des informations sur le système cible, en vue d'identifier ses vulnérabilités ou points faibles.
- R2L : Les attaques de type R2L (Remote to Local) sont conçues pour donner un accès local au système cible. Elles sont donc particulièrement dangereuses, car elles permettent à un attaquant de contourner les protections et d'obtenir un contrôle local sur la machine.
- U2R : Les attaques de type U2R (User to Root) permettent à un utilisateur normal d'obtenir un accès root. L'attaquant commence par accéder au compte d'utilisateur, puis exploite les vulnérabilités du système pour obtenir les privilèges root. Étant donné que l'accès root permet un contrôle total sur le système, les attaques U2R sont considérées comme les plus dangereuses parmi toutes les attaques de ce dataset.

3.2 Pré-traitement des données

Le prétraitement des données désigne le processus de préparation de données brutes afin de les rendre adaptées à l'analyse et à l'entraînement du modèle. Ce processus englobe une variété de méthodologies, telles que le nettoyage des données, la transformation des données et la sélection des caractéristiques :

3.2.1 Catégorisation des attaques

L'ensemble de données NSL-KDD contient plusieurs sous-classes d'attaques, qui ont été regroupées en quatre catégories distinctes, comme illustré dans le tableau 1.1.

La figure 1.8 montre la répartition des différents types d'enregistrements dans le jeu de données NSL-KDD après la catégorisation des attaques. Les données normales sont les plus représentées avec 97 278 échantillons, suivies des attaques DoS (38 834) et Probe (41 102). Tandis que, les attaques de type R2L (1 126) sont nettement moins fréquentes, et les attaques U2R (22) sont extrêmement rares.

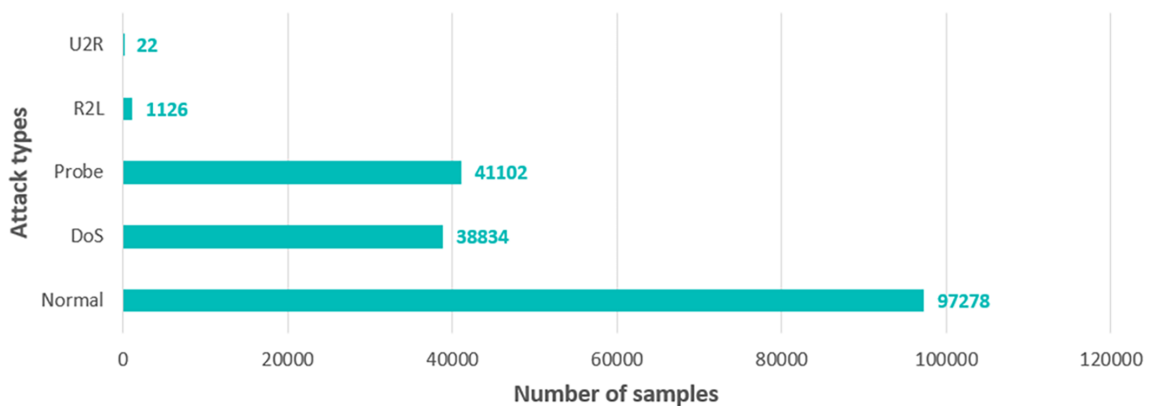


FIGURE 1.8 – Les nombres d'enregistrements d'attaques et d'enregistrements normaux dans le dataset NSL-KDD

3.2.2 Encodage des étiquettes et numérisation des caractéristiques

Au cours de cette phase, nous avons procédé à un encodage des étiquettes pour chaque type d'attaque. Une représentation numérique a été attribuée à chaque catégorie : 0 pour la catégorie "Normal", 1 pour "DoS", 2 pour "Probe", 3 pour "R2L" et 4 pour "U2R".

De plus, de nombreux algorithmes d'apprentissage automatique ne prennent pas en charge les données catégoriques, ce qui nécessite leur conversion en format numérique avant leur utilisation dans les modèles. Par conséquent, les attributs non numériques "protocol-type", "service" et "flag" ont été convertis en représentations numériques à l'aide de la fonction `factorize()` de `pandas`. Cela a permis de représenter les variables catégoriques à l'aide d'étiquettes numériques appropriées, compatibles avec les méthodes d'apprentissage automatique.

3.2.3 Normalisation

La standardisation, également appelée normalisation, est une méthode statistique utilisée pour mettre les données à l'échelle en les centrant autour de la moyenne et en ajustant leur écart type à 1. Le processus de standardisation revêt une importance capitale en raison des caractéristiques disparates présentes dans l'ensemble de données, telles que le temps, les octets source et les octets de destination, qui peuvent présenter des variations significatives entre leurs valeurs maximales et minimales respectives. Pour ce faire, l'algorithme `StandardScaler` [119] est utilisé pour calculer les valeurs standardisées.

$$z = \frac{x_i - \text{mean}(x)}{\text{stdev}(x)} \quad (1.13)$$

Où $\text{mean}(x)$ représente la moyenne de l'échantillon d'entraînement, $\text{stdev}(x)$ représente l'écart type de cet échantillon, et x_i représente la valeur spécifique de la caractéristique à standardiser.

3.3 La sélection des caractéristiques

Comme indiqué ci-dessus, nous utilisons les méthodes `Shap Values`, `Boruta` et `Anova f-test` pour la sélection des caractéristiques comme première étape d'exploration de nos datasets. Le but est d'étudier l'influence de chaque attribut dans la prédiction de la classe cible et de sélectionner le sous-ensemble de caractéristiques pertinentes en éliminant les caractéristiques non importantes.

3.3.1 Shap Values

La valeur SHAP a été utilisée pour éliminer les caractéristiques non pertinentes du jeu de données. Les caractéristiques du jeu NSL-KDD ayant les valeurs SHAP les plus élevées sont : `nsrc bytes`, `dst bytes`, `service`, `dst host diff srv rate`, `dst host serror rate`, `dst host rerror rate`, `dst host same src port rate`, `count`, `hot` et `duration` (figure 1.9).

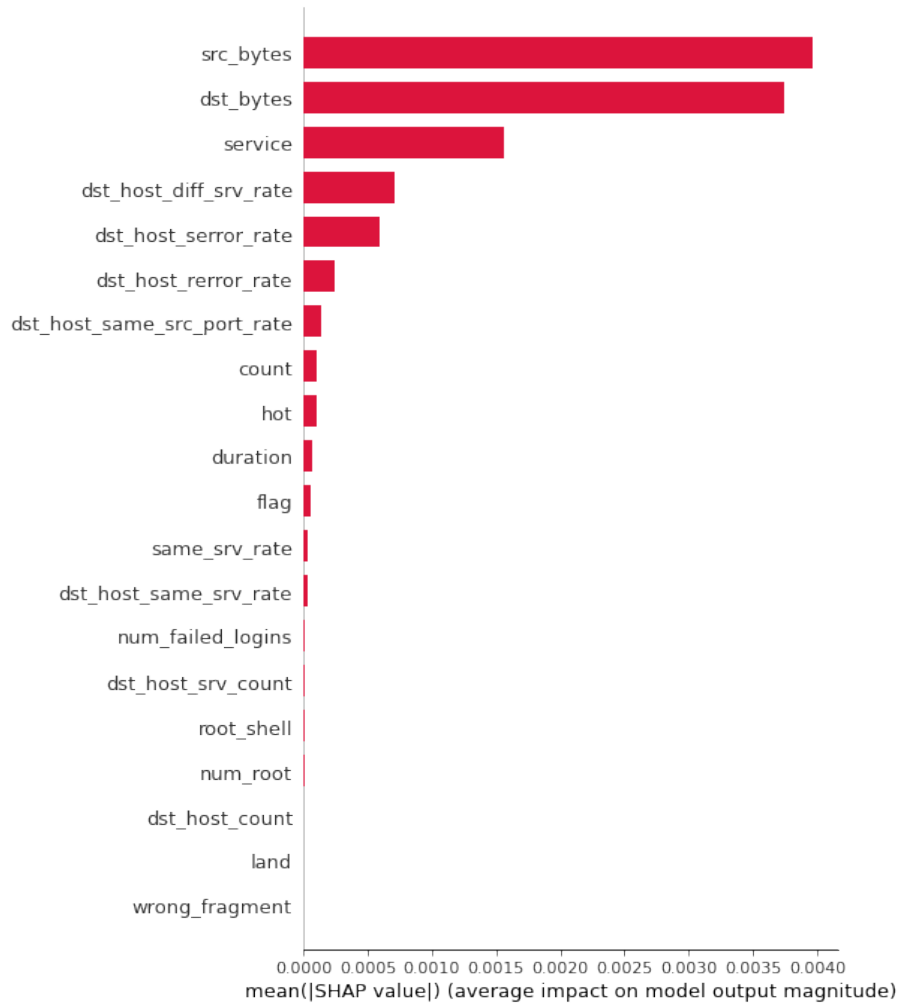


FIGURE 1.9 – La valeur SHAP indique l'importance des fonctionnalités dans NSL-KDD

3.3.2 Boruta

La méthode Boruta est une extension de la méthode de classification par forêts aléatoires. Elle cherche à inclure toutes les caractéristiques significatives qui peuvent exister dans le jeu de données par rapport à la variable de sortie.

Le tableau 1.2 présente les caractéristiques conservées après application de l'algorithme Boruta.

TABLE 1.2 – Les attributs sélectionnés par la méthode Boruta

Méthode	Caractéristiques sélectionnées
Boruta	duration, protocol_type, service, flag, src_bytes, dst_bytes, hot, num_failed_logins, logged_in, root_shell

3.3.3 Anova f-test

Le test ANOVA f-test est une technique de sélection de caractéristiques qui effectue un test statistique entre chaque caractéristique et la variable cible. Le test ANOVA f-test appliqué au jeu de données NSL-KDD est présenté dans la figure 1.10.

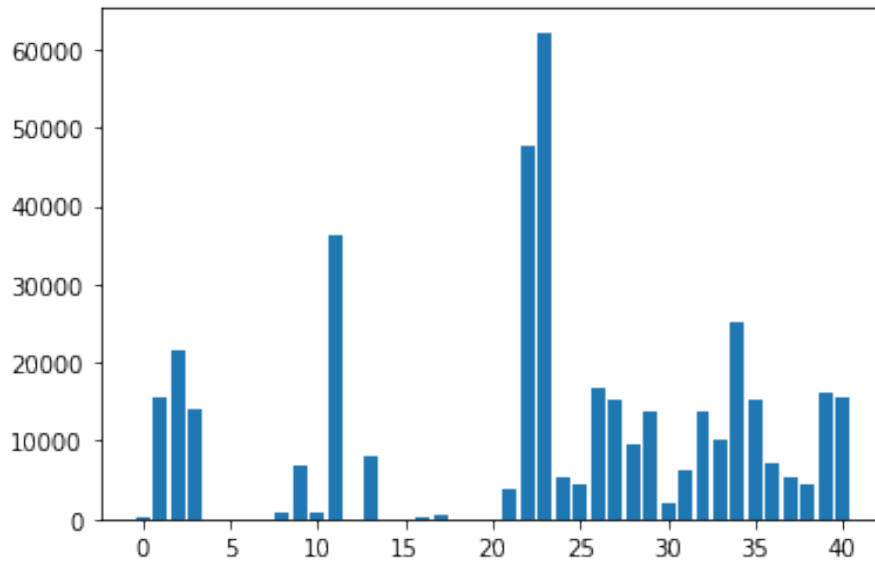


FIGURE 1.10 – Score d'Anova f-test

3.3.4 Discussion des résultats de sélection des features

Les trois méthodes de sélection des caractéristiques appliquées permis d'identifier des ensembles de caractéristiques pertinentes pour la détection d'intrusions (voir tableau 1.3). Certaines features se retrouvent systématiquement dans les résultats de chaque méthode, telles que `src_bytes`, `dst_bytes`, `service` et `duration`. Cette récurrence suggère qu'elles jouent un rôle clé dans la capture du trafic réseau, constituant ainsi un socle indispensable pour le modèle.

TABLE 1.3 – Les caractéristiques sélectionnées par différentes méthodes

Méthodes	Caractéristiques sélectionnées
Shap Values	<code>src_bytes</code> , <code>dst_bytes</code> , <code>service</code> , <code>dst_host_diff_srv_rate</code> , <code>dst_host_serror_rate</code> , <code>dst_host_rerror_rate</code> , <code>dst_host_same_src_port_rate</code> , <code>count</code> , <code>hot</code> , <code>duration</code>
Boruta	<code>duration</code> , <code>protocol_type</code> , <code>service</code> , <code>flag</code> , <code>src_bytes</code> , <code>dst_bytes</code> , <code>hot</code> , <code>num_failed_logins</code> , <code>logged_in</code> , <code>root_shell</code>
Anova f-test	<code>protocol_type</code> , <code>service</code> , <code>count</code> , <code>srv_count</code> , <code>rerror_rate</code> , <code>dst_host_rerror_rate</code> , <code>dst_host_diff_srv_rate</code> , <code>dst_host_same_src_port_rate</code> , <code>dst_host_rerror_rate</code> , <code>dst_host_srv_rerror_rate</code>

Cependant, chaque méthode met en lumière des aspects différents du jeu de données :

- **SHAP Values** : méthode basée sur l'attribution de scores d'importance à chaque variable selon leur impact sur la prédiction. Elle met en avant des métriques d'erreur de l'hôte distant, notamment `dst_host_diff_srv_rate` et `dst_host_serror_rate`, soulignant leur importance dans les décisions du modèle.
- **Boruta** : algorithme wrapper qui compare les variables originales à des «shadow features» générées aléatoirement. Il retient, en plus des quatre variables communes, des indicateurs système et comportementaux tels que `num_failed_logins` et `root_shell`, essentiels pour repérer les activités malveillantes hors du simple volume de trafic.

- Anova F-test : filtre univarié fondé sur la différence de moyennes entre classes. Il conserve notamment les taux d'erreur `error_rate`, `dst_host_error_rate` ainsi que `srv_count`, démontrant son efficacité à déceler les variables dont la distribution varie significativement entre flux normaux et intrusifs.

Après avoir utilisé les algorithmes valeurs SHAP, Boruta et Anova f-test pour identifier les caractéristiques les plus significatives, un total de trois ensembles de données réduits ont été dérivés. Ces ensembles ne contiennent que les 10 caractéristiques principales déterminées par les algorithmes en question.

Une fois les datasets réduits, et afin de mettre en œuvre notre modèle, les ensembles de données dérivés ont été divisés en un ensemble d'entraînement représentant 70% des données, tandis que 30% ont été attribués aux tests pour les catégories DoS, Normal et Probe. Toutefois, comme la catégorie R2L ne contient que 1126 enregistrements, une plus petite proportion de 10% a été réservée aux tests afin de conserver un volume suffisant pour l'entraînement.

3.4 Modèles proposés

Dans ce travail, nous proposons de mettre en place un modèle capable de modéliser la dépendance séquentielle entre les événements réseau consécutives. Ce modèle aura la capacité de conserver les informations issues des entrées passées, et identifier les différentes associations entre les événements qui sont dispersés dans la séquence d'entrée.

Nous avons entraîné les données en utilisant diverses variantes des algorithmes de réseaux de neurones récurrents, notamment : LSTM, BiLSTM et stacked-LSTM.

Nous construisons notre modèle de détection d'intrusions avec 10 neurones d'entrée, étant donné que chaque caractéristique d'entrée présente dans notre dataset sera représentée par son neurone d'entrée.

L'architecture de stacked-LSTM proposée est constituée des éléments suivants :

- Une première couche LSTM composée de 150 cellules. L'initialisation des poids et des biais est réalisée à l'aide de la méthode d'initialisation RandomUniform avec une graine fixée à 42 pour assurer la reproductibilité des résultats.
- Une couche de dropout avec un taux de 0,2 afin de réduire le surapprentissage.
- Deux autres couches LSTM, également composées de 150 unités chacune, configurées avec les mêmes paramètres de dropout (0.2) et recurrent dropout (0.2). Ces couches sont empilées afin de permettre un apprentissage hiérarchique des représentations des séquences.
- Une quatrième couche LSTM composée de 100 cellules, avec également un dropout et un recurrent dropout fixés à 0.2, permettant d'extraire les caractéristiques finales avant la classification.
- Une couche dense avec une seule unité de sortie et une fonction d'activation softmax afin de produire des prédictions multiclasse.

Le modèle est compilé à l'aide de l'optimiseur Nadam (Nesterov-accelerated Adaptive Moment Estimation) avec un taux d'apprentissage fixé à 0.002, des coefficients de moment β_1 de 0,9, β_2 de 0,999, une valeur epsilon de 1e-08, et une planification de décroissance du taux d'apprentissage fixée à 0.004. La fonction de perte utilisée est l'entropie croisée catégorielle, adaptée à la classification multiclasse.

Pour évaluer la robustesse du modèle, une validation croisée $K - Fold$ ($k=5$) a été appliquée, garantissant que chaque observation soit utilisée une fois pour la validation et $k - 1$ fois pour l'entraînement.

Le tableau 1.3 présentent les valeurs des paramètres utilisés dans les modèles LSTM, BiLSTM et Stacked-LSTM proposés.

TABLE 1.4 – *Tableau des paramètres des architectures LSTM, Stacked-LSTM et BiLSTM*

Paramètres	LSTM	Stacked-LSTM	BiLSTM
Nombre de caractéristiques	10	10	10
Nombre de couches	-	4	-
Nombre de neurones par couche	-	150 (3 couches), 100 (dernière couche)	-
Fonction d'activation	Softmax	Softmax	Softmax
Fonction de perte	Entropie croisée catégorielle	Entropie croisée catégorielle	Entropie croisée catégorielle
Optimiseur	Adam	Adam	Adam
Taux d'apprentissage	0.002	0.002	0.002
Epsilon	1e-08	1e-08	1e-08
Décroissance du programme	0.004	0.004	0.004
Époques	50	50	50
Dropout	0.3	0.3	0.3/0.2

Les architectures proposées ont été conçues pour offrir de bonnes performances tout en minimisant les risques de surapprentissage et en assurant une convergence rapide lors de l'entraînement.

3.4.1 Validation croisée k-fold (K-fold Cross Validation)

La validation croisée est une technique largement utilisée en apprentissage automatique, elle aide à atténuer les problèmes de surapprentissage (overfitting) et de sous-apprentissage (underfitting) [81].

En utilisant cette technique, les chercheurs peuvent évaluer efficacement la capacité du modèle à se généraliser à un jeu de données indépendant, améliorant ainsi la fiabilité et la robustesse du modèle. Dans notre étude, nous avons utilisé la méthode de validation croisée K-fold avec k fixé à 5. Cela signifie que le jeu de données est divisé en cinq plis, et le modèle est entraîné et testé en cinq itérations.

Le processus de validation croisée k-fold suit un enchaînement d'étapes [82], détaillées ci-dessous :

3. Approche proposée

1. Choisissez un nombre de plis, k . En général k est égale à 5 ou 10 (k soit un nombre inférieur à la longueur du jeu de données).
2. Divisez le dataset en k parties égales appelées plis.
3. À chaque itération, choisissez $k-1$ plis comme ensemble d'entraînement. Le pli restant sera l'ensemble de test.
4. Entraînez le modèle sur l'ensemble d'entraînement.
5. Effectuez la validation sur l'ensemble de test.
6. Enregistrez le résultat de la validation.
7. Répétez les étapes 3 à 6 k fois. À chaque fois, utilisez le pli restant comme ensemble de test. À la fin, vous aurez validé le modèle sur chaque pli.
8. Pour obtenir le score final, faites la moyenne des résultats obtenus à l'étape 6.

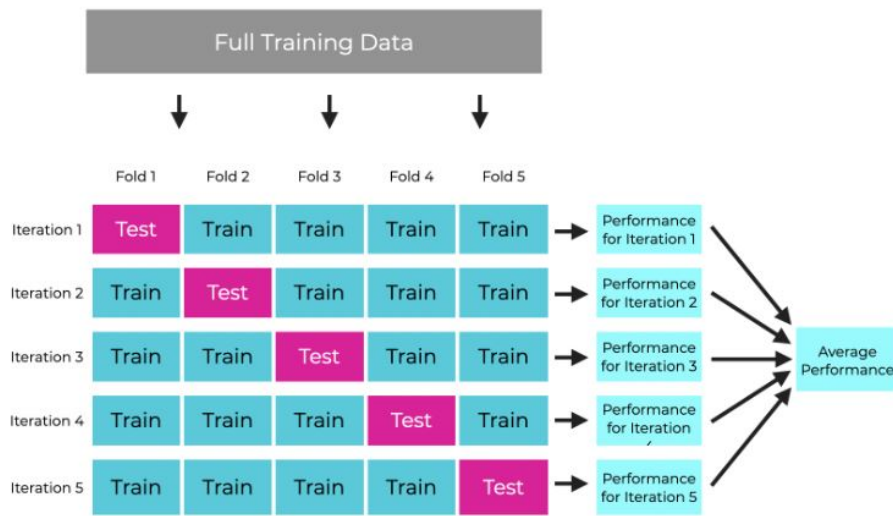


FIGURE 1.11 – Validation croisée k_{fold}

3.4.2 Mesures de performance

Nous avons utilisé un ensemble complet de métriques de performance, comprenant l'exactitude, la sensibilité, la précision, le score F1 et la matrice de confusion.

- Exactitude (Accuracy) : mesure le ratio des observations correctement prédites par rapport au nombre total d'observations.

$$Exactitude = \frac{TP+TN}{TP+FP+TN+FN} \quad (1.14)$$

où TP (True Positives) est le nombre de vrais positifs, TN (True Negatives) est le nombre de vrais négatifs, FP (False Positives) est le nombre de faux positifs et FN (False Negatives) est le nombre de faux négatifs).

- Précision (precision) : mesure le ratio des prédictions positives correctement prédites par rapport au nombre total des prédictions positives effectuées.

$$Precision = \frac{TP}{TP+FP} \quad (1.15)$$

- Rappel (Recall) : mesure la capacité du modèle à identifier les vrais positifs.

$$Recall = \frac{TP}{TP+FN} \quad (1.16)$$

- Score F1 (F1-score) : mesure la probabilité qu'une prédiction positive soit correcte.

$$F_1score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (1.17)$$

- Matrice de confusion : La matrice de confusion est un outil utilisé pour décrire les performances d'un modèle de classification. Elle présente les prédictions correctes et incorrectes pour chaque classe en comparant les résultats réels avec les résultats prédits.

Les faux positifs (FP) sont des événements qui sont classés à tort par le système IDS comme des attaques, mais qui représentent en réalité des comportements normaux du réseau ou des utilisateurs. Bien que ces erreurs de classification n'entraînent pas directement de dommages importants, elles peuvent engendrer une surcharge pour les équipes de sécurité, qui doivent examiner des alertes non pertinentes.

En revanche, les faux négatifs (FN) désignent des intrusions effectivement présentes mais classées à tort comme des événements normaux. Ces cas peuvent entraîner des conséquences graves, en autorisant à des attaques de se produire sans être détectées, ce qui expose l'organisation à des risques liés à la compromission de données ou de systèmes. Par conséquent, la métrique de Rappel (Recall) est particulièrement cruciale dans ce contexte, car elle mesure la capacité du modèle à détecter correctement les intrusions. Elle constitue ainsi une mesure d'évaluation essentielle pour apprécier l'efficacité du modèle de détection d'intrusions proposé.

En utilisant ces métriques de performance, nous pouvons analyser soigneusement notre modèle de classification et obtenir une compréhension précise de sa qualité et de ses performances dans divers situations.

4 Résultats et discussions

Nous présentons et discutons les résultats obtenus avec différents modèles. Nous avons testé nos modèle de classification en utilisant plusieurs techniques de sélection de caractéristiques, telles que Shap Values, Anova f-test et Boruta. Nous avons évalué les performances des modèles en termes d'exactitude, de rappel, de précision et de Score f1, pour le cas de classification multiclasse.

Les résultats des tests, présentés dans le tableau 1.5, comparent les différentes méthodes de réduction de caractéristiques appliquées à plusieurs modèles d'algorithmes d'apprentissage profond. Chaque expérience a été validée à l'aide de la méthode de validation croisée à cinq volets (5-fold cross-validation).

TABLE 1.5 – Résultats expérimentaux

Type	Métriques	Shap Values	Boruta	ANOVA-F Test
LSTM	Accuracy	0.9907	0.9214	0.9933
	Precision	0.9416	0.7065	0.7493
	Recall	0.8414	0.6178	0.7115
	F1-score	0.8564	0.6458	0.7277
BiLSTM	Accuracy	0.9913	0.9207	0.9937
	Precision	0.9581	0.6696	0.7600
	Recall	0.7486	0.6172	0.7153
	F1-score	0.8083	0.6349	0.7341
Stacked LSTM	Accuracy	0.9925	0.9647	0.9945
	Precision	0.9528	0.7269	0.7566
	Recall	0.8760	0.7245	0.7180
	F1-score	0.8969	0.7223	0.7345

L'étude a révélé que les classifieurs utilisant les valeurs SHAP comme méthode de sélection des caractéristiques produisent systématiquement les meilleurs résultats en termes de performances, comparés à d'autres méthodes de sélection. Le test ANOVA f-test, associé au Stacked LSTM, a quant à lui montré le plus haut niveau de précision. À l'inverse, l'algorithme Boruta a enregistré les scores les plus bas lorsqu'il est appliqué aux modèles LSTM, BiLSTM et Stacked LSTM.

Sur la base des résultats, nous concluons que les valeurs SHAP constituent l'approche la plus optimale pour la réduction des données, lorsqu'elles sont combinées aux algorithmes d'apprentissage profond présentés dans cette étude. Le modèle Stacked LSTM a offert la meilleure exactitude, le meilleur rappel (Recall) et le meilleur F1-score parmi tous les autres, en utilisant les valeurs SHAP pour sélectionner les caractéristiques.

En outre, pour mettre en évidence les performances de notre modèle proposé, nous présentons une visualisation de la matrice de confusion générée pour chaque modèle (voir figure 1.12), appliqué notre dataset NSL-KDD, et à partir de laquelle nous illustrons que notre modèle proposé a une bonne capacité à minimiser le nombre attaques classées comme normales, ce qui est d'une grande importance dans la vie réelle pour protéger les systèmes contre les intrusions.

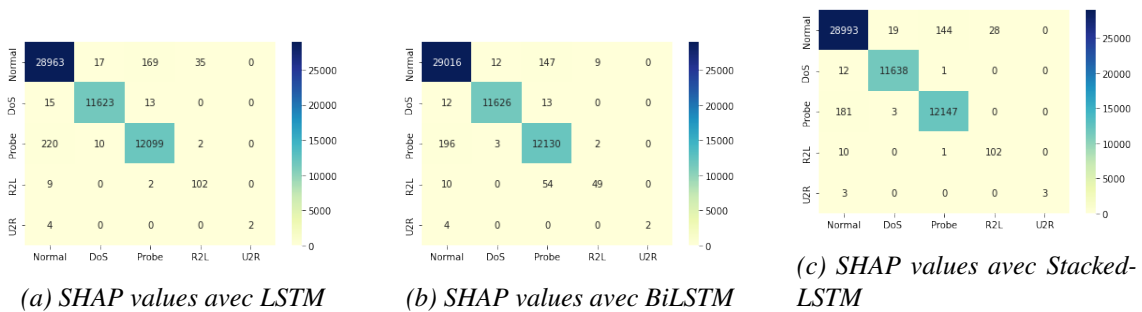


FIGURE 1.12 – Matrices de confusion des modèles

Enfin, il convient de souligner que les résultats obtenus sont très satisfaisants, avec des précisions de tests supérieurs à 95%. Ceci montre que notre modèle proposée est capable de gérer

efficacement le problème de classification et pourrait être utilisée avec succès dans des applications réelles.

5 Conclusion

Comparaison des résultats avec l'état d'art

Pour une meilleure interprétation des résultats expérimentaux, nous avons comparé ceux-ci avec ceux d'études antérieures (tableau 1.6). Cette comparaison est présentée à titre indicatif uniquement, car les environnements d'exécution, les méthodes de prétraitement des données et les procédures d'interprétation des IDS diffèrent souvent.

TABLE 1.6 – *Comparaison des performances*

Type	Accuracy	Precision	Recall	F1-score
CNN [32]	80.13	-	-	-
Adaptive Ensemble [33]	85.20	86.50	86.50	85.20
CNN+BiLSTM [34]	83.58	85.82	84.49	85.14
Shap Values + Stacked LSTM	0.9925	0.9528	0.8760	0.8969

En revanche, notre modèle surpasse les modèles de classification comparées, à savoir CNN, BiLSTM+CNN, indiquant que notre approche est mieux adaptée à ce type de situations.

Limites

Le modèle proposé comporte certaines limitations. Tout d'abord, l'emploi d'une unique couche LSTM s'est avéré inadéquat pour saisir de manière efficace les relations complexes présentes dans les données. En revanche, l'augmentation à quatre couches LSTM a entraîné une amélioration significative des performances, ce qui confirme l'importance de la profondeur du réseau. Cependant, cette approche augmente de manière significative les exigences en ressources, ce qui rend le modèle plus onéreux en termes de temps de calcul et de mémoire.

De plus, notre modèle éprouve des difficultés à classer de manière précise les attaques qui se caractérisent par un faible nombre d'enregistrements. Cette contrainte peut conduire à des inexactitudes dans la classification, en particulier pour les attaques rares ou peu communes, qui revêtent souvent une importance critique en matière de cybersécurité. Ces contraintes soulignent la nécessité d'optimiser davantage notre modèle à fin d'améliorer sa capacité à gérer divers scénarios d'attaques, tout en préservant un équilibre entre performance et efficacité computationnelle.

En plus, malgré l'efficacité des LSTM dans la capture des relations temporelles entre les événements, leur architecture présente des limitations lorsqu'il s'agit de traiter des séquences très longues. En réalité, à mesure que la longueur de la séquence augmente, ces modèles rencontrent des difficultés pour conserver l'ensemble des informations pertinentes, ce qui peut entraîner une perte d'informations et impacter la qualité des prédictions. En outre, leur aptitude à se généraliser est restreinte en présence de la variabilité et de l'évolution des schémas d'attaque, ce qui diminue leur résistance dans des scénarios complexes.

Dans le cadre de travaux futurs, nous envisageons d'explorer une approche novatrice de deep learning pour la détection d'intrusions, qui repose sur le modèle Paragraph Vector-Distributed Me-

mory (PV-DM). Cette approche permet une analyse contextuelle plus approfondie en représentant les attaques dans un espace vectoriel qui conserve à la fois les relations sémantiques et temporelles. Dans cette perspective, une comparaison sera établie avec les réseaux de neurones récurrents, en particulier les LSTM, dans le but d'examiner leurs avantages et inconvénients respectifs, et de déterminer les situations dans lesquelles chaque modèle est plus approprié.

2

Apprentissage Contextuel et Sémantique des attaques

Dans le travail précédent, nous avons élaboré un modèle pour représenter les relations temporelles présentes dans les flux de données réseau, et nous avons renforcé la capacité de détection d'intrusions en exploitant les réseaux de neurones récurrents de type LSTM. Les expériences ont mis en évidence que l'approche séquentielle de la modélisation des événements, en tenant compte du contexte des intrusions, permet au classifieur de détecter de manière plus précise les attaques au sein d'une séquence de données, ce qui se traduit par une amélioration significative des performances de prédiction. Néanmoins, malgré leur efficacité pour saisir les relations temporelles sur le court et le long terme, les LSTM se caractérisent par une structure complexe qui engendre une consommation importante de ressources computationnelles. En réalité, l'obligation de conserver une mémoire sur diverses échelles temporelles complexifie la phase d'apprentissage et d'inférence, ce qui peut restreindre leur utilisation dans des environnements avec des contraintes en ressources, en particulier pour les systèmes de détection d'intrusions en temps réel. Pour répondre à cette problématique, une approche alternative fondée sur PV-DM (ParagraphVector-Distributed Memory) est proposée dans ce chapitre. À la différence des LSTM qui recourent à des mécanismes récurrents pour saisir les dépendances temporelles, PV-DM produit une représentation vectorielle dense et significative des événements réseau en prenant en considération à la fois leur séquence et le contexte global des intrusions. Cette approche permet de générer des vecteurs de caractéristiques qui prennent en compte à la fois les données locales spécifiques à chaque événement et une perspective plus globale du trafic réseau, ce qui renforce la compréhension des attaques dans leur globalité. Une fois ces vecteurs créés, ils servent à entraîner un modèle de détection d'intrusions, permettant de classifier les événements avec une précision élevée. Les résultats de l'expérience démontrent que l'approche PV-DM présente des performances similaires à celles du modèle LSTM tout en offrant des avantages en matière de simplicité et de coût computationnel réduit.

1 Introduction

Dans le domaine de la détection des intrusions, les cyberattaques modernes ne sont plus de simples événements isolés, mais des suites d'actions coordonnées qui s'étendent sur une longue période. Cette évolution les rend particulièrement difficiles à détecter, car chaque étape prise individuellement peut sembler légitime. Ces attaques sont souvent multi-étapes, impliquant des dépendances complexes entre les différentes actions d'un attaquant, et nécessitent une analyse approfondie pour être identifiées. De plus, la détection des intrusions en temps réel exige un traitement rapide et efficace de flux de données massifs provenant de multiples sources, telles que les logs, les paquets réseau et les événements système.

Dans le chapitre précédent, nous avons étudié l'utilisation des réseaux de neurones récurrents de type LSTM pour modéliser ces attaques séquentielles. Ces modèles ont démontré leur capacité à capturer les relations temporelles entre des événements successifs, améliorant ainsi la précision des systèmes de détection des intrusions. Cependant, leur architecture présente plusieurs limitations qui peuvent nuire à leur efficacité dans un contexte opérationnel [83] :

- **Coût computationnel élevé** : Les LSTM nécessitent un nombre important d'opérations par cellule, notamment des multiplications de matrices et des fonctions d'activation impliquant plusieurs portes (entrée, sortie et oubli). Cela entraîne une complexité temporelle élevée, ce qui peut limiter leur applicabilité dans des environnements contraints en ressources.
- **Perte d'information dans les longues séquences** : Les LSTM doivent compresser l'ensemble de la séquence d'entrée $x_1, x_2, \dots, x_n$ en un seul vecteur de contexte c . Cette compression peut entraîner une perte d'information critique, rendant la détection des intrusions plus difficile, en particulier pour les séquences longues et complexes.

Pour pallier ces limitations, nous proposons dans ce chapitre une nouvelle approche de détection des intrusions basée sur PV-DM (ParagraphVector - Distributed Memory). Cette technique permet de générer une représentation vectorielle dense des événements réseau, en tenant compte à la fois de leur séquence et de leur contexte global. Contrairement aux LSTM, qui s'appuient sur des mécanismes récurrents, PV-DM apprend des représentations continues et distribuées des événements, ce qui permet :

- Une meilleure capture du contexte global des attaques, en intégrant des informations sur l'ensemble du flux de données, et non pas uniquement sur une fenêtre temporelle restreinte.
- Une réduction du coût computationnel, car PV-DM ne repose pas sur des calculs itératifs complexes, mais sur un modèle d'apprentissage de représentations plus efficace.
- Une meilleure généralisation des patterns d'attaque, grâce à des vecteurs de caractéristiques enrichis qui conservent des informations pertinentes sur les intrusions, sans souffrir des pertes d'information liées à la compression d'une séquence entière en un seul vecteur.

Les principales contributions abordées dans ce chapitre sont les suivantes :

1. Développement d'un système de détection d'intrusion (IDS) innovant, intégrant un processus de sélection de caractéristiques et exploitant PV-DM comme méthode d'extraction de représentations. Ce système combine plusieurs modèles d'apprentissage automatique pour

classifier les attaques avec une grande précision.

2. Implémentation de deux approches de réduction des caractéristiques : Information mutuelle (MI) pour évaluer la dépendance entre les variables d'entrée et la classe cible, et Valeurs SHAP pour interpréter l'importance des caractéristiques et éliminer celles qui n'apportent pas de valeur ajoutée à la classification.
3. Utilisation de PV-DM pour convertir les données d'intrusion en représentations vectorielles, lesquelles servent ensuite d'entrée aux modèles d'apprentissage automatique.
4. Évaluation expérimentale approfondie, comparant l'efficacité de différentes méthodes d'apprentissage automatique afin de déterminer le classificateur et la stratégie de sélection des caractéristiques les plus performants.
5. Comparaison avec les LSTM, afin de mettre en évidence les avantages et les limites de chaque approche, et de démontrer la pertinence de PV-DM pour la détection des intrusions en temps réel.

Le reste de ce chapitre est structuré comme suit :

- Section 2 : Présentation des concepts fondamentaux et des techniques utilisées dans ce travail.
- Section 3 : Description détaillée du modèle proposé et de sa méthodologie de mise en œuvre.
- Section 4 : Analyse des résultats expérimentaux, avec une comparaison des performances entre PV-DM et LSTM.
- Section 5 : Conclusion du chapitre et perspectives pour les recherches futures.

2 Travaux connexes

Dans cette section, nous examinerons plusieurs travaux de recherche académique récemment menés dans ce domaine. L'étude réalisée par [84] a utilisé une méthode générique de détection, appelée DOC2VEC, pour réaliser une analyse chronologique et une validation croisée des ensembles de données D3M (Drive-by-Download Malware Dataset) et MTA (Malicious Traffic Analysis). Ces ensembles de données contenaient du trafic capturé d'Exploit Kits (EK). Les résultats ont démontré un haut niveau d'efficacité, avec une précision de détection de 0,98 pour l'ensemble de données d'analyse chronologique et de 0,97 pour l'autre ensemble, mesurées par le F-mesure. Cette étude a illustré l'efficacité de la méthodologie pour identifier et détecter précisément le trafic actuel des Exploit Kits (EK).

L'étude menée par San [85] a utilisé la modélisation skip-gram, une variante de la technique word2vec, pour effectuer des tests. L'algorithme de détection d'intrusion a montré des performances notables, avec un taux de précision de 99,20%, un rappel de 82,07% et une précision globale de 91,02%. De plus, il a réussi à minimiser les faux positifs avec un faible taux de 0,61%. Les résultats ont affiché une amélioration notable par rapport aux méthodologies contemporaines.

La validation du modèle a été effectuée à l'aide de l'ensemble de données UNSW-NB15.

Les auteurs de l'étude [86] ont introduit un système combinant deux modèles d'apprentissage profond, à savoir les réseaux de mémoire à long et court terme (LSTM) et les réseaux neuronaux convolutifs (CNN). Le modèle CNN-LSTM a été évalué sur l'ensemble de données KDD99, atteignant un taux de précision de 99,78%. L'étude s'est principalement concentrée sur un seul paramètre de performance et a utilisé un seul ensemble de données pour évaluer le modèle proposé.

Les auteurs de l'étude [87] ont introduit un système de détection d'intrusion (IDS) novateur basé sur l'apprentissage profond, intégrant des Réseaux Neuronaux Convolutifs (CNN) et des Réseaux à Mémoire Longue et Courte (LSTM). Le DL-IDS a été évalué sur l'ensemble de données CICIDS2017 à l'aide d'une approche de classification multicatégories, obtenant une précision globale impressionnante de 98,67%. De plus, ils ont atteint un taux de précision supérieur à 99,50% pour chaque type d'attaque.

La recherche référencée [88] a présenté un système de détection d'intrusion novateur basé un réseau neuronal profond (DNN). L'évaluation du système a été effectuée sur l'ensemble de données KDD Cup 99, avec une architecture de réseau neuronal profond composée de quatre couches cachées. Le type de classification utilisé était binaire, distinguant entre les instances normales et les attaques. Le système a démontré un taux de précision louable de 99,08%.

