Université du Québec en Outaouais
Département d'Informatique et d'Ingénierie
Québec, Canada

Thèse

Analyse des cybermenaces basée sur l'apprentissage machine

Auteur
Mohamed El Amine Bekhouche

Superviseur
Pr. Kamel Adi

Jury d'évaluation

Pr. Alan Davoust	Président du jury
Pr. Mohamed Mejri	Membre externe du jury
Pr. Omer Landry Nguena Timo	Membre interne du jury

10 juin 2025

Remerciements

Tout d'abord, je rends grâce à Dieu Tout-Puissant, pour m'avoir accordé la santé, la patience et la persévérance tout au long de ce parcours académique. Sans Sa guidance et Sa bénédiction, l'accomplissement de cette thèse n'aurait pas été possible.

Je tiens à exprimer ma profonde gratitude à mon directeur de recherche, le professeur Kamel Adi, pour son encadrement précieux, ses conseils avisés, et son soutien indéfectible tout au long de ce projet. Son expertise et sa disponibilité ont été des éléments clés dans l'aboutissement de ce travail.

Mes pensées les plus profondes vont à ma chère mère, dont l'amour, le soutien inconditionnel et les sacrifices ont été une source d'inspiration et de motivation. À toute ma famille, merci pour votre patience, votre encouragement et votre présence constante.

Je tiens également à remercier tous mes amis, sans exception, pour leur amitié sincère, leur soutien moral, et les moments de partage qui ont été une bouffée d'air frais durant ces années de travail intense.

Je souhaite également exprimer ma profonde reconnaissance au Laboratoire de Recherche en Sécurité Informatique (LRSI) pour l'environnement scientifique stimulant, les échanges enrichissants, et les ressources précieuses mis à disposition tout au long de mon parcours doctoral. Travailler au sein de ce laboratoire a grandement contribué à mon développement académique et professionnel.

Je remercie chaleureusement mes enseignants de l'École Nationale Supérieure de Technologie (ENST) pour leur enseignement de qualité, leur dévouement, et les bases solides qu'ils m'ont transmises, lesquelles ont été essentielles pour le succès de ce projet.

Enfin, mes remerciements vont à l'équipe du Centre d'expertises en prototypage intelligent (CEPI) avec laquelle j'ai eu l'honneur de travailler. Votre collaboration, vos idées novatrices, et votre soutien ont grandement enrichi cette expérience de recherche.

Résumé

La multiplication, la diversification et la sophistication des cyberattaques posent de nos jours, des défis majeurs aux organisations, notamment en matière de détection, d'attribution et de réponse rapide aux menaces. Malheureusement, l'approche traditionnelle de la Cyber Threat Intelligence (CTI), basée sur l'analyse manuelle, est devenue insuffisante face à la complexité des cybermenaces modernes. En effet, ces méthodes peinent à exploiter efficacement les vastes quantités de données provenant de sources ouvertes, telles que les réseaux sociaux ou les rapports d'incidents de sécurité. La collecte, le prétraitement et l'analyse de ces données nécessitent des ressources considérables, limitant ainsi la capacité des organisations à réagir de manière proactive.

Dans ce contexte, la présente thèse propose une approche innovante et prometteuse permettant d'automatiser les étapes clés du cycle de vie de la CTI. La première étape de ce cycle de vie repose sur la collecte de données. Il est donc nécessaire d'élaborer des techniques permettant d'extraire efficacement et en temps réel des informations critiques sur les cybermenaces, notamment en ce qui concerne les techniques, tactiques et procédures employées dans les cyberattaques. Ces informations doivent être extraites à partir de l'Open-Source Intelligence (OSINT), notamment des rapports d'incidents. Cette extraction structurée permet d'automatiser les processus d'analyse de la menace en établissant, par exemple, des liens entre les cybermenaces et leurs acteurs et de prédire la gravité des incidents, permettant ainsi de prioriser les réponses aux cyberattaques et d'optimiser l'allocation des ressources.

Cette thèse vise donc à renforcer les capacités des organisations en matière d'analyse et de défense proactive. Ses principales contributions sont les suivantes :

- Une nouvelle méthode pour l'analyse et la collecte en temps réel des informations liées aux cybermenaces à partir des réseaux sociaux.
- Une nouvelle méthode pour l'extraction et la reconnaissance des entités liées à la Cyber Threat Intelligence dans les rapports d'incidents de sécurité.
- Mise en place d'un framework méthodologique pour l'attribution des cyberattaques.
- Un nouveau modèle de prédiction du niveau de gravité des cybermenaces pour une gestion optimisée des risques.

Abstract

The multiplication, diversification, and sophistication of cyberattacks pose major challenges to organizations today, particularly in terms of detection, attribution, and rapid threat response. Unfortunately, the traditional approach of Cyber Threat Intelligence (CTI), based on manual analysis, has become insufficient in the face of the complexity of modern cyber threats. Indeed, these methods struggle to effectively exploit the vast amounts of data from open sources, such as social networks or security incident reports. The collection, preprocessing, and analysis of this data require considerable resources, thereby limiting organizations' ability to react proactively.

In this context, the present thesis proposes an innovative and promising approach to automating the key steps of the CTI lifecycle. The first step of this lifecycle is data collection. It is therefore necessary to develop techniques to efficiently and in real time extract critical information about cyber threats, particularly regarding the techniques, tactics, and procedures employed in cyberattacks. This information must be extracted from Open-Source Intelligence (OSINT), particularly from incident reports. This structured extraction enables the automation of threat analysis processes by, for example, establishing links between cyber threats and their actors and predicting the severity of incidents, thereby allowing organizations to prioritize responses to cyberattacks and optimize resource allocation.

This thesis thus aims to strengthen organizations' capabilities in analysis and proactive defense. Its main contributions are as follows :

- A new method for the real-time collection and analysis of cyber threat-related information from social networks.
- A new method for extracting and recognizing entities related to Cyber Threat Intelligence in security incident reports.
- Design and implementation of a methodological framework for cyberattack attribution.
- A new model for predicting the severity level of cyber threats for optimized risk management.

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

API Application Programming Interface.
APT Advanced Persistent Threat.
ATT&CK Adversarial Tactics, Techniques, and Common Knowledge.

BERT Bidirectional Encoder Representations from Transformers.
Bi-LSTMs Bidirectional long-short term memory.
BPTT Backpropagation through time.

CFR Conditional Random Fields.
CNNs Convolutional Neural Networks.
CS cosine similarity.
CTI Cyber Threat Intelligence.
CVSS système commun de notation des vulnérabilités.

DBSCAN Density-Based Spatial Clustering of Applications with Noise.
DG Dependency Graph.
DL Deep Learning.
DNNs Deep neural networks.
DNP Distributed Network Protocol.
DRL Deep Reinforcement Learning.
DT Decision Tree.

IA Intelligence Artificielle.
ICSs Industrial Control Systems.
IoCs Indicateurs de Compromission.

KMeans KMeans.
KNN K-Nearest Neighbors.

LDA Latent Dirichlet Allocation.

LLMs Large Language Models.

LR Logistic Regression.

LSA Latent Semantic Analysis.

LSTMs Long Short-Term Memory.

MASINT Measurement And Signature Intelligence.

MITD Man In The Disk.

ML Machine Learning.

ModBus Modicon Communication Bus.

NB Naïve Bayes.

NER Named-Entity Recognition.

NLP Natural language processing.

OSINT Open-Source Intelligence.

PLCs Programmable Logic Controllers.

ReLU Rectified Linear Unit.

RF Random Forest.

RNNs Recurrent Neural Networks.

SIGINT Signals Intelligence.

SOC Security Operations Center.

STIX Structured Threat Information eXpression.

SVCE Security Vulnerability Concept Extractor.

SVM Support Vector Machine.

Tanh Tangente Hyperbolique.

TF-IDF Term Frequency-Inverse Document Frequency.

TTPs Tactiques, Techniques et Procédures.

UCO Unified Cybersecurity Ontology.

URL Uniform Resource Locator.

Chapitre 1

Introduction générale

1.1 Préambule

De nos jours, les cybercriminels posent des défis majeurs aux organisations qui cherchent à protéger leurs données et leurs systèmes contre les cyberattaques. Ces cybercriminels peuvent être des attaquants individuels ou des organisations criminelles bien structurées et utilisent souvent une large gamme de techniques pour mener des attaques, avec des objectifs divers, comme le vol d'identité ou d'autres informations sensibles. Une cyberattaque peut également viser à transformer la machine de la victime en "zombie", permettant ainsi de l'utiliser comme arme dans d'autres attaques. Bien que les méthodes d'attaques varient, les cyberattaques suivent généralement un cycle de vie similaire, allant de la phase de reconnaissance de la cible à l'exécution de charges virales via des vecteurs d'attaques.

Face à l'augmentation exponentielle du volume et de la diversité des cyberattaques, la tâche des analystes en cybersécurité est devenue extrêmement complexe, rendant ainsi l'approche manuelle ou semi-manuelle de détection et de réponse aux incidents largement inefficace. Il devient alors urgent de développer des approches innovantes de Cyber Threat Intelligence (CTI) pour contrer cette menace croissante. Traditionnellement, la CTI permet de produire, de manière manuelle ou semi-manuelle, des informations stratégiques sur le paysage de la cybermenace. Information destinée à aider les professionnels de la cybersécurité à identifier les Indicateurs de Compromission (IoCs) et à recueillir des détails sur les cyberattaques. Cette information est souvent extraite de rapports d'incidents et présentée sous la forme de tactiques, techniques et procédures Tactiques, Techniques et Procédures (TTPs) employées par les cybercriminels. Les TTPs fournissent des observables comportementaux permettant de détecter les cyberattaques et ceci en analysant les artefacts numériques collectés depuis les réseaux et les systèmes informatiques des victimes. Les TTPs sont alors utilisées pour élaborer des profils détaillés des attaquants, aidant ainsi les chercheurs à mieux anticiper les futures attaques et à renforcer les capacités de détection de menaces. Par ailleurs, la structure des TTPs permet aux analystes de comprendre quelles actions de

l'adversaire relèvent de procédures spécifiques, associées à des tactiques et techniques particulières et cela aide à identifier les objectifs des attaquants et à concevoir des stratégies de défense plus efficaces.

L'apprentissage machine (ML) peut avantageusement contribuer dans le domaine de la CTI en facilitant la mise en place d'outils performants et précis qui permettent la détection et l'analyse de la menace avec des objectifs multiples tels que l'évaluation des risques de cyberattaques, la mise en place d'une cyberdéfense proactive ou l'aide à l'élaboration de cyberréponses aux incidents, etc. Par exemple, les algorithmes ML peuvent être entraînés sur des données historiques afin de prédire de nouvelles menaces, permettant ainsi de libérer les analystes pour des tâches plus complexes et plus stratégiques. Autre exemple, le traitement automatique du langage naturel (NLP) est conçu pour permettre aux machines de comprendre, interpréter, et générer du langage naturel. Appliqué aux rapports d'incidents, le NLP peut extraire des informations pertinentes telles que le type et la gravité de l'incident, les systèmes ou actifs affectés, les logiciels malveillants impliqués, et les recommandations de remédiation. Cela aide les organisations à améliorer leurs processus de réponse et à prévenir des incidents similaires dans le futur. Par ailleurs, l'utilisation des modèles séquence-à-séquence, en particulier ceux basés sur des architectures de transformeurs, réputés pour leur capacité à gérer des séquences de données longues et complexes, sont idéaux pour analyser les tendances et comportements dans les données de cybersécurité. Ils permettent de traiter et d'interpréter de longues séquences d'incidents, offrant ainsi une capacité de prédiction des actions futures des cybercriminels. Cette capacité de prévision est fondamentale pour anticiper et contrer les attaques avant leur exécution, renforçant ainsi les stratégies de défense proactive des organisations. Enfin, l'apprentissage par renforcement (DRL) offre un potentiel important pour optimiser la prise de décision en CTI. En effet, en simulant des scénarios d'attaque et en apprenant de chaque interaction, les algorithmes d'apprentissage par renforcement sont capables de développer des stratégies de réponse aux incidents qui s'adaptent en permanence. Cette approche permet aux systèmes de réagir de manière dynamique aux nouvelles menaces adverses, améliorant ainsi l'efficacité et la réactivité des décisions prises par les équipes de réponse aux incidents. Cependant, nous pensons que l'intuition et l'expertise humaines demeurent essentielles dans ce domaine et l'association de l'intelligence humaine et de techniques avancées d'intelligence artificielle promet une approche plus robuste et intégrée face aux défis croissants de la cybersécurité.