Les auteurs de l'article [89] ont proposé un cadre d'apprentissage profond intégrant des Auto-Encodeurs et des Réseaux de Croyance Profonde (DBN) pour améliorer l'efficacité de la détection d'intrusion. L'utilisation d'un Auto-Encodeur a permis de réduire la dimensionnalité des données et d'identifier leurs caractéristiques saillantes, tandis que le Réseau de Croyance Profonde (DBN) s'est chargé de détecter les codes potentiellement malveillants. Le modèle a été évalué à l'aide de l'ensemble de données KDD Cup 99 et comparé à un Réseau de Croyance Profonde seul. Les résultats indiquent que le système hybride récemment développé présente une amélioration statistiquement significative en termes de précision et d'efficacité. Cependant, les auteurs n'ont pas fourni de justification détaillée pour leur décision d'intégrer les DBN et les Auto-Encodeurs dans leur modèle.

Dans leur étude, Kim et al. [90] ont réalisé une analyse comparative de diverses techniques de détection d'intrusion en utilisant l'ensemble de données KDD99. Les chercheurs ont découvert que le réseau LSTM-RNN offrait des performances supérieures par rapport à diverses autres méthodes telles que le Réseau Neuronal à Régression Généralisée (GRNN), le Réseau Neuronal basé sur le Produit (PNN), les k-Plus Proches Voisins (KNN), les Machines à Vecteurs de Support (SVM), les approches bayésiennes et d'autres encore.

Yakubu et al. [91] ont présenté une approche de type Mémoire Bidirectionnelle Longue et Courte (BiDLSTM) pour la détection d'intrusion. Les performances de ce système ont été évaluées à l'aide de l'ensemble de données NSL-KDD. Les résultats expérimentaux ont démontré que le modèle BiDLSTM surpassait les autres modèles contemporains, tels que LSTM, en termes de précision, rappel, score F et métriques pour la détection des attaques U2R et R2L.

Shen et al. [92] ont introduit une méthodologie hybride intégrant la machine d'apprentissage extrême (ELM) et l'algorithme de chauve-souris (BA) afin d'améliorer la précision de détection d'intrusion par optimisation de la sélection des caractéristiques. La machine d'apprentissage extrême (ELM) a été utilisée comme classificateur principal, tandis que l'algorithme de boosting (BA) a permis d'éliminer les caractéristiques redondantes, réduisant ainsi la dimensionnalité de l'ensemble de données. La méthode a été évaluée sur trois ensembles de données distincts

(KDD99, NSL-KDD, et Kyoto), avec une précision globale de 98,94%.

Dong [93] a introduit un modèle de détection d'intrusion novateur nommé AE AlexNet, qui utilise l'architecture de réseau neuronal AlexNet Auto-Encoder. Lors de l'évaluation avec l'ensemble de données KDD99, le modèle a atteint un taux de précision de 94,32%.

Les auteurs de l'étude [94] ont proposé DL-MAFID, un modèle de détection d'intrusion basé sur un système multi-agents, incorporant un Auto-Encodeur pour la réduction de dimension. Ce modèle a été spécifiquement développé pour répondre aux complexités des tâches de détection d'intrusion multicatégories. Ses performances ont été évaluées avec l'ensemble de données KDD-Cup'99. En plus d'utiliser l'Auto-Encodeur, les cinq catégories de l'ensemble de données ont été classées à l'aide de classificateurs superficiels tels que MLP et KNN. Les résultats expérimentaux montrent que DL-MAFID peut atteindre une précision de détection de 99,95%, tout en réduisant la durée nécessaire pour la détection.

Les chercheurs de l'étude [95] ont présenté trois systèmes distincts de détection d'intrusion basés sur des techniques d'apprentissage profond, à savoir les Réseaux Neuronaux Artificiels (ANN), les Réseaux Neuronaux Profonds (DNN), et les Réseaux Récurrents (RNN). L'évaluation des méthodologies proposées a été réalisée à l'aide de l'ensemble de données UNSW-NB15, en englobant des tâches de classification binaire et multicatégories.

Les auteurs de [96] proposent l'introduction de BFFA, une adaptation binaire de l'algorithme de fertilité des terres agricoles (FFA), spécifiquement développé pour la sélection de caractéristiques dans la classification des IDS. Les classificateurs DT, KNN et SVM ont été hybridés avec la technique de sélection des caractéristiques suggérée pour créer un IDS efficace et fiable. La méthodologie a été soumise à une analyse comparative avec plusieurs algorithmes (KNN, SVM, DT, RF, ADA_BOOST et NB). Les ensembles de données NSL-KDD et UNSW-NB15 ont été évalués, révélant que la technique suggérée surpassait les classificateurs précédents en termes de précision, de rappel et de temps d'exécution.

Dans leur étude, les chercheurs ont présenté MODHHO, une technique d'optimisation dynamique multi-objectif Harris Hawks, spécifiquement développée pour la détection de botnets dans les réseaux de l'Internet des Objets (IoT) [97]. En utilisant le puissant algorithme HHO, MODHHO détecte efficacement des caractéristiques significatives dans les ensembles de données IoT, affichant des performances améliorées grâce à la mise en œuvre d'une stratégie multi-objectifs. Le classificateur K-NearestNeighbor (KNN) a été utilisé pour évaluer les erreurs de classification, indiquant que MODHHO possède une précision élevée par rapport à ses concurrents dans l'identification des botnets IoT. Le modèle suggéré démontre systématiquement des performances supérieures par rapport aux algorithmes alternatifs, avec une amélioration moyenne de 26% sur tous les ensembles de données et critères d'évaluation.

D'après l'évaluation de la littérature (voir tableau 2.1), il est évident que la plupart des recherches manquent d'accent sur les modèles d'ingénierie des caractéristiques des données, qui jouent un rôle crucial dans l'identification des caractéristiques pertinentes. De plus, même lorsque certaines études utilisent une technique de réduction, elles ont tendance à se limiter à une seule méthode. En outre, la majorité des recherches ont évalué leurs modèles proposés en utilisant un seul ensemble de données et une classification binaire.

Il convient de noter que les recherches récentes ont accordé une attention significative à l'utilisation de systèmes de détection d'intrusion (IDS) basés sur des techniques d'apprentissage profond. Cependant, il est important de considérer que de tels systèmes peuvent présenter des com-

plexités inhérentes et nécessiter des investissements de temps substantiels. En revanche, les systèmes IDS pratiques devraient privilégier la facilité d'utilisation, l'efficacité opérationnelle et la capacité à détecter rapidement les anomalies. Par ailleurs, il est évident qu'aucune recherche antérieure n'a intégré l'architecture PV-DM avec les méthodes de réduction de dimension basées sur l'Information Mutuelle et les valeurs SHAP pour les classifications binaire et multicalasse.

TABLE 2.1 – *Aperçu des différents modèles bien établis dans les systèmes de détection d'intrusion (IDS)*

Réf.	Architecture	Dataset	Résultat	Limites
[88]	DNN	KDD Cup 99	Précision = 99%	L'étude a utilisé une approche de classification binaire.
[89]	Auto-Encodeur + DBN	KDD Cup 99	Précision = 92.10%, FPR = 1.58%, TPR = 92.20%	Les auteurs n'ont pas fourni de justification approfondie pour leur choix d'utiliser les réseaux de croyance profonds (DBN) avec un auto-encodeur dans leur modèle
[87]	CNN + LSTM	CICIDS2017	Précision = 98.67%, F1 = 93.32%, TPR = 97.21%, FPR = 0.47%	L'article ne contient pas d'analyse comparative du DL-IDS par rapport à d'autres méthodologies établies ou concurrentes. Une étude du modèle proposé en comparaison avec d'autres méthodes de détection d'intrusion (IDS) aurait pu offrir une meilleure compréhension de ses avantages distincts
[98]	Random Forest + Non supervisé	KDD Cup 99	Précision = 96.42%, FPR = 0.98%	Temps d'exécution long.
[99]	BLSTM-RNN	-	Mirai = 99%, UDP = 98%, DNS = 98%. Pertes : 0.0008, 0.1256, 0.1164.	Temps d'exécution long.
[100]	RF Regressor + ID3	CICIDS2019	Précision = 78%, Rappel = 65%, F1 = 69%	L'efficacité du modèle proposé est faible.
[101]	CFS-BA	NSL-KDD, CICIDS2017, AWID	NSL-KDD : 99.8%, CICIDS2017 : 99.9%, AWID : 99.5%	Les performances ne sont pas significativement améliorées par l'utilisation des méthodes d'ensemble, en particulier dans les situations où il existe un déséquilibre important dans la répartition des classes.

3 Concepts de base

Après avoir exploré les différentes approches proposés dans la littérature pour la détection d'intrusion, il est important de détailler les concepts spécifiques qui sous-tendent les méthodes utilisées dans ce chapitre. Cette section se concentre sur les méthodes utilisées dans ce chapitre, telles que le modèle d'apprentissage superficiel (shallow learning model) PV-DM (Paragraph Vector-Distributed et la méthode de sélection de caractéristique information mutuelle (MI).

3.1 Modèle shallow learning

Un modèle d'apprentissage superficiel, parfois appelé apprentissage automatique superficiel ou réseau neuronal superficiel, désigne un modèle d'apprentissage automatique caractérisé par un nombre restreint de couches ou une complexité architecturale limitée. Les modèles d'apprentissage superficiel se caractérisent généralement par leur profondeur limitée, ne comprenant généralement qu'une ou quelques couches de neurones. Ces modèles sont fréquemment utilisés pour des tâches ou des ensembles de données présentant des caractéristiques basiques, où les associations entre caractéristiques et cibles sont relativement simples.

Théoriquement, l'utilisation d'un nombre limité de couches dans l'apprentissage superficiel améliore l'interprétabilité des décisions, ce qui est crucial pour leur compréhension. La priorité accordée à l'interprétabilité est particulièrement importante dans des domaines critiques comme la détection d'intrusion. D'un point de vue pratique, les modèles d'apprentissage superficiel montrent des performances exceptionnelles dans des situations où les données sont limitées et nécessitent une quantité moindre de données étiquetées pour l'entraînement. En revanche, les méthodologies d'apprentissage profond peuvent exiger une quantité significative de données annotées pour atteindre une efficacité optimale, tout en impliquant des temps d'exécution plus longs en raison de l'utilisation de plusieurs couches.

De plus, les modèles superficiels présentent une complexité informatique réduite, ce qui renforce leur adéquation pour une utilisation dans des environnements aux ressources limitées. Dans le contexte de la détection d'intrusion, la priorité accordée à la détection en temps réel des menaces nécessite l'adoption d'un modèle superficiel offrant une meilleure interprétabilité. Cette approche s'avère plus faisable et efficace, notamment en tenant compte des limitations de l'apprentissage profond en termes de besoins en données et de temps d'exécution.

3.2 Word Embedding

La principale idée derrière le modèle de Word Embedding consiste à représenter les concepts sous forme de vecteurs, les relations binaires sous forme de matrices, et à modéliser l'application d'une relation à un concept par une multiplication matrice-vecteur, permettant ainsi d'obtenir une approximation du concept associé [102]. Le vecteur Word Embedding est généralement calculé à l'aide de réseaux de neurones selon le procédé suivant :

- La couche d'entrée (Input Layer) : dans cette couche, chaque mot appartenant au vocabulaire V est représenté par un vecteur one-hot de dimension $|V|$ (correspondant à la dimension du vocabulaire).
- La couche de projection (Hidden Layer) : dans cette couche, le vecteur one-hot associé à un mot est multiplié par une matrice $W \in \mathbb{R}^{V \times N}$ où W représente la matrice de Word Embedding

initialisée aléatoirement, $|V|$ correspond à la dimension du vocabulaire et N désigne la dimension des vecteurs Word Embedding du mot qu'on veut calculer (généralement comprise entre 20 et 100). En parcourant l'ensemble du corpus, à chaque étape t , le vecteur du mot cible et les paramètres de la matrice W sont ajustés afin que les mots ayant une signification similaire se rapprochent dans l'espace vectoriel. Après un certain nombre d'itérations, les vecteurs convergent vers des représentations significatives, attribuant des vecteurs proches aux mots partageant des similarités [103] [104].

- La couche de sortie : Cette couche prend la sortie de la couche de projection, multipliée par une matrice W' (également initialisée aléatoirement), et génère une distribution de probabilité sur les mots du vocabulaire à l'aide de la fonction Softmax.

La figure 2.1 illustre l'architecture générale du Word Embedding, où $x_1, x_2, \dots, x_V$ représente les vecteurs d'entrée, $H = h_1, h_2, \dots, h_N$ désigne les projections des entrées dans la couche cachée, et $Y = y_1, y_2, \dots, y_V$ correspond aux sorties calculées à l'aide de la fonction Softmax.

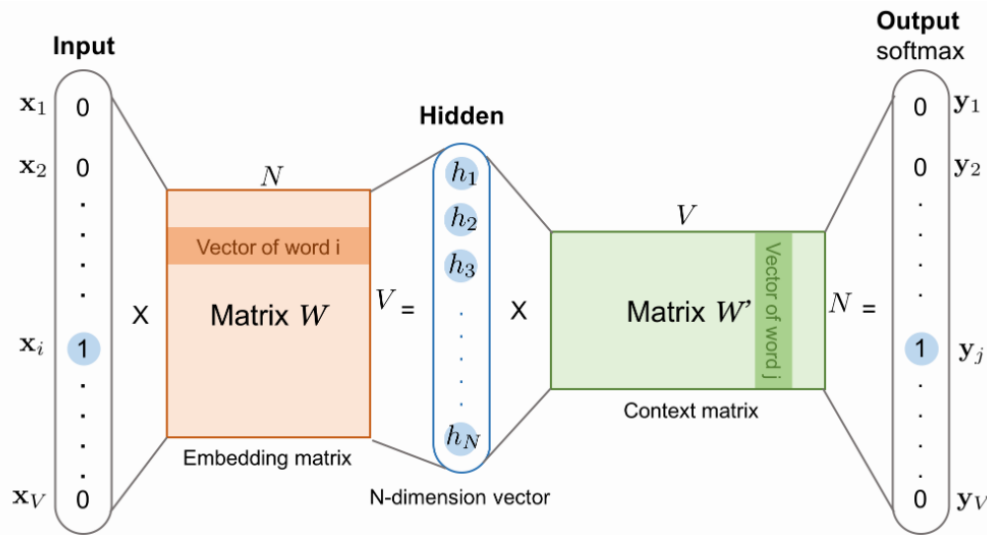


FIGURE 2.1 – word embedding

3.3 Word2vec

Word2Vec [103] [104] est un modèle non supervisé basé sur le Word Embedding. Il repose sur des perceptrons linéaires simples avec une seule couche cachée. L'objectif est de compresser le corpus en un dictionnaire de vecteurs denses de dimension significativement réduite, représentant les relations entre les mots et leur contexte. Word2Vec est l'un des modèles les plus populaires de Word Embedding [105] [106], et il existe en deux variantes : le Continuous Bag of Words (CBOW), qui prédit un mot cible à partir de son contexte, et le Skip-gram, qui prédit les mots du contexte à partir d'un mot cible donné. La fonction de perte est minimisée lorsque le modèle attribue des probabilités élevées aux mots qui appartiennent au contexte du mot cible et des probabilités faibles aux autres mots, considérés comme du bruit.

3.4 Apprentissage Superficiel : Paragraph Vector-Distributed Memory

L'apprentissage superficiel (Shallow learning), également appelé apprentissage automatique superficiel ou réseau neuronal superficiel, désigne une classe d'algorithmes dont les modèles comportent généralement une ou deux couches de traitement. Ces modèles sont généralement rapides,

interprétables et efficaces pour les tâches simples avec des données de dimensionnalité relative-ment faible, ces modèles incluent des algorithmes tels que la régression logistique, les machines à vecteurs de support (SVM) et les arbres de décision.

L'algorithme ParagraphVector-Distributed Memory (PV-DM) est un algorithme développé par Google [103]. C'est une technique basée sur les réseaux de neurones qui transforme les mots en vecteurs réels, en prenant en compte leur ordre, de sorte que les mots ayant des similarités sémantiques et syntaxiques se retrouvent proches dans l'espace vectoriel. Ces vecteurs peuvent ensuite être utilisés comme caractéristiques dans des algorithmes de classification et d'apprentissage automatique.

Le PV-DM s'inspire du modèle Word Embedding, en particulier du modèle CBOW de Word2Vec [104]. L'idée principale du PV-DM est qu'un paragraphe P peut être représenté par un vecteur qui contribue à la prédiction du mot manquant dans une phrase [107].

La figure 2.2 illustre le processus de prédiction dans le modèle PV-DM. Étant donné un ensemble de mots $x_{n+2}, x_{n+1}, x_{n-1}, x_{n-2}$ dans un paragraphe P , où le mot x_n est manquant, PV-DM prédit le vecteur du mot absent $V(x_n)$ en tenant compte des autres mots vecteurs $v(x_{n+2}), v(x_{n+1}), v(x_{n-1}), v(x_{n-2})$, ainsi que du vecteur de paragraphe $V(P)$, qui représente le contexte du mot que nous essayons de prédire.

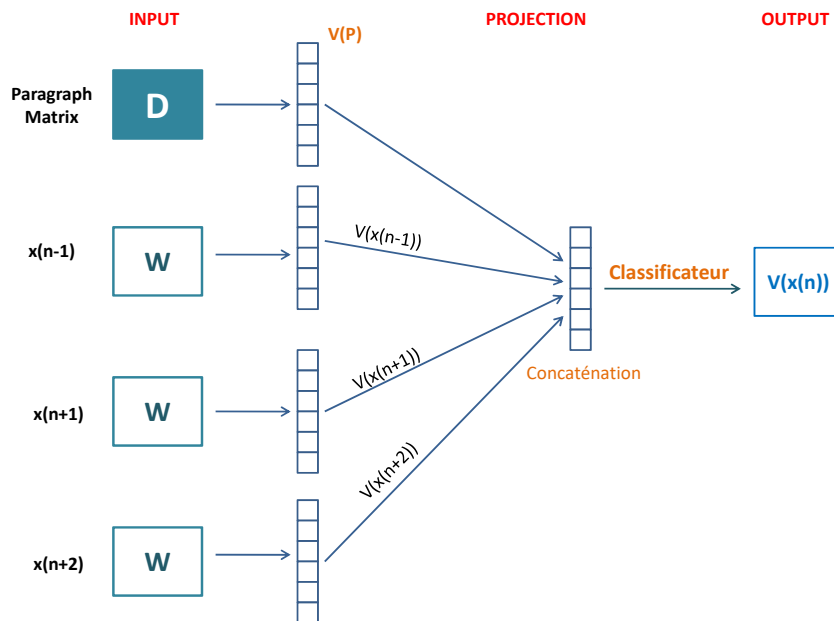


FIGURE 2.2 – Modèle de PV-DM pour l'apprentissage des vecteurs de paragraphes [108]

Dans le modèle PV-DM (voir Figure 10), chaque paragraphe est représenté par un vecteur unique, qui correspond à une colonne de la matrice D , et chaque mot est également représenté par un vecteur unique, situé dans une colonne de la matrice W [108]. Les vecteurs de paragraphe et de mots sont ensuite concaténés, et la prédiction est réalisée à l'aide d'un classificateur, tel que Softmax, comme suit :

$$p(v(x_n)|v(x_{n+2})...v(x_{n-2}), P, W, D) = \frac{e^{y_{v(x_n)}}}{\sum_i e^{y_{v(x_i)}}} \quad (2.1)$$

Chaque $y_{v(x_i)}$ est déterminé de la manière suivante :

$$y = b + U * h(v(x_n)|v(x_{n+2})...v(x_{n-2}), P, W, D) \quad (2.2)$$

Où b et U sont les paramètres du classificateur et h est construit à partir des matrices W et D .

L'utilisation de la méthode PV-DM pour la représentation vectorielle présente plusieurs avantages :

- Prise en compte des contextes : PV-DM tient compte des contextes dans lesquels les mots apparaissent, ce qui permet de mieux capturer les relations sémantiques entre les mots au sein d'un paragraphe ou d'une phrase.
- Capacité à traiter des phrases ou des paragraphes complets : PV-DM peut gérer des phrases ou des paragraphes entiers, préservant ainsi la structure syntaxique des phrases et offrant des informations sémantiques plus complètes.
- Meilleure performance pour les tâches de traitement du langage naturel : PV-DM se distingue par son efficacité dans les tâches de traitement du langage naturel, comme la classification et la génération de textes.

En résumé, PV-DM permet une meilleure compréhension des contextes et des relations sémantiques des mots, est adapté au traitement de phrases et de paragraphes complets, et améliore les performances dans les tâches de traitement du langage naturel.

3.5 Information Mutuelle (MI)

L'algorithme de l'information mutuelle (MI), basé sur la théorie de l'information, est utilisé pour quantifier la dépendance entre deux variables aléatoires [109]. Elle est souvent utilisée pour évaluer la pertinence des caractéristiques dans les problèmes d'apprentissage automatique, en mesurant la quantité d'information qu'une caractéristique fournit sur la variable cible.

L'information mutuelle entre deux variables aléatoires X et Y est définie comme suit :

$$I(X;Y) = \sum_{(x,y)} P_{XY}(x,y) \log \left(\frac{P_{XY}(x,y)}{P_X(x)P_Y(y)} \right) \quad (2.3)$$

Où $P_{XY}(x,y)$ est la probabilité conjointe de X et Y , $P_X(x)$ et $P_Y(y)$ sont les probabilités marginales : $P_X(x) = \sum_y P_{XY}(x,y)$.

L'information mutuelle est toujours non négative et ne sera nulle que si les deux variables sont indépendantes. Une grande valeur d'information mutuelle indique une forte dépendance entre les

variables, alors d'une valeur faible suggère une dépendance moindre.

Dans le cadre de la sélection des caractéristiques, l'information mutuelle est un outil utilisé pour évaluer la pertinence de chaque caractéristique par rapport à la variable cible. Pour ce faire, on évalue l'information mutuelle entre chaque caractéristique et la variable cible, puis on ordonne les caractéristiques selon leur score d'information mutuelle. Les attributs qui ont obtenu des scores élevés sont jugés comme étant plus pertinents et plus informatives pour la variable cible.

Dans notre recherche, nous avons choisi d'utiliser l'information mutuelle afin de sélectionner les caractéristiques les plus importantes, car cette méthode offre une évaluation robuste de la dépendance entre les caractéristiques et la variable cible, en considérant à la fois les relations linéaires et non linéaires.

En plus de MI, nous avons utilisé dans ce travail les valeurs Shap, qui sont déjà définies dans le chapitre 1 section 2.7.1.

3.6 Approche proposée

Comme indiqué ci-dessus, le modèle que nous proposons utilise en premier lieu un prétraitement des données qui est une étape primordiale pour obtenir des résultats fiables et améliorer les performances des algorithmes d'apprentissage automatique. Ensuite, une sélection de caractéristiques, dans le but de sélectionner les caractéristiques les plus importantes et de réduire le nombre de caractéristiques d'entrée avant de les introduire dans le modèle. Ensuite, le modèle à apprentissage superficiel PV-DM est utilisé afin de générer une représentation numérique à partir des flux réseau et des séquences d'événements, créant ainsi des vecteurs compacts où les indices contiennent implicitement des informations sur le contexte global de la séquence ainsi que sur les contextes locaux des événements. Enfin, les vecteurs de contexte générés par le modèle PV-DM ont été introduits comme entrées pour entraîner les classificateurs Random Forest et XGBoost, qui ont ensuite été utilisés pour classer les données. L'architecture du système proposé est illustrée dans la figure 2.3.

3.6.1 Ensembles de Données

La présente étude utilise deux ensembles de données, à savoir le NSL-KDD [110] et l'UNSW NB15 [111], afin de mener des expériences visant à valider notre approche proposée.

A- NSL-KDD

La description complète du jeu de données NSL-KDD a été fournie dans le premier chapitre, où les caractéristiques et les spécificités ont été détaillées.

B- UNSW-NB15(University of New South Wales–NB2015)

La création de l'ensemble de données UNSW-NB15 [111] a eu lieu en 2015 au Centre australien pour la cybersécurité (Australian Center for Cyber Security, ACCS), avec pour principal objectif de faciliter la recherche sur la détection d'intrusion. Les données ont été obtenues à l'aide d'un outil IXIA pour collecter une combinaison d'actions régulières et malveillantes.

L'ensemble de données comprend 49 caractéristiques classées en cinq catégories : Binaire, Flottant, Entier, Nominal, et Horodatage. Il couvre un large éventail de comportements typiques et aberrants pour permettre une investigation approfondie. L'UNSW-NB15 regroupe un total de neuf catégories distinctes d'attaques, comprenant : Fuzzers, Analyse, Backdoors, Déni de Service,

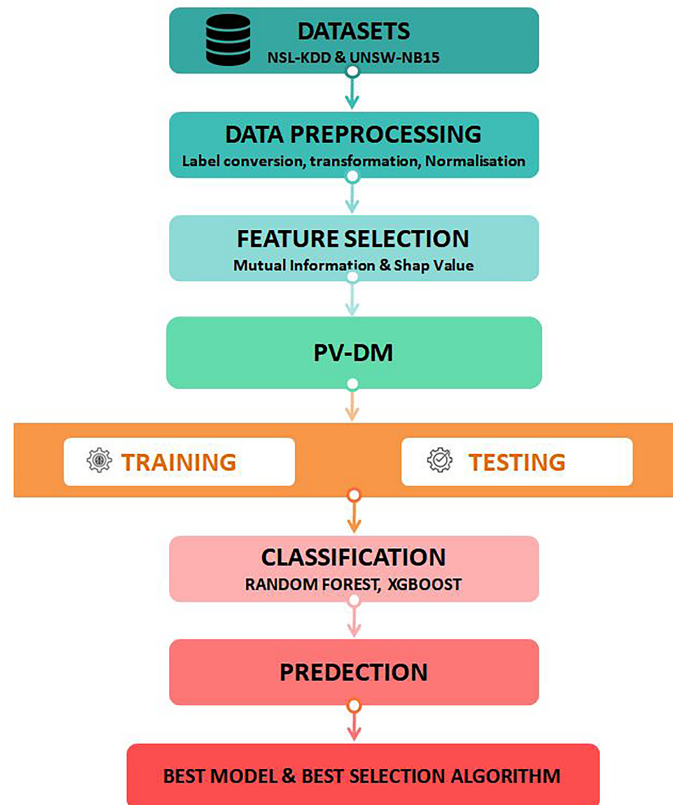


FIGURE 2.3 – L’architecture du modèle proposé pour la détection d’intrusion

Exploits, Génériques, Reconnaissance, Shell code, et Worms.

L’utilisation de cet ensemble de données a été largement adoptée par les chercheurs dans le domaine de l’évaluation des systèmes de détection d’intrusion. Une observation notable concernant cet ensemble est que le nombre d’enregistrements d’attaques dépasse celui des enregistrements normaux. En outre, il existe une disparité significative dans le nombre d’enregistrements entre les différents types d’attaques, ce qui engendre un problème de déséquilibre des données.

La répartition inégale des données dans l’ensemble peut poser des difficultés pour entraîner et évaluer efficacement les systèmes de détection d’intrusion. La représentation graphique de la répartition des enregistrements d’attaques et normaux dans les ensembles de données UNSW-NB15 et NSL-KDD est illustrée respectivement dans la figure 2.4 et la figure 2.5.

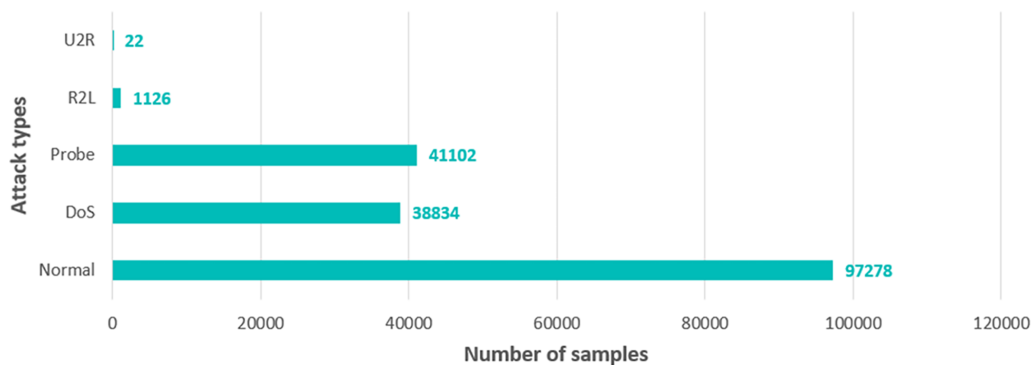


FIGURE 2.4 – Enregistrements d’attaques et normaux dans l’ensemble de données NSL-KDD

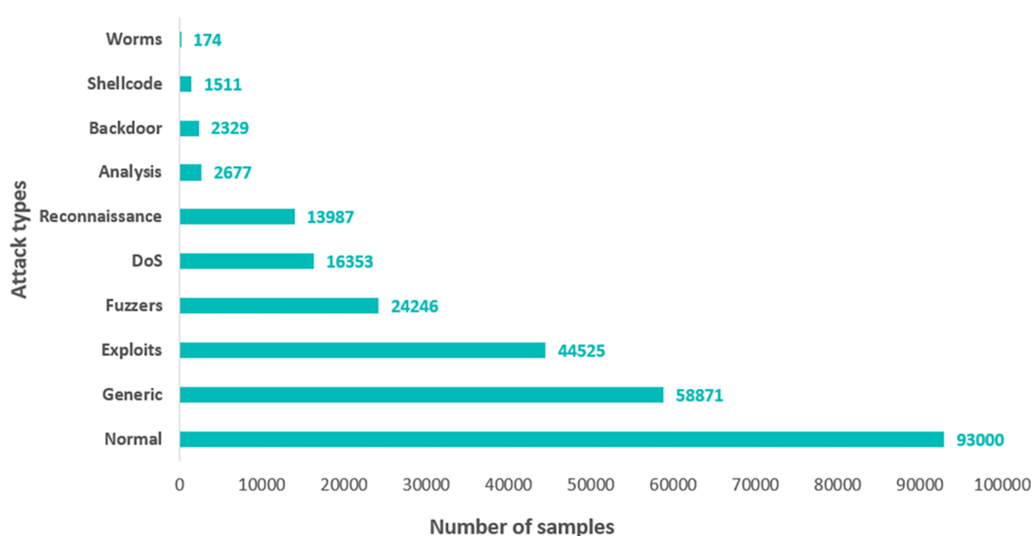


FIGURE 2.5 – Enregistrements d’attaques et normaux dans l’ensemble de données UNSW-NB15

En raison du faible nombre d’enregistrements pour certains types d’attaques dans les deux ensembles de données, comme expliqué précédemment et illustré dans les Figures 2.4 et 2.5, nous avons choisi de concentrer notre recherche sur les quatre premières catégories d’attaques dans chaque ensemble de données, comme indiqué dans le Tableau 2.2.

TABLE 2.2 – Les différentes catégories d’attaques dans les deux ensembles de données

Ensemble de données	Catégories d’attaques
NSL-KDD	Normale, DoS, Probe, R2L
UNSW-NB15	Normale, Générique, Exploits, Fuzzers

3.6.2 Prétraitement des Données

Transformation numérique des caractéristiques

Étant donné que de nombreux algorithmes de machine learning ne traitent pas directement les données catégorielles, il est essentiel de les convertir en valeurs numériques avant de les utiliser. Ainsi, les attributs non numériques à savoir protocol-type, service et flag pour NSL-KDD ; proto, service et state pour UNSW-NB15 ont été transformés à l’aide de la fonction `factorize()` de pandas. Cette transformation permet de représenter efficacement les variables catégorielles sous forme d’étiquettes numériques compatibles avec les modèles d’apprentissage automatique.

Normalisation

Nous avons ensuite utilisé le `StandardScaler`, qui standardise les caractéristiques, c’est-à-dire en les centrant autour de 0 et en les mettant à l’échelle de manière à ce qu’elles aient un écart-type de 1. Par la suite, nous avons mélangé les données afin de réduire la variance et de nous assurer que les modèles restent généraux et évitent le sur-apprentissage. Enfin, nous avons préparé les données pour qu’elles correspondent à la classification multiclasse.

Encodage des étiquettes

Dans cette étape, nous avons effectué un encodage des étiquettes pour chaque classe d'attaque. Pour le dataset NSL-KDD, nous avons attribué une valeur numérique à chaque classe : 0 pour Normal, 1 pour DoS, 2 pour Probe et 3 pour R2L. De même, pour le dataset UNSW-NB15, les catégories ont été codées comme suit : 0 pour Normal, 1 pour Generic, 2 pour Exploits et 3 pour Fuzzers.

Dans cette étape, nous avons effectué un encodage des étiquettes en 4 classes : Pour le dataset NSL-KDD, nous avons attribué une valeur numérique à chaque classe : 0 pour Normal, 1 pour DoS, 2 pour Probe et 3 pour R2L. De même, pour le dataset UNSW-NB15, les catégories ont été codées comme suit : 0 pour Normal, 1 pour Generic, 2 pour Exploits et 3 pour Fuzzers. Nous avons ensuite remplacé la caractéristique "label" de chaque dataset par ces chiffres, au lieu des noms d'attaque.

3.7 Résultats expérimentaux et discussion

Le langage de programmation utilisé dans cette étude était Python 3.7.1, et l'environnement de développement intégré employé était Google Colab. L'implémentation a fait appel à plusieurs bibliothèques, telles que NumPy, Pandas, Keras et Scikit-Learn (Sklearn).

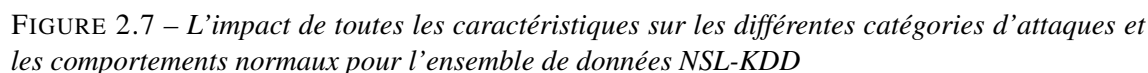
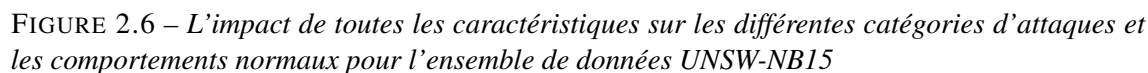
Les ensembles de données prétraités de l'UNSW-NB15 ont été divisés en un ensemble d'entraînement représentant 70% des données et un ensemble de test contenant les 30% restants. La division de l'ensemble de données NSL-KDD a été spécifiquement conçue pour tenir compte des variations dans le nombre d'enregistrements d'attaques. En particulier, 30% de l'ensemble de données ont été attribués aux tests pour les catégories DoS, Normal et Probe. En revanche, en raison de sa quantité relativement limitée de 1126 enregistrements, la catégorie R2L a reçu une proportion plus faible de 10% pour les tests.

3.7.1 Résultats de la sélection des caractéristiques

Importance de la sélection de caractéristiques

La sélection des caractéristiques joue un rôle crucial dans la phase de prétraitement des données [112]. La méthodologie proposée vise à améliorer la précision et l'efficacité des modèles d'apprentissage automatique en se concentrant spécifiquement sur les données pertinentes et en priorisant leur qualité.

Les Figures 2.6 et 2.7 illustrent la représentation visuelle des données issues des ensembles de données UNSW-NB15 et NSL-KDD, dans un espace de dimension réduite. Cette réduction est réalisée en utilisant l'Analyse en Composantes Principales (ACP) comme technique de transformation des données. Les graphiques illustrent les effets de diverses caractéristiques sur les différents types d'attaques. Il est évident que certaines caractéristiques, telles que Djit, smean et dinpkt dans l'ensemble de données UNSW-NB15, ont un degré de signification réduit par rapport à d'autres attributs. Par conséquent, l'application de méthodes de réduction des caractéristiques devient impérative pour réduire les données non informatives ou confuses.



Comme mentionné précédemment, l'information mutuelle mesure le niveau de dépendance entre deux variables. Dans cette étude, nous utilisons l'information mutuelle pour évaluer l'impact de chaque caractéristique sur la variable cible dans les deux ensembles de données.

54

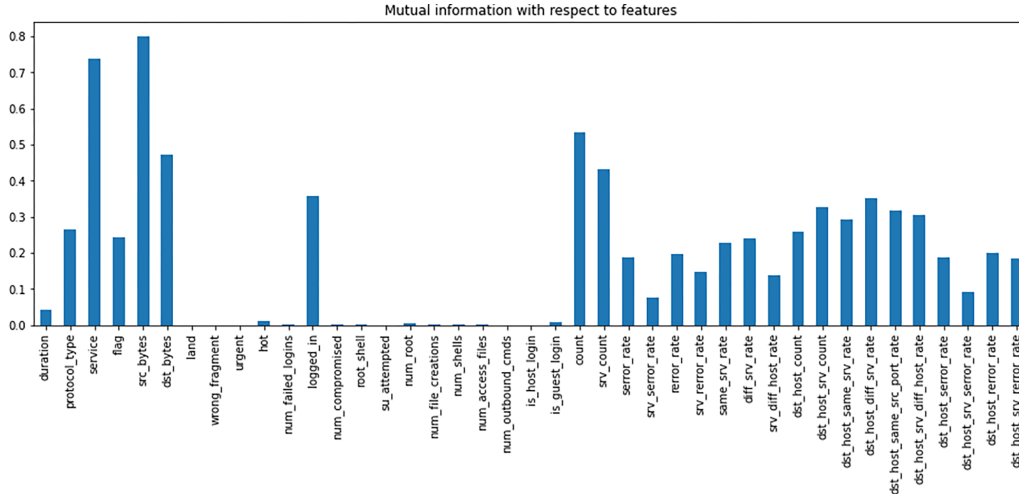


FIGURE 2.8 – Scores d'information mutuelle des caractéristiques de l'ensemble de données NSL-KDD

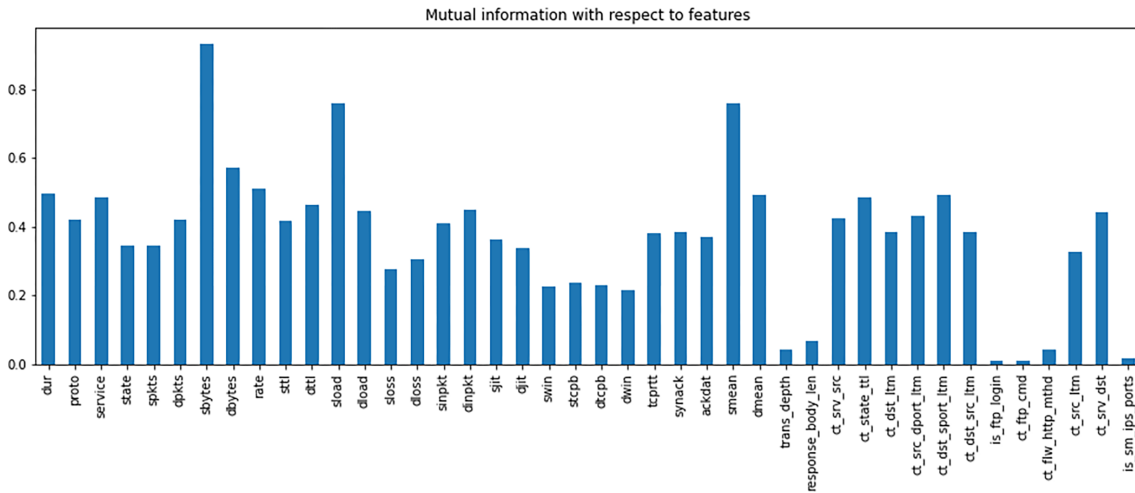


FIGURE 2.9 – Scores d'information mutuelle des caractéristiques de l'ensemble de données UNSW-NB15

svr_count, logged_in, dst_host_diff_srv_rate, rate, dst_host_srv_count, dst_host_same_src_port_rate, et dst_host_srv_diff_host_rate affichent les scores d'information mutuelle les plus élevés dans l'ensemble de données NSL-KDD. En revanche, on observe dans la figure 2.9 que les attributs sbytes, sload, smean, dbytes, rate, dur, ct_dst_sport_ltm, dmean, ct_state_ttl, et service dans l'ensemble UNSW-NB15 présentent les scores d'information mutuelle les plus élevés. À l'inverse, les autres caractéristiques des deux ensembles de données affichent des scores faibles, suggérant qu'elles ont une influence limitée sur la variable cible.

Shap Values

D'un autre côté, comme indiqué dans le chapitre précédent, les valeurs SHAP fournissent des informations sur les contributions individuelles de chaque caractéristique à la prédiction de la variable cible. Pour évaluer l'importance d'une caractéristique particulière, le modèle considère les résultats de toutes les combinaisons possibles de caractéristiques, comme illustré dans les figures 2.10 et 2.11.

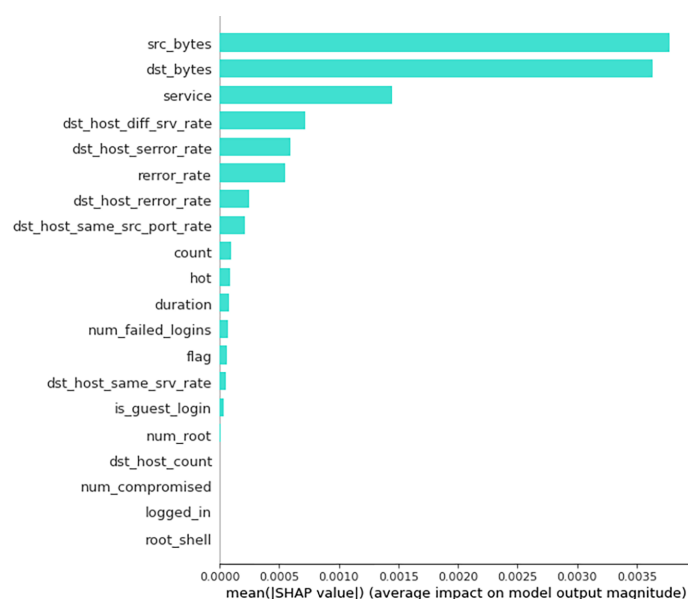


FIGURE 2.10 – Importance des caractéristiques dans l'ensemble de données NSL-KDD en utilisant SHAP

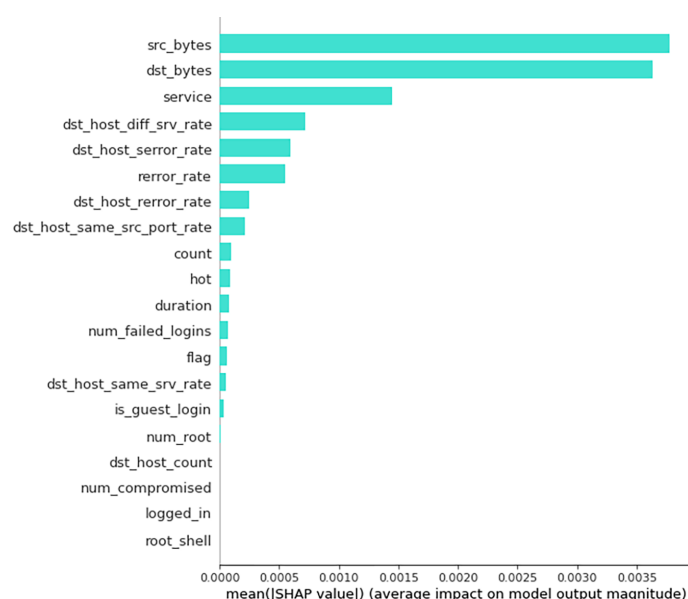


FIGURE 2.11 – Importance des caractéristiques dans l'ensemble de données UNSW-NB15 en utilisant SHAP

Selon les représentations visuelles des figures 2.10 et 2.11, il est observé que l'ensemble de données NSL-KDD affiche les valeurs SHAP les plus importantes pour les caractéristiques src_bytes, dst_bytes, service, dst_host_diff_srv_rate, dst_host_serror_rate, rerror_rate, dst_host_rerror_rate, dst_host_same_src_port_rate, count, et hot. En revanche, l'ensemble de données UNSW-NB15 met en évidence les 10 caractéristiques les plus notables suivantes : sttl, sbytes, ct_dst_src_ltm, proto, dur, ct_srv_dst, ct_state_ttl, response_body_len, sloss, et dbytes. L'influence de ces caractéristiques sur la prédiction de la cible a été déterminée en fonction de leurs valeurs SHAP respectives.

En outre, il peut être remarqué dans les figures 2.9 et 2.8 que la technique d'information mu-

tuelle utilisée pour l'ensemble de données NSL-KDD met en évidence plusieurs caractéristiques, notamment `srv_count`, `logged_in`, `dst_host_srv_count`, et `dst_host_srv_diff_host_rate`, comme faisant partie des 10 caractéristiques ayant les scores les plus significatifs. Cependant, les valeurs SHAP n'attribuent pas un classement élevé à ces caractéristiques dans le top 10. À la place, des caractéristiques telles que `dst_host_serror_rate`, `error_rate`, `dst_host_serror_rate`, et `hot` reçoivent des classements plus élevés. Cette divergence observée suggère que l'évaluation de la pertinence des caractéristiques peut différer entre l'information mutuelle et les valeurs SHAP, soulignant ainsi les perspectives et critères distincts adoptés par ces méthodologies.

Après avoir utilisé les algorithmes d'information mutuelle et de valeurs SHAP pour identifier les caractéristiques les plus significatives des deux ensembles de données, un total de quatre ensembles de données réduits ont été dérivés. Ces ensembles ne contiennent que les 10 caractéristiques principales déterminées par les algorithmes en question. Le résumé de ces quatre ensembles de données réduits est présenté dans le tableau 2.3.

TABLE 2.3 – *Caractéristiques des ensembles de données réduits*

Ensemble de données réduit	Caractéristiques incluses
Red_NSL_MI	5, 3, 23, 6, 24, 12, 35, 33, 36, 37
Red_NSL_ShapV	5, 6, 3, 35, 38, 27, 40, 36, 23, 10
Red_UNSW_MI	4, 9, 24, 5, 6, 1, 32, 25, 29, 41
Red_UNSW_ShapV	7, 4, 33, 40, 1, 38, 29, 27, 11, 5

Impact des caractéristiques sélectionnées sur les données

Les figures 2.12, 2.13, 2.14 et 2.15 illustrent l'influence des caractéristiques sélectionnées en utilisant les valeurs de l'Information Mutuelle (MI) et SHAP sur les deux ensembles de données.

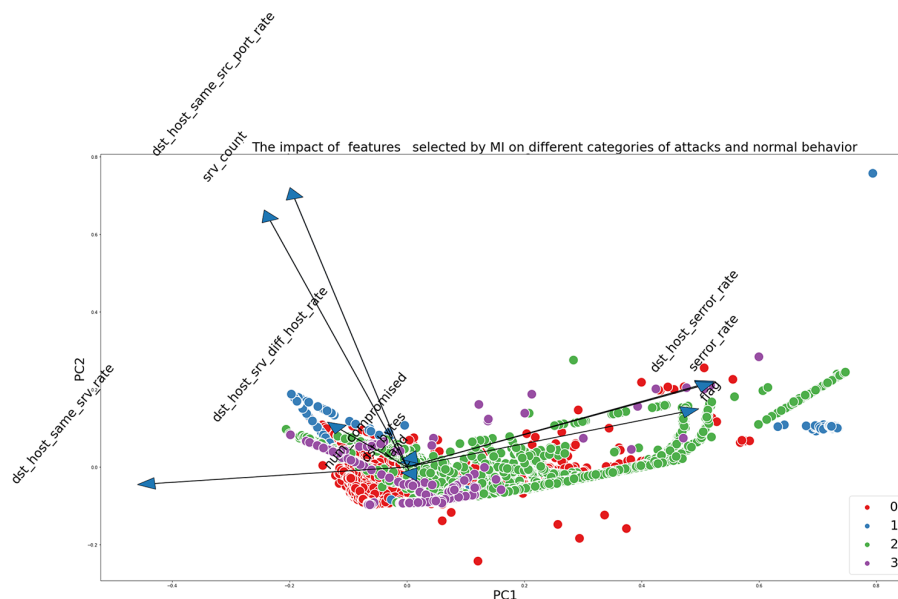
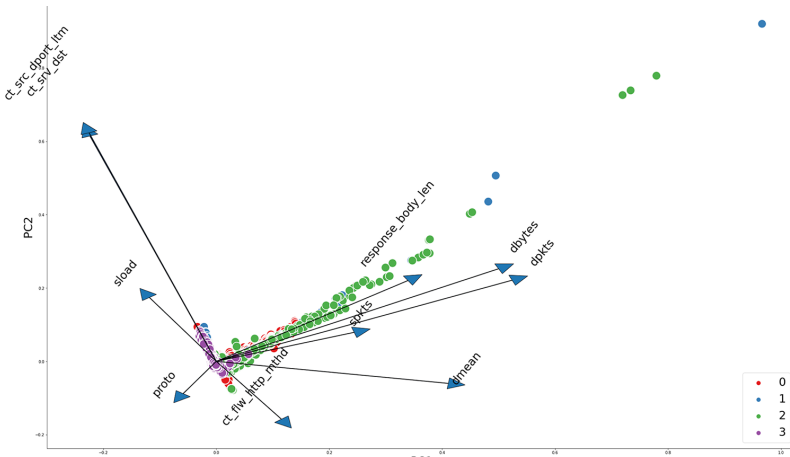
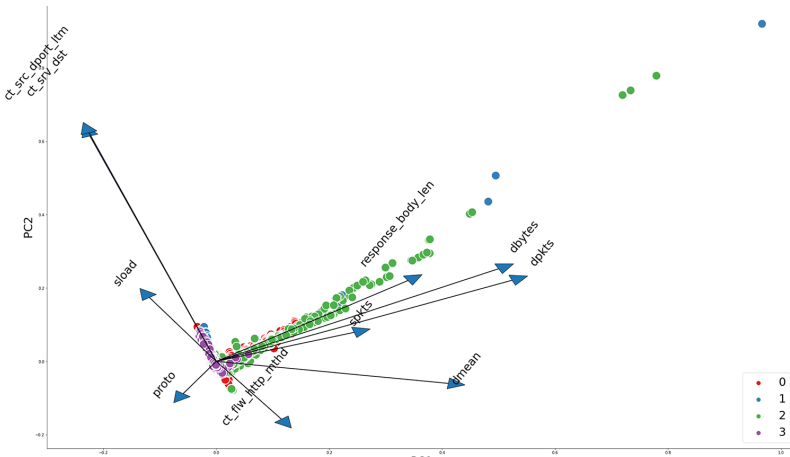
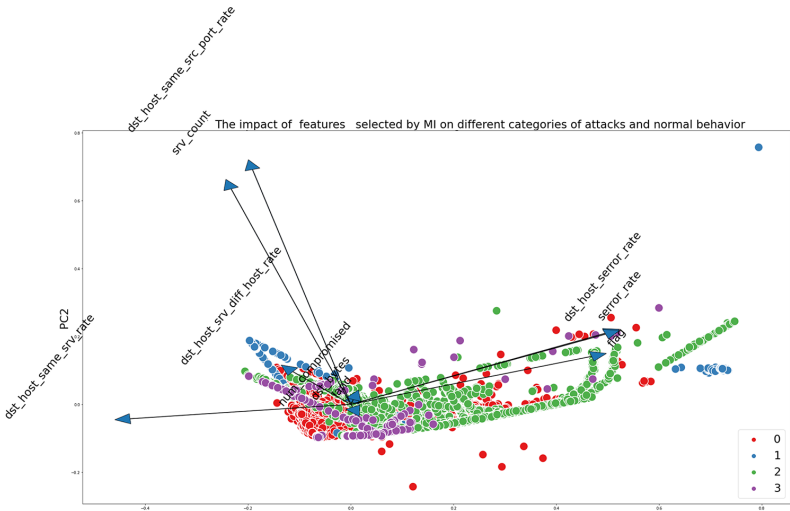


FIGURE 2.12 – *Impact des caractéristiques sélectionnées par MI sur les différentes catégories d'attaques et le comportement normal pour l'ensemble de données NSL-KDD*



Les deux ensembles de données démontrent que certaines caractéristiques exercent une influence substantielle sur des types spécifiques d'attaques. Dans le contexte de l'ensemble de données UNSW-NB15, en se concentrant spécifiquement sur l'attaque 2 (Exploits), il est évident que certaines caractéristiques, à savoir `response_body_len`, `dpkts`, `sbytes`, et `spkts`, présentent un impact notable sur la détection de cette attaque. Cette observation souligne l'importance de ces attributs pour identifier et comprendre efficacement cette forme spécifique d'attaque.

3.7.2 Résultats du modèle d'apprentissage superficiel PV-DM

Nous utilisons le modèle PV-DM (ParagraphVector-Distributed Memory) pour construire des vecteurs numériques qui sont ensuite utilisés comme entrées pour les modèles d'apprentissage automatique.

Un total de six ensembles de vecteurs d'embedding sont produits, comprenant :

- Quatre ensembles dérivés des ensembles de données réduits spécifiés dans le tableau 2.3.
- Deux ensembles obtenus à partir des ensembles de données originaux (NSL-KDD et UNSW-NB15), qui incluent toutes les caractéristiques.

Le modèle PV-DM est conçu avec les paramètres indiqués dans le tableau 2.4.

TABLE 2.4 – *Tableau des paramètres du modèle PV-DM (ParagraphVector-Distributed Memory)*

Paramètres du modèle PV-DM	Valeurs
Nombre de fonctionnalités	Toute, 10
<code>min_count</code>	5
<code>window</code>	8
<code>vector_size</code>	100
<code>sample</code>	1e-4
<code>negative</code>	5
<code>workers</code>	4
<code>alpha</code>	0.025
<code>min_alpha</code>	0.025
<code>epochs</code>	100

L'objectif de notre étude est d'améliorer notre approche en déterminant la méthode de sélection des caractéristiques et le classificateur les plus précis, en utilisant des métriques de performance telles que la précision, la précision spécifique (precision), le rappel, le score F1 et l'AUC.

Pour ce faire, nous avons divisé les vecteurs d'embedding PV-DM de chaque ensemble en ensembles distincts pour l'entraînement et les tests, comme indiqué dans le tableau 2.5. Les classificateurs Random Forest et XGBoost ont été utilisés pour classifier les données, car ils ont démontré leur efficacité dans la catégorisation des attaques dans plusieurs études, telles que celles référencées en [113] [114].

Les résultats de ces classifications sont présentés dans les figures 2.16 et 2.17.

TABLE 2.5 – Taille de l'ensemble d'entraînement et de l'ensemble de test pour chaque ensemble

Vecteurs d'embedding de	Taille du train	Taille du test
Red_NSL_MI	125061	53279
Red_NSL_ShapV	125061	53279
NSL-KDD	125061	53279
Red_UNSW_MI	176514	44128
Red_UNSW_ShapV	176514	44128
UNSW_NB15	176514	44128

Les métriques de performance, à savoir la précision (accuracy), le rappel (recall), la précision spécifique (precision), le score F1 et l'AUC, pour chaque modèle, sont illustrées dans les figures 2.16 et 2.17. De plus, la figure 2.18 représente la courbe ROC (Receiver Operating Characteristic) pour chaque classificateur.

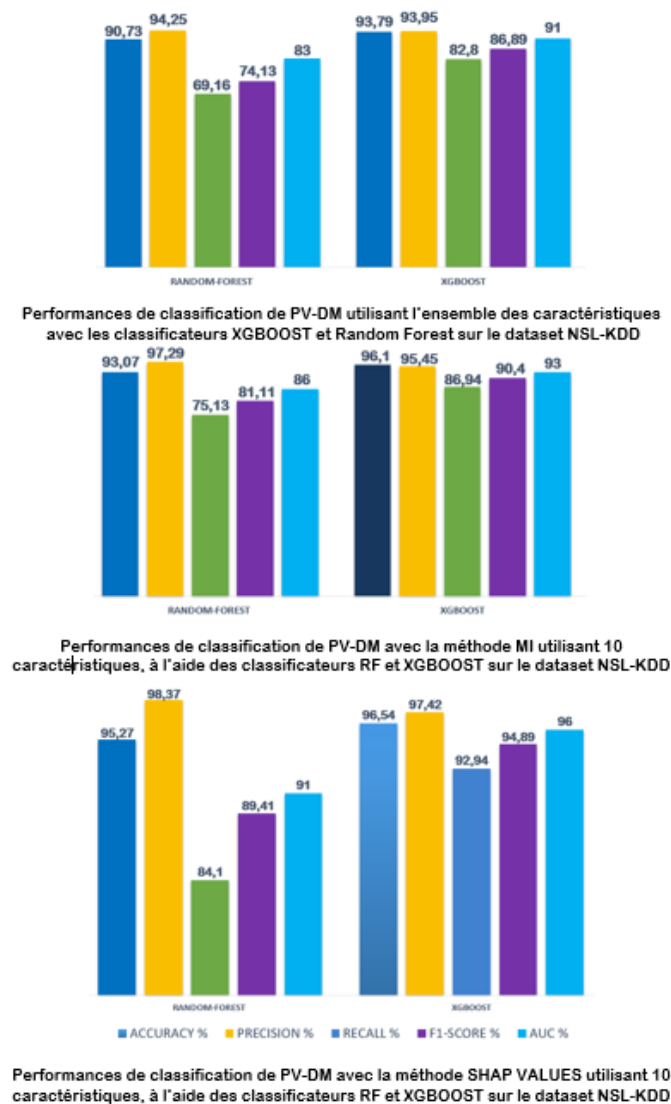


FIGURE 2.16 – Performances de classification sur les ensembles NSL-KDD

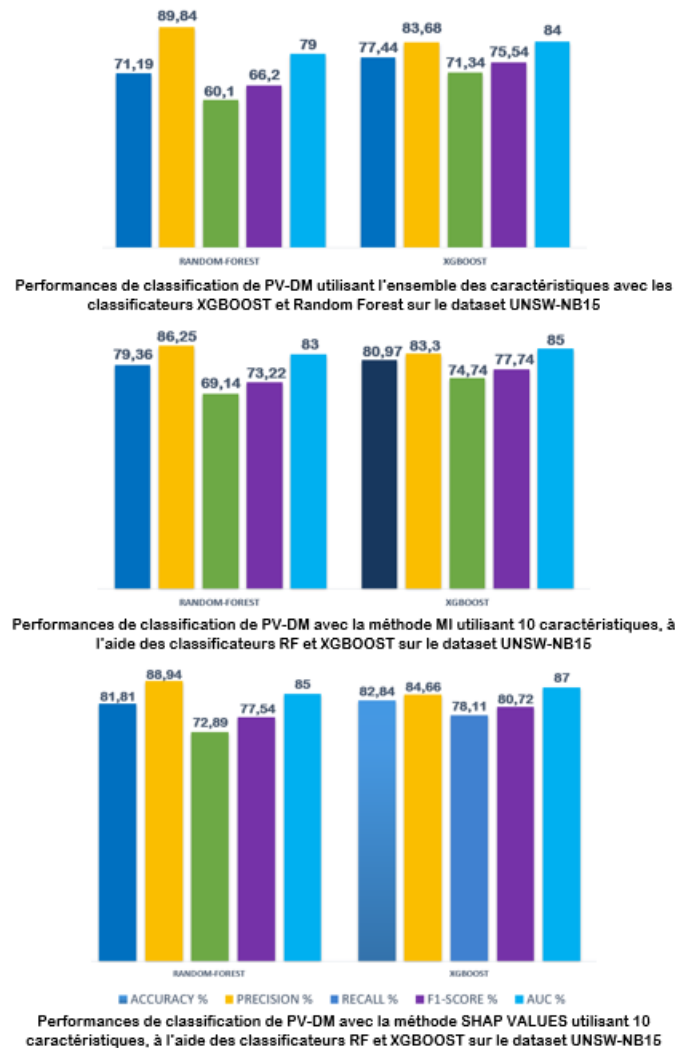


FIGURE 2.17 – Performances de classification sur les ensembles UNSW15.

Une première observation montre clairement que l'inclusion de données pertinentes améliore la précision de la catégorisation des attaques, comme le démontrent les résultats obtenus. Plus précisément, l'utilisation des techniques d'Information Mutuelle (MI) ou des valeurs SHAP conduit à des scores plus élevés dans les métriques d'évaluation, par rapport à la classification des données en utilisant toutes les caractéristiques disponibles.

De plus, d'après les résultats illustrés dans les figures 2.16 et 2.17, il est évident que le modèle PV-DM associé au classificateur XGBOOST a systématiquement montré une meilleure performance en termes de précision (accuracy), rappel (recall) et score F1 pour les deux ensembles de données, en comparaison avec le classificateur RANDOM-FOREST. L'évaluation de ces métriques revêt une importance significative pour analyser l'efficacité des résultats de classification.

En outre, il est important de souligner que le modèle PV-DM, lorsqu'il est associé à l'algorithme RANDOM-FOREST, a présenté des scores de précision notables, comme indiqué par les techniques d'Information Mutuelle (MI) et les valeurs SHAP. À l'inverse, il est évident que lorsque l'on compare l'efficacité des valeurs MI et SHAP en tant que techniques de sélection de caractéristiques, l'utilisation constante des valeurs SHAP conduit à des résultats de performance supérieurs.

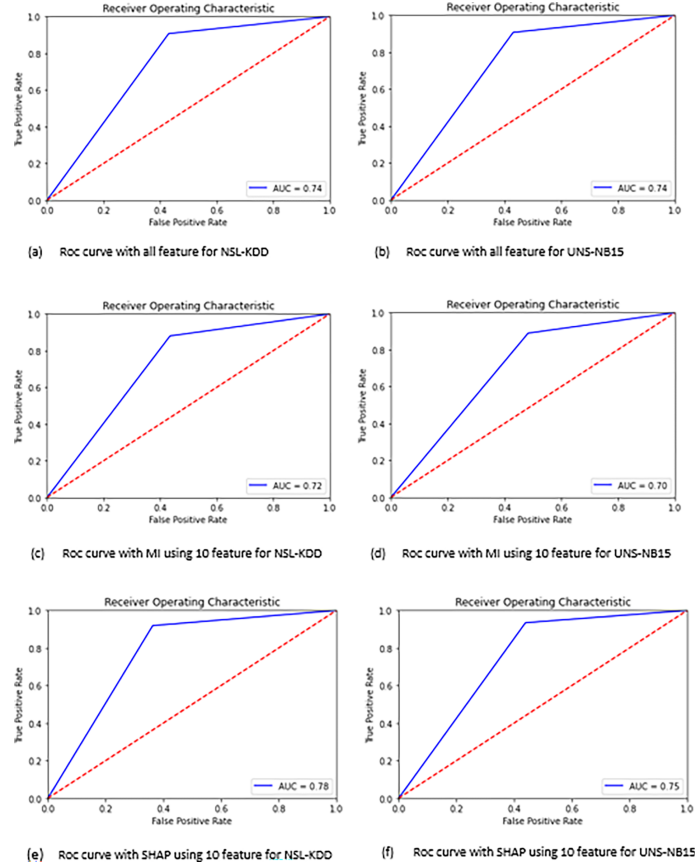


FIGURE 2.18 – Courbe ROC du classificateur XGBOOST.

Bien que l'Information Mutuelle (MI) puisse ne pas afficher le même niveau de performance que les valeurs SHAP, il apparaît que l'intégration de l'Information Mutuelle dans le modèle PV-DM améliore les performances sur plusieurs métriques et classificateurs par rapport à l'utilisation de PV-DM avec l'ensemble des caractéristiques.

Afin de mettre en évidence les performances de classification de notre système de détection d'intrusion proposé, nous fournissons une représentation visuelle de la courbe ROC (Receiver Operating Characteristic) pour chaque classificateur, comme illustré dans la figure 2.18.

D'après les courbes présentées, il est évident que notre modèle proposé démontre une capacité remarquable à réduire efficacement l'occurrence d'attaques incorrectement étiquetées comme des comportements normaux, tout en identifiant et capturant avec succès des attaques inhabituelles. Cette caractéristique revêt une importance significative dans des scénarios pratiques, en particulier pour les professionnels de l'informatique.

3.7.3 Comparaison de notre approche avec LSTM

Dans cette section, une analyse comparative est réalisée entre le modèle proposé et l'algorithme LSTM, largement reconnu comme une méthode très efficace et largement utilisée pour la détection d'intrusion. L'objectif de cette analyse comparative est d'évaluer l'efficacité de notre modèle, en accordant une attention particulière à la réduction des occurrences de faux négatifs.

Les réseaux LSTM ont été implémentés dans le but d'identifier des schémas, en utilisant les ensembles réduits tels que décrits précédemment dans le tableau 2.3. La conception architecturale

de ces réseaux comportait une seule couche cachée. Les valeurs des paramètres du modèle LSTM proposé sont présentées dans le tableau 2.6.

TABLE 2.6 – Paramètres du modèle LSTM

Paramètres	Valeurs LSTM
Fonction d'activation	Softmax
Fonction de perte	Entropie croisée catégorielle clairsemée
Optimiseur	Nadam
Taux d'apprentissage	0.002
Epsilon	1e-08
Déclin du calendrier	0.004
Numéro d'époque	20
Abandonner	0.3

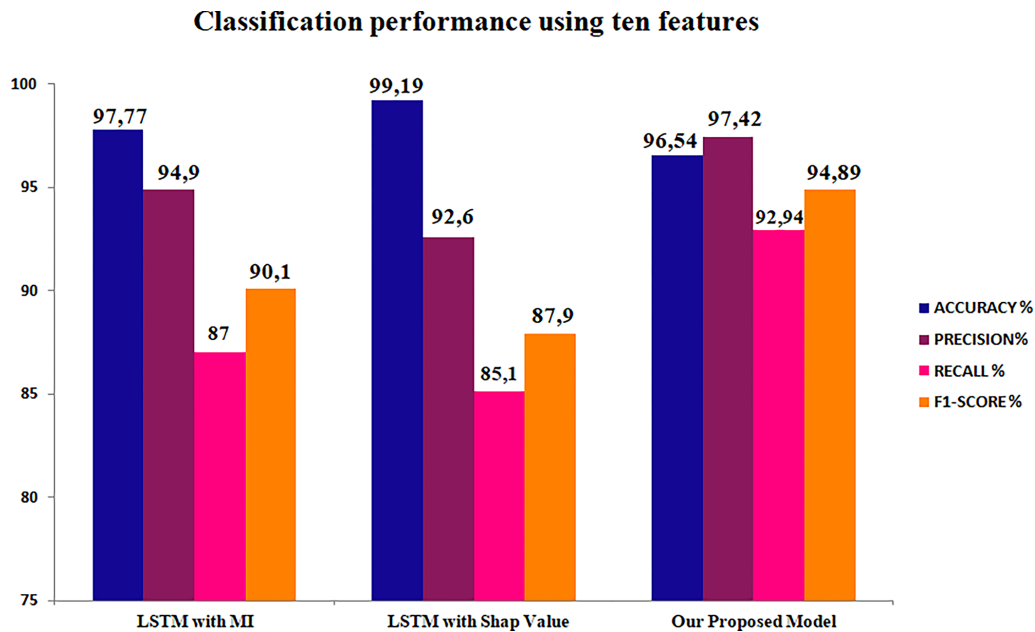


FIGURE 2.19 – Performances de classification du modèle LSTM sur les ensembles réduits de NSL-KDD.

Le modèle LSTM a démontré le niveau de précision le plus élevé lorsqu'il a été utilisé avec les 10 attributs fondamentaux identifiés via l'Information Mutuelle (MI) pour les ensembles de données NSL-KDD et UNSW-NB15 (figures 2.19 et 2.20). Cependant, notre modèle a systématiquement obtenu les meilleurs résultats en termes de précision spécifique (precision), rappel (recall) et score F1. De plus, en utilisant les valeurs SHAP comme technique de réduction de la dimensionnalité en combinaison avec LSTM, notre méthodologie proposée a affiché les niveaux de précision, rappel et score F1 les plus élevés pour l'ensemble de données réduit UNSW-NB15.

En revanche, il a été constaté que la performance du modèle Long Short-Term Memory (LSTM) sur l'ensemble de données NSL-KDD était comparable, à l'exception de la métrique de précision globale (accuracy), où LSTM a significativement surpassé nos méthodes.

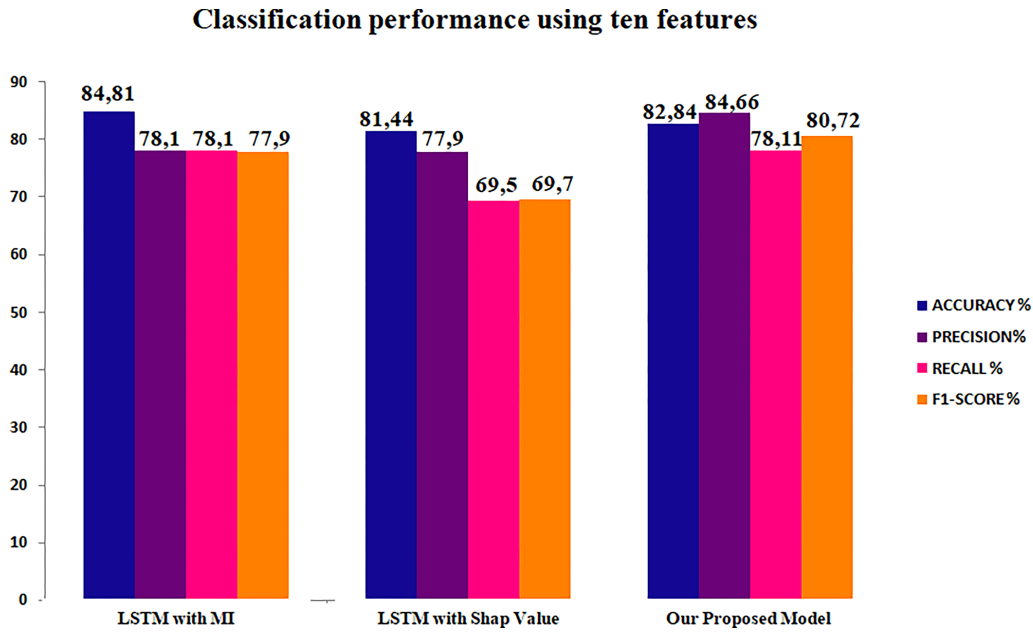


FIGURE 2.20 – Performances de classification du modèle LSTM sur les ensembles réduits de UNSW-NB15.

Les résultats suggèrent que la performance de LSTM est affectée négativement lorsque des attributs inutiles sont éliminés et que seules les caractéristiques pertinentes sont sélectionnées. Cette observation implique que les modèles Long Short-Term Memory (LSTM) rencontrent des difficultés pour distinguer efficacement les différents types d'attaques. Une raison plausible de ce phénomène est que le modèle LSTM, appartenant à la catégorie des réseaux neuronaux récurrents, nécessite un volume substantiel de données pour acquérir une compréhension complète des risques potentiels. De plus, le modèle LSTM considère chaque séquence comme une entité distincte, ce qui peut entraîner une baisse de performance lorsqu'il utilise des techniques de sélection des caractéristiques.

En revanche, notre méthodologie proposée, qui repose sur l'algorithme PV-DM, intègre à la fois le contexte global et local de chaque séquence. Même en présence d'ensembles de données déséquilibrés, notre approche démontre une meilleure efficacité dans la détection des attaques. Cela souligne l'efficacité de notre méthodologie pour détecter et analyser les attaques en prenant en compte les facteurs contextuels.

La performance de notre architecture suggérée, notamment en termes de valeur de rappel (recall), montre une plus grande efficacité en comparaison avec LSTM lorsque des méthodes de sélection des caractéristiques sont employées. Cette étape est d'une importance capitale dans la quête d'amélioration de la précision de détection des différentes formes d'attaques, tout en réduisant simultanément la durée nécessaire à l'entraînement et aux tests, et en augmentant le taux global de détection d'intrusion [115].

3.7.4 Évaluation des performances du modèle via une réduction supplémentaire des caractéristiques

Afin d'améliorer l'évaluation des performances de notre modèle, nous avons choisi de réduire le nombre de caractéristiques à 6 et 4, respectivement, en fonction de leurs scores correspondants dérivés de la méthodologie des valeurs SHAP. Cette réduction du nombre de caractéristiques nous permettra d'évaluer la capacité de notre modèle à maintenir ses performances dans ces circons-

tances modifiées.

Selon les résultats présentés dans le tableau 2.7 de l'analyse de l'ensemble de données NSL-KDD, il est évident que les performances de notre modèle montrent une amélioration significative en termes de précision (accuracy) et de précision spécifique (precision) lorsque le nombre de caractéristiques est réduit à 6 et 4, par rapport aux résultats obtenus avec 10 caractéristiques.

TABLE 2.7 – *Classification performance of our method with 6 and 4 attributes on NSL-KDD*

Caractéristiques	Précision (%)	Précision (%)	Rappel (%)	F1-SCORE (%)
06	98.76	98.86	95.55	97.10
04	98.92	98.92	95.44	96.77

TABLE 2.8 – *Performances de classification de notre méthode avec 6 et 4 caractéristiques sur l'ensemble de données UNSW-NB15*

Caractéristiques	Précision (%)	Précision (%)	Rappel (%)	F1-SCORE (%)
06	82.86	84.07	77.70	80.20
04	83.27	83.55	77.70	79.81

Selon les résultats présentés dans le tableau 2.8, il est évident qu'il y a une augmentation marginale de la précision globale pour l'ensemble de données UNSW-NB15, avec une valeur enregistrée de 82,86%. La précision spécifique est de 84,07%, tandis que le rappel est de 77,70% lorsqu'un ensemble de 6 caractéristiques est utilisé. En revanche, avec un ensemble de 4 caractéristiques, le modèle affiche le taux de précision le plus élevé à 83,27%, un taux de précision spécifique de 83,55% et maintient constamment un rappel de 77,70%.

Ces résultats suggèrent que les configurations avec 4 ou 6 caractéristiques présentent des performances similaires à celles de la configuration avec 10 caractéristiques. Cela souligne que la réduction du nombre de caractéristiques de 10 à 6 ou 4 n'a pas d'impact significatif sur les performances du modèle. Par conséquent, cela permet de proposer un modèle simplifié qui utilise un plus petit ensemble de caractéristiques tout en atteignant des performances de classification satisfaisantes.

De plus, en comparant les performances de notre approche sur les ensembles de données NSL-KDD et UNSW-NB15, nous avons constamment observé des mesures de performance supérieures pour l'ensemble de données NSL-KDD. La différence entre les deux ensembles peut être attribuée à la complexité plus grande de l'ensemble UNSW-NB15 sur plusieurs dimensions.

L'inclusion d'une vaste gamme de scénarios d'attaques actuels et de caractéristiques du trafic réseau typique dans l'ensemble UNSW-NB15 le rend plus complexe à identifier correctement. Cependant, il convient de noter que les attaques observées dans l'ensemble NSL-KDD, ainsi que leurs comportements habituels associés, ne reflètent peut-être pas avec précision le paysage actuel des menaces. En outre, la complexité de l'ensemble UNSW-NB15 est également due aux nombreuses similitudes entre les occurrences normales et celles des attaques à travers diverses variables [116]. Cette similarité pose un défi pour différencier les actions bénignes des actions nuisibles, ce qui entraîne une baisse de la précision de classification.

4. Conclusion

Notre modèle démontre des performances constantes même lorsque le nombre de caractéristiques est réduit à 6 ou 4, comme démontré ici. Néanmoins, il est essentiel de reconnaître qu'il existe des obstacles persistants nécessitant une attention particulière pour exécuter avec succès le cadre proposé dans des situations pratiques. Ces problèmes concernent essentiellement deux aspects fondamentaux : l'exécution en temps réel du système et la compréhension des vecteurs produits par le modèle PV-DM.

3.7.5 Analyse comparative : Évaluation de notre méthode par rapport aux travaux antérieurs

Pour enrichir l'analyse des résultats expérimentaux, un examen comparatif a été réalisé entre les résultats obtenus avec notre approche proposée et les conclusions des recherches antérieures, comme présenté dans les tableaux 2.9 et 2.10. Il est important de reconnaître que cette comparaison agit comme un outil illustratif en raison des variations potentielles dans l'environnement d'exécution, les procédures de prétraitement des données et les processus d'interprétation entre les différents systèmes de détection d'intrusion (IDS).

Néanmoins, les résultats de cette comparaison démontrent que notre modèle affiche des performances supérieures par rapport aux modèles pris en compte dans cette analyse, soulignant ainsi l'efficacité de notre méthode pour traiter ce problème spécifique. Cela implique que la méthodologie choisie est plus appropriée pour relever les défis associés à ce sujet.

TABLE 2.9 – Comparaison entre différents modèles et notre approche sur l'ensemble de données NSL-KDD

Modèle	Méthode de sélection des caractéristiques	Précision (%)	Précision (%)	Rappel (%)	F1-score (%)
LCNN-GRNN [117]	Saut de Grenouille Froissée Améliorée (ESFL)	92.56	-	-	-
DNN [118]	04 features	78.1	-	-	-
Decision Tree with 05 features [119]	CPIO (Optimiseur Inspiré des Pigeons Continu) (05 caractéristiques)	88.3	-	-	88.2
Two-stage classification algorithm (rotation forest and bagging) [120]	Hybride (GA+PSO+ACO) (37 caractéristiques)	85.79	88.0	86.8	-
Proposed model	Valeurs SHAP (04 caractéristiques)	98.92	98.92	95.44	96.77

TABLE 2.10 – Comparaison entre différents modèles et notre approche sur l'ensemble de données UNSW15

Modèle	Méthode de sélection des fonctionnalités	Précision (%)	Precision (%)	Rappel (%)	F1-score (%)
IGAN-IDS [121]	Réseau de Neurones à Propagation Avant	82.53	-	-	-
DNN [118]	-	64.5	61.4	64.5	58.6
Proposed model	Valeurs SHAP (06 caractéristiques)	82.86	84.07	77.70	80.20

4 Conclusion

Dans ce chapitre, nous avons proposé un nouveau système de détection d'intrusions basé sur le modèle ParagraphVector - Distributed Memory (PV-DM), avec intégration des techniques avancées de sélection de caractéristiques pour améliorer la performance du classificateur. Plus précisément, nous avons exploité les valeurs SHAP afin d'identifier les caractéristiques les plus discriminantes et d'optimiser ainsi la représentation des données d'intrusion. Le modèle résultant a ensuite été entraîné en utilisant XGBoost, un classificateur puissant et robuste face aux jeux de données déséquilibrés.

Nous avons également mené une étude comparative approfondie entre notre approche et la méthodologie LSTM, un modèle d'apprentissage profond largement utilisé pour la détection des

intrusions. Nos résultats expérimentaux démontrent que notre approche surpasse LSTM en termes de précision, de robustesse et d'efficacité computationnelle, et ce, même en prenant en compte la nature déséquilibrée des ensembles de données NSL-KDD et UNSW-NB15. L'utilisation d'un nombre réduit de caractéristiques pertinentes a permis d'améliorer l'interprétabilité du modèle tout en réduisant le coût de calcul.