1.2 Motivations

Le cycle de vie de la CTI, également appelé cycle d'intelligence, constitue un processus structuré et itératif visant à identifier, à analyser et à comprendre les cybermenaces. Son objectif est de fournir aux organisations des informations exploitables pour protéger leurs actifs critiques. Ce cycle se décompose en plusieurs phases essentielles : la planification des actions de collecte, la collecte des données, le prétraitement des don-

nées, l'analyse et la diffusion des renseignements stratégiques aux parties prenantes. Parmi ces étapes, celles de la collecte, du prétraitement et de l'analyse des données jouent un rôle central, car elles transforment des informations brutes en intelligence opérationnelle. Toutefois, chacune de ces phases est confrontée à des défis significatifs dans le contexte des cybermenaces modernes, nécessitant ainsi des solutions innovantes.

La collecte de données est une étape cruciale et représente souvent le point de départ du cycle d'intelligence. Dans le domaine de la CTI, cette collecte s'appuie largement sur l'intelligence en sources ouvertes, ou Open-Source Intelligence (OSINT). L'OSINT concerne l'extraction d'informations accessibles publiquement sur la menace à partir de sources diverses, telles que les articles de presse, les forums, les blogs, les réseaux sociaux et les sites web spécialisés. Parmi ces plateformes, Twitter se distingue comme une source importante en raison de la richesse et de la rapidité de propagation des informations partagées. En effet, chaque jour, des milliers de tweets contenant des informations critiques sur des vulnérabilités, des exploits et des cyberattaques en cours, sont publiés en temps réel par des professionnels et des chercheurs en cybersécurité. Ainsi, la capacité à extraire rapidement des informations pertinentes à partir de ces flux massifs est essentielle pour identifier les menaces émergentes et prévenir leurs impacts. Toutefois, exploiter efficacement ces informations contenues dans des flux de données volumineux et non structurées représente un défi majeur.

Une fois les données collectées, elles peuvent être prétraitées afin d'être exploitées efficacement. Ce prétraitement inclut notamment l'extraction des entités liées à la CTI tel que les TTPs ; ces informations comportementales jouant un rôle clé dans la compréhension des cyberattaques et des tactiques employées par les acteurs de menaces. Traditionnellement, cette extraction est principalement réalisée de manière manuelle par des analystes en cybersécurité, souvent à travers une analyse minutieuse des documents textuels. Cependant, cette méthode est extrêmement chronophage, surtout face à l'augmentation continue du nombre de rapports d'incidents générés quotidiennement. Il devient donc impératif de développer des méthodes automatiques capables de traiter ces documents de manière plus rapide et précise.

La phase d'analyse des données est le pivot du cycle d'intelligence, car elle permet de transformer les informations collectées et prétraitées en intelligence exploitable. Par exemple, parmi les cybermenaces les plus sophistiquées figurent les Advanced Persistent Threats (APT), des attaques menées par des entités organisées, souvent motivées par des objectifs politiques, économiques ou stratégiques. Ces attaques visent des informations sensibles ou des infrastructures critiques et se caractérisent par leur furtivité et leur persistance. Il est important de pouvoir attribuer ces cyberattaques à leurs auteurs ou commanditaires, un processus complexe. L'attribution repose sur des analyses approfondies pour tracer les acteurs malveillants à partir de divers indices, tels que les techniques utilisées. Cependant, les cybercriminels emploient des techniques sophistiquées d'évasion, telles que l'obfuscation, les faux drapeaux et l'utilisation d'outils partagés, compliquant ainsi considérablement ce processus d'attribution.

Dans le cadre de la phase d'analyse, un autre aspect essentiel est la prédiction de la gravité des cybermenaces. Classer ces derniers en fonction de leur gravité est

fondamental pour permettre aux organisations de prioriser leurs efforts de réponse. Cela est particulièrement crucial pour les grandes organisations confrontées à un volume élevé d'incidents, où une gestion efficace des ressources est indispensable. Une mauvaise priorisation pourrait entraîner une allocation sous-optimale des ressources, augmentant ainsi le risque d'impacts majeurs sur les systèmes critiques.

Face à ces défis, les approches basées sur l'apprentissage machine offrent des opportunités sans précédent pour automatiser et améliorer chaque phase du cycle de vie de la CTI.

1.3 Objectifs

Afin de relever les défis posés par la collecte, le prétraitement et l'analyse des données dans le cadre de la CTI, cette thèse vise à développer une approche innovante basée sur l'apprentissage machine. Les objectifs spécifiques de ce travail s'articulent autour des axes suivants :

1. Automatisation de la collecte d'informations en temps réel à partir des réseaux sociaux : nous avons proposé une approche capable de détecter automatiquement les informations pertinentes sur les cybermenaces à partir de plateformes de réseaux sociaux, avec un accent particulier sur Twitter. Notre approche est capable d'identifier les tweets contenant des informations critiques, telles que des vulnérabilités nouvellement découvertes ou des attaques en cours , en temps réel, permettant ainsi aux analystes de réagir plus rapidement et plus efficacement.
2. Extraction automatique des indicateurs comportementaux (TTPs) : pour répondre aux limitations des approches manuelles dans le prétraitement des données, nous avons développé un modèle d'apprentissage machine dédié à l'extraction automatique des (TTPs) à partir de rapports d'incidents non structurés. En automatisant cette tâche, notre modèle permet de structurer les données comportementales de manière rapide et précise, offrant ainsi un gain de temps significatif pour les analystes de cybersécurité et améliorant la capacité à traiter un grand volume de rapports.
3. Attribution des cyberattaques aux acteurs de menaces : nous avons introduit un framework permettant d'attribuer les cyberattaques à leurs auteurs, en analysant les caractéristiques des logiciels malveillants associés. Ce framework repose sur un réseau siamois capable de comparer de nouvelles menaces avec des profils existants d'acteurs malveillants. Si une menace ne correspond à aucun profil connu, notre méthode est capable de créer automatiquement un nouveau profil, renforçant ainsi les capacités des organisations à détecter et à répondre à des menaces inédites.
4. Prédiction de niveau de gravité des cybermenaces : pour aider les organisations à mieux prioriser leurs réponses aux incidents, nous avons développé un modèle capable d'analyser les rapports d'incidents et de classer les menaces selon leur

gravité. Notre approche permet d'identifier rapidement les menaces les plus critiques, aidant ainsi les équipes de sécurité à concentrer leurs efforts là où ils sont le plus nécessaire permettant ainsi une meilleure gestion des ressources.

1.4 Organisation du document

Ce premier chapitre a présenté le contexte général de notre recherche, les motivations, ainsi que les objectifs principaux de cette thèse. Le reste du document est organisé comme suit :

- **Chapitre** 2 **- Concepts généraux** : ce chapitre introduit les notions fondamentales liées à la CTI. Nous y présentons également des cadres de référence tels que le Framework ATT&CK, qui sert de base à de nombreuses analyses. Les concepts abordés permettront de mieux comprendre les concepts utilisés dans les chapitres subséquents.
- **Chapitre** 3 **- Principes de l'apprentissage automatique** : ce chapitre explore les algorithmes d'apprentissage automatique. Nous y abordons notamment les réseaux de neurones profonds, les réseaux convolutifs, les réseaux récurrents, ainsi que les modèles basés sur des architectures avancées tel que les transformeurs. Ces méthodes constituent base technique de nos contributions.
- **Chapitre** 4 **- Revue de la littérature** : ce chapitre propose une analyse critique des travaux existants dans le domaine de la CTI. Nous passons en revue les approches utilisées pour la collecte, le prétraitement, et l'analyse des données, en mettant en évidence leurs forces et leurs limites. Cette revue de la littérature permet de situer nos contributions par rapport à l'état de l'art.
- **Chapitres** 5, 6, 7 et 8 **- Contributions de recherche** : ces chapitres détaillent les contributions cette thèse. Ils incluent les travaux développés pour la collecte d'informations liée au cybermenace sur les réseaux sociaux, l'extraction automatique des indicateurs comportementaux (TTPs) à partir de rapports d'incidents, l'attribution des cyberattaques, ainsi que la prédiction de niveau de gravité pour la gestion des risques.
- **Chapitre** 9 **- Conclusion et perspectives** : le document se termine par une synthèse des principales contributions de cette thèse. Nous y discutons également des pistes de recherche futures pour améliorer davantage le travail proposé et répondre aux défis émergents dans le domaine de la cybersécurité.

Chapitre 2

Concepts généraux

2.1 Introduction

Avec la numérisation croissante à l'échelle mondiale, les cybermenaces sont devenues à la fois plus sophistiquées et plus fréquentes, imposant aux organisations de cyberprotection une connaissance approfondie des Tactiques, Techniques et Procédures (TTPs) employées par les attaquants. Cette connaissance repose sur un processus complet de collecte de données et d'analyse de la menace, englobant les Indicateurs de Compromission (IoCs) comportementaux des attaquants ainsi que d'autres informations stratégiques.

Parmi les outils les plus utilisés pour la Cyber Threat Intelligence (CTI) figure le framework Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK). Ce framework structure les TTPs des attaquants en tactiques et techniques, offrant un langage standardisé pour examiner et classifier les cybermenaces.

Une autre source majeure de CTI est l'OSINT. L'OSINT désigne la collecte et l'analyse d'informations accessibles publiquement, telles que les publications sur les médias sociaux, les articles de presse et les forums en ligne. L'OSINT peut fournir des informations précieuses sur les motivations, les capacités et les TTPs des adversaires, et s'avère utile pour compléter d'autres sources de CTI.

Dans ce chapitre, nous introduisons les fondements de la CTI ainsi que ses principaux composants, notamment les IoCs atomiques et comportementaux, le framework ATT&CK et l'OSINT. Nous examinerons comment ces éléments s'articulent pour offrir une vision globale des cybermenaces potentielles et comment ils peuvent être exploités pour élaborer des stratégies de défense efficaces. À la fin de ce chapitre, nous aurons acquis une compréhension approfondie de l'importance de la CTI dans le domaine de la cybersécurité.

2.2 Cyber Threat Intelligence

La Cyber Threat Intelligence (CTI) a été développée pour aider les professionnels de la sécurité à identifier les indicateurs de cybermenaces, à recueillir des détails précis sur ces attaques, et à y répondre de manière efficace et opportune.

Selon [14], la Cyber Threat Intelligence (CTI) correspond aux informations sur les cybermenaces une fois qu'elles ont été collectées, évaluées selon leur source et leur fiabilité, puis analysées par des experts à l'aide de techniques structurées et rigoureuses. Les experts disposent d'un accès à l'ensemble des sources d'informations pertinentes et doivent identifier des similitudes et des différences au sein de grandes quantités de données pour détecter les attaques et produire des renseignements précis et exploitables.

Une autre source [26] définit la CTI comme le produit final résultant de la collecte et du traitement ciblé d'informations concernant l'environnement, les capacités et les intentions des acteurs malveillants. L'objectif est d'identifier les menaces et de fournir aux professionnels de la cybersécurité les moyens d'exploiter ces informations pour renforcer les défenses.

Dans ce contexte, il est essentiel de bien distinguer deux notions clés : la cybermenace et la cyberattaque. Une cybermenace désigne tout événement ou tout acteur pouvant compromettre la confidentialité, l'intégrité ou la disponibilité d'un système informatique. Il s'agit donc d'une intention potentielle d'attaque. À l'inverse, une cyberattaque constitue la réalisation effective de cette menace : c'est un acte malveillant exécuté dans le but de causer un dommage.

Un défi majeur consiste à fournir cette intelligence de manière claire et exploitable lorsque d'importants volumes de données sont collectés ou générés par différentes sources de sécurité. Pour y parvenir, des techniques avancées d'analyse de données massives sont nécessaires pour exploiter, interpréter et extraire des connaissances pertinentes à partir des informations collectées. Ce processus fait appel au cycle d'intelligence [79], illustré à la figure 2.1, qui décrit la séquence d'activités permettant d'obtenir, d'assembler, puis de transformer les informations en intelligence pour les mettre à disposition des utilisateurs. Bien qu'il existe différentes représentations de ce cycle, on peut généralement le résumer en cinq phases principales :

Identification des besoins. cette phase consiste à définir les besoins en information, planifier la collecte de données en collaboration avec les organismes de collecte et assurer un suivi continu de leur performance.

Collecte des données. il s'agit d'exploiter les sources identifiées pour recueillir les informations nécessaires, puis de transmettre ces données à l'unité de traitement appropriée en vue de la production de renseignements exploitables.

Prétraitement des données. cette étape vise à transformer les informations brutes en données analysables, facilitant leur interprétation lors de la phase d'analyse.

Analyse des données. l'analyse consiste à intégrer, évaluer et interpréter les informations afin de produire des intelligences contextualisés et cohérentes, permettant une compréhension approfondie des cybermenaces.

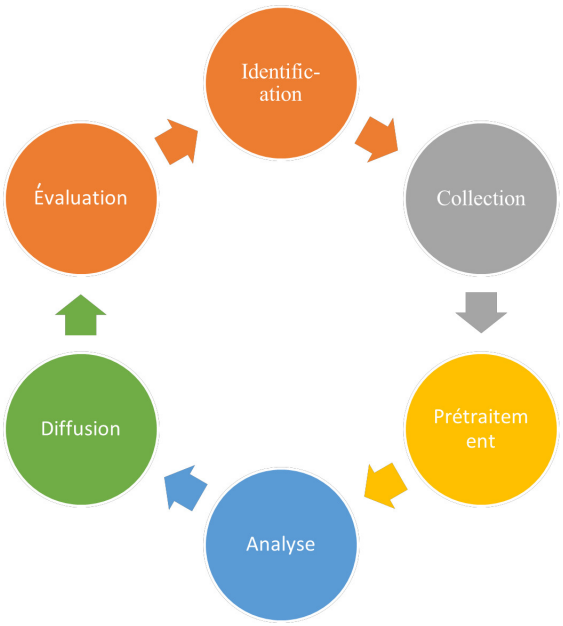


FIGURE 2.1 – Cycle de vie de la CTI [79].

Diffusion. les intelligences sont transmis aux parties prenantes dans un format adapté et en temps opportun, garantissant que l'information est utilisable et pertinente.

Évaluation. certaines études [89, 18] incluent une sixième étape : l'évaluation et le retour d'information après la diffusion des renseignements. Cette phase permet de corriger d'éventuels biais et incertitudes liés aux différences d'interprétation des analystes, et implique une nouvelle itération du cycle d'intelligence.

La CTI en tant que produit final est le résultat d'une série d'actions menées de manière séquentielle. Dans cette série, le cycle d'intelligence offre une explication simplifiée d'un processus complexe. Par exemple, une autorité ou un gouvernement peut avoir des besoins spécifiques en matière d'information pour prendre des décisions éclairées sur un sujet donné. Le cycle débute donc par l'identification de ces besoins, suivie de la planification de l'acquisition des informations qui seront ensuite traitées et analysées pour produire des renseignements exploitables.

Une fois la planification achevée, la prochaine étape est l'acquisition des données. Cette collecte peut être réalisée par le biais de différentes méthodes de collecte d'informations : Signals Intelligence (SIGINT), Open-Source Intelligence (OSINT), Measurement And Signature Intelligence (MASINT), etc. Bien que ces méthodes n'imposent pas de standard unique pour le format des données, elles définissent les catégories de sources à partir desquelles les informations peuvent être extraites, telles qu'un simple site web public, un forum, ou encore un artefact intercepté.

Après la collecte, un pipeline de traitement et d'exploitation est activé pour trans-

former les données en une forme adaptée à la production de renseignements. Ce pipeline inclut des tâches comme le décryptage, la traduction ou la conversion des données. Dans le cadre du cycle d'intelligence, l'étape d'analyse est indispensable : c'est lors de cette phase que les données traitées sont examinées pour générer le renseignement final. Cette analyse intègre les informations recueillies et traitées, quelle que soit leur source d'origine.

Enfin, une fois les renseignements produits, ils sont transmis aux utilisateurs ayant exprimé les besoins initiaux, sous une forme appropriée et par divers canaux de diffusion. Ce produit final est destiné à faciliter la prise de décision des analystes et, le cas échéant, à déclencher une nouvelle itération du cycle d'intelligence.

2.2.1 Types de CTI

L'analyse de la CTI s'articule souvent autour de la triade des acteurs, intentions et capacités, en tenant compte de leurs TTPs, de leurs motivations, ainsi que de leur accès aux cibles visées. L'étude de cette triade permet de formuler des évaluations à trois niveaux distincts : stratégique, opérationnel et tactique. La figure 2.2 illustre ces trois niveaux de CTI et leur application dans le contexte de la cybersécurité.

FIGURE 2.2 – Les différents niveaux de la CTI [36].

CTI tactique. La CTI tactique consiste en l'évaluation des événements, investigations et activités à court terme, fournissant un soutien opérationnel quotidien destiné à un public technique. Elle identifie des IoCs spécifiques permettant aux équipes de cybersécurité de détecter et de neutraliser des menaces ciblant un système informatique. Ces IoCs incluent des adresses IP malveillantes, des noms de domaine suspects, un trafic anormal, des alertes de connexion inhabituelles

ou des pics de demandes de fichiers. La CTI tactique est généralement automatisée et a une durée de vie souvent courte en raison de l'obsolescence rapide des IoCs [88].

CTI opérationnelle. La CTI opérationnelle se concentre sur l'évaluation des incidents potentiels liés à des événements ou à des activités spécifiques, en répondant aux questions : qui est derrière l'attaque, pourquoi a-t-elle eu lieu et comment a-t-elle été menée ? Ces informations permettent de cibler des incidents particuliers. Ainsi, ce type de CTI fournit des renseignements détaillés sur le comportement des cyberattaquants et leurs TTPs, souvent associés à des groupes, des logiciels malveillants ou des outils utilisés. La CTI opérationnelle demande plus de ressources que la CTI tactique et possède une durée de vie plus longue, car les cyberattaquants ne modifient pas leurs TTPs aussi facilement que leurs IoCs [83].

CTI stratégique. La CTI stratégique consiste en une analyse de haut niveau destinée à un public non technique. Elle intègre diverses sources d'information pour fournir une vision d'ensemble, permettant ainsi de formuler des décisions et des politiques sur des questions générales. Ce type d'intelligence offre une vue globale des intentions et des capacités des cybermenaces, en identifiant les acteurs, les outils et les TTPs utilisés, ainsi que les tendances et les risques émergents. Elle informe les décideurs et les alerte en temps réel face aux menaces potentielles [96].

Une exploitation optimale de la CTI peut offrir une meilleure compréhension des cybermenaces, permettant ainsi une réponse plus rapide et mieux ciblée en matière de développement et d'allocation des ressources. Par exemple, la CTI peut aider les analystes à déterminer les risques commerciaux acceptables, à établir des contrôles budgétaires et à prendre des décisions en matière d'équipement et de personnel (CTI stratégique) ; à fournir des informations précieuses pour guider et soutenir la réponse aux incidents ainsi que les activités post-incident (CTI opérationnelle et tactique) ; et à valider et prioriser l'utilisation des IoCs en précisant leurs durées de validité (CTI tactique).

En ce qui concerne le cycle de vie de la Cyber Threat Intelligence, notre travail se concentre spécifiquement sur trois étapes clés : la collecte, le prétraitement et l'analyse des données. Ces étapes constituent le socle initial du processus d'intelligence, permettant de transformer des données brutes en renseignements exploitables. Ainsi, notre contribution s'inscrit principalement dans les deux premiers niveaux de la CTI, à savoir la CTI opérationnelle et la CTI tactique. La première vise à fournir des indicateurs techniques à court terme pour la détection et la réponse aux menaces, tandis que la seconde s'intéresse aux modes opératoires des attaquants afin d'orienter les mesures de défense à moyen terme.

2.2.2 Indicateurs de Compromission

La CTI repose sur les IoCs comme éléments clés pour détecter et identifier plus rapidement les activités malveillantes au sein des infrastructures ciblées par les cyberattaquants. Les IoCs permettent de spécifier l'utilisation des capacités technologiques, telles que les outils, les artefacts, et les TTPs développées par les acteurs de la menace [74]. Selon [49], les IoCs peuvent être classés en trois catégories, basées sur leur niveau de complexité et la granularité des données qu'ils représentent.

IoCs Atomiques. Les indicateurs atomiques sont des éléments de base qui ne peuvent être décomposés en parties plus petites tout en conservant leur signification dans le contexte d'une intrusion. Des exemples d'indicateurs atomiques incluent les adresses IP et les noms de domaine.

IoCs Calculés. Les indicateurs calculés sont dérivés de données utilisées dans une cyberattaque. Les valeurs de hachage et les signatures de virus en sont des exemples, car ils nécessitent un processus de calcul pour identifier des caractéristiques spécifiques associées à des menaces.

IoCs Comportementaux. Les indicateurs comportementaux représentent une combinaison d'indicateurs atomiques et calculés, souvent caractérisés par une logique combinatoire ou une fréquence d'occurrence. Un exemple concret de séquence comportementale (TTPs) peut être observé dans les campagnes d'infection menées par le malware Emotet. Initialement conçu comme un cheval de Troie bancaire, Emotet a évolué vers un vecteur d'infection modulaire capable de déployer d'autres malwares comme TrickBot ou Ryuk. La chaîne d'attaque commence par une phase de reconnaissance (T1598.002), où les attaquants utilisent des courriels de phishing personnalisés contenant des liens malveillants déguisés en factures ou documents administratifs. Une fois l'utilisateur piégé, l'étape d'accès initial (T1566.002) est franchie par le téléchargement d'un document Word piégé contenant des macros. L'exécution (T1059.005) est ensuite assurée par un script VBScript, qui télécharge le binaire d'Emotet. Pour assurer sa persistance (T1547.001), le malware s'inscrit dans la base de registre. Vient ensuite une phase d'exploration du système (T1016), suivie du téléchargement de modules supplémentaires via PowerShell (T1059.001), tels que TrickBot, qui poursuit l'exploration des fichiers (T1083), voire une propagation latérale (T1021.002) si possible. Enfin, Emotet établit une connexion de commande et contrôle (T1071.001) en HTTPS avec son serveur distant. Cette séquence, illustrée dans la Figure 2.3, démontre la pertinence des TTPs comportementaux pour détecter et caractériser les actions d'un acteur malveillant au-delà des simples indicateurs techniques.

Selon la pyramide de la douleur [10] dans la figure 2.4, la modification des IoCs comportementaux, représentent un défi important pour les attaquants mais sont très efficaces contre les systèmes de défense. En revanche, les IoCs atomiques et calculés sont plus faciles à modifier par les attaquants mais sont également plus facilement détectés

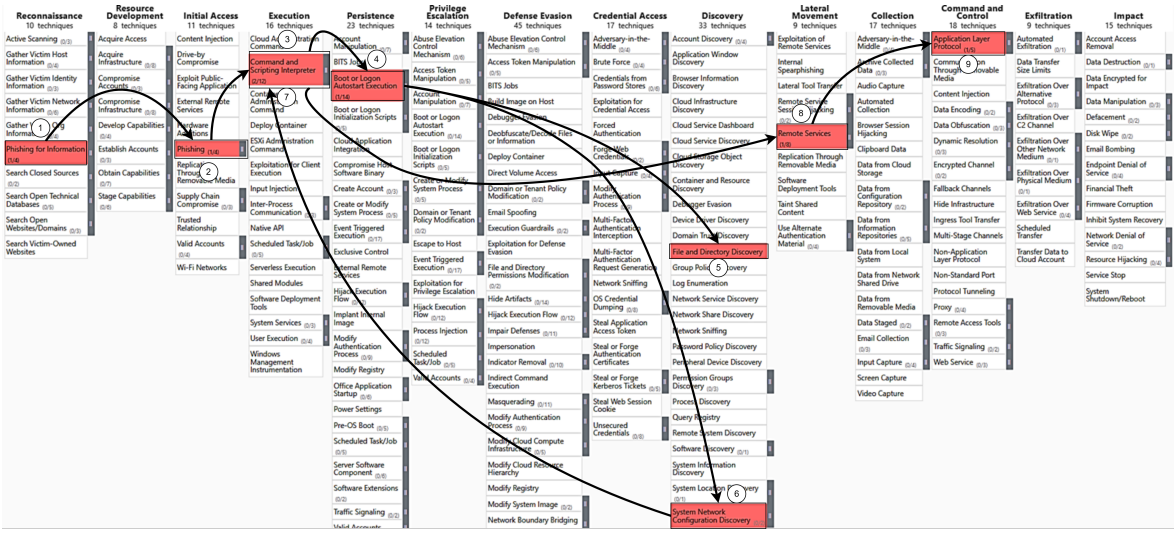


FIGURE 2.3 – Chaîne comportementale d'une campagne d'infection par Emotet basée sur les TTPs MITRE ATT&CK.

par les systèmes de sécurité. Ce contraste souligne la pertinence accrue et l'avantage stratégique de se concentrer sur les IoCs comportementaux dans les mécanismes de défense en cybersécurité.