De manière significative, notre approche a maintenu des performances élevées malgré une réduction substantielle du nombre de caractéristiques, mettant en évidence l'efficacité de notre méthodologie d'extraction et de sélection des caractéristiques :

- Sur l'ensemble de données UNSW-NB15 :
 - Nombre de caractéristiques sélectionnées : 6
 - Exactitude globale : 82,86%
 - Précision spécifique : 84,07%
 - Rappel : 77,70%
 - Score F1 : 80,20%
- Sur l'ensemble de données NSL-KDD :
 - Nombre de caractéristiques sélectionnées : 4
 - Exactitude globale : 98,92%
 - Précision spécifique : 98,92%
 - Rappel : 95,44%
 - Score F1 : 96,77%

Ces résultats indiquent que notre modèle parvient à capturer efficacement les schémas d'intrusion tout en conservant une grande stabilité même avec un jeu de caractéristiques fortement réduit. Cela souligne l'importance d'une sélection pertinente des features pour optimiser la classification sans compromettre la performance globale du modèle.

Dans les recherches futures, nous nous concentrerons sur l'amélioration de notre modèle afin de traiter plus efficacement le problème des classes minoritaires dans la détection des intrusions. En effet, l'un des défis majeurs des systèmes de détection actuels est l'identification précise des attaques rares, qui sont souvent sous-représentées dans les ensembles de données, entraînant un déséquilibre qui peut biaiser la classification. Plutôt que d'adopter des méthodes classiques telles que la rééchantillonnage (oversampling/undersampling) ou la génération artificielle de données (SMOTE), nous envisageons d'explorer une nouvelle approche basée sur l'extraction automatique des caractéristiques discriminantes propres à chaque type d'attaque. Cette méthodologie visera à réduire l'impact du déséquilibre des données tout en maintenant une fidélité élevée à la distribution réelle des cyberattaques.

De plus, une extension de cette étude pourrait inclure l'intégration d'autres modèles hybrides, combinant l'apprentissage profond et les techniques d'apprentissage supervisé classiques, afin d'améliorer encore davantage la détection des menaces émergentes. L'application de techniques telles que l'apprentissage auto-supervisé ou les graphes de relations entre événements pourrait également offrir des pistes prometteuses pour enrichir la compréhension des cyberattaques complexes.

En conclusion, notre approche basée sur PV-DM et la sélection intelligente des caractéristiques constitue une alternative performante et efficace aux modèles de réseaux récurrents comme LSTM, tout en réduisant le coût computationnel et la complexité. Les résultats obtenus démontrent le potentiel de cette approche pour la détection en temps réel des intrusions, ouvrant ainsi la voie à de nouvelles perspectives d'amélioration pour les systèmes de cybersécurité avancés.

3

Une approche efficace pour la détection des attaques rares

Les attaques rares ou à faible fréquence, telles que R2L (Remote to Local) et U2R (User to Root), bien que marginales en nombre, représentent un niveau de dangerosité extrêmement élevé. Ces attaques visent généralement à obtenir un accès non autorisé à un système ou à élever les privilèges d'un utilisateur compromis pour prendre le contrôle total d'un hôte, ce qui peut avoir des conséquences catastrophiques.

Un exemple frappant d'attaque U2R est l'exploitation de la vulnérabilité "Dirty COW" (CVE-2016-5195), une faille critique du noyau Linux restée exploitable pendant près de neuf ans avant d'être rendue publique. Elle permettait à un utilisateur local non privilégié de modifier des fichiers en lecture seule, conduisant à une élévation de privilèges jusqu'au niveau root. Cette faille a été activement exploitée dans des campagnes ciblées contre des serveurs Linux, y compris dans des environnements cloud et des infrastructures critiques. Les conséquences de ces attaques ont été multiples : exfiltration de données confidentielles, installation de backdoors persistantes, et prise de contrôle complète de systèmes stratégiques.

Sur le plan économique, plusieurs entreprises, notamment dans les secteurs de la finance et de la santé, ont subi des pertes importantes à cause de cette vulnérabilité : interruption de service, amendes réglementaires, et atteinte à leur réputation. Par exemple, certaines estimations ont évalué à plusieurs millions de dollars les pertes cumulées liées à des attaques exploitant cette faille, notamment à travers des intrusions suivies de ransomwares ou de vols de données critiques.

La rareté de ce type d'attaque dans les jeux de données d'apprentissage masque leur impact potentiel et complique considérablement leur détection. En apprentissage automatique, ce phénomène est connu sous le nom de déséquilibre de classe, où la classe minoritaire (les attaques critiques) est éclipsée par la classe majoritaire (le trafic légitime), ce qui pousse souvent les modèles à ignorer ou sous-estimer ces menaces.

L'objectif de ce chapitre est de proposer une méthode pour améliorer la détection de ces attaques à faible fréquence en sélectionnant des features spécifiques à chaque type d'attaque à l'aide de la méthode SHAP-RFECV, puis en utilisant XGBoost comme classifieur. Cette approche permet au modèle de se focaliser sur les attributs propres aux attaques rares, renforçant ainsi leur détection sans recourir à l'augmentation artificielle des données, tout en préservant l'intégrité des données originaux.

1 Introduction

Certaines intrusions, bien que rares en fréquence, peuvent engendrer des dommages considérables lorsqu'elles passent inaperçues. C'est notamment le cas des attaques de type ver informatique (worm), qui illustrent de manière frappante les risques liés aux classes minoritaires dans les systèmes de détection d'intrusion. Les vers sont des logiciels malveillants capables de se répliquer et se propager automatiquement via les réseaux, en exploitant des vulnérabilités logicielles ou des configurations réseau inadéquates. Contrairement à d'autres formes de malwares, ils n'ont pas besoin d'intervention humaine pour se propager, ce qui leur permet de se diffuser très rapidement d'une machine à l'autre.

Cette capacité d'auto-réplication peut entraîner une consommation massive de ressources réseau et système, provoquant des interruptions de service étendues, des pertes de données critiques, et des atteintes majeures à l'infrastructure. Parmi les actions nuisibles qu'ils peuvent exécuter, on retrouve la destruction de données, le vol d'informations sensibles, ou encore l'interruption de fonctions réseau vitales.

La détection automatique de ces intrusions, notamment celles issues de classes minoritaires comme U2R, R2L ou les vers, représente un défi de taille en cybersécurité. De nombreuses approches ont été proposées dans la littérature [122] [123] [124] [125] pour adresser ce problème, mais plusieurs facteurs continuent de compliquer la conception de systèmes de détection robustes. D'abord, l'évolution constante des techniques d'attaque rend rapidement obsolètes les modèles statiques. Ensuite, la nécessité d'une détection en temps réel impose des contraintes de performance et de réactivité. Enfin, les jeux de données fortement déséquilibrés, où les attaques rares représentent une infime minorité, posent un réel obstacle à l'apprentissage efficace des modèles de classification.

Ce déséquilibre de classes affecte profondément les performances des algorithmes de machine Learning, qui ont tendance à privilégier la classe majoritaire (trafic légitime) au détriment de la classe minoritaire (trafic malveillant). En conséquence, les modèles offrent souvent d'excellents résultats globaux, mais échouent à détecter les comportements atypiques, pourtant susceptibles de causer des dommages graves.

Dans ce chapitre, nous introduisons une approche innovante et efficace pour améliorer la détection des attaques rares dans les systèmes de détection d'intrusion réseau, sans recourir à l'augmentation artificielle de données. Notre méthode repose sur une démarche en plusieurs étapes :

1. Division des ensembles de données basée sur la fréquence des attaques afin d'individualiser le traitement des classes minoritaires.
2. Sélection de caractéristiques ciblées à l'aide d'une combinaison de valeurs SHAP (SHapley Additive Explanations) et de l'algorithme RFECV (RecursiveFeature Elimination with Cross-Validation). Cette intégration permet d'identifier avec précision les attributs les plus discriminants pour chaque type d'attaque.
3. Fusion intelligente de caractéristiques par fréquence, visant à améliorer leur capacité prédictive tout en conservant la cohérence des signaux d'origine.
4. Personnalisation du modèle XGBoost pour une meilleure gestion des jeux de données déséquilibrés, en mettant l'accent sur les métriques adaptées telles que la précision, le rappel, et le score F1, particulièrement pour les classes sous-représentées.

Nous avons validé notre approche à travers une évaluation rigoureuse sur l'ensemble de données UNSW-NB15, suivie d'une validation croisée sur les jeux de données CICIDS2019 et CI-

CIoT2023, reconnus pour leur diversité et leur richesse en scénarios d'attaques.

Les résultats expérimentaux ont démontré des améliorations significatives sur les performances de classification, en particulier pour les classes minoritaires. Ces résultats confirment l'efficacité de notre approche face aux défis posés par les données déséquilibrées, tout en préservant l'intégrité des données d'origine.

Cette contribution propose ainsi une solution complète et applicable à l'échelle opérationnelle pour améliorer la détection des attaques critiques sous-représentées, et offre une nouvelle perspective dans le traitement intelligent des jeux de données déséquilibrés en cybersécurité.

Ce chapitre est structuré comme suit :

- Section 2 : Présentation du problème de déséquilibre des classes dans les systèmes de détection d'intrusions ;
- Section 3 : Introduction aux concepts clés utilisés dans cette étude ;
- Section 4 : Détail de la méthodologie proposée ;
- Section 5 : Analyse et discussion des résultats expérimentaux ;
- Section 6 : Conclusion et perspectives de recherche futures.

2 Travaux connexes

Afin de relever le défi majeur du déséquilibre des classes dans les ensembles de données de sécurité réseau, en particulier pour les classes sous-représentées telles que les attaques Worm, R2L (Remote to Local) et U2R (User to Root), les chercheurs ont proposé un éventail de solutions innovantes. L'objectif de ces solutions est d'améliorer les capacités de détection des systèmes de détection d'intrusion (IDS) pour ces attaques moins fréquemment détectées mais particulièrement dangereuses.

Liu et al. [126] ont introduit une nouvelle approche appelée Difficult Set Sampling Technique (DSSTE) afin de résoudre le problème du déséquilibre des classes dans les ensembles de données tels que NSL-KDD et CSE-CIC-IDS2018. Cet algorithme génère des échantillons synthétiques pour les classes minoritaires, améliorant ainsi les performances des classificateurs.

Bedi et al. [127] ont proposé Siam-IDS, un système de détection d'intrusion (IDS) basé sur un réseau neuronal siamois (Siamese Neural Network). L'objectif principal de Siam-IDS est de traiter le problème du déséquilibre des classes sans recourir aux méthodes de rééquilibrage conventionnelles, telles que le suréchantillonnage (oversampling) et le sous-échantillonnage (undersampling). Leur méthodologie a abouti à des valeurs de rappel supérieures pour les catégories d'attaques R2L et U2R, démontrant le potentiel de l'apprentissage profond dans la gestion du déséquilibre des classes. Une autre avancée intéressante repose sur l'utilisation des réseaux antagonistes génératifs (GANs) pour renforcer les données dans les catégories sous-représentées.

Alabrah [128] a proposé l'utilisation d'un modèle GAN pour générer des exemples artificiels des catégories sous-représentées, améliorant ainsi l'efficacité des classificateurs d'apprentissage automatique sur l'ensemble de données UNSW-NB15. Cette méthode a montré des résultats pro-

metteurs en améliorant les niveaux de précision, de rappel et de score F1 dans la détection des attaques sur les classes minoritaires.

Elghalhoud et al. [129] ont introduit un système de détection d'intrusion réseau (NID) basé sur un réseau neuronal convolutif (CNN), spécialement conçu pour traiter le problème des ensembles de données déséquilibrés et permettant de classifier avec succès divers types d'attaques.

Kasongo et al. [130] ont développé un système de détection d'intrusion (IDS) intégrant plusieurs algorithmes d'apprentissage automatique, tels que la régression linéaire (LR), la forêt aléatoire (RF), le naïf Bayes (NB), l'arbre de décision (DT), les arbres supplémentaires (ET) et le Extreme Gradient Boosting (XGBoost). En outre, ils ont utilisé un algorithme génétique (GA) pour sélectionner les caractéristiques et ont incorporé la forêt aléatoire comme fonction d'évaluation de l'algorithme génétique. L'évaluation du modèle à l'aide de l'ensemble de données UNSW-NB15 a permis d'obtenir une valeur de l'aire sous la courbe (AUC) de 0,98 et une précision de 87,61%.

Gao et al. [131] ont introduit un système IDS basé sur l'apprentissage automatique en ensemble (ensemble learning), utilisant la réduction de dimension par analyse en composantes principales (PCA) pour l'extraction des caractéristiques. Leur méthodologie intègre plusieurs classificateurs, tels que l'arbre de décision (DT), la forêt aléatoire (RF), le plus proche voisin (KNN), le réseau neuronal profond (DNN) et MultiTree, à l'aide d'un algorithme d'ensemble basé sur le vote majoritaire, attribuant des poids à chaque classificateur afin d'améliorer la précision globale. L'évaluation expérimentale sur l'ensemble de données NSL-KDD a permis d'obtenir une précision de 85,2%, dépassant celle de chaque classificateur pris individuellement. Toutefois, l'efficacité du modèle est compromise lorsqu'il est appliqué à l'analyse d'attaques rares et peu fréquentes.

Kasongo et al. [132] ont examiné un système IDS novateur utilisant un réseau neuronal profond feed-forward (FFDNN) couplé à une unité d'extraction de caractéristiques basée sur un wrapper (WFEU). Les résultats ont montré que le système fonctionne bien en classification binaire, mais affiche des performances inférieures en classification multiclasse.

Les auteurs de l'étude [133] ont comparé les performances de divers classificateurs, notamment l'arbre de décision (DT), la régression logistique simple (SLR), le naïf Bayes (NB), le perceptron multicouche (MLP), la machine à vecteurs de support (SVM), la forêt aléatoire (RF) et Zero Rule. Ils ont conclu que l'arbre de décision (DT) était le plus performant pour détecter les intrusions.

Pham et al. [134] ont proposé un cadre IDS utilisant la sélection de caractéristiques par gain ratio, combinée à un modèle d'ensemble basé sur le bagging, avec l'algorithme J48 comme classificateur de base. L'algorithme J48 a permis de surmonter le biais en faveur des caractéristiques ayant un grand nombre de valeurs en appliquant la sélection par gain ratio. Leur méthodologie a été comparée à un ensemble utilisant le KNN et la sélection de caractéristiques par corrélation avec le naïf Bayes. Les résultats expérimentaux ont montré que leur cadre améliore la précision de classification et réduit le taux de fausses alertes.

Devikrishna et al. [135] ont implémenté l'architecture du perceptron multicouche (MLP) pour classifier les attaques en six catégories et détecter les intrusions. Cependant, l'approche MLP a été jugée inadéquate en raison de ses sorties non pertinentes.

Les auteurs de l'étude [136] ont proposé une nouvelle méthode de détection des intrusions qui divise les données réseau en sous-ensembles plus petits à l'aide de l'algorithme d'arbre de décision C4.5, suivi du développement de plusieurs modèles SVM pour chaque sous-ensemble.

Cette approche réduit efficacement la complexité temporelle et améliore le taux de détection des attaques inconnues.

Maseer et al. [137] ont développé un IDS hybride basé sur l'apprentissage profond pour l'IoT, utilisant un réseau de croyance profonde pondéré (DBN) combiné à un modèle Gaussien-Bernoulli, un réseau neuronal profond pondéré, et une machine de Boltzmann restreinte. L'ensemble de données CICIDS2017 a été utilisé pour évaluer les performances du modèle, qui a obtenu un taux de précision de 99,38 % pour les attaques Web et de 99,99 % pour l'ensemble des attaques.

Guezzaz et al. [138] ont amélioré la qualité des données des ensembles NSL-KDD et CIC-IDS2017 en développant un modèle IDS basé sur l'algorithme de l'arbre de décision (DT). Les résultats de la comparaison avec d'autres modèles utilisant les mêmes ensembles de données ont montré que le modèle proposé a atteint une précision de 99,42 % pour NSL-KDD et de 98,8 % pour CIC-IDS2017.

Les recherches susmentionnées apportent des contributions substantielles au domaine des systèmes de détection d'intrusion réseau (NIDS) en exploitant diverses techniques d'apprentissage automatique et approches de sélection des caractéristiques. Toutefois, plusieurs limites dans leurs méthodologies pourraient compromettre l'efficacité globale de la détection des intrusions, en particulier pour les attaques de classes minoritaires. Premièrement, l'approche consistant à sélectionner les variables pertinentes à partir de l'ensemble complet des données, comme adoptée dans ces études, peut ne pas être la plus efficace pour classer certains types d'attaques. Cette approche repose sur l'hypothèse que les caractéristiques les plus pertinentes pour l'ensemble du jeu de données le sont également pour chaque type d'attaque spécifique. Or, certaines attaques présentent des modèles ou comportements uniques nécessitant des ensembles de critères spécifiques pour une détection précise.

Le processus de détection d'intrusion peut ainsi être vulnérable si des caractéristiques significatives, essentielles à la détection d'attaques spécifiques, sont ignorées en raison de l'absence d'une sélection de caractéristiques adaptée à chaque type d'attaque. De plus, la majorité de ces recherches considèrent l'exactitude (accuracy) comme le critère principal d'évaluation de leurs modèles. Bien que l'exactitude soit indéniablement une mesure cruciale, elle ne fournit pas à elle seule une évaluation complète de l'efficacité d'un modèle dans les scénarios de détection d'intrusion. Dans le cadre des systèmes de détection d'intrusion réseau (NIDS), le paramètre du rappel (recall), également connu sous le nom de taux de vrais positifs, revêt une importance capitale. Le rappel mesure le ratio des attaques correctement détectées par le système par rapport au nombre total d'attaques réelles. Un rappel élevé est essentiel dans la détection d'intrusion, car l'échec de détection des attaques réelles (faux négatifs) peut entraîner des conséquences plus graves que la classification erronée du trafic normal en tant qu'attaque (faux positifs). En privilégiant l'exactitude, ces études peuvent involontairement minimiser l'importance de la réduction des attaques manquées, un élément crucial pour un NIDS efficace.

En outre, l'absence de prise en compte des attaques ciblant des classes minoritaires constitue une lacune majeure de ces recherches. Les ensembles de données de détection d'intrusion présentent souvent un déséquilibre des classes, où le nombre d'occurrences d'attaques, en particulier pour certains types spécifiques, est beaucoup plus faible que celui du trafic normal.

Cette disparité dans la distribution des classes constitue un défi pour les modèles d'apprentissage automatique, car ceux-ci ont tendance à favoriser la classe dominante (trafic normal), au détriment des classes moins représentées (attaques spécifiques). Si ce déséquilibre n'est pas cor-

TABLE 3.1 – *Travaux connexes et de leurs limites.*

Réf.	Architecture	Dataset	Limitation
[46]	LSTM	NSL-KDD	Le modèle présente des performances insuffisantes face aux attaques R2L et U2R.
[139]	LSTM + Attention	NSL-KDD	Le modèle rencontre des difficultés à classer correctement les attaques U2R, ce qui entraîne de nombreuses classifications erronées en tant qu'instances normales.
[122]	PCA-LSTM	KDD99	En raison du nombre limité d'instances associées aux attaques R2L et U2R par rapport aux autres catégories, les auteurs ont regroupé ces deux types d'attaques dans une seule et même classe.
[140]	MCA-LSTM	NSL-KDD, UNSW-NB15	Les auteurs n'ont pas clairement détaillé les stratégies mises en place pour éviter le surapprentissage lors de l'entraînement.
[141]	Réseau de neurones (ANN)	—	Temps d'exécution élevé.
[142]	KNN + sélection de caractéristiques	Bot-IoT	Faible performance globale.
[143]	MAD-RF (Random Forest)	NSL-KDD	L'exactitude de détection des attaques DDoS ICMP et UDP est inférieure et nécessite des améliorations.
[144]	Naïve Bayes + SVM	NSL-KDD, UNSW-NB15, CIC-IDS2017, Kyoto	Le modèle ne donne pas de meilleures performances pour l'ensemble des dataset, à l'exception de NSL-KDD.
[145]	KNN + RSL	ICS Cyber Attack Dataset	L'exactitude peut être améliorée.
[146]	Corrélation + FLDA	KDD99	La seule métrique utilisée pour évaluer les performances était l'exactitude.
[147]	PCA + Naïve Bayes	NSL-KDD	Pour améliorer la détection des classes d'attaques U2R et R2L, ils ont exclu d'autres types d'attaques.

rectement pris en compte, les modèles risquent de ne pas détecter des menaces rares mais potentiellement catastrophiques. Cette omission souligne la nécessité de stratégies qui ne se limitent pas à l'exactitude globale, mais qui garantissent également une détection efficace des attaques ci-

blant les classes minoritaires, un aspect fondamental pour établir une sécurité réseau robuste et exhaustive.

3 Le problème du déséquilibre des classes

3.1 Définition

L'un des principaux défis rencontrés dans le domaine de l'apprentissage automatique est celui des données déséquilibrées, également connu sous le nom de déséquilibre des classes. Ce problème se produit lorsque les ensembles de données n'illustrent pas une représentation équitable de toutes les classes. Par exemple, une classe peut contenir un nombre d'échantillons largement supérieur aux autres classes. La figure 3.1 illustre la distribution extrêmement déséquilibrée du dataset des attaques utilisé dans nos implémentations.

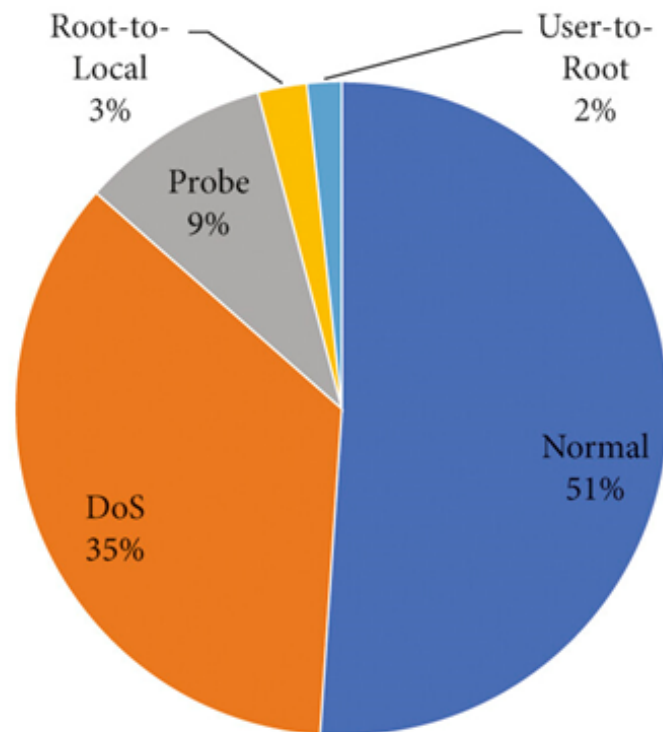


FIGURE 3.1 – *Distribution extrêmement déséquilibrée des classes*

Ce phénomène est souvent constaté dans les ensembles de données du monde réel, en particulier dans le secteur de la détection d'intrusion où les instances

3.2 Problème lié au déséquilibre de classe

Ce problème de déséquilibre des classes peut affecter la précision des modèles de machine learning, le modèle risquerait de favoriser la classe majoritaire, ce qui entraînerait des performances insatisfaisantes pour la classe minoritaire [148] [149]. Cela se produit car les méthodes d'apprentissage traditionnelles ne parviennent pas à classer correctement les nouvelles instances dans de tels contextes. Elles enregistrent une bonne précision uniquement pour les observations de la classe majoritaire, tandis que la précision pour la classe minoritaire reste inacceptable. En d'autres termes, en cas de déséquilibre des données, un classifieur peut prédire toutes les nouvelles observations comme appartenant à la classe majoritaire, en négligeant la classe minoritaire.

De ce fait, Il est donc crucial de prendre en compte le problème du déséquilibre des classes lorsqu'on travaille avec des ensembles de données et d'appliquer les techniques appropriées pour remédier à ce problème.

3.3 Approches proposées pour les données déséquilibrées

Plusieurs techniques ont été proposées dans la littérature pour atténuer les difficultés posées par les ensembles de données déséquilibrés. Ces techniques peuvent être classées en deux grandes catégories comme le montre le schéma 1 :

1. Techniques de ré-échantillonnage au niveau des données (prétraitement) : Cette catégorie de méthodes corrige le déséquilibre des classes en agissant directement sur les données, en les modifiant avant leur utilisation pour l'entraînement du modèle, autrement dit, des techniques de ré-échantillonnage sont employées pour modifier l'ensemble de données d'origine, soit en réduisant le nombre d'échantillons de la classe majoritaire (undersampling), soit en répliquant celui de la classe minoritaire (oversampling), ou encore en combinant ces deux approches [150] [151].
2. Approches d'apprentissage d'Ensemble au niveau algorithmique : Cette méthode traite le problème du déséquilibre des classes au niveau de l'algorithme de classification. Elle repose sur l'utilisation des méthodes d'Ensemble, qui combinent plusieurs classifieurs faibles pour former un classifieur plus performant, améliorant ainsi la classification des datasets déséquilibrés [152] [153] [154]. Parmi les stratégies les plus courantes, on retrouve le bagging et le boosting, qui seront détaillés dans les sections suivantes.

3.3.1 Approche de ré-échantillonnage (Data level)

Cette approche repose sur une phase de prétraitement des données initiales avant de transmettre les données en entrée au modèle prédictif. Ce technique de prétraitement est réalisé en adoptant une approche basée sur les données, connue sous le nom de rééchantillonnage, qui comprend trois stratégies principales : (a) le sous-échantillonnage (undersampling), (b) le sur-échantillonnage (oversampling), et (c) l'approche hybride qui combine les deux. Leur principal avantage est qu'elles sont compatibles avec tout type de classifieur.



FIGURE 3.2 – Méthode d'Oversampling et d'undersampling

A- Méthodes de sur-échantillonnage (Oversampling)

Les méthodes de suréchantillonnage visent à équilibrer les deux classes d'un jeu de données en augmentant le nombre d'observations au sein de la classe minoritaire (voir la figure 3.2). Le principal inconvénient de ces méthodes est qu'en cas de déséquilibre trop important, il est nécessaire

d'ajouter un grand nombre d'observations à la classe minoritaire, ce qui peut entraîner un surapprentissage et une perte d'informations sur la distribution des données. Une autre méthode pour équilibrer les classes tout en minimisant la perte d'informations de la classe majoritaire consiste à augmenter le nombre d'observations au sein de la classe minoritaire (voir la figure 3.2). Cette approche implique la sélection aléatoire d'observations à dupliquer. De plus, des stratégies de sur-échantillonnage contrôlé ont été conçues pour répliquer ou générer des instances dans des régions pertinentes de la variété de données. Quelques exemples de méthodes sur-échantillonnage seront abordés dans les sous-sections suivantes.

Sur-échantillonnage aléatoire

Cette technique consiste à augmenter aléatoirement le nombre d'observations de la classe minoritaire en les dupliquant [155]. Cette duplication n'apporte pas d'informations supplémentaires et peut conduire à un surapprentissage (Overfitting) de l'algorithme. De plus, le suréchantillonnage aléatoire allonge le temps d'apprentissage étant donné que l'équilibrage du dataset entraîne une augmentation artificielle de sa taille.

Synthetic Minority Oversampling Technique (SMOTE)

La technique SMOTE est une méthode de sur échantillonnage proposée pour la première fois en 2002 [156], qui consiste à générer des données "synthétiques" à partir des données de la classe minoritaire. Les étapes de cette méthode sont les suivantes :

- Sélection des K voisins les plus proches pour chaque instance de la classe minoritaire. K est par défaut égale à 5. Cela se fait en appliquant une distance (souvent la distance Euclidienne).
- Choisir un voisin au hasard parmi les k voisins proches.
- La nouvelle instance est générée en interpolant entre l'exemple d'origine et le voisin choisi. La formule de génération d'une nouvelle instance est la suivante :
nouvelle instance = instance d'origine + $\lambda \times$ (voisin instance d'origine) avec λ représente un facteur aléatoire entre 0 et 1.
- Les nouvelles instances générées sont intégrées à l'ensemble de données, augmentant ainsi la taille de la classe minoritaire.
- Ces étapes peuvent être répétées jusqu'à ce que l'ensemble de données soit équilibré.

Une des limites de la méthode SMOTE est qu'elle génère les instances synthétiques sans prendre en compte les éventuels éléments de la classe majoritaire à proximité, ce qui peut conduire à des chevauchements (Overlap) de catégories et une sur-généralisation du classifieur [157].

B- Méthodes de sous-échantillonnage (Undersampling)

Les techniques de sous-échantillonnage consistent à retirer des échantillons de la classe majoritaire dans le but de réduire le nombre d'échantillons de classe majoritaire afin de rendre le dataset équilibré (figure). Ces techniques sont particulièrement utilisées pour traiter le problème de déséquilibre de classe lorsque la taille de l'ensemble de donnée est grande, et que la réduction du nombre d'observations de la classe majoritaire peut améliorer considérablement l'efficacité des calculs et atténuer les problèmes de stockage. Plusieurs méthodes permettent de sélectionner les instances de données à supprimer. Parmi les techniques les plus couramment utilisées figurent le sous-échantillonnage aléatoire [155] et Tomek Links [158].

Sous-échantillonnage aléatoire

Le sous-échantillonnage aléatoire [155] est une technique simple, qui consiste à supprimer aléatoirement un nombre d'instances de la classe majoritaire afin d'équilibrer le dataset. Le principal inconvénient de cette méthode est qu'elle pourrait éliminer des instances riches d'informations et potentiellement cruciales pour le processus d'apprentissage du classifieur. La figure fournit une représentation illustrative de la procédure de sous-échantillonnage, où 50% de la classe majoritaire est supprimée.

Tomek Links (TL)

La technique Tomek Links a été conçue par Tomek [159] en 1976 et constitue une modification de la technique de sous-échantillonnage Condensed Nearest Neighbors (CNN). Cette technique suit une règle spécifique pour identifier et supprimer des paires d'instances (a et b) qui sont considérées comme Tomek Link si a et b sont les plus proches voisins l'un de l'autre et appartiennent à des classes différentes [160], l'une étant issue de la classe minoritaire et l'autre de la classe majoritaire (les paires de Tomek sont entourées en vert, comme cela est montré dans la figure 3.3. Une fois ces paires identifiées, les observations de la classe majoritaire présentes dans une Tomek Link sont supprimées afin de rééquilibrer les données. Cette approche permet d'améliorer la séparation entre les classes et d'éliminer les observations ambiguës, contribuant ainsi à une meilleure qualité des données pour les modèles d'apprentissage supervisé.

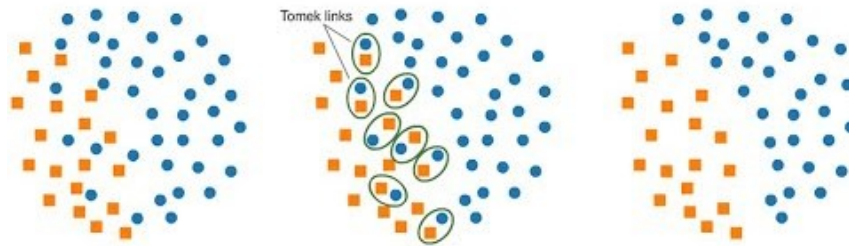


FIGURE 3.3 – Tomek links

C- Méthodes Hybride

Une méthode hybride implique l'intégration de techniques de suréchantillonnage et de sous-échantillonnage en éliminant des données de la classe majoritaire tout en ajoutant des données à la classe minoritaire afin d'ajuster la distribution des classes [160]. Par exemple, une méthode fréquente consiste à utiliser la technique SMOTE pour générer de nouvelles observations de la classe minoritaire, puis Tomek Links pour éliminer les instances ambiguës de la classe majoritaire, améliorant ainsi la séparation entre les classes.

3.3.2 Approche d'apprentissage d'Ensemble (Algorithmic level)

Pour faire face au problème de déséquilibre des classes, les méthodes « algorithm-level » représentent une alternative aux solutions « data-level » mentionnées dans la précédente section. Tandis que les techniques de rééchantillonnage modifient l'ensemble de données d'apprentissage pour lutter contre le déséquilibre, les méthodes « algorithm-level » visent à ajuster le processus d'entraînement des classificateurs traditionnels ou à suggérer de nouveaux modèles spécifiquement conçus pour traiter le déséquilibre.

L'approche d'Ensemble vise à améliorer les performances des modèles en combinant plusieurs classificateurs de base. Ces algorithmes reposent soit sur des stratégies adaptatives, comme le Boosting [161], soit sur des stratégies aléatoires, comme le Bagging [162]. De nombreuses études comparatives témoignent de leur efficacité dans divers domaines problématiques du monde réel [163] [164]. L'atout majeur des méthodes d'Ensemble réside dans leur capacité à garantir un taux d'erreur inférieur par rapport au meilleur classificateur individuel qu'elles regroupent. Les sous-sections suivantes explorent en détail les subtilités de ces deux stratégies d'ensemble.

Boosting

Le boosting est une méthode d'ensemble puissante conçue par Schapire [165]. Cette technique consiste à combiner plusieurs classificateurs faibles, également appelés classificateurs de base, pour créer un classificateur plus puissant, capable de produire de meilleurs résultats que ceux générés par un classificateur individuel. Le boosting entraîne les classificateurs faibles de manière séquentielle (voir figure 4.3), chaque classificateur tente de corriger son prédécesseur en ajoutant plus de poids aux données mal classées précédemment. Par conséquent, le classificateur suivant concentrera ses efforts sur les données les plus difficiles à prédire. Les algorithmes de boosting existants diffèrent par leur méthodologie de pondération des erreurs pour obtenir le prochain classificateur ou par les techniques d'agrégation elles-mêmes. Il existe de multiples exemples d'algorithmes de boosting tels qu'AdaBoost, GradientBoost, XGBoost, etc. L'algorithme original AdaBoost, pour Adaptive Boosting [166], est l'un des plus utilisés, souvent basé sur un algorithme d'arbre de décision.

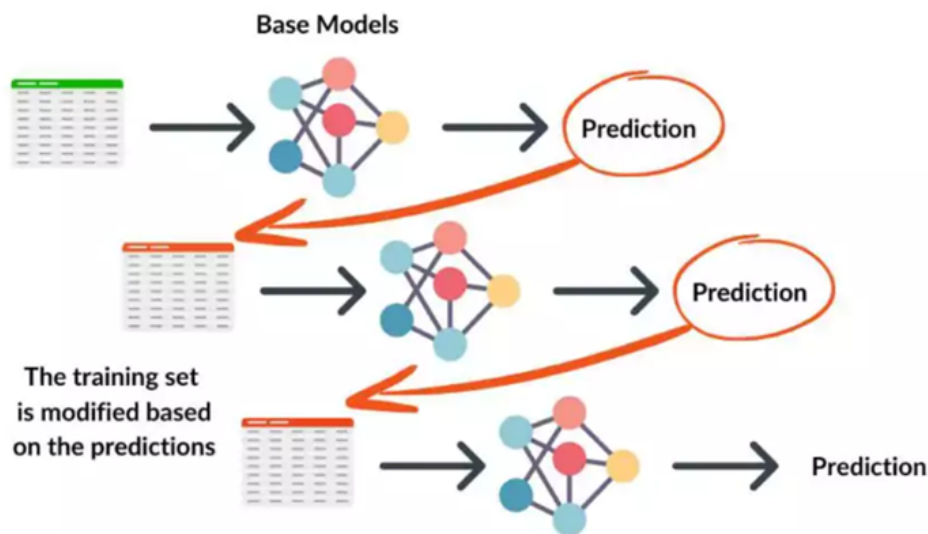


FIGURE 3.4 – *Aperçu de la technique Boosting*

Bagging

Le Bagging ou bien Bootstrap Aggregation, est une méthode proposée par L. Breiman en 1996 [167]. Elle repose sur la combinaison de plusieurs modèles de base (classificateurs faibles) pour obtenir un classificateur robuste. Contrairement au boosting, dans lequel les modèles sont entraînés séquentiellement, le bagging consiste à exécuter chaque classifieur en parallèle et les résultats sont combinés à la fin (voir figure 3.5), de sorte que chaque modèle de base s'entraîne indépendamment sur des sous-ensembles de données obtenus par échantillonnage aléatoire avec remplacement (bootstrap). Chaque modèle est par la suite utilisé sur des nouvelles données, et les

résultats finaux sont déterminés par un vote majoritaire (en cas de classification) ou par moyenne (dans le cadre de la régression) des classifications obtenues. L'avantage de bagging est l'amélioration des performances des classificateurs instables en diminuant leur sensibilité aux variations des données d'entraînement.

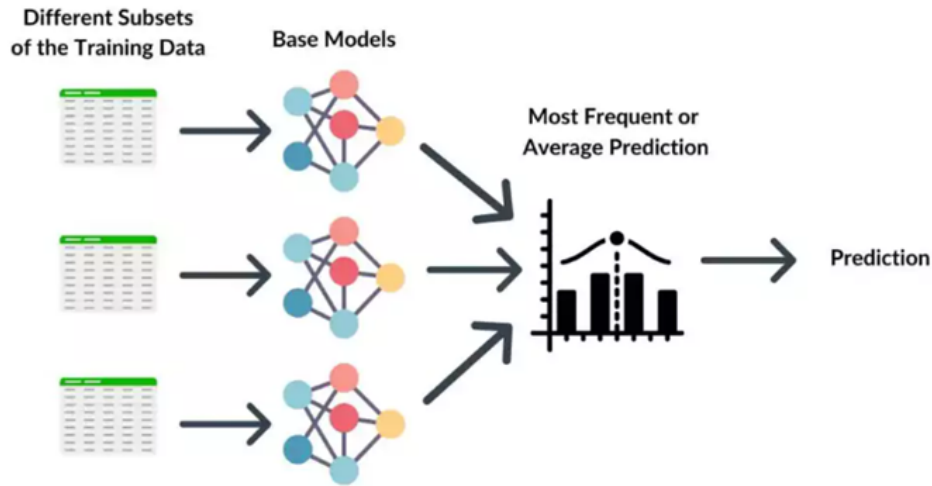


FIGURE 3.5 – Aperçu de la technique Bagging

3.4 Concepts de base

3.4.1 Élimination récursive des caractéristiques avec validation croisée (RFECV)

L'élimination récursive des caractéristiques (RFE) est une technique de sélection des caractéristiques qui améliore la performance d'un modèle en éliminant de manière méthodique les caractéristiques redondantes et peu significatives, réduisant ainsi leur influence sur l'erreur d'entraînement. Le RFE utilise une méthode appelée élimination des caractéristiques en arrière (backward feature elimination) [168], qui repose sur un processus itératif.

Considérons un ensemble de caractéristiques, X , où $X = X_1, X_2, \dots, X_n$ où n représente le nombre total de caractéristiques. Lors du processus d'entraînement, toutes les caractéristiques de l'ensemble de données sont utilisées pour entraîner le modèle. L'élimination récursive des caractéristiques (RFE) classe les caractéristiques en fonction de leurs scores d'importance. Ces scores peuvent être obtenus à partir de plusieurs métriques, en fonction du modèle utilisé, telles que les coefficients dans les modèles linéaires ou l'importance des caractéristiques dans les modèles basés sur les arbres de décision.

Soit $I = I_1, I_2, \dots, I_n$ l'ensemble des scores d'importance associés aux caractéristiques X .