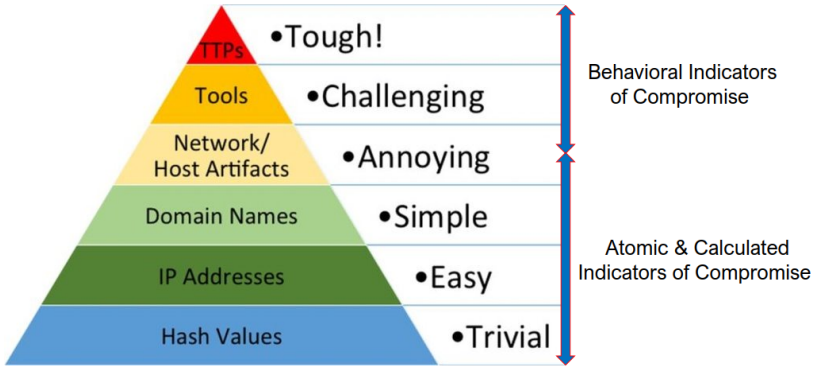


FIGURE 2.4 – Pyramide de la douleur [10].

Bien que les indicateurs comportementaux soient utiles, les indicateurs atomiques et calculés sont considérés par de nombreuses organisations comme les éléments les plus précieux de la CTI. La principale justification de cette perception est liée au fait que les intelligences tactiques sont généralement exprimées sous des formats lisibles par les machines, de sorte qu'ils peuvent facilement être chargés dans des dispositifs de sécurité, fournissant ainsi des résultats immédiats. En revanche, les flux des intelligences opérationnels ou stratégiques nécessitent dans la plupart des cas un traitement manuel par un être humain.

CTI Tactique	CTI Opérationnelle
Nature déductive.	Nature descriptive et caractérisation du comportement anormal.
Davantage de fausses alertes (false positives), spécifique à une attaque.	Moins de fausses alertes (false positives), couvrant une famille d'attaques basée sur le modèle comportemental.
Réactive.	Proactive.

TABLE 2.1 – Comparaison entre la CTI tactique et opérationnelle [25].

2.3 Framework ATT&ck

2.3.1 Définition

Le framework "Adversarial Tactics, Techniques, and Common Knowledge", (ATT&CK), est une base de connaissances qui répertorie les tactiques et techniques des cyberattaquants, fondée sur des observations issues du monde réel [86]. Elle offre une vue complète, évolutive et structurée des TTPs utilisées par les adversaires, et sert de framework de référence pour la recherche, l'analyse et l'atténuation des menaces. ATT&CK n'est pas limité à un secteur ou une technologie spécifique ; il a été conçu pour fournir aux professionnels de la cybersécurité un langage commun permettant de comprendre les TTPs utilisées par les acteurs de la menace dans leurs attaques. En décomposant les attaques en éléments distincts, les équipes de cybersécurité peuvent identifier les méthodes employées par les attaquants dans leurs opérations et évaluer l'impact de leurs activités afin de développer des stratégies de défense adaptées.

ATT&CK est également utilisé par les fournisseurs de sécurité et les chercheurs pour tester et valider l'efficacité de leurs produits et technologies. Cette utilisation a stimulé l'innovation dans le domaine de la cybersécurité, conduisant au développement de nouveaux outils et technologies mieux adaptés à la défense contre les menaces modernes.

La base de connaissances ATT&CK a été développée en 2013 par la MITRE Corporation, en réponse à la nécessité croissante pour un langage commun permettant de comprendre les TTPs utilisées par les adversaires. Le framework était initialement axé sur les groupes Advanced Persistent Threat (APT), mais il a depuis évolué pour couvrir un éventail plus large d'acteurs de la menace et de scénarios d'attaques.

2.3.2 Matrice ATT&ck

Le framework ATT&CK est composé d'un ensemble de techniques et de sous-techniques qui représentent les actions que les adversaires peuvent entreprendre pour

atteindre leurs objectifs. Ces objectifs sont regroupés sous des tactiques, auxquelles les techniques et sous-techniques sont associées. Cette représentation relativement simple établit un équilibre utile entre un niveau de détail technique suffisant (au niveau des techniques) et le contexte dans lequel ces actions se produisent (au niveau des tactiques).

La relation entre les tactiques, les techniques et les sous-techniques est visualisée dans la matrice ATT&CK. Par exemple, sous la tactique de Persistance (qui représente l'objectif de l'adversaire de maintenir une présence dans l'environnement cible), on trouve diverses techniques, telles que Hijack Execution Flow, Pre-OS Boot, Scheduled Task/Job, etc. Chacune de ces techniques représente une méthode spécifique que les adversaires peuvent utiliser pour atteindre l'objectif de persistance.

2.3.3 Évolution du framework ATT&ck

Au fil du temps, le framework ATT&CK a évolué pour suivre les menaces affectant diverses infrastructures. Différentes versions ont été développées pour les entreprises, les appareils mobiles et les systèmes de contrôle industriels.

ATT&CK pour les entreprises. Cette version est axée sur les techniques utilisées contre les systèmes Windows, Linux, Active Directory, les infrastructures cloud et d'autres environnements d'entreprise courants. Basé sur des observations du monde réel, ce framework propose une matrice qui associe les techniques aux tactiques spécifiques à ces attaques. Cela permet aux organisations d'évaluer leur posture de sécurité, de hiérarchiser leurs défenses et de mieux comprendre les risques auxquels elles sont confrontées.

ATT&CK pour les appareils mobiles. Cette version se concentre sur les appareils mobiles et les écosystèmes de gestion d'applications qui leur sont associés. Elle inclut les techniques utilisées pour exploiter les vulnérabilités des appareils mobiles, telles que le phishing, l'ingénierie sociale, les attaques basées sur les applications, ainsi que celles ciblant l'écosystème des applications, telles que les attaques de type Man In The Disk (MITD).

ATT&CK pour les systèmes de contrôle industriels (ICSs). Cette version cible les réseaux de systèmes de contrôle utilisés dans les infrastructures critiques, tels que les automates programmables (PLCs) et les protocoles de communication spécifiques comme ModBus et DNP. Elle inclut des techniques pour exploiter les vulnérabilités des ICSs, telles que le spearphishing et les infections par logiciels malveillants, ainsi que des méthodes pour manipuler ces réseaux, comme les actions de contrôle non autorisées et la manipulation de données.

De nouvelles techniques et tactiques utilisées par les acteurs de la menace sont régulièrement ajoutées, et le framework ATT&CK est mis à jour en continu pour refléter les nouveaux scénarios d'attaque ciblant les environnements visés. La figure 2.5 illustre l'évolution des TTPs dans le framework ATT&CK pour les entreprises.

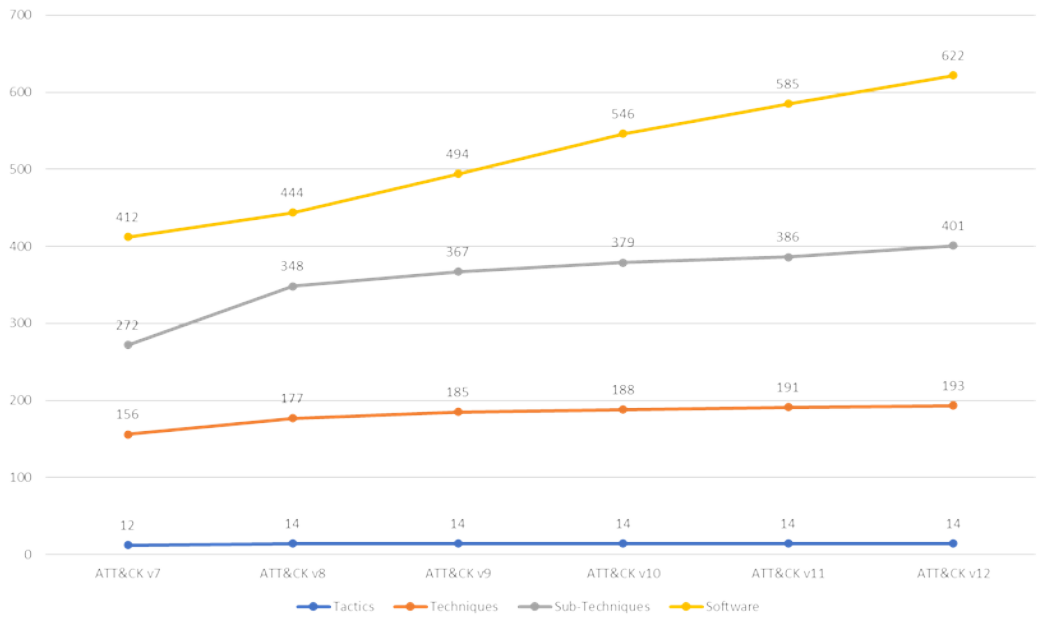


FIGURE 2.5 – Évolution du framework ATT&CK.

2.3.4 Framework ATT&ck pour la CTI

Selon [4], le framework ATT&CK est conçu pour servir le domaine de la CTI. Il peut être utilisé pour :

Émulation d'adversaires

L'émulation d'adversaires permet de tester la capacité d'un système de défense à détecter et atténuer l'activité malveillante à travers les différents points de l'organisation. ATT&CK peut être utilisé pour créer des scénarios d'attaque afin de tester et valider les défenses contre les techniques courantes des adversaires. Des profils spécifiques de groupes d'adversaires peuvent être élaborés à partir des informations documentées dans ATT&CK. Ces profils sont utiles aux équipes de défense pour aligner et renforcer les mesures défensives.

Enrichissement de la CTI

ATT&CK offre un environnement riche permettant de documenter les profils de groupes d'adversaires d'un point de vue comportemental, indépendamment des outils qu'ils utilisent. Les analystes peuvent ainsi mieux comprendre les comportements communs entre différents groupes et adapter leurs stratégies de défense en conséquence, en se posant des questions telles que : "Quelle est ma position défensive contre le groupe d'adversaires APT3 ?". En comprenant comment plusieurs groupes partagent des com-

portements techniques similaires, les analystes peuvent concentrer leurs efforts sur des défenses qui couvrent efficacement divers types de menaces. Le format structuré de ATT&CK enrichit les rapports de menace en classant les comportements au-delà des indicateurs standards. Par ailleurs, plusieurs groupes peuvent utiliser les mêmes techniques. Ainsi, il est déconseillé d'associer une activité à un groupe uniquement sur la base des techniques utilisées. L'attribution d'une activité à un groupe spécifique est un processus complexe qui intègre toutes les composantes du modèle en diamant, et pas seulement les TTPs.

Analyse comportementale

L'analyse comportementale va au-delà des IoCs traditionnels ou des signatures d'activité malveillante en identifiant une activité potentiellement malveillante au sein d'un système ou réseau, sans dépendre de la connaissance préalable des outils et indicateurs de l'adversaire. Elle permet de détecter des activités suspectes en s'appuyant sur la manière dont un adversaire interagit avec une plateforme spécifique, indépendamment des outils utilisés. ATT&CK peut être employé pour développer et tester l'analyse comportementale en vue de détecter des comportements adverses dans un environnement donné.

Évaluation d'un "SOC"

Le centre des opérations de sécurité (Security Operations Center (SOC)) d'une organisation est essentiel pour les moyennes et grandes entreprises qui surveillent en permanence les menaces actives sur leurs systèmes informatiques. Il est donc important de comprendre la maturité d'un SOC afin d'évaluer son efficacité. Ainsi, ATT&CK peut servir d'outil pour évaluer la capacité d'un SOC à détecter, analyser et répondre aux intrusions.

2.4 Open-Source Intelligence

L'Open-Source Intelligence (OSINT) est le processus de collecte d'informations provenant de sources accessibles au public pour éclairer la prise de décision. Ce domaine large englobe un éventail varié de sources d'information, telles que les médias sociaux, les articles de presse, les rapports gouvernementaux et les forums en ligne, entre autres. Dans le contexte de la CTI, OSINT permet de collecter des informations sur les menaces, les vulnérabilités et les attaques potentielles, ainsi que de surveiller les activités des acteurs malveillants et de suivre leurs TTPs.

En matière de CTI, une vaste gamme de sources de données est à la disposition des analystes pour extraire des informations précieuses sur les attaques modernes, notamment les IoCs traditionnels, les TTPs, et les outils utilisés par les adversaires. Ces informations permettent de renforcer les systèmes de défense. Les principales sources de données utilisées pour la CTI sont les suivantes :

— **Rapports APT**. Un rapport APT est une analyse approfondie d'un groupe APT spécifique, comprenant ses TTPs, ses motivations et ses cibles. Les APT représentent généralement des groupes de cybermenaces hautement qualifiés et dotés de ressources importantes, qui mènent des attaques ciblées et prolongées contre des organisations ou des individus spécifiques [33]. Les rapports APT contiennent généralement les informations suivantes :
  — Informations de base sur le groupe APT : ces informations incluent des détails sur les origines, l'historique et les affiliations connues du groupe.
  — Les TTPs du groupe APT : une analyse détaillée des tactiques, techniques et procédures du groupe, avec des informations sur les outils et méthodes utilisés pour accéder aux réseaux et systèmes cibles, maintenir leur présence et exfiltrer des données.
  — Motivations : les rapports peuvent également inclure des informations sur les motivations du groupe, qu'elles soient politiques, financières ou idéologiques.
  — Informations sur les cibles : ces rapports identifient les types d'organisations ou d'individus ciblés par le groupe, avec des détails sur des attaques spécifiques.
  — Recommandations d'atténuation : les rapports APT fournissent des recommandations pour aider les organisations à se protéger, comme l'implémentation de contrôles de sécurité spécifiques ou l'adoption de meilleures pratiques.
— **Rapports d'incident.** Un rapport d'incident, dans le domaine de la CTI, est un document qui fournit un compte rendu détaillé d'un incident ou d'une violation de la sécurité survenue au sein des réseaux, des systèmes ou des données d'une organisation. Ce rapport inclut généralement un résumé de l'incident, une description des actifs et systèmes affectés, une chronologie des événements ayant conduit à l'incident et de ceux s'étant déroulés pendant celui-ci, ainsi qu'une analyse de l'impact de l'incident sur l'organisation. Il détaille également les mesures prises pour atténuer l'incident et empêcher la récurrence de menaces similaires. Les rapports d'incidents proviennent de FireEye, Kaspersky, MISP, etc. Ils constituent un outil essentiel pour protéger les actifs et les données d'une organisation, aider à identifier et répondre aux menaces, et favoriser une amélioration continue des pratiques de sécurité.

  À titre d'exemple, la Figure 2.6 illustre un extrait d'un rapport d'incident publié par FireEye. On y trouve une analyse fine du comportement d'un malware, avec la mise en évidence explicite des tactiques et techniques employées par les attaquants selon le cadre MITRE ATT&CK. Par exemple, l'analyse montre que le malware établit une connexion SOCKS5 sur un port TCP spécifique après avoir exécuté des commandes système avec élévation de privilèges. Ce type de rapport fournit une vue comportementale complète de l'attaque, essentielle pour le renseignement sur les menaces. Il constitue un parfait exemple d'utilisation de données OSINT pour comprendre les méthodes d'intrusion et adapter les mécanismes de défense en conséquence.

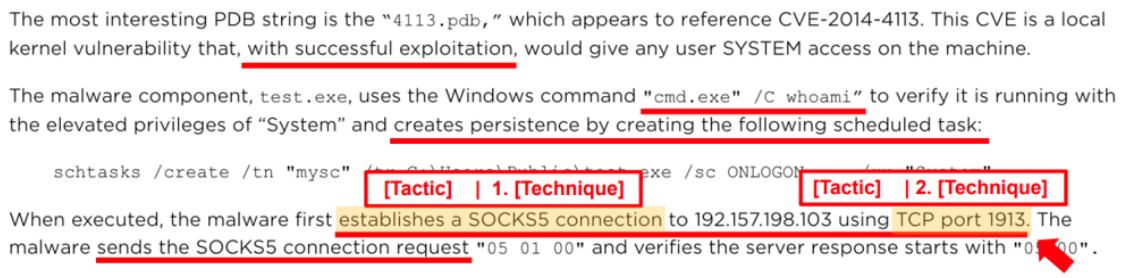


FIGURE 2.6 – Exemple d'extrait d'un rapport d'incident de FireEye illustrant les tactiques et techniques MITRE ATT&CK utilisées par un malware.

— **Réseaux sociaux.** Les réseaux sociaux peuvent constituer une source précieuse de renseignements pour la CTI. Ces plateformes permettent de collecter des informations en temps réel, de surveiller les tendances et d'observer les discussions autour des menaces, des vulnérabilités et des incidents de sécurité. En raison de leur caractère public et de leur rapidité de diffusion, les réseaux sociaux offrent aux analystes la possibilité de détecter des informations pertinentes pour la cybersécurité bien avant qu'elles ne soient rapportées dans des sources traditionnelles. Twitter, en particulier, peut être utilisé comme une source d'information précieuse [16]. Cette plateforme de médias sociaux permet aux utilisateurs de partager des informations en temps réel, incluant les dernières nouvelles sur les logiciels malveillants, les menaces émergentes et les vulnérabilités en matière de sécurité. Les analystes de cybersécurité peuvent utiliser Twitter pour surveiller les tendances et les discussions liées à la cybersécurité, suivre les activités des adversaires et recueillir des informations sur les attaques et les vulnérabilités potentielles.

2.5 Conclusion

Ce chapitre a établi les notions de base de la CTI, mettant en lumière la nécessité de comprendre les cybermenaces dans un contexte de numérisation croissante. Nous avons abordé les composants clés de la CTI, tels que les IoCs, les TTPs, le framework ATT&CK et l'OSINT, et montré comment ils se combinent pour fournir une compréhension approfondie des menaces et élaborer des stratégies de défense efficaces. Ces notions sont essentielles pour comprendre l'ensemble de notre travail tout au long de cette thèse.

Chapitre 3

Techniques d'apprentissage automatique

3.1 Introduction

L'apprentissage automatique, ou Machine Learning (ML) en anglais, est un sous-domaine de l'Intelligence Artificielle (IA) qui permet aux ordinateurs d'apprendre à partir de données sans être explicitement programmés. Plus précisément, il repose sur la construction de modèles capables d'identifier des schémas et de prendre des décisions en se basant sur un ensemble de données d'entraînement. Dans ce processus, un algorithme analyse les données, en extrait des caractéristiques pertinentes, puis ajuste ses paramètres afin d'améliorer progressivement ses performances. Une fois entraîné, le modèle peut être appliqué à de nouvelles données pour effectuer des prédictions ou automatiser certaines tâches.

L'apprentissage automatique se divise principalement en trois catégories : l'apprentissage supervisé, l'apprentissage non supervisé et l'apprentissage par renforcement. Dans l'apprentissage supervisé, l'algorithme est entraîné sur des données étiquetées pour prédire les étiquettes de nouvelles données inconnues. En apprentissage non supervisé, l'algorithme apprend à partir de données non étiquetées pour identifier des structures ou des regroupements dans les données. L'apprentissage par renforcement, quant à lui, consiste à apprendre comment prendre des décisions optimales en interagissant avec un environnement qui récompense ou pénalise les actions en fonction de leurs résultats.

L'apprentissage profond, ou Deep Learning (DL), est un sous-domaine de l'apprentissage automatique qui utilise des réseaux neuronaux profonds pour traiter de gros volumes de données. Inspirés de la structure du cerveau humain, les réseaux neuronaux sont composés de couches multiples, permettant au modèle d'apprendre des représentations de données de plus en plus abstraites et complexes. L'un des principaux avantages de l'apprentissage profond est sa capacité à extraire automatiquement des caractéristiques pertinentes à partir de données brutes, éliminant ainsi le besoin d'ingénierie

manuelle des caractéristiques. Cela a conduit à des avancées significatives dans des domaines tels que la vision par ordinateur, la reconnaissance vocale et le traitement du langage naturel. Cependant, l'apprentissage profond nécessite souvent des ressources matérielles et computationnelles considérables, ainsi que de grandes quantités de données pour l'entraînement. En outre, ces modèles peuvent être difficiles à interpréter, ce qui complique la compréhension du raisonnement derrière les prédictions.

Dans ce chapitre, nous allons explorer les réseaux de neurones profonds et leurs différentes architectures, notamment les "Deep neural networks" (DNNs), "Convolutional Neural Networks" (CNNs), et "Recurrent Neural Networks" (RNNs), ainsi que les mécanismes d'attention, les transformeurs, et les grands modèles de langage (LLMs) tels que BERT. Nous expliquerons le fonctionnement de ces modèles et leurs applications dans des domaines tels que la vision artificielle, le traitement du langage naturel, et bien d'autres.

Les concepts présentés dans ce chapitre s'appuient sur les références suivantes : [48, 3, 56, 71].

3.2 Réseaux de neurones denses

Les réseaux de neurones denses (DNNs) sont un type de réseau neuronal artificiel dans lequel chaque neurone d'une couche est connecté à tous les neurones de la couche précédente et de la couche suivante. Ce type de réseau est conçu pour résoudre des problèmes d'apprentissage supervisé, tels que la classification ou la régression, et il se compose d'une couche d'entrée, de N couches cachées, et d'une couche de sortie.

L'optimisation d'un réseau neuronal est une tâche cruciale en apprentissage automatique, car elle affecte directement les performances du modèle. L'objectif principal de l'optimisation est de trouver l'ensemble de poids et de biais qui minimise une fonction de perte, mesurant l'écart entre les sorties prédites et les sorties réelles. Les poids et les biais, qui sont les paramètres qui déterminent comment les entrées sont transformées en sorties, sont ajustés au cours du processus d'entraînement à l'aide d'algorithmes d'optimisation.

Les poids contrôlent la force des connexions entre les neurones du réseau. Chaque neurone reçoit des entrées des couches précédentes, qui sont multipliées par les poids pour calculer l'activation du neurone via une fonction d'activation. Les biais, quant à eux, déterminent le seuil d'activation du neurone. Une valeur de biais est ajoutée à la somme des entrées pondérées avant l'application de la fonction d'activation. Les biais permettent au réseau d'apprendre différentes limites de décision pour chaque neurone et sont également ajustés au cours de l'entraînement.

3.2.1 Entraînement d'un réseau de neurone dense

Le processus d'entraînement d'un réseau de neurones dense illustré dans la figure 3.1, comprend généralement les étapes suivantes :

1. Initialisation des poids du réseau : les poids du réseau sont initialisés à des valeurs aléatoires ou selon une méthode spécifique pour optimiser la convergence.
2. Propagation avant (forward pass) : les données d'entrée sont propagées à travers le réseau pour produire une sortie prédite.
3. Rétropropagation du gradient de l'erreur (backpropagation) : le gradient de l'erreur est calculé à partir de la différence entre la sortie prédite et la sortie réelle, puis propagé en arrière à travers le réseau.
4. Mise à jour des poids et des biais du réseau : les poids et les biais sont ajustés en fonction des gradients calculés pour minimiser l'erreur, en utilisant un algorithme d'optimisation tel que la descente de gradient.