L'élimination itérative consiste à supprimer la caractéristique la moins importante, notée X_{min} , en utilisant l'algorithme RFE. Le modèle est ensuite réentraîné sur l'ensemble de caractéristiques réduit, $X = X_{min}$, et les scores d'importance des caractéristiques restantes sont recalculés. Ce processus est répété jusqu'à atteindre le nombre prédéfini de caractéristiques k .

4 Approche proposée

Après avoir détaillé le problème du déséquilibre des classes ainsi que les solutions proposées dans la littérature, cette section présente une nouvelle approche visant à relever ce défi sans recourir aux méthodes classiques telles que le rééchantillonnage et autres approches similaires.

4.1 Dataset utilisé

Le dataset utilisé dans cette deuxième contribution est UNSW-NB15 (décrit dans le chapitre 2). Il constitue une référence essentielle dans la littérature pour évaluer l'efficacité des systèmes de détection d'intrusion réseau (NIDS). Ce dataset comprend 09 catégories uniques d'attaques, présentées dans le tableau 3.2 : Analysis, Fuzzers, Backdoors, DoS, Exploits, Reconnaissance, Generic, Shellcode et Worms. Chaque catégorie représente un défi particulier pour les défenses en cybersécurité [171] [172].

TABLE 3.2 – *Distribution des données d'entraînement et de test par type de connexion à partir du jeu de données UNSW-NB15*

Type	Train	Test
Normal	56000	37000
Analysis	2000	677
Backdoor	1746	583
DoS	12264	4089
Exploits	33393	11132
Fuzzers	18184	6062
Generic	40000	18871
Reconnaissance	10491	3496
Shellcode	1133	378
Worms	130	44

- **Fuzzers** : Cette technique d'attaque consiste à submerger un système avec une grande quantité de données aléatoires afin de découvrir des failles pouvant entraîner son plantage. L'objectif est d'exploiter les vulnérabilités des logiciels, systèmes d'exploitation ou réseaux.
- **Scanning (Reconnaissance)** : Une forme précoce d'intrusion impliquant des techniques telles que la recherche de ports (port scanning) et la diffusion de fichiers spam ou HTML. Son objectif est d'identifier les faiblesses pouvant être exploitées pour pénétrer des applications en ligne.
- **Backdoors** : Stratégies clandestines utilisées par les attaquants pour contourner les protocoles d'authentification standards, leur permettant d'obtenir un accès distant illicite à un système tout en évitant la détection.
- **DoS (Denial of Service)** : Une attaque intentionnelle visant à perturber les ressources d'un système en inondant sa mémoire avec un nombre excessif de requêtes, rendant le service indisponible pour les utilisateurs légitimes.
- **Exploits** : Ces attaques impliquent l'exécution de code malveillant exploitant des vulnérabilités, faiblesses ou comportements involontaires dans les systèmes hôtes ou réseaux, afin d'obtenir un accès ou un contrôle non autorisé.

- **Generic** : Stratégie qui applique des fonctions de hachage collisionnelles à tous les blocs de chiffrement, quelle que soit leur configuration.
- **Reconnaissance** : Une tactique utilisée dans les cyberattaques pour collecter des informations en étudiant minutieusement les systèmes informatiques et réseaux, dans le but de contourner les mesures de sécurité.
- **Shellcode** : Technique permettant aux attaquants d'obtenir un accès non autorisé à un système en injectant un petit morceau de code ou un shell, leur permettant de prendre le contrôle de la machine compromise.
- **Worms (Vers informatiques)** : Un logiciel malveillant capable de se répliquer et se propager à travers les réseaux informatiques. Il exploite fréquemment des vulnérabilités pour se diffuser et obtenir un accès non autorisé aux systèmes.

La répartition inégale des différentes catégories d'attaques dans l'ensemble de données UNSW-NB15 est illustrée dans la figure 3.6. Ce graphique met en évidence un déséquilibre considérable, puisque les enregistrements normaux représentent plus d'un tiers (31,94%) des données. En revanche, les données relatives aux attaques de type worm ne constituent qu'une infime partie (0,007%) du total.

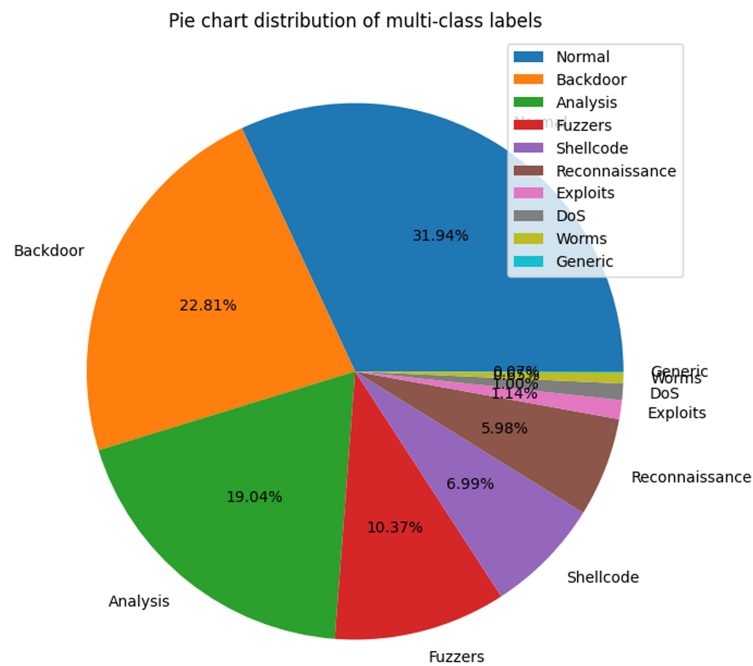


FIGURE 3.6 – Distribution des données d'entraînement et de test par type de connexion à partir du jeu de données UNSW-NB15.

L'ensemble de données UNSW-NB15 offre de nombreux avantages pour la recherche en détection d'intrusions réseau [176]. Il est particulièrement pertinent face aux défis actuels en cybersécurité, car il combine des activités d'attaques contemporaines générées de manière synthétique avec des événements réels du réseau. De plus, ce dataset couvre un large éventail de données collectées à la fois à partir du contenu des paquets et des métadonnées, permettant ainsi une analyse approfondie du trafic réseau.

4.2 Mise en oeuvre de l'approche

L'approche proposée, illustrée dans la figure 3.7, se compose de quatre étapes : le prétraitement des données, la division des données, la sélection des caractéristiques et la classification.

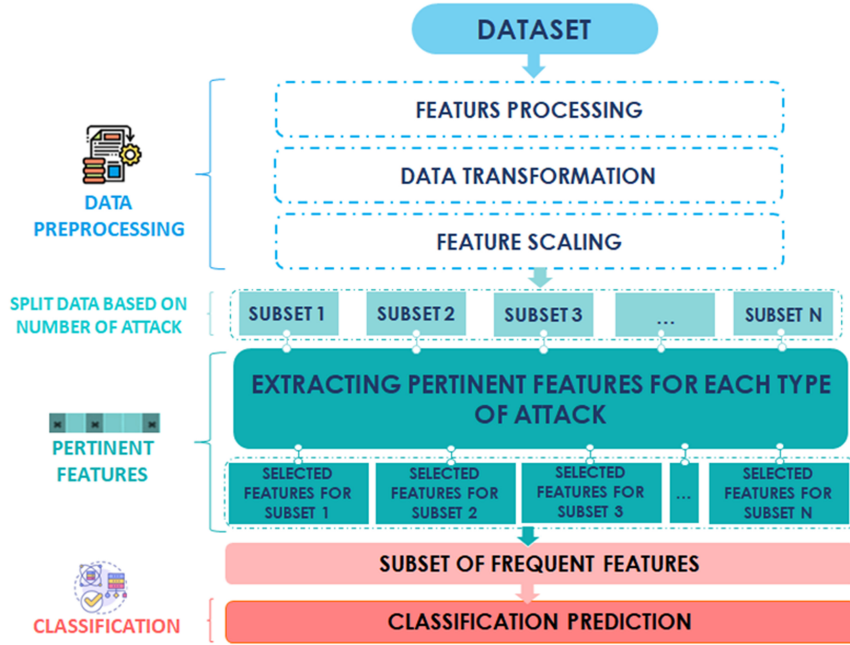


FIGURE 3.7 – L'approche proposée.

A-Prétraitement des données

Les données de trafic réseau contiennent souvent du bruit, des redondances, ainsi que des types de données variés, ce qui constitue des défis majeurs pour l'exploration de connaissances et la modélisation des données. Par conséquent, la préparation des données est essentielle afin d'assurer leur utilisabilité pour le développement de tout algorithme d'apprentissage.

Dans cette étude, la procédure de prétraitement comprend les étapes suivantes :

- **Suppression de caractéristiques non pertinentes** : L'ensemble de données UNSW-NB15 comprend 45 attributs et deux étiquettes. L'attribut 'id' a été supprimé en raison de son influence insignifiante sur la classification du trafic réseau.
- **Transformation des données catégorielles en variables numériques** : Les attributs catégoriels ont été convertis en valeurs numériques afin de les rendre exploitables par les algorithmes d'apprentissage automatique.
- **Normalisation des données** : La normalisation des données consiste à standardiser les valeurs des caractéristiques afin de garantir qu'elles soient sur une échelle comparable. Cela permet d'éviter qu'un modèle considère à tort les caractéristiques ayant des valeurs plus élevées comme ayant une influence plus importante sur les résultats. Pour cette expérience, la méthode de normalisation Min-Max a été choisie. Cette méthode ajuste chaque caractéristique pour qu'elle s'inscrive dans l'intervalle numérique [0,1], en utilisant la formule mathématique suivante :

$$X_n = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \quad (3.1)$$

Où : X_n représente la valeur normalisée des données, X est la valeur originale à traiter, $X_{\min}$

et X_{max} indiquent respectivement les valeurs maximale et minimale des données pour la caractéristique actuelle.

Une fois le prétraitement des données terminé, l'ensemble de données est divisé en neuf sous-ensembles, en fonction du type d'attaque (voir tableau 3.3).

TABLE 3.3 – *Partitionnement de l'ensemble de données en sous-ensembles*

Sous-ensemble	Attaques	Nombre d'instances
Subset 1	Generic	40000
Subset 2	Exploits	33393
Subset 3	Fuzzers	18184
Subset 4	DoS	12264
Subset 5	Reconnaissance	10491
Subset 6	Analysis	2000
Subset 7	Backdoor	1746
Subset 8	Shellcode	1133
Subset 9	Worms	130

Par exemple, le subset 1 comprend les instances du type d'attaque "Generic", permettant une analyse ciblée selon la catégorie d'attaque spécifique.

B-Sélection des caractéristiques pour chaque attaque en utilisant l'élimination récursive des caractéristiques (RFE) basée sur SHAP

Dans les domaines de l'apprentissage automatique et de la science des données, la création de modèles prédictifs précis et efficaces repose sur une sélection minutieuse des caractéristiques les plus pertinentes. Un trop grand nombre de caractéristiques peut entraîner un surapprentissage (overfitting), des coûts de calcul élevés, et une dégradation des performances du modèle. L'objectif est d'identifier les caractéristiques les plus pertinentes, qui améliorent la capacité du modèle à fournir des prédictions précises, tout en éliminant les composants superflus [173].

L'intégration de la méthodologie issue de la théorie des jeux, SHAP (SHapley Additive Explanations), dans la méthode standard RFE (Recursive Feature Elimination) permet d'améliorer son efficacité. Dans cette approche, RFECV (Recursive Feature Elimination with Cross-Validation) et SHAP sont utilisés pour évaluer et sélectionner l'ensemble optimal de caractéristiques en se basant sur les scores d'importance de SHAP. Le modèle Light Gradient Boosting Machine (LGBM) est employé comme classificateur principal.

Le processus de chaque itération de RFECV-SHAP est le suivant :

- Optimisation des hyperparamètres du modèle via la recherche aléatoire (Randomized Search CV) pour un ajustement optimal des hyperparamètres.
- Utilisation de la validation croisée pour entraîner le modèle sur l'ensemble d'entraînement et évaluer sa performance sur les données de validation à l'aide de la métrique de l'aire sous la courbe (AUC).
- Élimination des k caractéristiques les moins importantes selon les scores SHAP.

- Visualisation des performances de la méthode de sélection des caractéristiques RFECV-SHAP avec le classificateur LGBM, en traçant l'AUC obtenue.

Cette stratégie garantit que les caractéristiques sélectionnées apportent une contribution significative à la capacité prédictive du modèle, tout en répondant aux défis liés à l'ajustement des hyperparamètres du modèle et à la réduction du biais en faveur des caractéristiques ayant un grand nombre de valeurs distinctes dans la méthode standard de Recursive Feature Elimination (RFE).

Les figures présentées (figure 3.8) illustrent les performances de l'algorithme RFECV-SHAP appliqué aux neuf sous-ensembles de données. La performance est évaluée en calculant la métrique de l'AUC, qui est ensuite tracée en fonction du nombre de caractéristiques pour les ensembles d'entraînement et de validation.

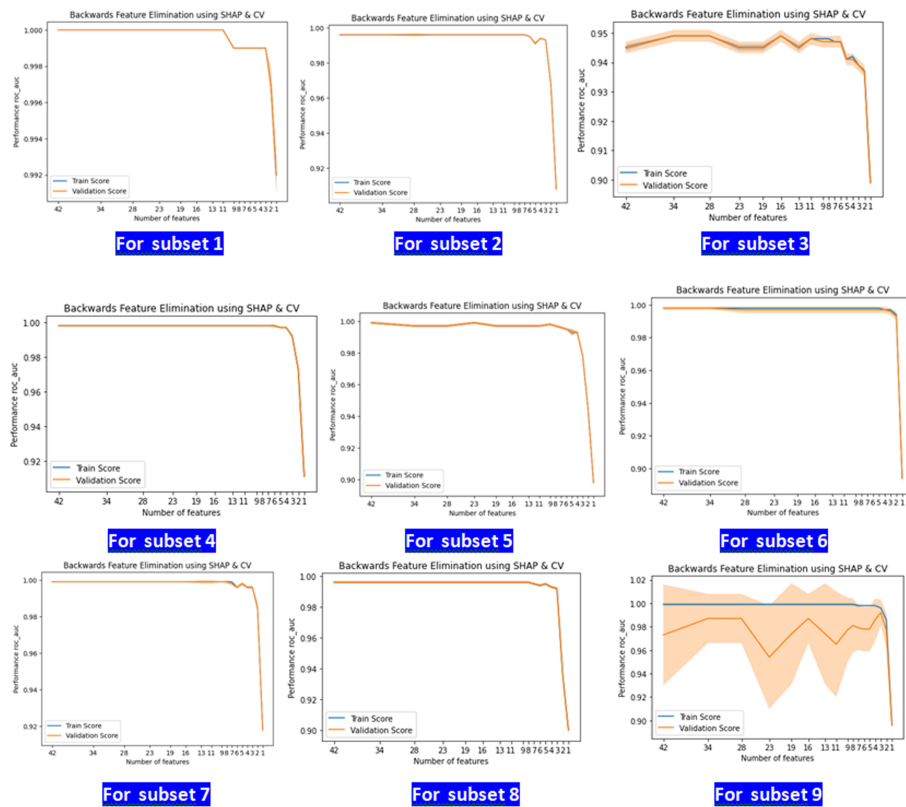


FIGURE 3.8 – L'approche proposée.

Chaque sous-ensemble présente une plage optimale de caractéristiques où la performance de l'AUC atteint son niveau maximal. Cette plage varie légèrement selon les sous-ensembles, soulignant l'importance d'une sélection spécifique des caractéristiques pour chaque ensemble afin d'obtenir des performances optimales.

- Sous-ensembles 1 et 2 : Ces ensembles affichent une AUC initiale élevée, qui reste stable jusqu'à une réduction significative du nombre de caractéristiques, après quoi le score de validation diminue fortement. Cela indique que quelques caractéristiques cruciales ont une influence majeure sur les performances du modèle.
- Sous-ensemble 3 : Présente une baisse progressive du score de validation, accompagnée de variations importantes, suggérant la présence de caractéristiques aux niveaux d'importance variables et d'interactions complexes qui influencent la stabilité du modèle.

- Sous-ensembles 4, 5, 6 et 7 : Ils présentent une tendance similaire aux sous-ensembles 1 et 2, avec des valeurs d'AUC constantes jusqu'à la suppression d'un nombre critique de caractéristiques, ce qui suggère l'existence de caractéristiques essentielles qui ont un impact significatif sur les performances du modèle.
- Sous-ensemble 8 : Montre une diminution constante des performances de validation à mesure que les caractéristiques sont éliminées, indiquant une dispersion progressive de l'importance des caractéristiques, sans qu'un ensemble de caractéristiques dominantes ne soit identifiable.
- Sous-ensemble 9 : Présente des variations notables de l'AUC de validation, suggérant la présence d'un ensemble de caractéristiques complexe avec des interactions ayant un impact non linéaire sur le modèle.

Les résultats de l'analyse RFECV-SHAP soulignent l'importance d'une sélection personnalisée des caractéristiques pour chaque sous-ensemble afin d'optimiser les performances du modèle.

L'existence de seuils critiques pour les caractéristiques dans les différents sous-ensembles met en évidence la nécessité de suivre de près l'élimination des caractéristiques, afin d'éviter une détérioration significative des performances.

Les caractéristiques sélectionnées pour chaque sous-ensemble sont répertoriées dans le tableau 3.4.

TABLE 3.4 – *Caractéristiques sélectionnées pour chaque sous-ensemble*

Sous-ensemble	Caractéristiques sélectionnées
Subset 1	'dinpkt', 'sload', 'ct_dst_sport_ltm', 'dbytes', 'ct_dst_src_ltm', 'dttl', 'proto', 'sbytes', 'sttl', 'ct_state_ttl', 'service'
Subset 2	'ct_dst_src_ltm', 'ct_srv_dst', 'dbytes', 'dmean', 'service', 'proto', 'sttl'
Subset 3	'dmean', 'sttl', 'sinpkt', 'smean', 'sloss', 'dloss', 'proto', 'ct_srv_dst', 'ct_dst_src_ltm', 'state', 'dload'
Subset 4	'dttl', 'ct_srv_dst', 'dbytes', 'ct_dst_src_ltm', 'sttl', 'proto'
Subset 5	'proto', 'service', 'sbytes', 'dbytes', 'sttl', 'sloss', 'smean', 'ct_dst_src_ltm', 'ct_srv_dst'
Subset 6	'proto', 'service', 'sttl', 'response_body_len'
Subset 7	'proto', 'ct_dst_src_ltm', 'sbytes', 'ct_srv_dst', 'dbytes', 'dload', 'dttl', 'synack'
Subset 8	'ct_src_dport_ltm', 'sbytes', 'ct_dst_src_ltm', 'ct_srv_dst', 'dbytes', 'sttl', 'spkts', 'smean', 'dur'
Subset 9	'service', 'sttl', 'ct_srv_dst'

C- Agrégation des Caractéristiques Fréquentes

Après avoir identifié les caractéristiques les plus pertinentes pour chaque sous-ensemble, nous avons mené une analyse approfondie afin de déterminer la fréquence d'occurrence de chaque caractéristique dans ces sous-ensembles, comme illustré dans le tableau 3.5. L'évaluation a consisté à classer les caractéristiques selon leur fréquence d'occurrence afin d'évaluer leur importance relative. Notre attention s'est portée sur les caractéristiques présentes dans au moins trois sous-groupes distincts. Cette méthodologie a permis de cibler les caractéristiques qui influençaient systématiquement les résultats des prévisions dans divers sous-groupes.

Un ensemble de caractéristiques ainsi que leurs fréquences d'apparition respectives à travers divers sous-ensembles sont présentés dans le tableau 3.5. Ces informations ont guidé notre processus de sélection, en privilégiant les caractéristiques qui apparaissent de manière récurrente et apportent une contribution significative aux prédictions dans chaque sous-groupe.

Les caractéristiques peuvent être classées en trois catégories : Caractéristiques à fréquence élevée, Caractéristiques à fréquence modérée et Caractéristiques à faible fréquence. En particulier, les

TABLE 3.5 – *Fréquence d'occurrence des caractéristiques*

Noms des caractéristiques	Fréquence des caractéristiques
sttl	8
proto	7
ct_dst_src_ltm	7
ct_srv_dst	7
dbytes	6
service	5
sbytes	4
dttl	3
smean	3
sloss	2
dload	2
dmean	2

caractéristiques "sttl" (présente huit fois), "proto", "ct_dst_src_ltm", et "ct_srv_dst" (chacune présente sept fois) ont été identifiées comme hautement influentes. Leur sélection fréquente suggère qu'elles exercent une influence constante et significative sur la performance prédictive à travers divers sous-ensembles.

Par ailleurs, les caractéristiques "dbytes", "service", et "sbytes", qui apparaissent entre quatre et six fois, indiquent un impact modéré mais néanmoins important sur les capacités prédictives du modèle.

Les caractéristiques qui n'ont pas été utilisées plus de trois fois, telles que "dload", "dmean" et "sloss", ont été considérées comme moins influentes. Par conséquent, elles ont été généralement exclues de l'ensemble final de caractéristiques afin d'améliorer la performance du modèle et de réduire le bruit.

Le tableau 3.6 présente l'ensemble final de caractéristiques utilisées pour la classification, dans le but d'améliorer les taux de détection. Vous trouverez ci-dessous une description de chaque caractéristique.

TABLE 3.6 – *Caractéristiques finales sélectionnées*

Caractéristique	Description
sttl	Temps de vie du paquet de la source à la destination
proto	Protocole de transaction
ct_dst_src_ltm	Nombre de connexions de la même source 1 et de la destination 3 parmi 100 connexions selon le dernier temps 26
ct_srv_dst	Nombre de connexions contenant le même service (14) et l'adresse de destination (3) parmi 100 connexions selon le dernier temps (26)
dbytes	Octets de la destination vers la source
service	http, ftp, ssh, dns, ..., else (-)
sbytes	Octets de la source vers la destination
dttl	Temps de vie du paquet de la destination vers la source
smean	Moyenne de la taille des paquets de flux transmis par la source

D- Classification

Dans cette section, la performance de classification du modèle XGBoost est analysée. Le modèle est entraîné dans deux scénarios distincts : en utilisant toutes les caractéristiques disponibles et en se concentrant exclusivement sur les caractéristiques les plus influentes identifiées dans le tableau 3.6. Les tableaux 3.7 et 3.8 offrent une comparaison complète des résultats obtenus par notre modèle proposé et ceux du modèle utilisant toutes les caractéristiques.

Dans cette section, la performance de classification du modèle XGBoost est analysée afin d'évaluer son efficacité dans différents scénarios d'apprentissage. Le modèle a été entraîné dans deux configurations distinctes :

1. Utilisation de l'ensemble complet des caractéristiques disponibles, afin d'exploiter toutes les données disponibles sans réduction de dimensionnalité.
2. Utilisation exclusive des caractéristiques les plus influentes, telles qu'identifiées dans le tableau 3.6.

TABLE 3.7 – Résultats expérimentaux

	Accuracy	Precision	Recall	F1-score
Toutes les fonctionnalités	0.7289	0.6205	0.4097	0.3929
Fonctionnalités sélectionnées	0.8353	0.7708	0.6388	0.6512

Le tableau 3.7 présente les métriques de performance globales des deux modèles. Les caractéristiques sélectionnées améliorent considérablement l'exactitude du modèle, la faisant passer de 0,7289 à 0,8353. Cela indique que l'approche de sélection des caractéristiques renforce de manière significative la capacité du modèle à catégoriser de manière fiable les occurrences dans l'ensemble du dataset.

De plus, la précision du modèle s'améliore, passant de 0,6205 lorsqu'on utilise toutes les caractéristiques à 0,7708 lorsque seules les caractéristiques sélectionnées sont utilisées. Une valeur de précision plus élevée signifie que le modèle, en utilisant les caractéristiques sélectionnées, présente un taux de faux positifs plus faible.

Cela suggère que les prédictions du modèle proposé sont globalement plus fiables. L'utilisation des caractéristiques sélectionnées dans le modèle entraîne une amélioration substantielle du rappel, qui passe de 0,4097 à 0,6388. Cette augmentation montre que le processus de sélection des caractéristiques aide le modèle à identifier plus précisément les occurrences réelles positives dans l'ensemble du dataset, réduisant ainsi le nombre de faux négatifs.

De plus, le score F1, qui mesure l'équilibre entre la précision et le rappel, connaît une augmentation significative, passant de 0,3929 à 0,6512 lorsque les caractéristiques choisies sont utilisées. Cette hausse substantielle indique que le modèle, intégrant les caractéristiques sélectionnées, parvient à un meilleur équilibre entre la précision et le rappel, ce qui conduit à une amélioration globale de la performance de classification.

Les analyses suivantes démontreront la performance de ce modèle pour chaque type d'attaque spécifique, offrant une compréhension plus détaillée de leur efficacité à identifier une variété d'attaques.

TABLE 3.8 – *Les mesures de performance des deux modèles dans différentes catégories.*

Class	Metric	All Features	Proposed Model
Normal	Precision	0.9572	0.9369
	Recall	0.8608	0.9206
	F1-score	0.9065	0.9287
Generic	Precision	0.9952	0.9972
	Recall	0.9764	0.9810
	F1-score	0.9857	0.9890
Exploits	Precision	0.7033	0.6424
	Recall	0.5440	0.9108
	F1-score	0.6135	0.7534
Fuzzers	Precision	0.6565	0.7531
	Recall	0.6725	0.7447
	F1-score	0.6644	0.7489
DoS	Precision	0.2441	0.4377
	Recall	0.8063	0.1627
	F1-score	0.3747	0.2372
Reconnaissance	Precision	0.6923	0.9165
	Recall	0.0029	0.7528
	F1-score	0.0057	0.8266
Analysis	Precision	0.6444	0.5806
	Recall	0.0483	0.1091
	F1-score	0.0899	0.1837
Backdoor	Precision	0.6000	0.8837
	Recall	0.0115	0.1322
	F1-score	0.0225	0.2300
Shellcode	Precision	0.2115	0.7505
	Recall	0.0971	0.8448
	F1-score	0.1331	0.7949
Worms	Precision	0.5000	0.8095
	Recall	0.0769	0.8293
	F1-score	0.1333	0.8193

Le tableau 3.8 présente les caractéristiques de performance des deux modèles à travers différentes classes. Le modèle proposé affiche une légère amélioration du score F1 et du rappel, ce qui indique une performance plus équilibrée. Les deux modèles, utilisant les caractéristiques sélectionnées, obtiennent une précision et un rappel élevés pour la classe "Normal". Les deux modèles montrent également des performances exceptionnelles dans la catégorie "Generic", mettant en évidence leur efficacité à identifier les attaques génériques, avec seulement de légères variations dans

les métriques.

Le modèle proposé améliore nettement le score F1 et le rappel pour la classe "Exploits", bien qu'il entraîne une légère diminution de la précision, ce qui signale une amélioration générale des capacités de détection. Dans la classe "Fuzzers", le modèle proposé atteint de meilleures valeurs de précision et de rappel, ce qui permet une détection plus précise des attaques par fuzzing et une amélioration du score F1.

L'approche proposée améliore la précision pour la classe "DoS", mais cela entraîne une réduction du rappel, ce qui indique que certaines instances de DoS peuvent être ignorées malgré une diminution des faux positifs. Certains enregistrements de DoS sont mal classés comme des attaques d'exploitation (Fig. 4), probablement en raison d'un chevauchement de caractéristiques entre les deux groupes. Ce problème est lié au chevauchement intrinsèque des données dans le dataset UNSW-NB15, où les attributs communs parmi les types d'attaques compliquent la classification précise et conduisent à des erreurs.

Le modèle proposé montre une amélioration significative du score F1 et du rappel pour la classe "Reconnaissance", facilitant ainsi une meilleure détection de ces attaques, jusqu'alors sous-représentées, lorsqu'on utilise toutes les caractéristiques.

Les classes telles que "Analysis", "Backdoor", "Shellcode" et "Worms" montrent des améliorations notables en précision, rappel et score F1 avec le modèle proposé. Ces résultats soulignent la capacité supérieure du modèle à détecter des types d'attaques rares ou complexes, qui posent souvent des difficultés pour les méthodes de classification traditionnelles.

La matrice de confusion pour le modèle utilisant toutes les caractéristiques (figure 3.9) montre un grand nombre d'instances correctement classées pour des classes moins complexes telles que "Normal" et "Generic" (composants diagonaux). Cependant, des catégories plus complexes telles que "Exploits", "Fuzzers" et "DoS" présentent des erreurs de classification considérables (éléments hors-diagonale). Par exemple, bien que les cas "DoS" atteignent un rappel élevé, la faible précision met en évidence une prédominance de faux positifs.

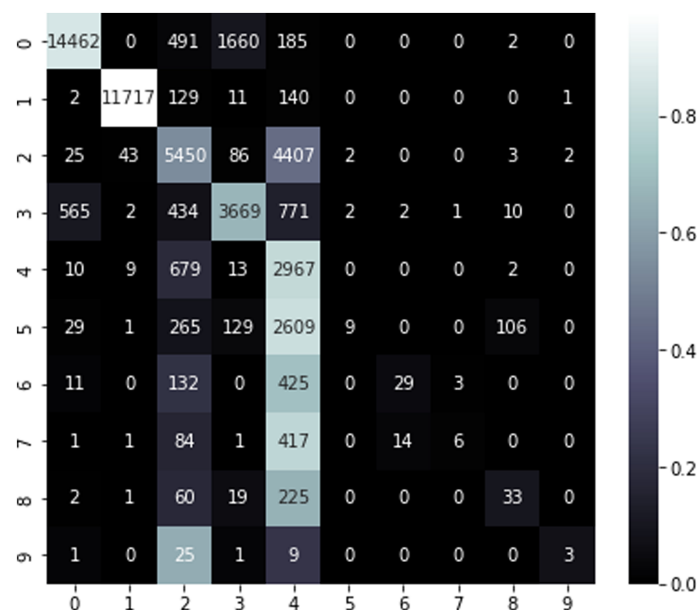


FIGURE 3.9 – La matrice de confusion utilisant toutes les fonctionnalités

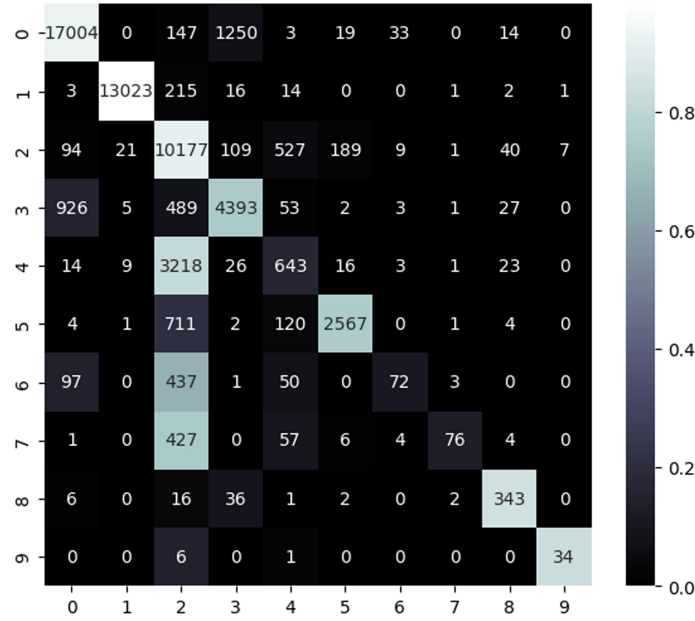


FIGURE 3.10 – La matrice de confusion du modèle proposé

La matrice de confusion pour le modèle proposé utilisant les caractéristiques sélectionnées (figure 3.10) montre une amélioration de la performance de classification à travers plusieurs classes. Les classifications correctes sont principalement alignées le long de la diagonale, ce qui indique une réduction des erreurs. Les métriques de précision et de rappel pour les classes "Reconnaissance", "Shellcode", "Backdoor", "Generic" et "Worms" reflètent des améliorations substantielles, mettant en évidence l'efficacité accrue du modèle proposé dans un contexte multi-classes.

En résumé, les résultats suggèrent que l'utilisation des caractéristiques sélectionnées améliore non seulement la précision et le rappel dans de nombreuses classes, mais permet également une détection plus précise des attaques sous-représentées et complexes. Cela montre l'efficacité du modèle proposé pour améliorer la détection des intrusions dans des contextes variés, tout en réduisant les erreurs de classification.

E- Validation de l'approche proposée

Pour valider le modèle proposé, nous avons utilisé deux ensembles de données contemporains et pertinents : CICIDS2019 [177] et CICIOT2023 [178]. Ces ensembles de données offrent une vaste gamme de scénarios d'attaque et corrigent les lacunes identifiées dans l'ensemble de données UNSW-NB15. L'ensemble de données CICIOT2023 est particulièrement important car il inclut des attaques DoS, qui sont susceptibles d'être mal classées dans UNSW-NB15, permettant ainsi une évaluation complète du modèle sur un éventail plus large de types d'attaques.

L'ensemble de données CICIDS2019, reconnu pour sa représentation des classes minoritaires et des attaques modernes, a été utilisé pour valider notre approche. Cet ensemble comprend des attaques DDoS exploitant les protocoles de couche application et transport (par exemple, UDP, TCP, ou leurs mélanges). Notre évaluation s'est concentrée sur le sous-ensemble Day-1, qui contient 88 caractéristiques et 569 650 enregistrements classifiés comme suit : Syn (352 968 instances), Portmap (42 606), LDAP (41 801), UDP (33 695), MSSQL (25 280), NetBIOS (16 252), UDPLag (83), et trafic normal (56 965). L'existence de classes minoritaires, telles que UDPLag, NetBIOS et MSSQL, a ajouté de la complexité, soulignant la capacité du modèle à identifier des types d'attaque peu fréquents et difficiles.

L'ensemble de données CICIoT2023, publié en 2023 [178], provient de dispositifs IoT réels et comprend 46 686 579 instances réparties sur 46 caractéristiques. Il englobe huit catégories principales : une pour le trafic standard et sept pour différents types d'attaques, à savoir DDoS, DoS, Reconnaissance, Web-based, Brute Force, Spoofing, et Mirai. Une grande difficulté réside dans le déséquilibre des classes, avec les attaques Web-based et Brute Force étant particulièrement sous-représentées par rapport aux catégories DoS et DDoS. Cette disparité a offert une occasion d'évaluer la résilience du modèle face à des distributions de données déséquilibrées.

L'évaluation du modèle sur ces ensembles de données variés et déséquilibrés a confirmé sa polyvalence dans différents scénarios d'attaque. Le tableau 3.9 résume les résultats, mettant en évidence l'efficacité du modèle à travers plusieurs catégories d'attaques.

TABLE 3.9 – *Évaluation des performances du modèle proposé sur différentes classes dans les jeux de données CICIDS2019 et CICIoT2023*

Dataset	Class	Precision	Recall	F1-score
CICIDS2019	Normal	0.9993	0.9998	0.9998
	Syn	0.9989	0.9978	0.9979
	Portmap	0.9387	0.9543	0.9389
	LDAP	0.9380	0.9289	0.9412
	UDP	0.9900	0.9792	0.9858
	MSSQL	0.9521	0.9789	0.9683
	NetBIOS	0.9256	0.9489	0.9654
	UDPLag	0.7623	0.7745	0.7965
CIC IoT 2023	Normal	0.8791	0.9418	0.9094
	DDoS	0.9997	0.9998	0.9997
	DoS	0.9998	0.9989	0.9994
	Mirai	0.9997	1.0000	0.9999
	Spoofing	0.8638	0.8050	0.8333
	Recon	0.8921	0.9070	0.9134
	BruteForce	0.8471	0.7558	0.7989
	Web-based	0.8167	0.7313	0.7717

Cette évaluation montre que le modèle proposé est robuste et adaptable, capable de traiter des scénarios complexes et déséquilibrés, tout en offrant de bons résultats de détection même pour des attaques rares ou peu représentées.

Les résultats démontrent les performances exceptionnelles du modèle sur les deux ensembles de données. Avec CICIDS2019, le modèle a atteint des précisions, rappels et scores F1 presque parfaits dans des classes telles que Normal et Syn, tout en gérant efficacement des attaques complexes comme NetBIOS (score F1 : 0.9654), MSSQL (0.9683) et LDAP (0.9412). Le modèle s'est également illustré dans la détection des classes minoritaires, telles que UDPLag, avec un score F1 de 0.7965, démontrant ainsi sa capacité à identifier des types d'attaques rares.

Avec CICIOT2023, le modèle a montré une proficiences exceptionnelles dans l'identification des attaques DoS, DDoS et Mirai, obtenant des scores pratiquement parfaits. Cette performance est particulièrement significative pour les attaques DoS, qui étaient auparavant sujettes à des difficultés de classification en raison du chevauchement des caractéristiques dans d'autres ensembles de données. La définition explicite des attaques DoS dans CICIOT2023 a permis au modèle d'atteindre des performances remarquables. De plus, le modèle s'est révélé performant pour identifier des attaques sous-représentées, notamment les attaques Web-based et Brute Force, avec des scores F1 respectifs de 0.7717 et 0.7989, mettant en évidence son efficacité dans la gestion des données déséquilibrées.

La capacité du modèle à classifier avec précision à la fois les groupes majoritaires et minoritaires renforce sa pertinence face aux défis réels de cybersécurité. Ces résultats confirment l'efficacité du modèle dans des environnements variés, complexes et déséquilibrés, soutenant ainsi les conclusions tirées de l'ensemble de données UNSW-NB15.

F- Comparaison

L'approche proposée, intégrant XGBoost avec ShapRFECV, montre une efficacité accrue par rapport à de nombreux modèles actuels, et ce, sur différents ensembles de données (tableau 3.10). Sur UNSW-NB15, elle atteint une précision de 83,53 %, légèrement inférieure à celle de l'arbre de décision (84,8 %), mais surpasse largement ce dernier en termes de précision (77,08 % contre 63,3 %), indiquant une réduction notable des faux positifs. En ce qui concerne CICIDS2019, la méthode atteint une précision de pointe de 99,25 %, surpassant les autres modèles en termes de précision (93,81 %) et de score F1 (94,92 %), démontrant ainsi une performance équilibrée et fiable. Sur CICIOT2023, les résultats sont équivalents ou supérieurs à ceux de la méthode basée sur le Deep Neural Network (DNN), avec une précision de 99,2 % et un score F1 de 93,55 %, dépassant de manière significative le rappel du DNN (93,15 % contre 67,94 %). Contrairement à des modèles comme la régression logistique, qui dépend de l'analyse en composantes principales et affiche des baisses de précision et de rappel, ou le LSTM sur NSL-KDD, favorisé par un jeu de données moins complexe, la méthode proposée maintient constamment un équilibre dans ses métriques de performance à travers des ensembles de données complexes et déséquilibrés, démontrant sa robustesse et son efficacité.