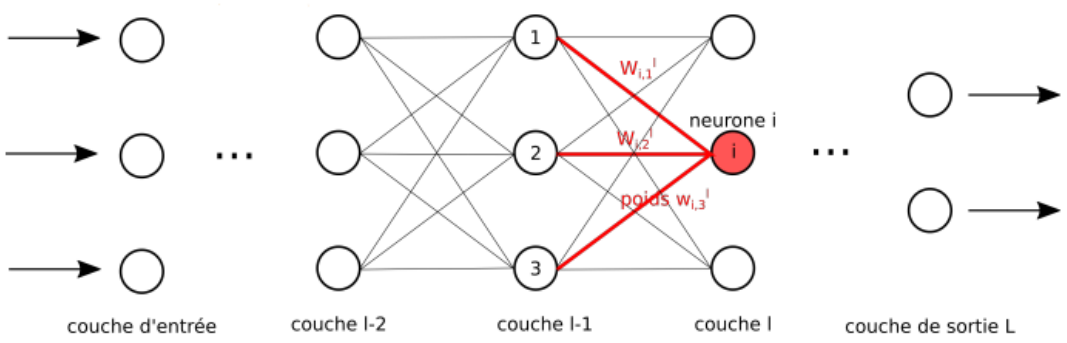


FIGURE 3.1 – Illustration des poids et des biais des connexions synaptiques entre un neurone de la couche l et un neurone de la couche précédente $(l-1)$ [34].

Initialisation des poids et des biais

Les poids et les biais d'un réseau de neurones représentent l'ensemble des paramètres des connexions synaptiques à travers toutes les couches du réseau. Une connexion synaptique correspond à la liaison entre deux neurones artificiels (également appelés unités), par analogie avec les structures observées en neurobiologie.

En termes de notation, considérons le neurone numéro 2 de la couche l. Le poids w correspondant à sa connexion avec le neurone numéro 1 de la couche précédente $(l-1)$ est noté $w^l_{2,1}$. De manière plus générale, $w^l_{i,j}$ désigne le poids reliant le neurone i de la couche l au neurone j de la couche $(l-1)$.

La somme P^l_i des poids associés au neurone i de la couche l est la somme de tous les poids des connexions synaptiques entre les neurones de la couche $(l-1)$ et ce neurone i. Cela s'exprime par l'équation suivante :

$$P^l_i = \sum_j w^l_{i,j} \tag{3.1}$$

L'initialisation des poids et des biais est une étape cruciale de la phase d'apprentissage. Un choix judicieux des poids initiaux peut déterminer si un réseau converge rapidement ou non. Une mauvaise initialisation peut entraîner une non-convergence, malgré un grand nombre d'itérations, et prolonger l'entraînement pendant des jours, voire des semaines. Pour remédier à ce problème, plusieurs techniques d'initialisation ont été développées, notamment la méthode de Xavier/Glorot, qui vise à améliorer la convergence du réseau.

Propagation avant (Forward pass)

Pour bien comprendre le principe d'un réseau de neurones dense, considérons l'exemple d'un perceptron multicouche, comme illustré dans la figure 3.1, qui est un réseau à propagation avant (feedforward neural network). Lors de cette étape, l'information se déplace uniquement dans une direction, de l'entrée vers la sortie, en passant par les couches cachées.

Les notations utilisées sont les suivantes :

— L : nombre total de couches du réseau de neurones.
— σ : fonction d'activation.
— $w_{i,j}^l$: poids reliant le neurone i de la couche l au neurone j de la couche précédente $(l-1)$.
— w^l : matrice des poids de la couche l, de dimensions $i \times j$, où i est le nombre de neurones de la couche l et j celui de la couche $(l-1)$.
— b_i^l : biais associé au neurone i de la couche l.
— b^l : vecteur des biais de la couche l.

La sortie du neurone i de la couche l est calculée à partir des sorties des neurones de la couche précédente $(l-1)$, des poids des connexions synaptiques correspondantes, et du biais du neurone.

Le processus de propagation avant implique deux étapes principales : l'agrégation des entrées et l'application de la fonction d'activation.

— z_i^l : valeur agrégée du neurone i de la couche l, calculée comme suit :

$$z_i^l = \sum_{j=1}^{n} w_{i,j}^l a_j^{l-1} + b_i^l \tag{3.2}$$

où a_j^{l-1} représente la sortie du neurone j de la couche $(l-1)$.

— a_i^l : valeur d'activation du neurone i de la couche l, obtenue en appliquant la fonction d'activation σ à la valeur agrégée z_i^l :

$$a_i^l = \sigma(z_i^l) \tag{3.3}$$

Ce processus est répété pour chaque neurone à travers les différentes couches du réseau, jusqu'à obtenir la sortie finale.

Rétropropagation (Backpropagation)

L'objectif de l'apprentissage est de trouver une configuration optimale des poids et des biais qui minimise l'erreur de prédiction. La rétropropagation du gradient permet de calculer l'erreur pour chaque neurone, de la couche de sortie vers la couche d'entrée, en ajustant les paramètres du réseau grâce à l'algorithme de la descente de gradient.

La rétropropagation se déroule en deux phases principales :

1. Calcul de l'erreur δ^{sortie} : cette étape consiste à comparer la sortie du réseau avec la valeur réelle (cible) et à calculer l'erreur correspondante pour la couche de sortie.
2. Propagation de l'erreur en arrière : l'erreur est ensuite propagée de couche en couche vers l'arrière, permettant de calculer les gradients pour ajuster les poids et les biais à chaque étape.

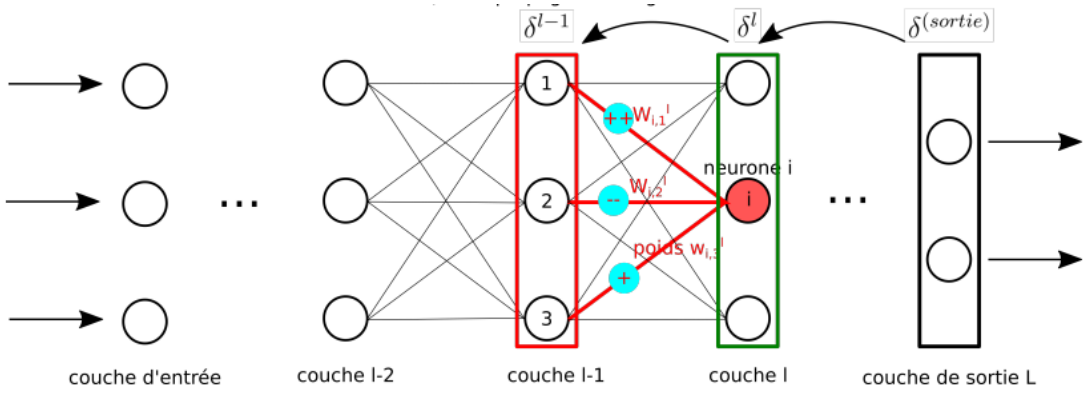


FIGURE 3.2 – Rétropropagation du gradient de l'erreur depuis la couche de sortie jusqu'aux poids des connexions synaptiques du neurone i de la couche l, issus du neurone j de la couche précédente $(l-1)$ [34].

Calcul de l'erreur pour la couche de sortie

Pour mesurer l'erreur de prédiction, on utilise une fonction appelée fonction de coût, fonction de perte ou fonction d'erreur, notée C. Elle est définie en fonction du type de problème à résoudre (régression ou classification). Grâce à l'algorithme de rétropropagation du gradient, nous pouvons calculer l'erreur individuelle $\frac{\partial C}{\partial w_{i,j}^{l}}$ et $\frac{\partial C}{\partial b_{i}^{l}}$ pour chaque poids et biais du réseau, et ainsi évaluer leur contribution à l'erreur finale δ^{sortie}.

Pour les poids de la couche de sortie L, l'équation suivante s'applique :

$$\frac{\partial C}{\partial w_{i,j}^{L}} = \frac{\partial z_{i}^{L}}{\partial w_{i,j}^{L}} \frac{\partial a_{i}^{L}}{\partial z_{i}^{L}} \frac{\partial C}{\partial a_{i}^{L}} \tag{3.4}$$

Chaque terme de cette équation peut être détaillé comme suit :

— $\frac{\partial C}{\partial a_i^L}$: la variation de la fonction de coût en fonction de la sortie du neurone, soit la dérivée de la fonction de coût par rapport à la sortie.
— $\frac{\partial a_i^L}{\partial z_i^L}$: la dérivée de la fonction d'activation par rapport à la valeur agrégée z_i^L.
— $\frac{\partial z_i^L}{\partial w_{i,j}^L}$: la dérivée de la fonction d'agrégation par rapport au poids $w_{i,j}^L$.

Rétropropagation de l'erreur

La variation de la fonction de coût par rapport à la sortie du neurone i de la couche l est exprimée comme :

$$\frac{\partial C}{\partial a_i^l} = \sum_k \frac{\partial z_k^L}{\partial a_i^l} \frac{\partial C}{\partial z_k^L} \tag{3.5}$$

Avec les relations suivantes :
— $\frac{\partial C}{\partial z_k^L} = \delta_k^L$, où δ_k^L représente l'erreur pour le neurone k dans la couche L.
— $\frac{\partial z_k^L}{\partial a_i^l} = w_{k,i}^L$, où $w_{k,i}^L$ est le poids reliant le neurone i de la couche l au neurone k de la couche L.

Mise à jour des poids et des biais

L'entraînement d'un réseau de neurones consiste à mettre à jour de manière itérative les poids $w_{i,j}^l$ et les biais b_i^l jusqu'à ce que la fonction de perte $C(w, b)$ converge vers un minimum. Pour cela, on utilise l'algorithme de descente de gradient.

Principe de l'algorithme de descente de gradient L'algorithme de descente de gradient repose sur les gradients des poids et des biais obtenus par rétropropagation pour les mettre à jour. L'objectif est de minimiser la fonction de coût $C(X)$ en ajustant les paramètres X de manière itérative, jusqu'à atteindre un point où le gradient de la fonction de coût est nul, ce qui correspond à un minimum local ou global. Cela permet de stabiliser la configuration du réseau neuronal.

Après chaque itération, les erreurs sont corrigées proportionnellement à l'importance de la contribution des neurones à l'erreur totale. Les poids synaptiques ayant un impact significatif sur l'erreur sont ajustés plus fortement que ceux ayant un impact marginal. Les mises à jour des poids se font selon l'équation suivante :

$$w_{i,j}^l(t+1) \leftarrow w_{i,j}^l(t) - \alpha \frac{\partial C}{\partial w_{i,j}^l(t)} \tag{3.6}$$

où α est le taux d'apprentissage, un hyperparamètre compris entre 0 et 1, déterminant la taille des pas de mise à jour.

Le calcul de la descente de gradient peut être effectué de plusieurs manières selon la taille des données utilisées à chaque itération :
— Gradient global (*batch gradient*) : toutes les données sont utilisées en une seule fois pour calculer le gradient, et les poids et biais sont ajustés après chaque itération.

— Gradient par lots (*mini-batch gradient*) : les données sont divisées en petits lots, et le gradient est calculé pour chaque lot. Les erreurs moyennes des lots sont ensuite utilisées pour mettre à jour les poids et biais.
— Gradient stochastique (*stochastic gradient*) : chaque donnée est utilisée individuellement pour calculer le gradient, et les poids sont ajustés après chaque donnée.

En pratique, le gradient par lots (*mini-batch gradient*) est souvent préféré car il combine les avantages des deux autres méthodes. Il offre une meilleure convergence que le gradient stochastique, tout en évitant les limitations de mémoire associées au gradient global.

L'entraînement comprend plusieurs cycles d'ajustement, appelés "époques", où les données d'entrée sont transmises au réseau par lots. À chaque époque, le modèle est évalué sur un ensemble de validation pour éviter le surapprentissage (*overfitting*). Une fois l'entraînement terminé, le modèle est testé sur un ensemble de test pour évaluer ses performances sur des données inédites.

3.2.2 Fonctions d'activation

En neurobiologie, lorsqu'un neurone atteint un seuil critique de potentiel électrique, il génère un potentiel d'action qui se propage le long de l'axone, déclenchant ainsi la transmission d'un message nerveux vers d'autres neurones. Cette dynamique biologique inspire directement la notion de fonction d'activation en apprentissage profond. Une fonction d'activation décide, en effectuant une transformation linéaire ou non linéaire des entrées, si un neurone doit transmettre ou non un signal, selon la valeur de sortie obtenue.

Dans les réseaux de neurones profonds, les fonctions d'activation sont essentiellement non linéaires. Cette non-linéarité est cruciale, car elle permet de capturer des relations complexes entre les données, rendant possible la séparation de données non linéairement séparables. En l'absence de cette non-linéarité, l'application répétée de transformations linéaires ne ferait qu'ajouter des couches sans enrichir la capacité du modèle à résoudre des problèmes complexes.

Un autre avantage des fonctions d'activation non linéaires est leur capacité à gérer les valeurs extrêmes. Dans les réseaux neuronaux, les données traitées peuvent atteindre des amplitudes élevées. Si une fonction linéaire était utilisée, les valeurs transmises de couche en couche pourraient croître de manière incontrôlée, rendant les calculs plus lourds et instables. Les fonctions d'activation non linéaires atténuent cet effet en ramenant les valeurs de sortie à des échelles plus gérables, souvent en les normalisant sous forme de probabilités. Cela facilite non seulement les calculs, mais améliore aussi la convergence du réseau lors de l'entraînement.

Fonction sigmoïde

La fonction sigmoïde est définie par l'équation suivante :

$$\text{Sig}(x) = \frac{1}{1 + e^{-x}} \quad \text{où} \quad x \in \mathbb{R} \tag{3.7}$$

Pour des valeurs de x très grandes et positives, e^{-x} tend vers 0, ce qui fait que $f(x)$ approche 1. Inversement, pour des valeurs de x très grandes et négatives, e^{-x} tend vers l'infini, rendant $f(x)$ proche de 0. Seules les valeurs de x proches de 0 influencent significativement la variation de la sortie. La fonction sigmoïde est couramment utilisée dans les couches de sortie des réseaux neuronaux pour produire des probabilités comprises entre 0 et 1. Elle est également déployée dans les couches cachées pour introduire de la non-linéarité, permettant ainsi au réseau d'apprendre des modèles plus complexes. Un de ses avantages majeurs est la continuité et la douceur de sa courbe, facilitant l'apprentissage grâce aux algorithmes d'optimisation basés sur le gradient. Cependant, la fonction sigmoïde présente aussi un inconvénient notable : le problème de la disparition des gradients ("vanishing gradients"). Lorsque x prend des valeurs très grandes ou très petites, le gradient de la fonction devient extrêmement faible, ce qui peut ralentir considérablement la convergence du réseau lors de l'entraînement.

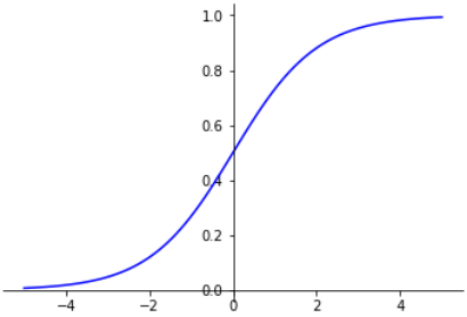


FIGURE 3.3 – Représentation de la fonction d'activation sigmoïde [50].

Tangente hyperbolique

La fonction d'activation tangente hyperbolique ($Tanh$) est définie par l'équation suivante :

$$\text{Tanh}(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{3.8}$$

Contrairement à la fonction sigmoïde, qui produit des valeurs de sortie entre 0 et 1, la fonction $Tanh$ mappe les valeurs d'entrée dans l'intervalle $[-1, 1]$. Cette caractéristique en fait une option populaire pour les couches cachées des réseaux neuronaux,

introduisant une non-linéarité essentielle pour modéliser des relations complexes dans les données. Un des principaux avantages de $Tanh$ par rapport à la sigmoïde est que sa sortie est centrée autour de zéro, ce qui facilite l'apprentissage en évitant les biais dans les mises à jour de poids. De plus, le gradient de la fonction $Tanh$ est plus important dans les régions proches de zéro, ce qui peut accélérer l'apprentissage pendant l'entraînement du réseau.

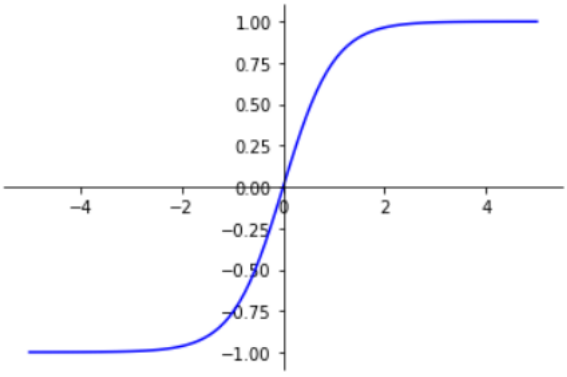


FIGURE 3.4 – Représentation de la fonction d'activation tangente hyperbolique [50].

Rectified Linear Unit (ReLU)

La fonction Rectified Linear Unit (ReLU), définie par l'équation suivante, est une fonction d'activation non linéaire largement utilisée dans les réseaux neuronaux artificiels :

$$\text{ReLU}(x) = \max(0, x) \tag{3.9}$$

La $ReLU$ transforme les valeurs d'entrée négatives en zéro, tout en conservant les valeurs positives inchangées. Cette simplicité contribue à sa popularité, en particulier dans les réseaux neuronaux de grande envergure. L'un des principaux avantages de $ReLU$ est sa capacité à atténuer le problème de la disparition des gradients, souvent rencontré avec des fonctions comme la sigmoïde ou la $Tanh$. En effet, $ReLU$ fournit un gradient constant pour les entrées positives, facilitant ainsi l'apprentissage des réseaux profonds. Cependant, il convient de noter que $ReLU$ peut souffrir du problème des neurones "morts" (dead neurons), où certains neurones peuvent cesser de s'activer pendant l'entraînement si leurs poids sont mis à jour de manière à toujours produire des sorties négatives.

Softmax

La fonction d'activation softmax est couramment utilisée dans la couche de sortie des réseaux neuronaux pour les problèmes de classification multi-classes. Elle trans-

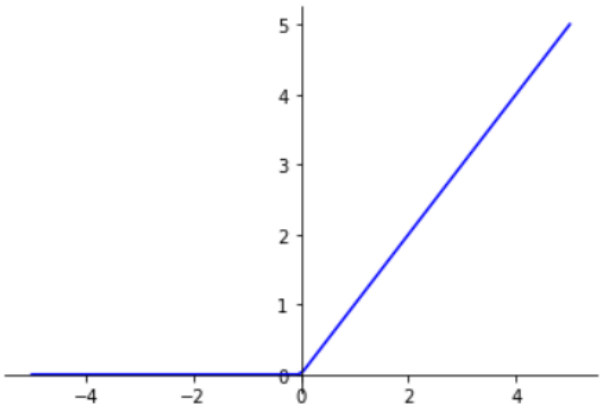


FIGURE 3.5 – Représentation de la fonction d'activation (ReLU) [50].

forme un vecteur d'entrées en une distribution de probabilité, chaque valeur de sortie représentant la probabilité que l'entrée appartienne à une classe particulière.

$$\sigma_k(u_k) = \frac{e^{u_k}}{\sum_{i=1}^{n} e^{u_i}} \tag{3.10}$$

La fonction softmax normalise les sorties de manière à ce que la somme de toutes les probabilités soit égale à 1. Cela permet d'interpréter chaque valeur de sortie comme une probabilité associée à une classe donnée. La classe avec la probabilité la plus élevée est celle prédite par le modèle. L'un des principaux atouts de softmax est qu'elle fournit un gradient continu et différentiable, ce qui facilite l'apprentissage à l'aide d'algorithmes d'optimisation comme la rétropropagation et la descente de gradient. Cette fonction est largement utilisée dans des domaines tels que la classification d'images, le traitement du langage naturel, et la reconnaissance vocale.

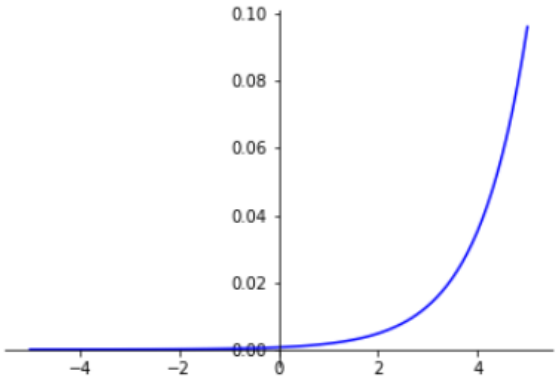


FIGURE 3.6 – Représentation de la fonction d'activation Softmax [50].

3.3 Réseaux de neurones convolutifs

Les réseaux de neurones convolutifs (CNNs) [67], sont une catégorie d'algorithmes d'apprentissage profond particulièrement adaptés aux tâches de vision par ordinateur, telles que la reconnaissance d'images, la détection d'objets, et la segmentation d'images. Plus généralement, sont conçus pour traiter des données structurées en tableaux multidimensionnels, ce qui leur permet de capturer efficacement les structures spatiales et les dépendances locales présentes dans les données.

3.3.1 Architecture d'un réseau de neurones convolutif

L'architecture d'un CNN est composée de plusieurs types de couches essentielles : les couches convolutives, les couches de regroupement (*pooling*), et les couches de sortie.

- Couches convolutives : ces couches appliquent des filtres ou noyaux convolutifs sur les données d'entrée pour générer des cartes de caractéristiques. Chaque filtre est une petite matrice glissant sur l'entrée, effectuant des multiplications élémentaires suivies d'une sommation. Ce processus permet d'extraire les motifs locaux et les dépendances spatiales des données, tout en réduisant leur dimensionnalité.
- Couches de regroupement (*Pooling*) : ces couches réduisent encore la dimensionnalité des cartes de caractéristiques en sous-échantillonnant les données. Le regroupement peut être effectué en prenant soit la valeur maximale (*max pooling*), soit la valeur moyenne (*average pooling*) sur une fenêtre glissante. Cela aide à réduire la sensibilité aux petites variations dans les données d'entrée, contribuant à l'invariance à la translation.
- Couches de sortie : la couche finale du CNN, souvent une couche entièrement connectée, prend les cartes de caractéristiques extraites et les utilise pour prédire la probabilité d'appartenance à différentes classes. Cette couche applique des poids optimisés, déterminés au cours de l'apprentissage, pour produire la distribution de probabilité sur les classes possibles.

Le processus d'apprentissage ajuste les poids des filtres et des couches denses en utilisant la rétropropagation pour minimiser l'erreur entre les prédictions du modèle et les sorties réelles.

Couches de convolution

La couche de convolution est généralement placée immédiatement après la couche d'entrée dans un réseau convolutif. Elle agit comme un extracteur de caractéristiques, appliquant des filtres convolutifs à l'entrée pour extraire des informations pertinentes qui seront transmises aux couches suivantes pour la prédiction des classes de l'entrée. La convolution consiste à faire glisser un noyau (ou filtre) sur l'entrée, en réalisant des opérations de multiplication et de sommation. Cette opération peut être effectuée de deux manières principales :

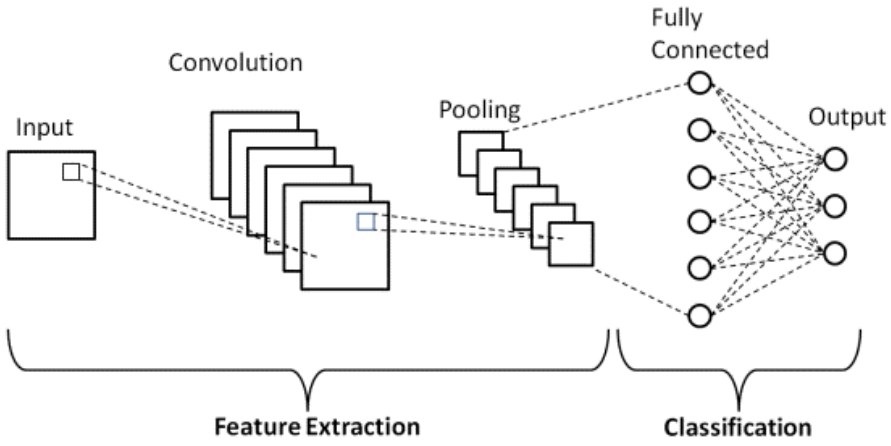


FIGURE 3.7 – Architecture typique de base d'un réseau de neurones convolutif [17].