TABLE 3.10 – Analyse comparative des résultats avec les modèles existants

Model	Feature Selection	Dataset	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
Logistic regression [179]	PCA + SMOTE	UNSW-NB15	68.2	57.8	58.0	54.8
Decision Tree [?]	Correlation attribute	UNSW-NB15	84.8	63.3	60.9	61.9
SVM [180]	XGBoost	UNSW-NB15	–	53.95	61.52	51.31
ANN [180]	XGBoost	UNSW-NB15	77.51	–	–	–
DNN [181]	DT	–	82.1	–	–	–
LSTM [182]	SHAP Value	NSL-KDD	99.74	95.42	94.42	94.90
DNN [113]	–	CICIoT2023	99.11	90.66	67.94	69.72
Proposed Approach	ShapRFECV	UNSW-NB15	83.53	77.08	63.88	65.12
		CICIDS 2019	99.25	93.81	94.53	94.92
		CICIoT2023	99.20	93.70	93.15	93.55

5 Conclusion et perspectives

Ce chapitre a présenté une méthodologie innovante combinant le classifieur XGBoost avec une stratégie de sélection de caractéristiques basée sur SHAP-RFECV (SHapley Additive Explanations avec RecursiveFeature Elimination et validation croisée), dans le but de renforcer la détection des intrusions réseau, en particulier celles issues des classes minoritaires. Appliquée à l'ensemble de données UNSW-NB15, cette approche a démontré son efficacité en atteignant une précision de 77,08% et une exactitude de 83,53%, révélant sa capacité à détecter les attaques rares tout en maintenant un faible taux de faux positifs.

Le modèle s'est montré particulièrement robuste face à des attaques complexes et peu fréquentes, telles que les reconnaissances, portes dérobées, shellcodes et vers, souvent ignorées par les approches classiques en raison de leur faible représentation dans les jeux de données. Pour tester la généralisation du modèle à d'autres contextes, une évaluation complémentaire a été réalisée sur les ensembles CICIDS2019 et CICIoT2023, intégrant des scénarios d'attaque récents et réalistes. Les résultats ont confirmé la fiabilité, la stabilité et la transférabilité de notre approche dans des environnements variés.

En synthèse, la méthode proposée constitue une réponse efficace et évolutive aux défis posés par le déséquilibre des données dans les systèmes de détection d'intrusion. En mettant l'accent sur la sélection de caractéristiques explicables et spécifiques aux attaques rares, cette approche permet d'améliorer la détection sans compromettre l'intégrité des données ni recourir à des techniques d'augmentation artificielle.

Dans la continuité de ce travail, plusieurs pistes d'amélioration peuvent être explorées. Une première direction prometteuse consiste à intégrer des représentations basées sur des graphes, afin d'exploiter les relations contextuelles entre les événements réseau, les hôtes, les ports, ou les séquences d'actions malveillantes. Les graphes de dépendances ou de communication permettraient notamment de détecter des comportements anormaux non évidents à partir de simples vecteurs de caractéristiques, en capturant des motifs d'attaque plus complexes et dynamiques.

Par ailleurs, l'utilisation de modèles graph-based (tels que les Graph Neural Networks – GNNs) pourrait enrichir considérablement la capacité du système à comprendre le contexte d'une intrusion, en tenant compte de la topologie réseau ou de la propagation temporelle d'une attaque. Cette intégration ouvrirait également la voie à une détection plus fine des attaques coordonnées, souvent indétectables individuellement.

Enfin, des travaux futurs pourraient s'intéresser à l'apprentissage semi-supervisé ou non supervisé, afin de détecter des attaques zéro-day ou inconnues, en combinant des techniques de clustering, d'analyse de similarité ou d'auto-encodage avec l'architecture proposée.

4

Détection d'attaques basée sur la représentation des données en graphe

Dans nos travaux précédents, nous avons employé un modèle de réseaux de neurones récurrents de type LSTM afin d'identifier les comportements d'attaque, en nous appuyant sur l'historique des événements réseau. Parallèlement, l'approche PV-DM (Paragraph Vector - Distributed Memory) a été exploitée pour apprendre des représentations vectorielles enrichies, intégrant de manière implicite le contexte global des séquences ainsi que les contextes locaux propres à chaque événement. Ces représentations ont permis de capturer des patterns temporels et sémantiques pertinents pour la détection d'anomalies.

Dans la continuité de ces travaux, nous avons proposé un modèle innovant visant spécifiquement la détection des attaques rares, souvent négligées par les méthodes classiques en raison de leur faible fréquence d'occurrence. Cette problématique soulève une interrogation centrale : ne serait-il pas pertinent de fusionner les différentes approches explorées afin de concevoir un modèle unique, capable à la fois de caractériser finement le contexte des attaques et d'améliorer la détection des cas rares ?

Dans cette perspective, nous avons exploré les réseaux de neurones convolutionnels sur graphes (GCN) comme solution unificatrice pour capturer à la fois les dynamiques globales et locales des données. Le paradigme du graphe offre en effet une représentation flexible et expressive des relations complexes entre événements, permettant de modéliser des interactions non linéaires souvent invisibles aux approches traditionnelles. En structurant les données sous forme de graphes, il devient possible de tirer parti de la connectivité implicite entre les instances, d'enrichir les représentations contextuelles et de faciliter la détection de comportements atypiques, notamment ceux associés aux attaques rares. Le GCN permet ainsi une propagation efficace de l'information au sein du graphe, tout en conservant la topologie locale. Ce cadre ouvre la voie à de nouvelles opportunités pour renforcer la robustesse et la précision des systèmes de détection.

1 Introduction

Les systèmes de détection d'intrusion (IDS) actuels, tels que présentés dans l'état de l'art, reposent majoritairement sur des modèles d'apprentissage automatique qui exploitent les caractéristiques statistiques extraites des flux réseau. Ces approches se concentrent soit sur une classification directe à partir de ces caractéristiques, soit sur la génération de signatures d'attaques connues. Toutefois, cette focalisation sur des attributs isolés limite leur portée [183]. En effet, en négligeant les relations sémantiques et temporelles entre les événements réseau, ces modèles peinent à saisir le contexte global des comportements observés, ce qui compromet leur efficacité face aux attaques complexes, furtives ou rares [184].

Par ailleurs, la diversité et l'évolution constante des techniques d'intrusion posent un défi majeur pour la généralisation des modèles. Lorsqu'ils sont confrontés à des attaques inconnues ou peu représentées dans les données d'entraînement, ces modèles tendent à produire un grand nombre de faux positifs. Cette limitation nuit non seulement à leur précision mais aussi à leur déploiement opérationnel dans des environnements réels, où la fiabilité est cruciale [185].

Face à ces défis, les réseaux neuronaux graphiques (GNN) apparaissent comme une alternative prometteuse. En tant que technologie émergente en apprentissage profond, les GNN sont conçus pour traiter efficacement des données représentées sous forme de graphes, en capturant les interactions complexes entre entités. Contrairement aux architectures classiques, ils permettent de modéliser explicitement les relations et dépendances dans les données, offrant ainsi un potentiel accru pour des applications où la structure relationnelle est essentielle, telles que les réseaux sociaux, la bio-informatique ou les systèmes de recommandation [186].

Dans cette optique, ce chapitre introduit une nouvelle approche de représentation des flux réseau sous forme de graphes, exploitant les propriétés structurelles et topologiques des données pour renforcer la détection d'intrusions, notamment celles peu fréquentes. L'approche proposée repose sur une combinaison de techniques éprouvées : un prétraitement par clustering via K-means pour segmenter les données, une construction de graphes intra-clusters à l'aide de l'algorithme K plus proches voisins (KNN) implémenté avec FAISS, et enfin une classification des nœuds par un réseau de neurones convolutionnels sur graphes (GCN).

Les principales contributions de ce chapitre sont les suivantes :

- ✓ Clustering des flux réseau avec K-means : permettant de regrouper les données similaires afin de mieux structurer l'espace d'entrée pour la suite du traitement.
- ✓ Génération de graphes locaux via KNN/FAISS : chaque cluster est représenté sous forme de graphe, où les relations de proximité entre les instances sont explicitement modélisées.
- ✓ Classification par GCN : les graphes ainsi constitués sont exploités par un GCN pour apprendre des représentations contextuelles et classifier les événements réseau, y compris les comportements rares.

La structure du chapitre est organisée comme suit : la section 2 présente les travaux connexes et les approches existantes. La section 3 introduit les concepts théoriques fondamentaux et décrit les méthodes utilisées. La section 4 détaille le modèle proposé et son architecture. Les résultats expérimentaux sont exposés et discutés dans la section 5. Enfin, la section 6 conclut le chapitre et ouvre des perspectives pour de futures recherches.

2 Synthèse des limites et contributions

Dans cette section, nous présentons une revue critique de plusieurs travaux récents portant sur l'utilisation des réseaux de neurones graphiques (GNN) pour la détection d'intrusions. Ces études illustrent les différentes stratégies adoptées pour représenter les données issues du trafic réseau sous forme de graphes, ainsi que les performances associées à ces approches. Nous mettons particulièrement l'accent sur les opportunités offertes par ces modèles, tout en identifiant leurs limites structurelles et conceptuelles.

L'étude [187] introduit BotGM, une approche non supervisée exploitant des graphes construits à partir de données NetFlow pour détecter les botnets. En modélisant les dépendances entre flux, BotGM parvient à identifier des sous-graphes anormaux avec un taux de précision atteignant 95%, surpassant largement les méthodes fondées sur des représentations individuelles de flux. Cette étude met en lumière le potentiel des représentations globales pour améliorer la sensibilité aux comportements collectifs malveillants.

Dans [188], les auteurs proposent une représentation orientée hôte, où les attaques sont modélisées par des graphes de connexions inter-hôtes. Cette méthode permet de capturer des motifs structurels liés aux attaques grâce à un GNN spécialisé. Testée sur CIC-IDS2017, l'approche atteint un F1-score de 0,99, rivalisant avec les modèles d'apprentissage traditionnels tout en conservant une robustesse appréciable face aux variations de caractéristiques.

L'étude [189] explore une stratégie originale : construire des graphes uniquement à partir de la topologie du réseau, sans intégrer de caractéristiques explicites. Bien que cette approche simplifie la modélisation, elle limite la capacité des GNN à apprendre des comportements complexes, puisque l'information contextuelle issue des flux ou des entités n'est pas exploitée. Cette limitation remet en question l'efficacité de représentations topologiques isolées pour la détection d'attaques sophistiquées.

Dans le domaine des infrastructures critiques, [190] propose un modèle hybride combinant GCN et LSTM (GC-LSTM) pour surveiller les réseaux électriques face aux cyberattaques. En s'appuyant sur une modélisation des flux temporels et structurels, le modèle atteint une précision supérieure à 96%, illustrant l'intérêt des architectures hybrides pour des contextes dynamiques et sensibles.

Les travaux présentés dans [191] [192] [193] [194] adoptent une représentation graphique classique, où les utilisateurs sont modélisés comme des nœuds et les flux de communication comme des arêtes. Cette structure présente plusieurs limites : elle peut être contournée car elle repose sur les adresses IP, nécessite une phase de prétraitement pour être compatible avec les algorithmes GNN, et ne fournit pas d'informations topologiques pertinentes concernant les attaques distribuées ou en plusieurs étapes.

L'étude [195] traite du problème de la détection d'intrusions dans les réseaux de capteurs sans fil en utilisant des données simulées. Un modèle de GNN y est défini pour une classification binaire. Cependant, cette méthode n'est pas applicable à une topologie de réseau non structurée, et l'intervalle de temps nécessaire pour associer une intrusion est élevé.

Les travaux [196] [197] proposent des modèles GCN pour la détection de botnets dans les systèmes réseau. Afin de capturer les dépendances dans des architectures de botnets étendues, les auteurs ont utilisé 12 couches GCN. Toutefois, la construction d'un modèle GCN avec un aussi grand nombre de couches le rend vulnérable au phénomène de sur-lissage (over-smoothing) [198],

ce qui réduit l'efficacité de ses prédictions.

Malgré les avancées significatives permises par l'introduction des réseaux neuronaux graphiques dans le domaine de la cybersécurité, plusieurs limitations majeures persistent dans les approches actuelles.

1. Premièrement, un grand nombre de travaux reposent sur des représentations graphiques simplifiées, souvent construites à partir de la seule topologie du réseau ou en utilisant des abstractions limitées comme les adresses IP ou les relations hôte-à-hôte. Or, ces représentations ignorent la richesse informationnelle des caractéristiques des flux, ainsi que les dynamiques locales et globales du trafic réseau. Cette approche limite fortement la capacité des modèles à capturer des comportements malveillants complexes, en particulier dans les cas d'attaques furtives, distribuées ou à faible fréquence.
2. Deuxièmement, plusieurs méthodes existantes appliquent des GNN à des graphes mono-granulaires, sans tenir compte de la variabilité contextuelle entre différents groupes de trafic. Cela empêche une spécialisation locale du modèle, le forçant à généraliser sur des structures hétérogènes, ce qui peut entraîner une baisse de performance ou une sensibilité accrue au bruit. D'autres modèles, à l'inverse, utilisent des architectures excessivement profondes (par exemple avec plus de 10 couches GCN), ce qui les rend vulnérables au sur-lissage (over-smoothing), un phénomène où les représentations des nœuds deviennent indistinctes et peu informatives.
3. Troisièmement, un nombre non négligeable d'approches adoptent une vision binaire de la détection (attaque vs. non-attaque), sans capacité à discriminer les types ou sous-types d'attaques. Cette limitation réduit considérablement l'utilité de ces systèmes pour les analystes en sécurité, qui ont besoin de connaissances précises pour prendre des décisions adaptées.

Afin de répondre à ces limites, nous proposons une approche innovante articulée autour de plusieurs contributions :

- Représentation contextuelle enrichie : en représentant les flux réseau sous forme de graphes construits localement à partir de clusters homogènes (via KMeans), nous garantissons que chaque graphe encode un contexte spécifique, ce qui améliore la cohérence des relations modélisées. Cela permet de capturer les dépendances fines entre événements au sein d'un sous-ensemble cohérent de données.
- Construction de graphes pertinents : en utilisant KNN via FAISS pour créer des arêtes uniquement au sein de chaque cluster, nous évitons les connexions arbitraires entre points dissemblables. Cela renforce la structure locale du graphe, réduit le bruit, et préserve la sémantique des relations entre les nœuds, favorisant une propagation plus fiable de l'information dans le GCN.
- Approche modulaire et contrôlée : en évitant une profondeur excessive du GCN, notre modèle est conçu pour prévenir le sur-lissage, tout en conservant une capacité d'apprentissage suffisante. Chaque couche exploite l'information de voisinage immédiat sans effondrement des représentations.

- Capacité à détecter des attaques rares : la structuration fine des graphes et la richesse des représentations apprises permettent à notre modèle de mieux différencier les comportements anormaux subtils, souvent associés à des attaques peu fréquentes ou émergentes.
- Scalabilité et généralisabilité : la combinaison clustering + FAISS permet une construction efficace de graphes à grande échelle, compatible avec des environnements réseau dynamiques, comme ceux des réseaux IoT ou des systèmes distribués.

En résumé, en combinant structuration locale, efficacité computationnelle et apprentissage guidé par le contexte, notre approche dépasse les limitations observées dans l'état de l'art, et ouvre la voie à des systèmes de détection plus robustes, adaptatifs et précis.

3 Concept de base

3.1 Les réseaux neuronaux graphiques (GNN)

Avant d'aborder le fonctionnement des réseaux neuronaux graphiques, il est fondamental de comprendre ce qu'est un graphe et pourquoi cette structure joue un rôle central dans la modélisation de données complexes, interconnectées ou non-euclidiennes.

3.1.1 Définition et structure d'un graphe

Dans de nombreux domaines d'application, les données ne peuvent être efficacement représentées par des structures tabulaires classiques. C'est notamment le cas lorsque les relations entre entités ont une importance significative. Un graphe est une structure de données permettant de modéliser ces relations : les nœuds (ou sommets) représentent les entités, tandis que les arêtes (ou liens) modélisent les interactions ou connexions entre elles.

Mathématiquement, un graphe [199] est défini comme $G = (V, E)$ où :

- V est l'ensemble des nœuds ;
- $E \subseteq V \times V$ est l'ensemble des arêtes reliant ces nœuds.

Chaque nœud et chaque arête peut être associé à un vecteur de caractéristiques décrivant ses propriétés (par exemple, une adresse IP pour un nœud, un débit ou un protocole pour une arête). Il existe plusieurs types de graphes (voir figure 4.1) :

- Graphes non orientés : les arêtes ne possèdent pas de direction ;
- Graphes orientés : les arêtes ont un sens, ce qui est utile pour modéliser des flux d'informations ou des dépendances causales ;
- Graphes pondérés : chaque arête est associée à un poids numérique exprimant une intensité ou une importance.

Les graphes sont utilisés dans des domaines variés : réseaux sociaux, biologie moléculaire, systèmes de transport, ou encore Internet des objets (IoT).

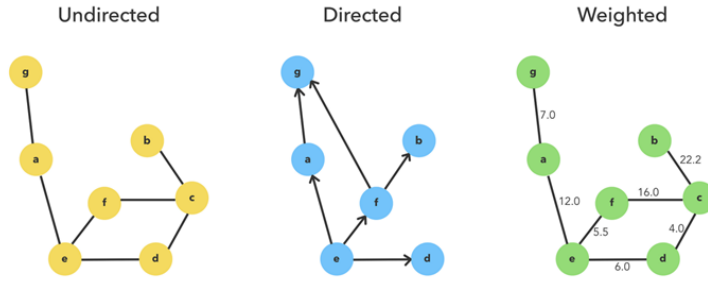


FIGURE 4.1 – Principaux types de structures de graphes

3.1.2 Les réseaux neuronaux graphiques

Les réseaux neuronaux graphiques sont une famille de modèles d'apprentissage profond conçus pour apprendre à partir de données structurées en graphes. Introduits initialement en 2005 [200], les GNN ont connu un essor considérable grâce à leur capacité à exploiter la structure du graphe et à capturer les dépendances entre nœuds, ce que ne permettent pas les réseaux de neurones classiques, conçus pour des données euclidiennes (images, textes, etc).

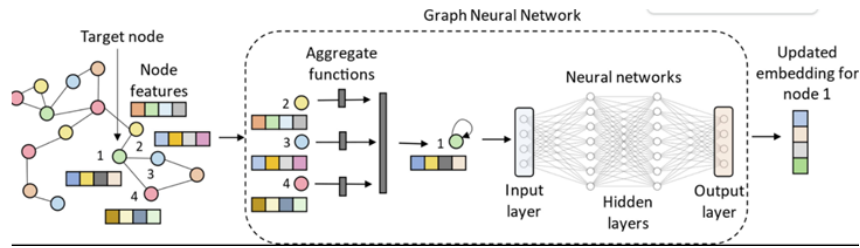


FIGURE 4.2 – Schéma fonctionnel d'un réseau neuronal graphique (GNN)

Le cœur des GNN repose sur un mécanisme appelé message passing. À chaque itération (ou couche), un nœud met à jour sa représentation (Feature_Vector) en agrégeant les informations provenant de ses voisins immédiats et éventuellement de lui-même [200] [201]. Cette mise à jour suit deux étapes principales : l'agrégation et la mise à jour.

- Fonction d'agrégation (AGGREGATE) : Elle consiste à collecter les représentations des nœuds voisins à l'itération $k-1$, soit

$$\mathbf{a}_v^{(k)} = \text{AGGREGATE}^{(k)} \left(\left\{ \mathbf{h}_u^{(k-1)} \mid u \in \text{Neighbors}(v) \right\} \right) \quad (4.1)$$

Avec :

- $\mathbf{h}_u^{(k-1)}$: est le feature du nœud u de l'itération précédente.
- $\text{Neighbors}(v)$: désigne l'ensemble des voisins du nœud v .

Les fonctions d'agrégation peuvent inclure : somme, moyenne, maximum, ou des variantes apprises.

- Fonction de mise à jour (Update) : Elle combine l'information agrégée $\mathbf{a}_v^{(k)}$ avec l'ancienne représentation du nœud pour en produire une nouvelle :

$$\mathbf{h}_v^{(k)} = \text{UPDATE}^{(k)} \left(\mathbf{h}_v^{(k-1)}, \mathbf{a}_v^{(k)} \right) \quad (4.2)$$

Avec :

- $\mathbf{h}_v^{(k-1)}$ est le feature du nœud v de l'itération précédente.
- $\mathbf{a}_v^{(k)}$ est le feature agrégée issue de la première étape.

En général, cette mise à jour est implémentée à l'aide d'une couche de réseau de neurones (par exemple une couche linéaire suivie d'une activation non linéaire comme ReLU).

L'agrégation vise à recueillir des informations provenant des nœuds voisins, dans le but de capturer les structures locales et les caractéristiques environnantes. Les fonctions d'agrégation les plus courantes (somme, moyenne, max) permettent de résumer efficacement l'environnement immédiat d'un nœud (souvent jusqu'à deux sauts). La mise à jour combine ensuite ces informations avec les attributs propres au nœud afin de produire un vecteur de caractéristiques enrichi, tenant compte de son contexte.

Ce processus, appelé message passing, regroupe l'ensemble des étapes d'agrégation et de mise à jour. Il est répété sur plusieurs couches, permettant une propagation de l'information à travers le graphe. Ainsi, chaque nœud peut progressivement intégrer des informations en provenance de régions plus éloignées du graphe, ce qui est crucial pour la détection de patterns complexes ou distribués [202] [203].

Capacités des GNN

Les GNN offrent plusieurs types de prédiction (figure 4.3), en fonction de l'objectif d'apprentissage :

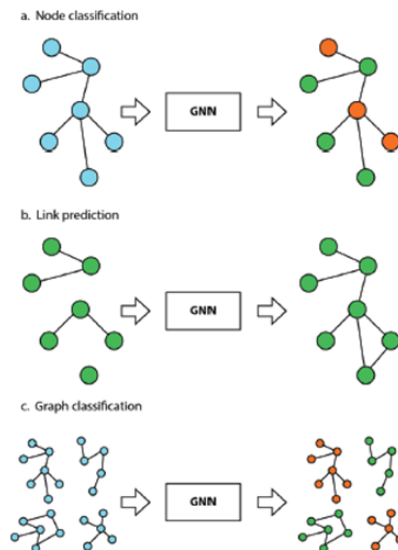


FIGURE 4.3 – Les tâches courantes sur les graphes incluent la classification de nœuds, la prédiction de liens et la classification du graphe

- Classification de nœuds : prédire une étiquette pour chaque nœud (ex. : normal vs. Compromis).
- Classification de graphes : assigner une classe à un graphe entier (ex. : type de trafic ou scénario d'attaque).
- Prédiction de liens : estimer la probabilité qu'une arête (relation) existe entre deux nœuds donnés (utile pour la détection d'interactions suspectes).

Grâce à leur capacité à exploiter les structures locales et globales, les GNN représentent une avancée majeure pour la modélisation des données relationnelles. Ils s'avèrent particulièrement adaptés à la détection d'intrusions, où les interactions entre événements peuvent révéler des comportements malveillants qui ne seraient pas détectables à partir des données individuelles seules.

3.2 K-means clustering (K-means)

Le clustering K-means est l'un des algorithmes de regroupement les plus populaires en apprentissage non supervisé. Il est couramment utilisé pour partitionner un ensemble de données en un nombre prédéfini de groupes (ou clusters) homogènes (voir figure 4.4). L'objectif principal de K-means est de minimiser la variance intra-cluster (la dispersion des points autour du centre de leur groupe) tout en maximisant la séparation inter-clusters [204] [205].

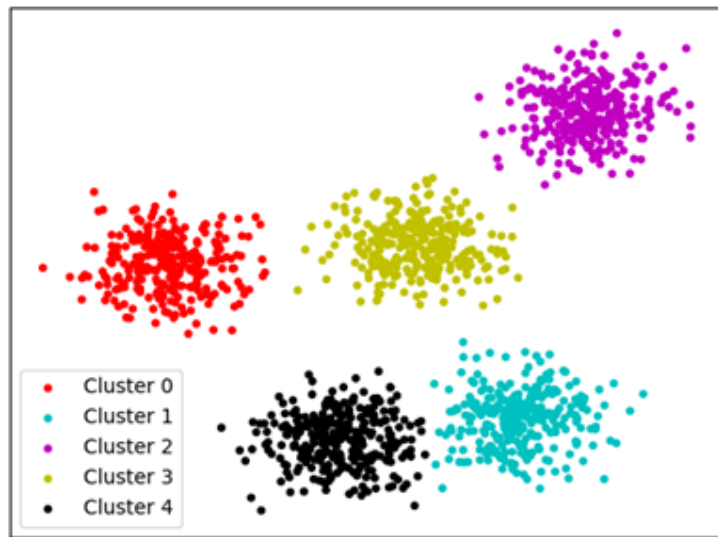


FIGURE 4.4 – Exemple de résultats de clustering

3.2.1 Principe de fonctionnement

L'algorithme K-means repose sur une approche itérative en trois étapes principales :

1. Initialisation : sélection aléatoire de k centres initiaux (appelés centroïdes) dans l'espace des données.
2. Affectation : chaque point de données est affecté au cluster dont le centroïde est le plus proche, généralement selon la distance euclidienne.
3. Mise à jour des centroïdes : pour chaque cluster, un nouveau centroïde est calculé comme la moyenne des points qui y sont assignés.

Ces étapes sont répétées jusqu'à convergence, c'est-à-dire lorsque les affectations de points ne changent plus ou que le changement de la fonction objectif devient négligeable.

3.2.2 Fonction objective

L'objectif de K-means est de minimiser la fonction de coût suivante :

$$E = \sum_{i=1}^K \sum_{x \in C_i} \|x - \mu_i\|^2 \quad (4.3)$$

Où :

- E représente l'erreur totale (ou la fonction de coût à minimiser, c'est-à-dire la somme des distances au carré),
- K représente le nombre de clusters, • C_i représente l'ensemble des points assignés au i-ème cluster,
- x est un point de données appartenant au i-ème cluster,
- μ_i est le centroïde (ou la moyenne) du i-ème cluster.

La distance utilisée est la distance euclidienne, dont la formule est la suivante :

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (4.4)$$

Où :

- $d(x, y)$: distance euclidienne entre les points x et y.
- x_i, y_i : i-ème composante des vecteurs x et y.
- n : dimension de l'espace.

3.2.3 Avantages et limites

L'algorithme K-means est apprécié pour sa simplicité, sa rapidité de convergence et sa scalabilité, ce qui le rend particulièrement adapté à de grands ensembles de données. Cependant, il présente également certaines limitations :

- Il nécessite de spécifier le nombre de clusters k à l'avance.
- Il est sensible à l'initialisation des centroïdes (différentes initialisations peuvent donner différents résultats).
- Il n'est pas robuste aux outliers et suppose des clusters sphériques de taille similaire.

3.2.4 Application dans notre approche

Dans notre travail, l'algorithme K-means est utilisé en phase de prétraitement afin de regrouper les points de données réseau en clusters homogènes, ce qui permet de construire des graphes locaux cohérents. Cette structuration préalable renforce la qualité des relations inter-nœuds modélisées par le GNN, tout en réduisant la complexité et le bruit dans la phase d'apprentissage.

3.3 Recherche des plus proches voisins avec KNN via FAISS

Après la phase de regroupement par K-means, une étape essentielle dans notre approche consiste à établir des connexions entre les données au sein de chaque cluster, en vue de construire une structure graphique pertinente. Pour cela, nous utilisons l'algorithme des k plus proches voisins (K-Nearest Neighbors, KNN), couplé à la bibliothèque FAISS (Facebook AI Similarity Search), optimisée pour des recherches efficaces dans des espaces vectoriels de grande dimension.

3.3.1 Principe du K plus proches voisins (KNN)

L'algorithme KNN [206] consiste pour chaque point de données à identifier les k instances les plus proches dans l'espace de caractéristiques en utilisant généralement la distance euclidienne comme métrique de similarité. Cette méthode est largement utilisée pour construire des graphes de similarité, réaliser des prédictions en classification ou en régression, et représenter des relations de proximité dans des contextes non supervisés [207] [208].

Dans le cadre de la construction de graphes, les résultats du KNN déterminent les arêtes du graphe : chaque nœud est connecté à ses k plus proches voisins, créant ainsi une topologie qui reflète la structure locale des données.

3.3.2 FAISS (Recherche de similarité IA sur Facebook)

FAISS (Facebook AI SimilaritySearch) est une bibliothèque open-source développée par Facebook AI Research, conçue pour effectuer des recherches de similarité rapide dans des grands ensembles de vecteurs à haute dimension. Elle prend en charge plusieurs algorithmes de recherche [209] :

- Recherche brute exacte,
- Indexation hiérarchique (IVF, PQ),
- Optimisations GPU pour accélérer le traitement.

Dans ce travail de recherche, nous avons utilisé FAISS pour construire l'index et effectuer la recherche des k plus proches voisins (IndexFlatL2), ce qui nous permet de générer efficacement la topologie du graphe (liste des arêtes) à l'intérieur de chaque cluster.

Dans notre approche, FAISS est utilisé pour :

1. Réaliser une recherche rapide et scalable des voisins dans chaque cluster,
2. Réduire le coût computationnel de la recherche KNN classique,
3. Garantir une qualité constante des liens inter-nœuds, même dans des jeux de données volumineux ou complexes.

3.3.3 Réseau de Neurones Convolutif sur Graphes (GCN)

Les GCN [210] constituent une extension des réseaux neuronaux convolutifs (CNN), initialement développés pour la vision par ordinateur, mais adaptés ici au traitement de données non-euclidiennes, telles que les graphes. Leur objectif est de généraliser l'opération de convolution aux structures relationnelles, en exploitant explicitement la topologie du graphe pour apprendre des représentations enrichies des nœuds [211].

Contrairement aux réseaux traditionnels où les connexions sont fixes et régulières, les GCN appliquent la convolution sur des graphes, où les nœuds peuvent avoir un nombre variable de voisins. Chaque nœud est initialement représenté par un vecteur de caractéristiques ; à travers une série de couches convolutives, ces représentations sont transformées pour intégrer progressivement l'information structurelle et contextuelle de leur voisinage.

Le mécanisme fondamental des GCN repose sur le message passing, un processus itératif au cours duquel chaque nœud met à jour sa représentation en agrégeant les informations de ses voisins, puis en les combinant avec ses propres attributs. Ces deux étapes peuvent être décrites comme suit [210] :

- Agrégation : collecte des représentations des nœuds voisins ;
- Mise à jour : application d'une transformation non linéaire (souvent via une couche neuronale) pour générer une nouvelle représentation.

Ce processus peut être formalisé par l'équation suivante :

$$h_v^{(k)} = \sigma \left(\mathbf{W}^{(k)} \cdot \text{AGGREGATE}^{(k)} \left(\left\{ h_u^{(k-1)} \mid u \in \mathcal{N}(v) \cup \{v\} \right\} \right) \right) \quad (4.5)$$

Où :

$h_v^{(k)}$: est la représentation du nœud v à la k -ième couche,
 $\mathcal{N}(v)$: représente l'ensemble des voisins du nœud v ,
 $\mathbf{W}^{(k)}$: est la matrice de poids de la couche k ,
 σ : est une fonction d'activation non linéaire (ex. ReLU).

Ce processus est répété à chaque couche du GCN, permettant à un nœud d'intégrer de l'information provenant de voisins de plus en plus éloignés. Après k couches, un nœud est donc informé par ses k -hop voisins, ce qui enrichit sa représentation avec un contexte [212].

4 Approche proposée

Notre pipeline de détection se compose de trois étapes interdépendantes mais modulaires (voir figure 4.5) :

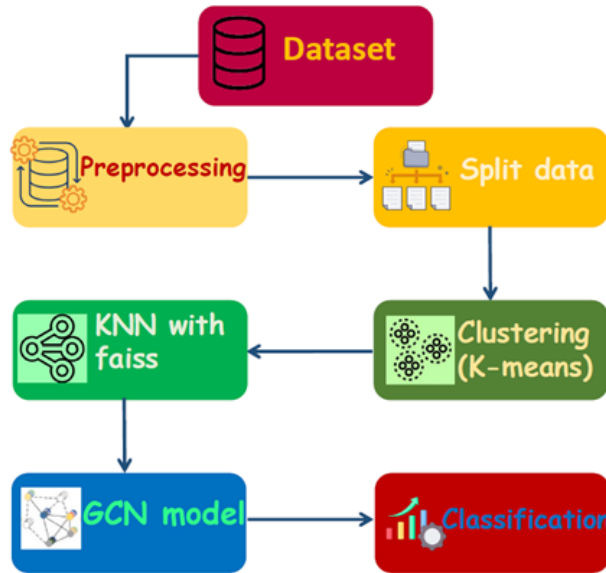


FIGURE 4.5 – Schéma du modèle proposé

1. Clustering via K-means : segmentation des flux réseau en sous-groupes homogènes ;
2. Construction de graphes locaux via KNN + FAISS : structuration de chaque cluster en graphe de proximité ;

3. Classification via GCN : apprentissage supervisé des représentations et classification des nœuds.

Cette architecture permet une détection fine, contextualisée et extensible, adaptée à des environnements complexes et dynamiques, comme les réseaux informatiques hétérogènes ou les systèmes distribués.

4.1 Description du dataset

Le dataset IoTID20 [213] a été conçu pour l'évaluation des systèmes de détection d'intrusions dans un contexte réaliste d'Internet des objets (IoT). Il a été généré à l'aide d'une plateforme expérimentale simulant un environnement domestique intelligent typique. Cette plateforme comprend divers dispositifs connectés [214], parmi lesquels :

- Le haut-parleur intelligent SKT NUGU, basé sur l'intelligence artificielle,
- Une caméra de surveillance Wi-Fi EZVIZ,
- Des équipements clients standards tels que smartphones, tablettes et ordinateurs portables, le tout interconnecté via un routeur Wi-Fi de maison intelligente.

Dans cette configuration, le SKT NUGU et la caméra EZVIZ sont utilisés comme dispositifs IoT cibles, tandis que les autres équipements jouent le rôle d'attaquants. Ce cadre permet de reproduire fidèlement des scénarios d'attaques plausibles dans un réseau domestique IoT hétérogène. Les fichiers de trafic brut ont été collectés à l'aide du mode moniteur d'un adaptateur sans fil, garantissant ainsi une capture complète des paquets échangés sur le réseau. Divers types d'attaques ont été simulés dans ce contexte (voir tableau 4.1), notamment :

- Des attaques par déni de service (DoS),
- Des attaques de balayage réseau (scanning),
- Des attaques de type Man-in-the-Middle (MitM), réalisées à l'aide d'outils classiques comme Nmap.

En complément, des paquets d'attaque de type botnet Mirai ont été générés puis modifiés afin de simuler une provenance légitime depuis des objets connectés, renforçant la complexité et le réalisme des scénarios.

TABLE 4.1 – Description et nombres des instances par catégorie d'attaque

Type d'attaque	Description	Nombre d'instances
DoS	Attaque par déni de service visant à rendre un service ou un réseau indisponible	59 391
Mirai	Botnet d'objets connectés utilisé pour réaliser des attaques par déni de service distribué (DDoS)	415 677
MITM	Attaque de type Man-in-the-Middle, où l'attaquant intercepte et manipule les communications entre deux parties	35 377
Scan	Analyse des ports ou des vulnérabilités pour identifier des cibles potentielles	75 265
Normal	Instances normales	40 073

Les traces réseau enregistrées ont été transformées en flux de données au format lisible en machine à l'aide de l'outil CICFlowMeter [214], qui extrait :80 caractéristiques réseau (features) représentant des métriques de flux (durée, paquets, octets, flags, etc.), et trois types d'étiquettes :Binaire (attaque vs. normal),Catégorie (type d'attaque),Sous-catégorie (sous-type spécifique d'attaque) [215].

Dans notre étude, nous nous focalisons spécifiquement sur l'étiquette de niveau "catégorie", qui regroupe les attaques selon leur typologie générale, permettant une analyse plus ciblée et contextuelle.

Le nombre d'instances par classe d'attaque est présenté dans le tableau 4.1, mettant en évidence une répartition déséquilibrée typique des environnements IoT, où certaines attaques apparaissent beaucoup plus fréquemment que d'autres, un défi central pour les systèmes de détection performants.

4.2 Prétraitement des données

Le prétraitement constitue une étape cruciale dans tout pipeline d'apprentissage automatique, notamment lorsqu'il s'agit de données complexes comme celles issues de flux réseau. Dans le cadre de cette étude, les opérations suivantes ont été réalisées pour garantir la qualité, la cohérence et la compatibilité des données avec les modèles d'apprentissage supervisé :

- **Suppression des attributs non pertinents** : Les colonnes Sub_Cat et Label, qui fournissent des informations redondantes ou non exploitées dans le cadre de notre classification par catégories générales d'attaques, ont été retirées. Cette étape vise à réduire la dimensionnalité et à éviter toute confusion dans l'apprentissage supervisé.
- **Élimination des doublons** : Un nettoyage approfondi a permis d'éliminer les lignes dupliquées au sein des classes cibles. Cette opération évite la redondance dans l'espace des caractéristiques et préserve la diversité des instances, ce qui est essentiel pour un apprentissage efficace et non biaisé.
- **Encodage des variables catégorielles** : Les champs de type texte tels que Flow_ID, Src_IP, Dst_IP et Timestamp, ont été transformés en valeurs numériques à l'aide du LabelEncoder de la bibliothèque scikit-learn. Bien que certaines de ces variables n'aient pas de signification sémantique directe, leur encodage permet leur inclusion dans la phase exploratoire ou leur exclusion via une analyse de pertinence.
- **Codage des classes cibles** : La variable Cat, correspondant aux catégories d'attaques, a été convertie en valeurs entières (ex. : Normal = 0, DoS = 1, Scan = 2, etc.). Cette transformation est indispensable pour permettre l'utilisation des fonctions de perte catégorielles (comme cross-entropy) et pour faciliter l'interprétation des résultats de classification.
- **Gestion des valeurs aberrantes et manquantes** : Le dataset a été examiné pour détecter la présence de valeurs manquantes ou infinies, susceptibles d'altérer l'entraînement du modèle. Ces valeurs ont été supprimées ou corrigées, garantissant ainsi l'intégrité statistique des données.
- **Normalisation des caractéristiques** : Un StandardScaler a été appliqué sur les variables numériques afin de centrer les distributions autour de zéro et de les ramener à une variance

unitaire. Cette étape est fondamentale pour les modèles sensibles à l'échelle des données (comme les GNN ou les réseaux denses), et elle améliore la vitesse de convergence durant l'apprentissage.

Une fois ces étapes de prétraitement complétées, l'ensemble de données a été divisé en trois sous-ensembles :

- Ensemble d'entraînement : 60% des données, utilisé pour ajuster les paramètres du modèle,
- Ensemble de validation : 20%, utilisé pour ajuster les hyperparamètres et éviter le surapprentissage,
- Ensemble de test : 20%, utilisé pour évaluer la généralisation du modèle sur des données non vues.

Ce protocole rigoureux de préparation garantit que les données sont prêtes pour une phase de modélisation fiable, tout en assurant une représentativité équilibrée et réaliste des différents types d'attaques.

4.3 Construction du graphe pour l'apprentissage GCN

Afin d'exploiter pleinement la capacité des réseaux neuronaux graphiques à modéliser des relations complexes entre entités, nous avons adopté une stratégie de construction de graphe guidée par le contexte local. Plutôt que de construire un graphe global sur l'ensemble du dataset, ce qui introduirait des connexions peu pertinentes ou bruitées. Nous avons structuré le graphe en deux étapes principales, s'appuyant sur le clustering K-means et la recherche de voisinage par KNN via FAISS.

4.3.1 Clustering local avec K-means

Dans une première phase, les données sont segmentées en k clusters homogènes à l'aide de l'algorithme K-means. Cette segmentation a pour but de capturer des structures locales cohérentes, où les points de données (ou nœuds) partagent des caractéristiques similaires. Chaque cluster peut être interprété comme une communauté locale de comportements réseau, permettant de limiter l'analyse à des sous-ensembles thématiquement ou statistiquement homogènes.

4.3.2 Construction locale du graphe avec KNN via FAISS

Une fois les clusters définis, un graphe est construit au sein de chaque cluster selon les étapes suivantes :

- Pour chaque point de données, nous identifions ses k -plus proches voisins dans le même cluster (avec $k=5$) en utilisant l'algorithme KNN.
- Les connexions (arêtes) sont créées entre ce point et ses voisins directs, sur la base de la distance euclidienne (L2).
- Le processus est accéléré grâce à l'implémentation optimisée de FAISS (Facebook AI SimilaritySearch), qui permet une recherche de voisinage rapide et scalable même dans des espaces à haute dimension.

L'utilisation combinée de K-means et KNN via FAISS permet une construction efficace de graphes locaux : les arêtes sont définies uniquement entre les nœuds d'un même cluster, évitant ainsi des connexions incohérentes entre régions de données dissemblables. Cette approche renforce la cohérence topologique de chaque sous-graphe, facilite la propagation de l'information au sein du GNN, et contribue à réduire le bruit et le surapprentissage. Elle est particulièrement adaptée à des contextes où les relations locales sont plus informatives que la structure globale brute, comme c'est souvent le cas en détection d'anomalies réseau.

Le résultat est un ensemble de graphes locaux dans lesquels :

- ✓ Les nœuds correspondent aux points de données (flux réseau),.
- ✓ Les arêtes traduisent des relations de proximité intra-cluster.

Cette approche présente plusieurs avantages : elle préserve la structure communautaire implicite des données, renforce la cohérence de la propagation des messages dans le GNN, et réduit le bruit et le risque de surapprentissage en se focalisant sur des relations significatives.

4.3.3 Modélisation GCN sur les graphes construits

Une fois les graphes locaux construits à partir des clusters et des relations de voisinage identifiées via KNN et FAISS, nous appliquons un réseau de neurones convolutif sur graphes (GCN) pour effectuer la classification des nœuds. Cette étape constitue le cœur du processus d'apprentissage supervisé de notre modèle.

Entrée du modèle

Les données sont encodées dans un objet Data de PyTorchGeometric (PyG), contenant :

- `x` : les caractéristiques des nœuds, issues des flux réseau prétraités,
- `edge_index` : la matrice de connectivité, définie par les relations KNN intra-cluster.

Propagation de l'information

Le GCN applique plusieurs couches de convolution graphique (GCNConv), au sein desquelles chaque nœud agrège les représentations de ses voisins. Ces représentations sont ensuite pondérées et transformées à l'aide de fonctions d'activation (ReLU). L'information est ainsi progressivement propagée à travers les couches, ce qui permet à chaque nœud d'enrichir sa représentation avec un contexte de plus en plus large, pouvant s'étendre jusqu'à k -sauts dans le graphe.

Grâce à la cohérence structurelle introduite par le clustering, les voisins sont généralement issus de comportements similaires (ex. : même type d'attaque), ce qui renforce la pertinence des représentations apprises.

Apprentissage et prédiction

Le modèle apprend des embeddings de nœuds qui intègrent à la fois leurs attributs individuels et le contexte relationnel local.

À la dernière couche, le GCN génère des log-probabilités par classe, utilisées pour la prédiction du type d'intrusion (ex. : Normal, DoS, Scan, etc.).

Paramètres du modèle

Le modèle GCN a été entraîné avec les paramètres décrits dans le tableau 4.2.

TABLE 4.2 – *Configuration des hyperparamètres du modèle GCN*

Paramètres	Valeurs
Nombre de couches GCN	3
Fonction d'activation	ReLU
Optimiseur	Adam
Taux d'apprentissage	0,01
Nombre d'époques	200
Taux de dropout	0,5

L'utilisation du dropout permet de régulariser le modèle, en évitant le surajustement, tandis que l'optimiseur Adam assure une convergence stable et rapide.

Avantages de l'approche GCN dans notre système

L'intégration du GCN dans notre pipeline offre plusieurs bénéfices notables :

- Robustesse accrue face aux attaques contextuelles ou peu fréquentes, en exploitant la cohérence locale des graphes construits ;
- Réduction du surapprentissage, grâce à des clusters homogènes et une topologie maîtrisée ;
- Propagation contrôlée de l'information, évitant le sur-lissage rencontré dans les architectures trop profondes ;
- Scalabilité et modularité, rendues possibles par l'indépendance des graphes locaux et l'efficacité de FAISS dans la recherche de voisinage.

5 Résultats expérimentaux et discussion

L'évaluation expérimentale du modèle proposé, basé sur une architecture GCN appliquée à des graphes construits localement via clustering K-means et KNN intra-cluster, a permis de mettre en évidence des performances remarquables, tant sur le plan global que par classe. L'apprentissage a été effectué sur 200 époques avec une fonction de perte Cross-Entropy, et les résultats ont été mesurés à l'aide de métriques classiques de classification multiclasse : précision, rappel, F1-score, coefficient de corrélation de Matthews (MCC) et indice de Kappa de Cohen.

La Figure 4.6, illustrant l'évolution de la fonction de perte sur les ensembles d'entraînement et de validation, montre une convergence fluide et symétrique des deux courbes. La perte chute rapidement au cours des premières itérations pour se stabiliser à des valeurs proches de zéro dès la 100e époque, ce qui indique une convergence rapide du modèle et une bonne capacité de généralisation. L'absence de divergence entre les deux courbes exclut tout surajustement manifeste, ce qui confirme que l'architecture GCN, combinée à la structuration en graphes locaux, favorise un



FIGURE 4.6 – Courbe de perte pendant l'entraînement

apprentissage efficace, même dans un espace de décision complexe.

Les performances quantitatives globales du modèle sont très élevées (table 4.3). Une accuracy de 98,57% et un F1-score de 98,09% témoignent d'un équilibre entre la précision (98,52%) et le rappel (97,67%), ce qui est essentiel dans un contexte de détection d'intrusions où les faux négatifs sont critiques. Les indicateurs de robustesse globale, tels que le Cohen Kappa (97,74%) et le MCC (97,75%), montrent que le modèle conserve une qualité de prédiction élevée même face à la distribution déséquilibrée des classes dans le dataset IoTID20.

TABLE 4.3 – Résultats globaux de performance du modèle GCN

Accuracy(%)	Recall(%)	Precision(%)	F1-Score(%)	Cohen Kappa(%)	Matthews Corr Coef(%)
98,57	97,67	98,52	98,09	97,74	97,75

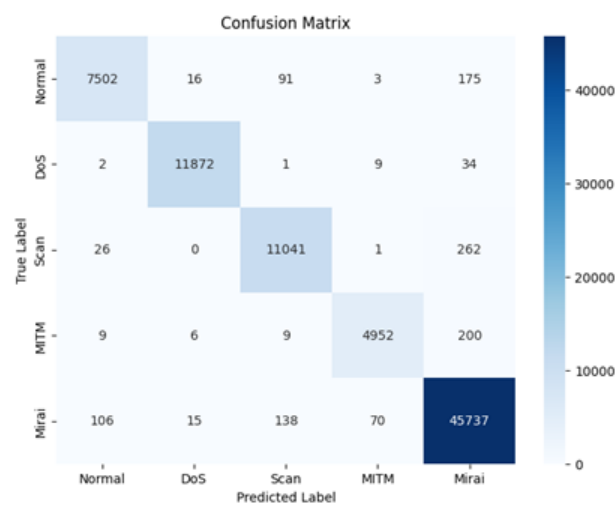


FIGURE 4.7 – Courbe de perte pendant l'entraînement

L'analyse fine de la matrice de confusion (Figure 4.7) corrobore ces résultats. La classe DoS

est détectée de façon presque parfaite avec 11 872 instances correctement classées sur un total quasi équivalent, un taux de précision de 99,69% et un F1-score de 99,65%. La classe Mirai, qui représente un volume important du dataset, est également bien identifiée, avec 45 737 exemples prédits correctement et un F1-score de 98,92%, malgré quelques erreurs de classification vers "Normal" (106 cas) ou "Scan" (138 cas), ce qui peut être attribué à une similarité comportementale dans certaines fenêtres temporelles.

TABLE 4.4 – *Performances du modèle par classe*

Classe	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
DoS	99,61	99,69	99,61	99,65
Mirai	99,29	98,55	99,29	98,92
MITM	95,67	98,35	95,67	96,99
Scan	97,45	97,88	97,45	97,66
Normal	96,34	98,13	96,34	97,23

Les classes Scan et MITM montrent aussi de bonnes performances, avec des F1-scores de 97,66% et 96,99% respectivement (table 4.4). Cependant, la classe MITM présente une sensibilité légèrement inférieure (rappel = 95,67%), probablement en raison du nombre plus restreint d'exemples d'attaque et de leur variabilité comportementale. Ces résultats soulignent la difficulté intrinsèque à détecter des attaques furtives ou latérales, mais démontrent aussi que notre modèle reste performant même dans ces conditions.

Concernant la classe Normal, bien que globalement bien distinguée (F1-score = 97,23%), on observe une confusion marginale avec des paquets d'attaque, en particulier "Scan" et "Mirai". Cela peut s'expliquer par la proximité statistique de certains flux légitimes avec ceux générés lors d'activités malveillantes dissimulées, mettant en évidence l'importance de la représentation contextuelle introduite par la structuration en graphes.

La Figure 4.8, qui présente une projection t-SNE des représentations latentes des nœuds (embeddings) extraits après l'entraînement, montre une séparation nette entre les différentes classes. Les points appartenant aux catégories DoS, Scan et Mirai forment des clusters denses et bien séparés, ce qui valide la capacité du modèle à apprendre des représentations discriminantes. Quelques recouvrements entre les classes MITM et Normal apparaissent, illustrant les confusions détectées dans la matrice de confusion, et soulignant que l'amélioration future de ces zones limites pourrait reposer sur des représentations plus fines ou des couches de discrimination supplémentaires.

Ces résultats démontrent la pertinence du choix méthodologique : la construction locale des graphes via K-means et KNN permet de concentrer l'apprentissage sur des contextes homogènes, tout en évitant la pollution des relations inter-clusters. La combinaison de cette structuration avec l'architecture GCN assure une propagation de l'information contextuellement guidée, renforçant la capacité du modèle à distinguer les comportements normaux des comportements malveillants, même lorsque les signaux sont subtils ou partiellement masqués.

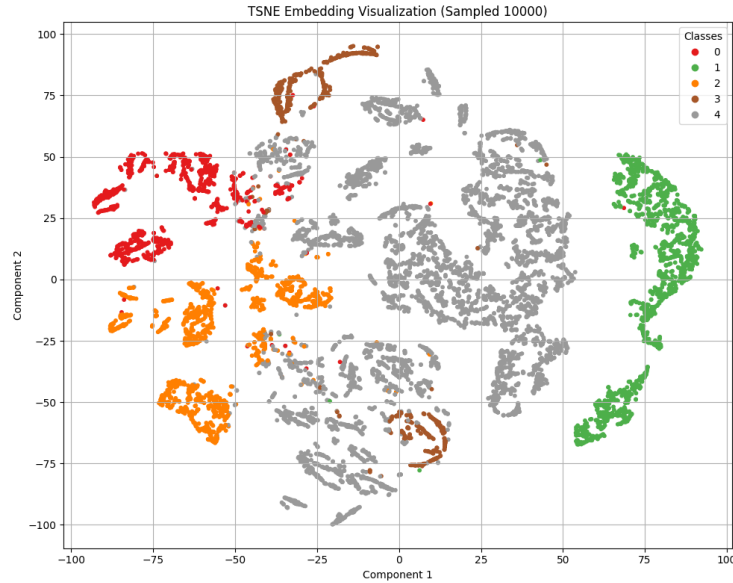


FIGURE 4.8 – *Projection bidimensionnelle des embeddings des nœuds, montrant la séparation entre les différentes catégories de trafic.*

6 Évaluation sur le dataset CICIoT2023

Pour évaluer davantage la robustesse et la capacité de généralisation du modèle proposé basé sur les GCN (Graph Convolutional Networks), nous avons élargi nos expériences en incluant le jeu de données CICIoT2023, un référentiel moderne constitué de trafic IoT réel. Ce jeu de données complet, publié en 2023, contient environ 46 millions d’enregistrements et 46 attributs, couvrant à la fois des activités bénignes et de nombreux vecteurs d’attaque, notamment DDoS, DoS, reconnaissance, web-based attacks, brute force, spoofing et Mirai.

Une difficulté majeure posée par CICIoT2023 est le déséquilibre important entre les classes, certaines catégories d’attaque comme les attaques web-based et brute force étant fortement sous représentées par rapport aux classes prédominantes telles que DDoS et DoS. Cette caractéristique fait de CICIoT2023 un banc d’essai exemplaire pour évaluer la capacité du modèle à maintenir de hautes performances de classification, tant sur les classes majoritaires que minoritaires.

En dépit de ce déséquilibre, le modèle a produit des résultats exceptionnels (tableau 4.5), avec une accuracy globale de 99,27%, une précision (precision) de 99,09%, un recall de 98,87%, un score F1 de 98,98%. Il a également affiché des mesures élevées avec un coefficient Kappa de Cohen et un coefficient de corrélation de Matthews atteignant chacun 98,84%. L’analyse par classe a validé l’efficacité du modèle dans la détection du trafic normal (score F1 : 98,47%), tout en obtenant des performances quasi parfaites sur les attaques DDoS (99,83%) et DoS (98,80%).

De même, des scores F1 élevés ont été enregistrés pour les attaques Mirai (98,87%) et Spoofing (98,30%), indiquant une résilience face aux menaces de type botnet et d’usurpation. Le modèle s’est montré particulièrement performant face aux attaques de reconnaissance furtives (99,48%) et a démontré une robustesse notable face aux attaques peu représentées telles que la force brute (81,86%) et les attaques web (97,00%).

Ces résultats confirment que le modèle GCN n’offre pas seulement d’excellentes performances sur des jeux de données équilibrés comme IoTID20, mais qu’il s’adapte également de manière

7. Comparaison des performances avec les approches existantes

efficace à des environnements réseau complexes, déséquilibrés et réalistes comme CICIoT2023.

TABLE 4.5 – *Performance du modèle GCN sur le dataset CICIoT2023*

Type	Catégorie	Accuracy	Précision	Rappel	F1-score	Kappa	MCC
Globale	Toutes classes	99.27%	99.09%	98.87%	98.98%	98.84%	98.84%
	Normal	99.00%	99.13%	97.82%	98.47%	–	–
Par catégorie	DDoS	98.13%	99.91%	99.76%	99.83%	–	–
	DoS	98.70%	98.74%	98.86%	98.80%	–	–
	Mirai	98.00%	98.81%	98.88%	98.87%	–	–
	Spoofing	98.30%	98.29%	98.32%	98.30%	–	–
	Reconnaissance	99.70%	99.36%	99.59%	99.48%	–	–
	Brute force	92.72%	73.28%	92.72%	81.86%	–	–
	Web-based	97.97%	97.67%	97.32%	97.00%	–	–

7 Comparaison des performances avec les approches existantes

L’analyse comparative des performances de notre modèle avec celles de plusieurs techniques de l’état de l’art (table 4.6) appliquées au même jeu de données IoTID20 révèle des écarts significatifs en faveur de notre approche, tant en termes de précision que de robustesse. Contrairement aux modèles classiques comme Random Forest ou Gradient Boosting, qui reposent sur des décisions univariées issues d’un ensemble d’attributs isolés, notre modèle s’appuie sur une structuration relationnelle des données sous forme de graphes. Ce changement de paradigme est fondamental : il permet de passer d’une vision individuelle des événements à une modélisation contextuelle et interconnectée, où les comportements malveillants peuvent être identifiés à travers leurs interactions plutôt que leurs seules caractéristiques intrinsèques.

TABLE 4.6 – *Comparaison des performances des différentes techniques de classification*

Technique	Accuracy	Recall	Precision	F1-Score
Gradient Boosting [216]	0,8841	0,8190	0,8287	0,8200
Random Forest [217]	0,8400	0,8400	0,8500	0,8400
CNN-LSTM [218]	0,8296	0,8354	0,8525	0,8414
CNN-GRU [218]	0,8676	0,8525	0,8728	0,8555
Approche proposée	0,9857	0,9767	0,9852	0,9809

Par ailleurs, les architectures CNN-LSTM ou CNN-GRU, bien que capables de modéliser des séquences temporelles, restent limitées à une vision linéaire et globale des données. Elles ne tiennent pas compte de la structure locale du trafic réseau ni des relations topologiques entre flux, deux dimensions particulièrement critiques dans les environnements IoT, où les comportements

malveillants peuvent se propager ou se dissimuler dans des agrégats de trafic similaires. Notre méthode, en revanche, introduit une modularisation locale par clustering (K-means), permettant de traiter chaque cluster comme une communauté homogène. À l'intérieur de ces communautés, les connexions sont établies uniquement entre points voisins (via FAISS), ce qui garantit que la propagation de l'information dans le GCN est guidée par la similarité réelle des comportements, et non par une topologie arbitraire.

Le gain notable en F1-score (+12 à +16 points), par rapport aux modèles de référence, souligne précisément cette capacité à maintenir un équilibre optimal entre rappel (sensibilité) et précision, deux critères essentiels pour la détection d'intrusions. En effet, un haut rappel signifie que peu d'attaques sont manquées, tandis qu'une haute précision indique que peu de faux positifs sont générés, un compromis difficile à atteindre dans les contextes IoT, où certaines attaques sont furtives ou sous-représentées.

De plus, les indicateurs comme le Cohen Kappa et le MCC, qui mesurent la concordance entre prédictions et classes réelles tout en tenant compte des déséquilibres de distribution, confirment que notre modèle est à la fois stable, discriminant et généralisable. Ces résultats traduisent une capacité d'adaptation à des classes minoritaires, souvent mal détectées par des modèles globaux.

Enfin, la visualisation t-SNE des représentations apprises par le GCN met en évidence une clarté structurelle dans l'espace latent, avec des clusters compacts et bien séparés. Cela témoigne de la capacité du modèle à extraire des embeddings informatifs, qui facilitent la classification et réduisent les ambiguïtés. Cette propriété est cruciale pour des extensions futures en détection non supervisée, classification fine ou apprentissage semi-supervisé.

En somme, les résultats obtenus ne relèvent pas d'une simple amélioration quantitative, mais découlent d'une conception méthodologique plus adaptée à la nature relationnelle et contextuelle des données IoT. Notre modèle n'est pas seulement performant : il est conceptuellement mieux aligné avec la structure réelle du problème, ce qui en fait une solution prometteuse, extensible et fondamentalement plus robuste face à la complexité croissante des cybermenaces dans les environnements connectés.

8 Conclusion

Dans cette thèse, nous avons proposé un modèle novateur pour la détection d'intrusions dans les environnements IoT, basé sur une architecture combinant clustering local (K-means) et réseaux de neurones convolutifs sur graphes (GCN). En nous appuyant sur le jeu de données IoTID20, nous avons démontré que cette approche permet non seulement d'améliorer les performances de classification des attaques réseau, mais aussi de mieux modéliser la complexité relationnelle inhérente aux systèmes distribués et hétérogènes.

La principale originalité de notre approche réside dans la structuration locale du graphe : en regroupant les données en clusters homogènes et en créant des connexions uniquement au sein de ces clusters via KNN, nous avons construit des graphes adaptés à une propagation de l'information cohérente et contextualisée dans le GCN. Cela nous a permis d'éviter les effets négatifs des connexions arbitraires entre entités dissemblables, souvent sources de bruit et de surapprentissage dans les modèles graphiques globaux.

Les résultats expérimentaux ont confirmé l'efficacité de notre méthode : le modèle a surpassé plusieurs techniques classiques et profondes de l'état de l'art, avec une accuracy de 98,57% et

un F1-score de 98,09%, tout en conservant une forte robustesse face à des classes déséquilibrées. L'analyse des représentations latentes et de la matrice de confusion a montré une meilleure séparation des classes et une capacité renforcée à distinguer des attaques similaires mais contextuellement différentes.

Au-delà des performances numériques, ce travail met en évidence un changement de paradigme dans la détection d'intrusions IoT : plutôt que de traiter chaque flux comme une entité isolée, notre approche propose une modélisation structurelle, contextuelle et communautaire du trafic réseau. Ce cadre théorique ouvre plusieurs pistes prometteuses pour les recherches futures.

9 Perspectives de recherche

1. Graphes dynamiques et temporels : Les graphes utilisés dans notre approche sont statiques, or les réseaux IoT évoluent constamment. L'introduction de GNN temporels ou de Dynamic GCN pourrait permettre de capturer les évolutions du comportement réseau au fil du temps, améliorant la détection d'attaques évolutives ou furtives.
2. Apprentissage en ligne et détection en temps réel : Adapter notre architecture à un cadre d'apprentissage continu ou incrémental permettrait d'intégrer de nouveaux comportements au fur et à mesure, ce qui est crucial pour une détection en temps réel sur des systèmes déployés, avec des contraintes de latence et de mémoire.
3. Amélioration de la robustesse aux attaques adverses : Les modèles graphiques peuvent être sensibles aux manipulations subtiles des connexions ou des attributs. L'intégration de techniques de défense contre les attaques adversaires sur graphes renforcerait la sécurité du système lui-même.
4. Généralisation à d'autres contextes IoT et edgecomputing : Tester le modèle sur d'autres jeux de données IoT, incluant des topologies variées, des dispositifs mobiles ou des réseaux industriels, permettrait d'évaluer sa généricité et de l'adapter à des environnements contraints, notamment en edgecomputing ou en fogcomputing.
5. Fusion de vues multiples (multi-view learning) : Intégrer plusieurs sources d'information (attributs statiques, comportements dynamiques, contexte topologique) dans un cadre multi-modal ou multi-graphe ouvrirait la voie à une détection plus fine, capable de capturer des corrélations complexes entre dimensions.

En conclusion, les résultats obtenus dans cette thèse illustrent non seulement les avantages pratiques d'une architecture GCN contextualisée, mais aussi son potentiel scientifique en tant que fondement méthodologique pour la cybersécurité des systèmes IoT. Cette approche pourrait, à terme, s'inscrire dans une stratégie globale de détection intelligente, évolutive et explicable, capable d'anticiper les cybermenaces émergentes dans des écosystèmes numériques de plus en plus interconnectés.

Conclusion générale

Conclusion

La sécurité des systèmes d'information demeure un enjeu stratégique majeur, d'autant plus crucial que la fréquence, la complexité et la diversité des attaques informatiques ne cessent de croître. Les intrusions constituent une menace sérieuse pour les organisations, susceptibles de causer des pertes économiques considérables, des atteintes à la réputation, ainsi que des fuites de données sensibles. Face à ces défis, la détection d'intrusions s'impose depuis plusieurs années comme un axe prioritaire de la recherche en cybersécurité.

Malgré les progrès significatifs réalisés, de nombreuses études émanant d'organismes de référence tels que le MITRE ou le NIST mettent en évidence les limites persistantes des systèmes classiques, en termes de précision, de réactivité et d'adaptabilité. Dans ce contexte, l'essor de l'intelligence artificielle, et en particulier de l'apprentissage automatique, constitue une opportunité majeure pour repenser en profondeur les mécanismes de détection.

Les travaux présentés dans cette thèse s'inscrivent dans cette dynamique et s'attachent à explorer, concevoir et expérimenter de nouvelles approches de détection d'intrusions fondées sur des techniques avancées de machine Learning et de Deep Learning. Plusieurs contributions majeures ont été apportées :

1. Modélisation du contexte d'attaque à partir de données historiques, en exploitant les capacités des réseaux LSTM pour capturer les dépendances temporelles. Bien que performante, cette approche présente des contraintes en matière de complexité computationnelle et de conservation de l'information dans les longues séquences.
2. Représentation vectorielle via l'approche PV-DM, combinée à une sélection optimisée des caractéristiques par les valeurs SHAP, et une classification à l'aide de XGBoost. Ce modèle se distingue par sa robustesse, sa simplicité d'utilisation et des performances comparables, voire supérieures, aux modèles récurrents classiques.
3. Détection des classes rares sans recours au suréchantillonnage, grâce à une stratégie de segmentation basée sur la fréquence des attaques, l'analyse des caractéristiques discriminantes communes, et une classification performante avec XGBoost. Cette méthode respecte l'intégrité des données originales tout en assurant une meilleure généralisation.
4. Classification par réseaux de neurones de graphes (GCN), où les données sont représentées sous forme de graphes construits à partir de clusters K-Means et de connexions KNN. Cette approche permet de capturer les relations topologiques complexes entre événements réseau, notamment dans les environnements IoT, et montre une capacité supérieure à détecter les comportements subtils et anormaux.

Ces contributions permettent de franchir plusieurs limites identifiées dans l'état de l'art, en particulier celles liées à la gestion des données déséquilibrées, à l'interprétation des attaques et à

l'adaptabilité aux nouvelles menaces. Les résultats expérimentaux obtenus sur des jeux de données de référence confirment la pertinence des approches proposées.

Néanmoins, plusieurs perspectives de recherche restent ouvertes et méritent d'être explorées :

1. **Généralisation des modèles :** L'évaluation a été menée sur des jeux de données publics ou simulés. Il serait essentiel de tester ces modèles sur des données réelles pour valider leur robustesse, fiabilité et efficacité dans des contextes opérationnels, tout en affinant les indicateurs de performance.
2. **Exploration de nouvelles architectures de DeepLearning :** Des modèles tels que GRU, CNN, ou leurs variantes hybrides, pourraient offrir des performances accrues pour le traitement de données temporelles. Le réglage fin des hyperparamètres restera un levier important pour optimiser les résultats.
3. **Interprétabilité des modèles :** Comprendre les décisions prises par les modèles de DeepLearning est fondamental en cybersécurité. Cette compréhension améliore non seulement la confiance dans les systèmes, mais peut également permettre de dériver des règles exploitables pour enrichir les outils existants.
4. **Étude de la résilience aux attaques adversaires :** Bien que cette thèse n'ait pas explicitement ciblé les attaques adversaires, ces dernières représentent une menace croissante pour les modèles de détection fondés sur l'apprentissage automatique. Une piste de recherche future consisterait à étudier la robustesse des approches proposées face à des exemples manipulés dans le but de contourner les systèmes IDS, et à développer des mécanismes de défense adaptés.

En conclusion, cette thèse propose un cadre méthodologique et expérimental solide pour la conception de systèmes de détection d'intrusion plus intelligents, robustes et adaptatifs. Elle constitue une avancée dans l'intégration des technologies d'apprentissage automatique au service de la cybersécurité, tout en ouvrant la voie à des recherches futures axées sur la généralisation des modèles, l'exploration de nouvelles architectures neuronales, et l'amélioration de l'interprétabilité des systèmes de détection.

Bibliographie

- [1] Stallings, W. (2016), *Network Security Essentials : Applications and Standards, 6th Edition*, Pearson Education, ISBN : 978-0134527338.
- [2] Anderson, R. (2020), *Security Engineering : A Guide to Building Dependable Distributed Systems*, 3rd Edition. WileyPublishing, ISBN : 978-1119642787.
- [3] Morgan, S. (2024), *Cybercrime To Cost The World 10.5 Trillion Annually By 2025. Cybersecurity Ventures Annual Report*, <https://cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016/> <https://cybersecurityventures.com/cybercrime-damages-6-trillion-by-2021/>.
- [4] dataconomy, <https://fr.dataconomy.com/2024/02/22/couts-mondiaux-des-cyberattaques-en-2024-et-dans-le-futur/>.
- [5] statista, <https://fr.statista.com/statistiques/573396/cybercriminalite-perte-moyenne-des-entreprises-dans-une-selection-de-pays/>.
- [6] cybersecurityventures, <https://cybersecurityventures.com/cybersecurity-spending-2021-2025/>.
- [7] lematin, <https://lematin.ma/economie/40-des-ordinateurs-ics-au-maroc-victimes-de-cyberattaques-depuis-janvier/246461>.
- [8] AUSIM. (2024), *Baromètre de la cybersécurité au Maroc 2024 - Livre blanc de l'Ausimètre. Association des Utilisateurs des Systèmes d'Information au Maroc*, <https://ausimaroc.com/premier-barometre-de-cybersecurite-au-maroc-en-partenariat-avec-le-cabinet-pwc-au-maroc/>.
- [9] expressvpn, <https://www.expressvpn.com/fr/blog/the-true-cost-of-cyber-attacks-in-2024-and-beyond/>.
- [10] Yu, C.-F. and Gligor, V. D. 1988, *A formal specification and verification method for the prevention of denial of service*, IEEE Symposium on Security and Privacy, 187202.
- [11] github, <https://github.blog/news-insights/company-news/ddos-incident-report/>.
- [12] Peter Denning, *The science of computing : Computer viruses.*, American Scientist, 76(3) :236238, May 1988.
- [13] Peter Denning, *Computers under Attack : Intruders, Worms and Viruses*. AddisonWesley, Reading, Mass., 1990.
- [14] Sharp, R. (2017), *An Introduction to Malware*.
- [15] Cacciapaglia, K. (2018), *Analyse sur les différentes cyberattaques informatiques*, (Doctoral dissertation, Haute école de gestion de Genève).
- [16] Cekerevac Z, *IoT and MITM attacks*, Security and economic risks MEST Journal Vol. 5 No. 2 pp. 15-25.
- [17] Paliwal, S., & Gupta, R. (2012), *Denial-of-service, probing & remote to user (R2L) attack detection using genetic algorithm*, International Journal of Computer Applications, 60(19), 57-62.
- [18] M. E.-S. GADELRAH, *Evaluation des systèmes de détection d'intrusion*, PhD thesis, université de Toulouse III, 2008.

- [19] H. A. Jabou Chaouki, Schillings Michaël, *Ter détection d'anomalies sur le réseau*, Rapport de projet, 2009.
- [20] firewall, , [https ://www.syloe.com/glossaire/firewall-pare-feu/](https://www.syloe.com/glossaire/firewall-pare-feu/).
- [21] Ferguson, Paul and Geoff Huston, *What Is a VPN ?* , Part I. (1998).
- [22] R. G. Yende, *Support de cours de sécurité informatique et crypto* , 2018.
- [23] DMZ, [https ://www.techtarget.com/searchsecurity/definition/DMZ](https://www.techtarget.com/searchsecurity/definition/DMZ).
- [24] Illy, Poulmanogo. (2018), *Les systèmes de détection d'intrusion (IDS)*, 10.13140/RG.2.2.10055.04001.
- [25] J. P. Anderson, *Computer security threat monitoring and surveillance*, Technical Report, James P. Anderson Company, Fort Washington, Pennsylvania, USA, April 1980.
- [26] Koral Ilgun, Richard A. Kemmerer and Phillip A. Porras, *State Transition Analysis : A Rule-Based Intrusion Detection Approach*, Software Engineering, Vol. 21, pp. 181-199, 1995.
- [27] P. A. Porras and P. G. Neumann, *EMERALD : Event Monitoring Enabling Responses to Anomalous Live Disturbances*, Proceeding of 20th NIST National Information Systems Security Conference, Baltimore, Maryland, USA, pp. 353-365, 1997.
- [28] Vern Paxson, *Bro : a System for Detecting Network Intruders in Real-time*, Computer Networks (Amsterdam, Netherlands : 1999), Vol. 31, pp. 2435-2463, 1999.
- [29] Martin Roesch, *Snort-Lightweight Intrusion Detection for Networks*, Proceedings of the 13th USENIX Systems Administration Conference (LISA'99), Seattle, Washington, USA, pp. 229-238, 1999.
- [30] Earl Carter, *Cisco Secure Intrusion Detection System*, Cisco Press, 2001.
- [31] IBM Internet Security Systems (ISS) products (2008), , [http ://www-935.ibm.com/services/us/index.wss/offerfamily/iss/a1029097](http://www-935.ibm.com/services/us/index.wss/offerfamily/iss/a1029097).
- [32] V. Jyothsna, R. Prasad and K. M. Prasad, *A review of anomaly based intrusion detection systems*, Int. J. Comput. Appl., vol. 28, no. 7, pp. 26-35, 2011.
- [33] V. Bukac, P. Tucek and M. Deutsc, *Advances and challenges in standalone host-based intrusion detection systems*, Proc. 9th Int. Conf. Trust Privacy Secur. Digit. Bus., pp. 105-117, 2012.
- [34] A. Thakkar and R. Lohiya, *A survey on intrusion detection system : Feature selection model performance measures application perspective challenges and future research directions*, Artif. Intell. Rev., vol. 55, no. 1, pp. 453-563, Jan. 2022.
- [35] Y. Kumar and B. Subba, *Stacking ensemble-based HIDS framework for detecting anomalous system processes in windows based operating systems using multiple word embedding*, Comput. Secur., vol. 125, Feb. 2023.
- [36] K. He, D. D. Kim and M. R. Asghar, *Adversarial machine learning for network intrusion detection systems : A comprehensive survey*, IEEE Commun. Surveys Tuts., vol. 25, no. 1, pp. 538-566, 1st Quart. 2023.
- [37] DE Denning, *An intrusion-detection model*, IEEE Transactions on software engineering (1987), 222-232.
- [38] Nathalie Dagorn, *Détection et prévention d'intrusion : présentation et limites*, [Rapport de recherche] 2006. inria-00084202.
- [39] H. Satilmiş, S. Akleyek and Z. Y. Tok, *A Systematic Literature Review on Host-Based Intrusion Detection Systems*, in IEEE Access, vol. 12, pp. 27237-27266, 2024, doi : 10.1109/ACCESS.2024.3367004.
- [40] Y. Kumar and B. Subba, *Stacking ensemble-based HIDS framework for detecting anomalous system processes in windows based operating systems using multiple word embedding*, Comput. Secur., vol. 125, Feb. 2023.

-
- [41] H. Hindy, D. Brosset, E. Bayne, A. K. Seeam, C. Tachtatzis, R. Atkinson, et al, *A taxonomy of network threats and the effect of current datasets on intrusion detection systems*, IEEE Access, vol. 8, pp. 104650-104675, 2020.
- [42] Illy, Poulmanogo. (2018), *Les systèmes de détection d'intrusion (IDS)*, 10.13140/RG.2.2.10055.04001.
- [43] T.H. Ptacek and T.N. Newsham, *Insertion, avasion, and denial of service : Eluding network intrusion detection*, tech/ rep., Secure Networks inc Calgary Alberta, 1998/.
- [44] Garcia-Teodoro, Pedro, J. Diaz-Verdejo, Gabriel M.; Enrique V, *Anomaly-based network intrusion detection : Techniques, systems and challenges*, computers & security, vol.28, no. 1, pp. 18, 28, 2009.
- [45] Aljanabi, Mohammad & Ismail, Mohd Arfian & Ali, Ahmed. (2021), *Intrusion Detection Systems, Issues, Challenges, and Needs. International Journal of Computational Intelligence Systems*, 14. 10.2991/ijcis.d.210105.001.
- [46] El Asry, Chadia & Ouahidi, Bouabid & Douzi, Samira. (2024), *A Deep Learning Model for Intrusion Detection with Imbalanced Dataset*, 261-271. 10.1007/978-3-031-47672-3_26.
- [47] El Asry, Chadia & Benchaji, Ibtissam & Ouahidi, Bouabid & Douzi, Samira. (2024), *A robust intrusion detection system based on a shallow learning model and feature extraction techniques*, PLOS ONE. 19. e0295801. 10.1371/journal.pone.0295801.
- [48] El Asry, Chadia & Benchaji, Ibtissam & Ouahidi, Bouabid & Douzi, Samira. (2024), *Effective Approaches for Intrusion Detection Systems in the Face of Low-Frequency Attacks*, Journal of Advances in Information Technology. 15. 1070-1078. 10.12720/jait.15.9.1070-1078.
- [49] Asry C EL, Benchaji I, Douzi S, Ouahidi B EL (2025) *Enhancing cybersecurity : A high-performance intrusion detection approach through boosting minority class recognition*. PLOS ONE 20(3) : e0317346. <https://doi.org/10.1371/journal.pone.0317346>.
- [50] J. Olivain, *Plate-forme de détection d'intrusions : Analyse et corrélation temporelle d'événements en temps réel. Rapport de stage, Laboratoire Spécification et Vérification*, ENS Cachan, France, novembre, décembre 2003.
- [51] Amaizu, G. C., Nwakanma, C. I., Lee, J. M., & Kim, D. S. (2020, October), *Investigating network intrusion detection datasets using machine learning*, International Conference on Information and Communication Technology Convergence (ICTC), pp. 1325-1328.
- [52] Bhati, B. S., Rai, C. S. (2020), *Analysis of support vector machine-based intrusion detection techniques*, Arabian Journal for Science and Engineering, 45(4), pp 2371-2383.
- [53] Ramkumar Devendiran and Anil V Turukmane *Dugat-LSTM : Deep learning based network intrusion detection system using chaotic optimization strategy*, <https://doi.org/10.1016/j.eswa.2023.123027>, <https://www.sciencedirect.com/science/article/pii/S0957417423035297>.
- [54] Sukhvinder Singh Bamber, Aditya Vardhan Reddy Katkuri, Shubham Sharma, Mohit Angurala, *A hybrid CNN-LSTM approach for intelligent cyber intrusion detection system*, Computers & Security, Volume 148, 2025, 104146, ISSN 0167-4048, <https://doi.org/10.1016/j.cose.2024.104146>.
- [55] Torre, Damiano and Chennamaneni, Anitha and Jo, JaeYun and Vyas, Gitika and Sabrsula, Brandon, *Toward Enhancing Privacy Preservation of a Federated Learning CNN Intrusion Detection System in IoT : Method and Empirical Study*, 2025, Association for Computing Machinery, <https://doi.org/10.1145/3695998>.
- [56] <https://ai.plainenglish.io/multiclass-vs-multilabel-classification-in-a-minute-261b6e320061>.
-

- [57] Steven Walczak, Narciso Cerpa, *Artificial Neural Networks*, Encyclopedia of Physical Science and Technology (Third Edition), Academic Press, 2003, Pages 631-645, <https://doi.org/10.1016/B0-12-227410-5/00837-1>.
- [58] David E *Learning representations by back-propagating errors*. *nature Artificial Neural Networks*, vol. 323, no. 6088, pages 533–536, 1986.
- [59] Jeffrey L Elman *Finding structure in time*, *Cognitive science*, vol. 14, no. 2, pages 179–211, 1990.
- [60] Yoshua Bengio, Patrice Simard et Paolo Frasconi *Learning long-term dependencies with gradient descent is difficult*, *IEEE transactions on neural networks*, vol. 5, no. 2, pages 157–166, 1994.
- [61] Justin Simon Bayer *Learning sequence representations*, Thèse de Doctorat, Technische Universität München, 2015.
- [62] Alex Graves et Navdeep Jaitly *Towards end-to-end speech recognition with recurrent neural networks*, In *International conference on machine learning*, pages 1764–1772. PMLR, 2014.
- [63] Ilya Sutskever, Oriol Vinyals et Quoc V Le *Sequence to sequence learning with neural networks*, *Advances in neural information processing systems*, vol. 27, 2014.
- [64] Zhao, Z., Chen, W., Wu, X., Chen, P. C. and Liu, J *Lstm Network : A Deep Learning Approach for Short-Term Traffic Forecast*, *IET Intelligent Transport Systems*, 2017, 11, 68-75. <https://doi.org/10.1049/iet-its.2016.0208>.
- [65] Qinghe Zheng, Xinyu Tian, Zhiguo Yu, Nan Jiang, Abdussalam Elhanashi, Sergio Saponara, Rui Yu *Application of wavelet-packet transform driven deep learning method in PM2.5 concentration prediction : A case study of Qingdao, China*, *Sustainable Cities and Society*, Volume 92, 2023, 104486, ISSN 2210-6707, <https://doi.org/10.1016/j.scs.2023.104486>.
- [66] Sepp Hochreiter et Jürgen Schmidhuber *Long short-term memory*, *Neural computation*, vol. 9, no. 8, pages 1735–1780, 1997.
- [67] Elfaik, Hanane & Nfaoui, El Habib. (2020) *Deep Bidirectional LSTM Network Learning-Based Sentiment Analysis for Arabic Text*, *Journal of Intelligent Systems*. 30. 395-412. 10.1515/jisys-2020-0021.
- [68] Hosseini, Fatemeh & Soleimani Gharehchopogh, Farhad & Masdari, Mohammad. (2022). *MOAEOSCA : an enhanced multi-objective hybrid artificial ecosystem-based optimization with sine cosine algorithm for feature selection in botnet detection in IoT*, *Multimedia Tools and Applications*. 82. 10.1007/s11042-022-13836-6.
- [69] Scott M Lundberg and Su-In Lee. *A Unified Approach to Interpreting Model Predictions*, In *Advances in Neural Information Processing Systems* 30. Curran Associates, Inc., 4768–4777. ,2017.
- [70] Magesh, P. R., Myloth, R. D., & Tom, R. J. (2020). *An explainable machine learning model for early detection of Parkinson’s disease using LIME on DaTSCAN imagery*, . *Computers in Biology and Medicine*, 126, 104041.
- [71] Molnar, C. (2020). *Interpretable machine learning*. , Lulu. com.
- [72] Breiman, L. *Random Forests* , *Mach. Learn.* 2001, 45, 5–32.
- [73] Hamed Ahmadpour , Ommolbanin Bazrafshan, Elham Rafiei-Sardooi , Hossein Zamani and Thomas Panagopoulos *Gully Erosion Susceptibility Assessment in the Kondoran Watershed Using Machine Learning Algorithms and the Boruta Feature Selection*.
- [74] Sánchez-Marono, N.; Alonso-Betanzos, A.; Calvo-Estévez, R.M *A Wrapper Method for Feature Selection in Multiple Classes Datasets* Springer : Berlin/Heidelberg, Germany, 2009 ; pp. 456–463.

-
- [75] Mercier, L.-M. (2017) *Modifications de l'algorithme SELECT() appliquées à une problématique du cancer du sein* Mémoire de maîtrise, École Polytechnique de Montréal]. PolyPublie. <https://publications.polymtl.ca/2862/>.
 - [76] Md Morshed Ali, Md Saiful Islam, Mohammed Nasir Uddin, Md. Ashraf Uddin *A conceptual IoT framework based on Anova-F feature selection for chronic kidney disease detection using deep learning approach, Intelligence-Based Medicine* Volume 10,2024,100170,ISSN 2666-5212, <https://doi.org/10.1016/j.ibmed.2024.100170>.
 - [77] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, *A Detailed Analysis of the KDD CUP 99 Data Set* Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), ,2009.
 - [78] Dhanabal L, Shantharajah S PA *study on NSL-KDD dataset for intrusion detection system based on classification algorithms*[J].International Journal of Advanced Research in Computer andcommunication Engineering, vol.4, no.6, pp.446-452,2015.
 - [79] S. Revathi and A. Malathi, *A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection*, Int. J. Eng. Res. Technol., vol. 2, no. 12, pp. 1848-1853, 2013.
 - [80] StandardScaler algorithm, <https://www.datacorner.fr/feature-scaling/>.
 - [81] Repeated K-Fold cross-validation with Python, URL <https://machinelearningmastery.com/repeated-k-fold-cross-validation-with-python>.
 - [82] K-Fold cross-validation <https://neptune.ai/blog/cross-validation-in-machine-learning-how-to-do-it-right>.
 - [83] Ibtiissam benchajy. *Vers un Deep Learning Système de Détection des Fraudes sur cartes bancaires*. <https://toubkal.imist.ma/xmlui/77/handle/123456789/34208>.
 - [84] Mimura, Mamoru & Tanaka, Hidema. *Reading Network Packets as a Natural Language for Intrusion Detection*. 10.1007/978-3-319-78556-1 19, 2018.
 - [85] Rafael San Miguel Carrasco, Miguel-Angel Sicilia. *Unsupervised intrusion detection through skip-gram models of network behavior*. Computers and Security Volume 78,2018, Pages 187-197, ISSN 0167-4048, <https://doi.org/10.1016/j.cose.2018.07.003>.
 - [86] Kottapalle, Prasanna. *A CNN-LSTM Model for Intrusion Detection System from High Dimensional Data*. Journal of Information and Computational Science, 10. 1362-1370., 2020.
 - [87] Sun, Pengfei & Liu, Pengju & Li, Qi & Liu, Chenxi & Lu, Xiangling & Hao, Ruochen & Chen, Jinpeng. *DL-IDS : Extracting features using CNN-LSTM hybrid network for intrusion detection system networks*. Security and Communication Networks, 2020. 1-11. 10.1155/2020/8890306.
 - [88] Jin Kim, Nara Shin, Seung Yeon Jo and Sang Hyun Kim. *Method of Intrusion Detection using Deep Neural Network*. In IEEE, 2017.
 - [89] Li Y., Ma R. and Jiao R., *A Hybrid Malicious Code Detection Method Based on Deep Learning*. International Journal of Security and Its Applications (IJSIA), vol. 9, no. 5, pp. 205–216, 2015.
 - [90] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, *Longshort term memory recurrent neural network classifier for intrusion detection*. In Proceedings of the 2016 International Conference on Platform Technology and Service (PlatCon), IEEE, Jeju, South Korea , February 2016.
 - [91] Imrana, Yakubu & Xiang, Yanping & Ali, Liaqat & Abdul-Rauf, Zaharawu. *A bidirectional LSTM deep learning approach for intrusion detection*. Expert Systems with Applications. 185. 115524. 10.1016/j.eswa.2021.115524, 2021.
 - [92] Shen Y, Zheng K, Wu C, Zhang M, Niu X, Yang Y. *An ensemble method based on selection using bat algorithm for intrusion detection*. Comput J, 2018 ;61(4) :526–38.
-

- [93] Dong Y, Wang R, He J. *Real-Time Network Intrusion Detection System Based on Deep Learning*. IEEE 10th International Conference on Software Engineering and Service Science (ICSESS), 2019. <https://doi.org/10.1109/icse47205.2019.9040718>.
- [94] F. Louati and F. B. Ktata, *A deep learning-based multi-agent system for intrusion detection*, Social Netw. Appl. Sci., vol. 2, no. 4, pp. 1–13, Apr. 2020.
- [95] A. M. Aleesa, Mohammed Younis, Ahmed A. Mohammed, Nan m. Sahar, *deep-intrusion detection system with enhanced unsw-nb15 dataset based on deep learning techniques* Journal of Engineering Science and Technology, 2021.
- [96] Naseri, Touraj&SoleimanianGharehchopogh, Farhad. (2022). *A Feature Selection Based on the Farmland Fertility Algorithm for Improved Intrusion Detection Systems*. Journal of Network and Systems Management. 30. 40. 10.1007/s10922-022-09653-9.
- [97] SoleimanianGharehchopogh, Farhad & Abdollahzadeh, Benyamin &Barshandeh, Saeid & Arasteh, Bahman. (2023). *A Multi-Objective Mutation-based Dynamic Harris Hawks Optimization for Botnet Detection in IoT*. Internet of Things. 100952. 10.1016/j.iot.2023.100952.
- [98] J. Li, et al., *Ai-based two-stage intrusion detection for software defined iot networks*, IEEE Internet Things J. 6 (2) (2018) 2093–2102.
- [99] C.D. McDermott, F. Majdani, A.V. Petrovski, *Botnet detection in the Internet of Things using deep learning approaches*, in : Proceedings of the International Joint Conference on Neural Networks (IJCNN), IEEE, 2018.
- [100] I. Sharafaldin, A.H. Lashkari, S. Hakak, A.A. Ghorbani, *Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy*, in : In 2019 International Carnahan Conference on Security Technology (ICCST), IEEE, 2019, pp. 1–8.
- [101] Y. Zhou, G. Cheng, S. Jiang, M. Dai, *Building an efficient intrusion detection system based on feature selection and ensemble classifier*, Comput. Netw. 174 (2020), 107247.
- [102] Alberto Paccanaro, Geoffrey Hinton, *Learning Distributed Representations of Concepts using Linear Relational Embedding*, IEEE Transactions on Knowledge and Data Engineering, Pages 232- 244 (April 2001).
- [103] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean, *Distributed Representations of Words and Phrases and their Compositionality* , arXiv :1310.4546v1 [cs.CL] Page 9.(16 Oct 2013).
- [104] Mikolov, Tomas & Chen, Kai & Corrado, G.s & Dean, Jeffrey, *Efficient Estimation of Word Representations in Vector Space*. Proceedings of Workshop at ICLR, 2013.
- [105] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, Christian Jauvin, *A Neural Probabilistic Language Model*, Journal of Machine Learning Research 3 1137–1155,(2003).
- [106] Crawford, Elisabeth Koprinska, Irena Patrick, Jon, *Phrases and Feature Selection in E-Mail Classification*, Proceedings of the Ninth Australasian Document Computing Symposium 5962,(December 2004).
- [107] Le, Quoc & Mikolov, Tomas. *Distributed Representations of Sentences and Documents*. In 31st International Conference on Machine Learning, ICML 2014.
- [108] Douzi, Samira *Deep Learning Systèmes De Detection des Intrusions Malveillantes*. https://toubkal.imist.ma/xmlui/77/bitstream/handle/123456789/15329/THESE_DOUZI.pdf?sequence=1.
- [109] Cover TM, Thomas JA. *Information theory and statistics. Elements of information theory*. 2nd edn. Wiley ,2005. <https://doi.org/10.1002/047174882X.ch11>.
- [110] NSL-KDDdataset, [online]Available :<http://nsl.cs.unb.ca/nsl-kdd/>.

-
- [111] Moustafa, Nour, and Jill Slay. *UNSW-NB15 : a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)*. Military Communications and Information Systems Conference (MilCIS), 2015. IEEE, 2015.
- [112] Jie Cai, Jiawei Luo, Shulin Wang, Sheng Yang, *Feature selection in machine learning : A new perspective*, Neurocomputing, Volume 300, 2018, Pages 70-79, ISSN 0925-2312, <https://doi.org/10.1016/j.neucom.2017.11.077>.
- [113] Viegas, E., Lemos, J. M., Ferreira, P., & Pereira, J. M. (2014). *A comparative study of machine learning techniques for intrusion detection*. Procedia Technology, 16, 1266-1275.
- [114] Chen, Y., Zhang, Z., Yu, G., & Zeng, W. (2019). *Intrusion detection model based on XG-Boost*. Journal of Physics : Conference Series, 1349(4), 042073.
- [115] M. Alazab, S. Venkatraman, P. Watters and M. Alazab, *Zero-day malware detection based on supervised learning algorithms of API call signatures*, Proc. 9th Australas. Data Mining Conf., vol. 121, pp. 171-182,,2011.
- [116] Nour Moustafa & Jill Slay, *The evaluation of Network Anomaly Detection Systems : Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 dataset*, 2016,1-14. 10.1080/19393555.2015.1125974.
- [117] Ramadan RA, Yadav K. *A novel hybrid intrusion detection system (IDS) for the detection of internet of things (IoT) network attacks*. Ann Emerg Technol Comput, 2020. <https://doi.org/10.33166/aetic.2020.05.004>.
- [118] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat and S. Venkatraman, *Deep Learning Approach for Intelligent Intrusion Detection System*, in IEEE Access, vol. 7, pp. 41525-41550, 2019, doi : 10.1109/ACCESS.2019.2895334.
- [119] Wu, K. ; Chen, Z. ; Li, W. *A Novel Intrusion Detection Model for a Massive Network Using Convolutional Neural Networks*, IEEE Access ,2018, 6, 50850–50859.
- [120] Adhi Tama Bayu, Comuzzi Marco, Rhee Kyung Hyune, *TSE-IDS : A Two-Stage Classifier Ensemble for Intelligent Anomaly-based Intrusion Detection System*, IEEE Access 7. 10.1109/ACCESS.,2019.2928048.
- [121] Huang, S., Lei, K., 2020. *IGAN-IDS : an imbalanced generative adversarial network towards intrusion detection system in ad-hoc networks*. Ad Hoc Netw. 105, 102177.
- [122] Laghrissi, Fatimaezzahra & Douzi, Samira & Khadija, Douzi & Hssina, Badr. (2021). *Intrusion detection systems using long short-term memory (LSTM)*. Journal of Big Data. 8. 10.1186/s40537-021-00448-4.
- [123] El Asry, Chadia & Benchaji, Ibtiham & Douzi, Samira & Ouahidi, Bouabid. (2025). *An intrusion detection system using paragraph vector-distributed memory approach*. 10.1201/9781003587552-12.
- [124] Douzi, Samira & Alshahwan, Feda & Lemoudden, Mouad & Ouahidi, Bouabid. (2020). *Hybrid Email Spam Detection Model Using Artificial Intelligence*. International Journal of Machine Learning and Computing. 10. 316-322. 10.18178/ijmlc.2020.10.2.937.
- [125] Andy Reed, Laurence Dooley, Soraya Kouadri Mostefaoui, *SA-IDS : A single attribute intrusion detection system for Slow DoS attacks in IoT networks*, Internet of Things, Volume 30, 2025, 101512, ISSN 2542-6605, <https://doi.org/10.1016/j.iot.2025.101512>.
- [126] L.Liu, P.Wang, J.Lin, & L. Liu, *Intrusion Detection of Imbalanced Network Traffic Based on Machine Learning and Deep Learning*. IEEE Access, 9, 7550-7563. <https://doi.org/10.1109/ACCESS.2020.3048198>, 2021.
- [127] P.Bedi, N., Gupta, & D. Jindal, *Siam-IDS : Handling class imbalance problem in Intrusion Detection Systems using Siamese Neural Network*. Procedia Computer Science, 171, 7pp 80-789. <https://doi.org/10.1016/j.procs.2020.04.085>, 2020.
-

- [128] A. Alabrah, *A Novel Study : GAN-Based Minority Class Balancing and Machine-Learning-Based Network Intruder Detection Using Chi-Square Feature Selection*. Applied Sciences. <https://doi.org/10.3390/app122211662>, 2022.
- [129] O. Elghalhoud, S. Naik, M. Zaman, & S. RicardoManzano, *Data Balancing and CNN based Network Intrusion Detection System*. *IEEE Wireless Communications and Networking Conference (WCNC)*, 1-6. <https://doi.org/10.1109/WCNC55385.2023.10118702>, 2023.
- [130] S. M. Kasongo, *An advanced intrusion detection system for IIoT based on GA and tree based algorithms*, *IEEE Access*, vol. 9, pp. 113199–113212, 2021.
- [131] X. Gao, C. Shan, C. Hu, Z. Niu, Z. Liu, *An Adaptive Ensemble Machine Learning Model for Intrusion Detection*. *IEEE Access*, 7, 82512–82521, 2019.
- [132] SM, Kasongo, YSun, *A deep learning method with wrapper based feature extraction for wireless intrusion detection system*. *Comput Secur* 92 :101752. <https://doi.org/10.1016/j.cose.2020.101752>, 2020.
- [133] E. Anthi; L. Williams; M. Słowińska; G. Theodorakopoulos; P. Burnap, *A supervised intrusion detection system for smart home IoT devices*. *Internet Things J*, 6, 9042–9053, 2019.
- [134] N.T. Pham, E. Foo, S. Suriadi, H. Jeffrey, H.F.M. Lahza, D. Abramson, *Improving performance of intrusion detection system using ensemble methods and feature selection*. In : *Proceedings of Australian Computer Science Week (ACSW)*, pp. 1–6.
- [135] K. S. Devikrishna, and B. B. Ramakrishna, *An Artificial Neural Network based Intrusion Detection System and Classification of Attacks*, *International Journal of Engineering Research and Applications (IJERA)*, Jul-Aug 2013, 3(4) : 1959-1964.
- [136] G. Kim, S. Lee, S. Kim, *A novel hybrid intrusion detection method integrating anomaly detection with misuse detection*. *Expert Syst. Appl.* 2014, 41, 1690–1700.
- [137] Z. K. Maseer, R. Yusof, S. A. Mostafa, N. Bahaman, O. Musa, and B. A. S. Al-rimy, *DeepIoT.IDS : Hybrid deep learning for enhancing IoT network intrusion detection*, *Computers, Materials & Continua*, vol. 69, no. 3, pp. 3945–3966, 2021.
- [138] A. Guezaz, S. Benkirane, M. Azrou, and S. Khurram, *A reliable network intrusion detection approach using decision tree with enhanced data quality*, *Security and Communication Networks*, vol. 2021, p. 1230593, 2021.
- [139] Laghrissi, Fatimaezzahra & Douzi, Samira & Khadija, Douzi & Hssina, Badr. (2021). *IDS-attention : an efficient algorithm for intrusion detection systems using attention mechanism*. *Journal of Big Data*. 8. 10.1186/s40537-021-00544-5.
- [140] Dong R.H., Li X.Y., Zhang Q.Y., Yuan H. *Network intrusion detection model based on multivariate correlation analysis-long short-time memory network* *IET Inf. Secur.*, 14 (2) (2019), pp. 166-174.
- [141] [19] S. Li, et al., *An Improved Information Security Risk Assessments Method for Cyber-Physical-Social Computing and Networking*, 6, *IEEE Access*, 2018, pp. 10311–10319.
- [142] M. Habib, et al., *Multi-objective particle swarm optimization for botnet detection in Internet of Things*. *Evolutionary Machine Learning Techniques*, Springer, 2020, pp. 203–229.
- [143] P. Verma, S. Tapaswi, W.W. Godfrey, *An adaptive threshold-based attribute selection to classify requests under DDoS attack in cloud-based systems*, *Arab. J. Sci. Eng.* 45 (4) (2020) 2813–2834.
- [144] J. Gu, S. Lu, *An effective intrusion detection approach using SVM with naïve Bayes feature embedding*, *Computers & Security* 103 (2021), 102158.
- [145] Derhab A, Guerroumi M, Gumaei A, Maglaras L, Ferrag MA, Mukherjee M, Khan FA. *Blockchain and random subspace learning-based IDS for SDN-enabled industrial IoT security*. *Sensors*. 2019 Jul 15;19(14) :3119.

-
- [146] P. G. Jeya, M. Ravichandran, and C. S. Ravichandran, *Efficient classifier for R2L and U2R attacks*, Int. J. Comput. Appl., vol. 45, no. 21, p. 29, 2012.
 - [147] Fauzi, Rafeef & Al-Shammari, Rafeef. (2018). *NEW APPROACH FOR CLASSIFICATION R2L AND U2R ATTACKS IN INTRUSION DETECTION SYSTEM*.
 - [148] Gustavo EAPA Batista, Andre CPLF Carvalho et Maria Carolina Monard. *Applying one-sided selection to unbalanced datasets*. In *Mexican International Conference on Artificial Intelligence*, pages 315–325. Springer, 2000.
 - [149] *The effect of class imbalance handling on datasets toward classification algorithm performance*. Matrik : jurnal manajemen, teknik informatika, dan rekayasa komputer, 22(2) :227–238, 2023. doi : 10.30812/matrik.v22i2.2515.
 - [150] Andrea Dal Pozzolo, Olivier Caelen, Reid A. Johnson, and Gianluca Bontempi. *Calibrating probability with undersampling for unbalanced classification*. In 2015 IEEE Symposium Series on Computational Intelligence, pages 159–166. IEEE, 2015.
 - [151] Chris Drummond and Robert C. Holte. C4.5, *class imbalance, and cost sensitivity : why undersampling beats over-sampling*. In Workshop on Learning from Imbalanced Datasets II, volume 11, pages 1–8. Citeseer, 2003.
 - [152] Mienye, Domor & Sun, Yanxia. (2022). *A Survey of Ensemble Learning : Concepts, Algorithms, Applications, and Prospects*. IEEE Access. PP. 1-1. 10.1109/ACCESS.2022.3207287.
 - [153] H. Kaur, H.S. Pannu, and A.K. Malhi, *A systematic review on imbalanced data challenges in machine learning : applications and solutions*, ACM Computing Surveys, vol. 52(4), pp. 1-36, 2019.
 - [154] H. He and E.A. Garcia, *Learning from imbalanced data*, IEEE Transactions on Knowledge and Data Engineering, vol. 21(9), pp. 1263-1284, 2009.
 - [155] Emmanuel Remy, Vanessa Verges, Emilie Dautremer, Bruna Martin-Cabanais. *Revue des principales approches de résolution du problème de déséquilibre des classes*. Congrès Lambda Mu 22 “ Les risques au cœur des transitions ” (e-congrès)- 22e Congrès de Maîtrise des Risques et de Sécurité de Fonctionnement, Institut pour la Maîtrise des Risques, Oct 2020, Le Havre (e-congrès), France. hal-03480661.
 - [156] <https://www.ibm.com/frfr/topics/upsampling> : :text=La%20technique%20de%20sur%2D%C3%A9chantillonnage,la%20classe%20de%20la%20minorit%C3%A9.
 - [157] H. He and E.A. Garcia, *Learning from imbalanced data*, IEEE Transactions on Knowledge and Data Engineering, vol. 21(9), pp. 1263-1284, 2009.
 - [158] Ivan Tomek. *A generalization of the k-nn rule*. IEEE Transactions on Systems, Man, and Cybernetics, (2) :121–126, 1976.
 - [159] Ivan Tomek. *A generalization of the k-NN rule*. IEEE Transactions on Systems, Man, and Cybernetics, no. 2, pages 121–126, 1976.
 - [160] Gustavo EAPA Batista, Ronaldo C. Prati, and Maria Carolina Monard. *A study of the behavior of several methods for balancing machine learning training data*. ACM SIGKDD Explorations Newsletter, 6(1) :20–29, 2004.
 - [161] Jerome H. Friedman. *Stochastic gradient boosting*. Computational Statistics & Data Analysis, 38 (4) :367–378, 2002.
 - [162] Leo Breiman. *Bagging predictors*. Machine Learning, 24(2) :123–140, 1996b.
 - [163] Badih Ghattas. *Agrégation d’arbres de classification*. Revue de Statistique Appliquée, 48(2) : 85–98, 2000.
 - [164] Giorgio Valentini and Francesco Masulli. *Ensembles of learning machines*. In Italian Workshop on Neural Nets, pages 3–20. Springer, 2002.
-

- [165] Robert E Schapire. *The strength of weak learnability*. Machine learning, vol. 5, no. 2, pages 197–227, 1990.
- [166] Yoav Freund, Robert Schapire et Naoki Abe. *A short introduction to boosting*. Journal-Japanese Society For Artificial Intelligence, vol. 14, no. 771780, page 1612, 1999.
- [167] Leo Breiman. Bias, variance, and arcing classifiers. *Rapport technique*, Tech. Rep. 460, Statistics Department, University of California, Berkeley ..., 1996.
- [168] Neha V Sharma, Narendra Singh Yadav, *An optimal intrusion detection system using recursive feature elimination and ensemble of classifiers*, Microprocessors and Microsystems, Volume 85, 2021, 104293, ISSN 0141-9331, <https://doi.org/10.1016/j.micpro.2021.104293>.
- [169] Fu, Yanfang & Du, Yishuai & Cao, Zijian & Li, Qiang & Xiang, Wei. (2022). *A Deep Learning Model for Network Intrusion Detection with Imbalanced Data*. Electronics. 11. 898. 10.3390/electronics11060898.
- [170] Imrana, Yakubu & Xiang, Yanping & Ali, Liaqat & Abdul-Rauf, Zaharawu & Hu, Yu-Chen & Kadry, Seifedine & Lim, Sangsoon. (2022). *2-BidLSTM : A Feature Driven Intrusion Detection System Based on 2 Statistical Model and Bidirectional LSTM*. Sensors. 22. 2018. 10.3390/s22052018.
- [171] Cloudstor. Unsw-nb15 csv files ; 2020, [cited 2021 Jun 30.] Available from : <https://cloudstor.aarnet.edu.au/plus/index.php/s/2DhnLGDdEECo4ys?path=%2FUNSW-NB15%20-%20CSV%20Files>.
- [172] Choudhary S, Kesswani N. *Analysis of KDD-Cup’99, NSL-KDD and UNSW-NB15 Datasets using Deep Learning in IoT*. Procedia Computer Science. 2020;167 :1561–73.
- [173] Lundberg SM, Erion GG, Lee SI. *Consistent individualized feature attribution for tree ensembles*. arXiv preprint. 2018.
- [174] Sharafaldin I, Lashkari AH, Hakak S, Ghorbani AA. *Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy*. 2019 International Carnahan Conference on Security Technology (ICCST). 2019. p. 1–8.
- [175] Neto ECP, Dadkhah S, Ferreira R, Zohourian A, Lu R, Ghorbani AA. *CICIoT2023 : A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment*. Sensors (Basel). 2023;23(13) :5941. pmid :37447792.
- [176] Moustafa N, Slay J. *A hybrid feature selection for network intrusion detection systems : Central points*. Proceedings of the 16th Australian Information Warfare Conference. 2015.
- [177] I. Sharafaldin, A.H. Lashkari, S. Hakak, A.A. Ghorbani, *Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy*, In 2019 International Carnahan Conference on Security Technology (ICCST), IEEE, 2019, pp. 1–8.
- [178] Neto, E.C.P.; Dadkhah, S.; Ferreira, R.; Zohourian, A.; Lu, R.; Ghorbani, A.A. *CICIoT2023 : A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment*. Sensors 2023, 23, 5941. <https://doi.org/10.3390/s23135941>.
- [179] HA. Ahmed, A. Hameed, NZ. Bawany. *Network intrusion detection using oversampling technique and machine learning algorithms*. PeerJ Comput Sci. 2022 Jan 7;8 :e820. doi : 10.7717/peerj-cs.820.
- [180] S.M.Kasongo, Y. Sun, *Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset*. J Big Data 7, 105. <https://doi.org/10.1186/s40537-020-00379-6>, 2020.
- [181] AD. Eunice, Q. Gao, M-Y. Zhu, Z. Chen, L. Na, *Network anomaly detection technology based on deep learning*. In 2021 IEEE 3rd International Conference on Frontiers Technology of Information and Computer (ICFTIC). IEEE, pp. 6–9, 2021.

- [182] El Asry, Chadia & Benchaji, Ibtissam & Douzi, Samira & Ouahidi, Bouabid. *Effective Approaches for Intrusion Detection Systems in the Face of Low-Frequency Attacks*. Journal of Advances in Information Technology. 15. 1070-1078. 10.12720/jait.15.9.1070-1078. 2024.
 - [183] P. V. Pandit et al., *Implementation of Intrusion Detection System Using Various Machine Learning Approaches with Ensemble learning*, in International Conference on Advancement in Computation and Computer Technologies (InCACCT), pp. 468–472, May, 2023.
 - [184] J. Navarro, A. Deruyver, and P. Parrend, *A Systematic Survey on Multi-step Attack Detection*, Computers & Security, vol. 76, Mar. 2018.
 - [185] S. Walling and S. Lodh, *A Survey on Intrusion Detection Systems : Types, Datasets, Machine Learning methods for NIDS and Challenges*, in 13th International Conference on Computing Communication and Networking Technologies (ICCCNT), pp. 1–7, Oct., 2022.
 - [186] Hamdi Friji. *Graph neural network-based intrusion detection for secure edge networks*. Computer science. Institut Polytechnique de Paris, 2024. English. NNT : 2024IPPAS030. tel-04986862.
 - [187] Sofiane Lagraa, Jérôme François, Abdelkader Lahmadi, Marine Miner, Christian Hammer-schmidt, and Radu State. 2017. *BotGM : Unsupervised graph mining to detect botnets in traffic flows*. In 2017 1st Cyber Security in Networking Conference (CSNet). IEEE, 1–8.
 - [188] David Pujol-Perich, José Suárez-Varela, Albert Cabellos-Aparicio, and Pere BarletRos. 2022. *Unveiling the potential of graph neural networks for robust intrusion detection*. ACM SIGMETRICS Performance Evaluation Review 49, 4 (2022), 111–117.
 - [189] Zhou, J.; Xu, Z.; Rush, A.M.; Yu, M. *Automating Botnet Detection with Graph Neural Networks*. arXiv 2020, arXiv :2003.06344.
 - [190] A. Presekal, A. Stefanov, V. S. Rajkumar and P. Palensky, *Attack graph model for cyber-physical power systems using hybrid deep learning*, IEEE Trans. Smart Grid, vol. 14, no. 5, pp. 4007-4020, Sep. 2023.
 - [191] A. Presekal, A. Stefanov, V. S. Rajkumar and P. Palensky, *Attack graph model for cyber-physical power systems using hybrid deep learning*, IEEE Trans. Smart Grid, vol. 14, no. 5, pp. 4007-4020, Sep. 2023.
 - [192] Lo, W.W., Layeghy, S., Sarhan, M., Gallagher, M., Portmann, M. : *EGraphSAGE : A Graph Neural Network based Intrusion Detection System*. CoRR abs/2103.16329 (Apr 2021), <https://arxiv.org/abs/2103.16329>.
 - [193] Pei, Y., Huang, T., van Ipenburg, W., Pechenizkiy, M. : *ResGCN : Attention-based Deep Residual Modeling for Anomaly Detection on Attributed Networks*. In : 2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA). pp. 1–2 (2021). <https://doi.org/10.1109/DSAA53316.2021.9564233>.
 - [194] Qin, K., Zhou, Y., Tian, B., Wang, R. : *AttentionAE : Autoencoder for Anomaly Detection in Attributed Networks*. In : 2021 International Conference on Networking and Network Applications (NaNA). pp. 480–484 (2021). <https://doi.org/10.1109/NaNA53684.2021.00089>.
 - [195] Narasimha Prasad, S., Senthamil Selvan, K., and Lakshmi Dhevi, B. (2022). *Intrusion detection system in wireless sensor networks and fair resource allocation using geometric deep learning techniques*. Wireless Personal Communications, pages 1–12.
 - [196] Zhou, J.; Xu, Z.; Rush, A.M.; Yu, M. *Automating Botnet Detection with Graph Neural Networks*. arXiv 2020, arXiv :2003.06344.
 - [197] Zhang, B.; Li, J.; Chen, C.; Lee, K.; Lee, I. *A Practical Botnet Traffic Detection System Using GNN*, Springer : Berlin/Heidelberg, Germany, 2021 ; pp. 66–78.
 - [198] Rusch, T.; Bronstein, M.; Mishra, S. *A Survey on Oversmoothing in Graph Neural Networks*. arXiv 2023, arXiv :2303.10993.
-

- [199] Guillermo Cobo Arróniz *Detection of network attacks using Graph Neural Networks*, 2023.
- [200] Benjamin Sanchez-Lengeling, Emily Reif, Adam Pearce, and Alexander B. Wiltschko. *An introduction to Graph Neural Networks*. September 2, 2022. <https://distill.pub/2021/gnn-intro/>.
- [201] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, M. Sun *Graph neural networks : A review of methods and applications AI Open*, 1 (2020), pp. 57-81, 10.1016/j.aiopen.2021.01.001.
- [202] S. Jegelka, *Theory of graph neural networks : Representation and learning*, in : 2022.
- [203] Sajith Wettewa, Lei Hou, Guomin Zhang, *Graph Neural Networks for building and civil infrastructure operation and maintenance enhancement*, Advanced Engineering Informatics, Volume 62, Part D, 2024, 102868, ISSN 1474-0346, <https://doi.org/10.1016/j.aei.2024.102868>.
- [204] Zhu Ming. *Data Mining*. Press of University of Science and Technology.2008.
- [205] Zhang, Hong & Jiang, Yi. (2011). *The Improved K-Means Algorithm in Intrusion Detection System Research*. Advanced Engineering Forum. 1. 204-208. 10.4028/www.scientific.net/AEF.1.204.
- [206] H. M. Hussain, K. Benkrid, A. T. Erdogan, and H. Seker, *Highly parameterized k-means clustering on fpgas : Comparative results with gpps and gpus*, in 2011 International Conference on Reconfigurable Computing and FPGAs, pp. 475–480, Nov 2011.
- [207] Q. Chen, D. Li, and C. Tang, *Knn matting*, in 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 869–876, June 2012.
- [208] S. Liu, X. Liang, L. Liu, X. Shen, J. Yang, C. Xu, L. Lin, and S. Yan, *Matching-cnn meets knn : Quasi-parametric human parsing*, in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1419–1427, June 2015.
- [209] <https://medium.com/@DataPlayer/scalable-approximate-nearest-neighbour-search-using-googles-scann-and-facebook-s-faiss-3e84df25ba>.
- [210] Zhang, S. ; Tong, H. ; Xu, J. ; Maciejewski, R. *Graph convolutional networks : A comprehensive review*. Comput. Soc. Netw. 2019, 6, 11.
- [211] J. Zhou, Z. Xu, A. Rush, and M. Yu, *Automating Botnet Detection with Graph Neural Networks*, in 4thWorkshop on Machine Learning and Systems (MLSys), 2020.
- [212] Tran, D.-H.; Park, M. *FN-GNN : A Novel Graph Embedding Approach for Enhancing Graph Neural Networks in Network Intrusion Detection Systems*. Appl. Sci. 2024, 14, 6932. <https://doi.org/10.3390/app14166932>.
- [213] H. Kang, D. H. Ahn, G. M. Lee, J. D. Yoo, K. H. Park and H. K. Kim, *IoT Network Intrusion Dataset*, IEEE DataPort, 2019. Available : <https://ieeedataport.org/open-access/iot-network-intrusion-dataset>. [Accessed 2020].
- [214] I. Ullah and Q. H. Mahmoud, *A Scheme for Generating a Dataset for Anomalous Activity Detection in IoT Networks*, In : Goutte C., Zhu X. (eds) Advances in Artificial Intelligence. Canadian AI 2020. Lecture Notes in Computer Science, vol. 12109, pp. 508-520, 2020.
- [215] Ullah, Imtiaz & Mahmoud, Qusay. (2020). *A Scheme for Generating a Dataset for Anomalous Activity Detection in IoT Networks*. Lecture Notes in Computer Science. 508-520. 10.1007/978-3-030-47358-7_52.
- [216] Hammood, Baseem & Sadiq, Ahmed. (2023). *ENSEMBLE MACHINE LEARNING APPROACH FOR IOT INTRUSION DETECTION SYSTEMS*. Iraqi Journal for Computers and Informatics. 49. 93-99. 10.25195/ijci.v49i2.458.

- [217] Ullah, Imtiaz & Mahmoud, Qusay. (2020). *A Scheme for Generating a Dataset for Anomalous Activity Detection in IoT Networks*. Lecture Notes in Computer Science. 508-520. 10.1007/978-3-030-47358-7_52
- [218] Qaddos, A., Yaseen, M.U., Al-Shamayleh, A.S. et al. *A novel intrusion detection framework for optimizing IoT security*. Sci Rep 14, 21789 (2024). <https://doi.org/10.1038/s41598-024-72049-z>.

Résumé

Les systèmes de détection d'intrusion (IDS) occupent une place centrale dans la cybersécurité moderne. Leur mission consiste à surveiller et analyser les événements d'un système informatique ou d'un réseau afin d'identifier toute activité malveillante ou non autorisée. Bien que renforcés par l'apprentissage automatique, ils font face à des défis majeurs notamment la contextualisation des attaques, la détection des attaques rares, le traitement en temps réel et la réduction des faux positifs et faux négatifs. Dans cette thèse, nous développons plusieurs approches innovantes basées sur l'apprentissage automatique pour renforcer la détection des intrusions réseau. Nous commençons par exploiter les séquences d'événements à l'aide d'un réseau de neurones récurrent de type LSTM, afin de modéliser les comportements malveillants. Pour améliorer la qualité des données en entrée, l'algorithme SHAP est utilisé pour identifier les caractéristiques les plus pertinentes, permettant ainsi de réduire le bruit, d'optimiser l'apprentissage et d'accroître la capacité du modèle à détecter les anomalies. Dans une seconde contribution, nous adaptons le modèle Paragraph Vector–Distributed Memory (PV-DM) pour générer des représentations denses (embeddings) résumant les séquences d'événements. Les dimensions d'embedding les plus pertinentes sont sélectionnées à l'aide de SHAP, puis utilisées pour entraîner un classifieur XGBoost. Cette combinaison assure un encodage contextuel riche, tout en maîtrisant la complexité du modèle et en assurant une classification efficace. Afin de traiter la détection des classes rares sans recourir aux méthodes classiques de rééchantillonnage, nous proposons une approche originale reposant sur la subdivision des données selon la fréquence des attaques. Nous identifions ensuite les caractéristiques propres à chaque type d'attaque grâce à l'algorithme ShapRFECV, avant d'entraîner XGBoost sur ces sous-ensembles optimisés. Cette méthode améliore la précision de détection tout en conservant l'intégrité des données. Enfin, nous introduisons une approche basée sur les graphes pour exploiter les relations structurelles entre événements réseau. Les données sont d'abord regroupées en clusters via l'algorithme K-Means, puis connectées à leurs plus proches voisins par KNN pour former les arêtes d'un graphe. Un réseau de neurones convolutionnel pour graphes (GCN) est ensuite entraîné pour classifier les événements en comportements normaux ou intrusifs, en exploitant la topologie du réseau et les corrélations entre entités. Ces contributions forment une chaîne cohérente, allant de l'analyse temporelle à la représentation topologique, aboutissant à des IDS plus robustes, intelligents et adaptatifs face à l'évolution des menaces.

Mots-clefs (5) : Intelligence artificielle, Système de détection d'intrusion, PV-DM, sélection de caractéristiques, Deep Learning, déséquilibre des classes, Graphes.

Abstract

Intrusion Detection Systems (IDS) play a central role in modern cybersecurity. Their mission is to monitor and analyze the events of a computer system or network to identify any malicious or unauthorized activity. Increasingly, these systems rely on machine learning algorithms capable of processing data flows in real time. However, several challenges remain, including attack contextualization, the detection of rare threats, real-time processing, and the reduction of false positives and false negatives. In this thesis, we develop several innovative machine learning-based approaches to enhance network intrusion detection. We begin by modeling sequences of network events using a Long Short-Term Memory (LSTM) recurrent neural network to capture malicious behavior. To improve the quality of input data, the SHAP algorithm is used to identify the most relevant features, helping to reduce noise, optimize learning, and enhance anomaly detection. In a second contribution, we adapt the ParagraphVector–Distributed Memory (PV-DM) model to generate dense vector representations (embeddings) of event sequences. The most significant embedding dimensions are selected using SHAP and used to train an XGBoost classifier. This combination provides rich contextual encoding while controlling model complexity and ensuring efficient classification. To address the detection of rare classes without relying on traditional resampling techniques, we propose an original approach based on data subdivision according to attack frequency. The specific features of each attack type are identified using the ShapRFECV algorithm, and XGBoost is then trained on these optimized subsets. This method improves detection accuracy while preserving data integrity. Finally, we introduce a graph-based approach to capture structural relationships between network events. Data is first clustered using the K-Means algorithm, then connected to their nearest neighbors via KNN to form graph edges. A Graph Convolutional Network (GCN) is trained to classify events as normal or intrusive by leveraging the network topology and inter-entity correlations. These contributions form a coherent and complementary processing pipeline, ranging from temporal analysis to topological representation, ultimately leading to more robust, intelligent, and adaptive IDS capable of keeping pace with evolving threats.

Keywords (5) : Artificial Intelligence, Intrusion Detection System, PV-DM, Feature Selection, Deep Learning, Class Imbalance, Graphics.