1. Convolution non causale (souvent utilisée dans les CNNs classiques).
2. Convolution causale.

Convolution non causale - Corrélation croisée

La convolution non causale, également appelée corrélation croisée, est une opération où les échantillons futurs de l'entrée sont pris en compte lors de la convolution. Cela signifie que la sortie à un moment donné peut dépendre des échantillons futurs de l'entrée. Par exemple, si la sortie $y(0)$ dépend de l'entrée future $x(1)$, le système est considéré comme non causal. Considérons une entrée de longueur n, notée x, et un noyau de longueur k, noté h. La convolution non causale avec un décalage de s est définie comme suit :

$$y(n) = \begin{cases} \sum_{i=0}^{k} x(n+i)h(i), & \text{si } n = 0 \\ \sum_{i=0}^{k} x(n+i+(s-1))h(i), & \text{sinon} \end{cases} \tag{3.11}$$

Convolution causale

Dans les CNNs, la convolution causale est largement utilisée, en particulier pour les systèmes en temps réel, car elle ne prend en compte que les échantillons passés de l'entrée. Cela signifie que la sortie ne dépend que des données passées et présentes, rendant possible la prédiction en temps réel. Considérons une entrée de longueur n, notée x, et un noyau de longueur k, noté h. La convolution causale avec un décalage de s est définie comme suit :

$$y(n) = \begin{cases} \sum_{i=0}^{k} x(n-i)h(i), & \text{si } n = k-1 \\ \sum_{i=0}^{k} x(n-i+(s-1))h(i), & \text{sinon} \end{cases} \tag{3.12}$$

La convolution causale est souvent préférée dans les applications pratiques car elle permet une prédiction en temps réel et ne nécessite pas de connaître les données futures. Elle est également plus facile à implémenter dans des systèmes qui requièrent des calculs

en temps réel, car elle se base uniquement sur les données disponibles jusqu'à l'instant actuel.

Couches de regroupement (Pooling)

La couche de regroupement, ou pooling, est une méthode utilisée dans les CNNs pour réduire la dimensionnalité des caractéristiques extraites par les couches de convolution, ce qui diminue la complexité computationnelle du réseau. Elle est généralement placée entre les couches de convolution et les couches de classification (couches de sortie). Le regroupement aide également à rendre le réseau plus résilient aux variations mineures des données d'entrée. La technique de pooling la plus couramment utilisée est le "max pooling", qui consiste à sélectionner la valeur maximale dans une fenêtre de taille f. Cette fenêtre glisse sur l'entrée avec un pas de s après chaque opération de pooling. Une autre variante est le "average pooling", où la moyenne des valeurs dans la fenêtre est calculée.

Couches de sortie

Dans les CNNs, les couches de convolution et de pooling sont responsables de l'extraction des caractéristiques et de la réduction du nombre de paramètres. Cependant, pour produire une sortie finale, il est nécessaire d'utiliser des couches entièrement connectées, qui réalisent les tâches de classification ou de régression. Les couches de sortie appliquent une fonction de perte adaptée à la tâche en question (classification ou régression), permettant de calculer l'erreur de prédiction. Une fois la propagation avant terminée, la rétropropagation est utilisée pour ajuster les poids et les biais, minimisant ainsi l'erreur et la fonction de perte.

3.4 Réseaux de neurones récurrents

Les réseaux de neurones récurrents (Recurrent Neural Networks (RNNs)) [80], sont des architectures de réseaux neuronaux conçues pour capturer les dépendances et les motifs dans des séquences de données. Ces séquences peuvent inclure de l'écriture manuscrite, du texte ou des séries temporelles numériques. Les RNNs sont également adaptés aux images lorsqu'elles sont décomposées en une série de tâches, traitées comme des séquences. À un niveau plus avancé, les RNNs trouvent des applications dans la modélisation du langage, la génération de texte, la reconnaissance vocale, la description d'images ou l'annotation de vidéos.

Ce qui distingue les RNNs des réseaux neuronaux denses, ou perceptron multicouches, est la manière dont l'information est propagée. Alors que les réseaux denses transmettent l'information sans rétroaction, les RNNs utilisent des cycles pour réinjecter les informations dans le réseau. Cela leur permet de tenir compte des entrées précédentes $X_{0:t-1}$ en plus de l'entrée actuelle X_t. Les RNNs peuvent également comporter plusieurs couches cachées, organisées en blocs.

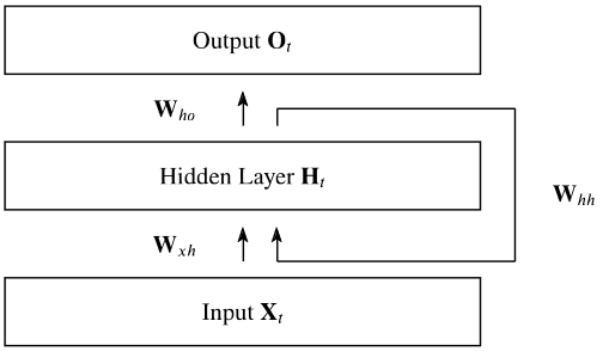


FIGURE 3.8 – Architecture d'un réseau de neurones récurrent [58].

Le processus de transmission d'information dans un RNNs repose sur l'état caché H_t et l'entrée X_t au temps t. Les notations sont les suivantes : $H_t \in \mathbb{R}^{n\times h}$ pour l'état caché, $X_t \in \mathbb{R}^{n\times d}$ pour l'entrée, où n est le nombre d'échantillons, d le nombre d'entrées par échantillon, et h le nombre d'unités cachées. Les matrices de poids sont $W_{xh} \in \mathbb{R}^{d\times h}$ pour les entrées vers les états cachés, et $W_{hh} \in \mathbb{R}^{h\times h}$ pour les états cachés vers eux-mêmes. Le vecteur de biais est $b_h \in \mathbb{R}^{1\times h}$. Ces paramètres sont appliqués à une fonction d'activation ϕ, souvent une sigmoïde, une fonction logistique ou une tangente hyperbolique.

Les équations suivantes représentent les calculs de l'état caché et de la sortie :

$$H_t = \phi_h(X_t W_{xh} + H_{t-1} W_{hh} + b_h) \tag{3.13}$$

$$O_t = \phi_o(H_t W_{ho} + b_o) \tag{3.14}$$

L'état caché H_t est influencé récursivement par H_{t-1}, ce qui permet au RNNs de capturer les dépendances séquentielles à travers tous les états précédents. Cela contraste avec les réseaux neuronaux denses, où les calculs sont définis par :

$$H = \phi_h(XW_{xh} + b_h) \tag{3.15}$$

$$O = \phi_o(XW_{ho} + b_o) \tag{3.16}$$

Ces équations soulignent la capacité unique des RNNs à traiter des données séquentielles en maintenant une mémoire des états précédents pour chaque calcul actuel.

Dans un réseau de neurones récurrents, l'algorithme Backpropagation through time (BPTT) (Backpropagation Through Time) est utilisé pour calculer les gradients de l'erreur par rapport aux poids et biais à chaque étape temporelle. Ce processus démarre à partir de la sortie du dernier instant et remonte chronologiquement. Cela est essentiel en raison des connexions récurrentes entre les unités cachées à différents moments, permettant aux RNNs de traiter des séquences d'entrées. Toutefois, cela implique que le gradient à chaque étape temporelle dépend de celui de l'étape précédente, nécessitant un calcul récursif des gradients.

L'un des défis majeurs des RNNs réside dans la capacité limitée à conserver les informations sur de longues périodes. Cela peut rendre difficile la capture des dépendances à long terme dans les données séquentielles. Ce problème, connu sous le nom de disparition et explosion des gradients, se manifeste particulièrement pendant la rétropropagation dans le temps (BPTT). La disparition des gradients survient lorsque les valeurs des gradients deviennent extrêmement faibles à mesure que la rétropropagation remonte le temps, entravant l'apprentissage des relations à long terme. À l'inverse, l'explosion des gradients se produit lorsque les valeurs des gradients augmentent de manière exponentielle, rendant les mises à jour des poids trop abruptes, ce qui peut mener à une instabilité du réseau et empêcher la convergence.

Pour surmonter ces défis, les réseaux récurrents à mémoire longue et courte durée (LSTMs) ont été développés. Ces réseaux sont spécialement conçus pour atténuer le problème de disparition des gradients, permettant ainsi aux RNNs d'apprendre efficacement sur de longues séquences.

3.4.1 Réseaux récurrents à mémoire longue et courte durée

Les LSTMs [23] sont une extension des RNNs classiques, capables de conserver des informations sur des intervalles de temps étendus en utilisant des mécanismes appelés "portes". Ces portes contrôlent le flux d'informations à chaque étape de temps, permettant aux LSTMs de stocker et de réguler les informations de manière efficace. Une cellule LSTMs dispose de trois portes principales : la porte de sortie O_t, la porte d'entrée I_t, et la porte d'oubli F_t. Ces portes sont définies par les équations suivantes :

$$O_t = \sigma(X_t W_{xo} + H_{t-1} W_{ho} + b_o) \tag{3.17}$$

$$I_t = \sigma(X_t W_{xi} + H_{t-1} W_{hi} + b_i) \tag{3.18}$$

$$F_t = \sigma(X_t W_{xf} + H_{t-1} W_{hf} + b_f) \tag{3.19}$$

Ici, W_{xi}, W_{xf}, $W_{xo} \in \mathbb{R}^{d \times h}$ et W_{hi}, W_{hf}, $W_{ho} \in \mathbb{R}^{h \times h}$ représentent les matrices de poids, tandis que b_i, b_f, $b_o \in \mathbb{R}^{1 \times h}$ sont les vecteurs de biais correspondants. La fonction sigmoïde σ est utilisée pour restreindre les sorties entre 0 et 1, permettant de réguler efficacement les flux d'informations.

Les cellules LSTMs contiennent également une mémoire candidate $\overline{C}_t \in \mathbb{R}^{n \times h}$, calculée avec une fonction d'activation $Tanh$ pour limiter les valeurs entre -1 et 1 :

$$\overline{C}_t = \tanh(X_t W_{xc} + H_{t-1} W_{hc} + b_c) \tag{3.20}$$

Les poids $W_{xc} \in \mathbb{R}^{d \times h}$, $W_{hc} \in \mathbb{R}^{h \times h}$ et les biais $b_c \in \mathbb{R}^{1 \times h}$ sont spécifiques à cette mémoire candidate.

Pour mettre à jour la mémoire de la cellule C_t, on combine l'ancien contenu C_{t-1} avec les nouvelles informations, contrôlées par les portes d'entrée et d'oubli :

$$C_t = F_t \odot C_{t-1} + I_t \odot \overline{C}_t \tag{3.21}$$

Enfin, l'état caché H_t est calculé à partir de la mémoire mise à jour, modulée par la porte de sortie :

$$H_t = O_t \odot \tanh(C_t) \tag{3.22}$$

Ces mécanismes permettent aux LSTMs de conserver une mémoire stable sur de longues séquences, surmontant ainsi les limitations des RNNs classiques et permettant des performances supérieures sur des tâches séquentielles complexes.

3.4.2 Réseau récurrent bidirectionnel à mémoire courte et longue durée

Les réseaux Bidirectional long-short term memory (Bi-LSTMs) sont une extension des réseaux LSTMs qui traitent les séquences d'entrée dans les deux directions : de manière avant (forward) et arrière (backward). Cette architecture permet de capturer les dépendances contextuelles passées et futures au sein d'une séquence d'entrée.

L'idée clé des Bi-LSTMs est d'intégrer à la fois le contexte précédent et suivant, ce qui est particulièrement bénéfique pour des tâches comme la traduction automatique, la reconnaissance vocale, et l'analyse des sentiments. En exploitant les informations des deux directions, le modèle est capable d'apprendre des dépendances complexes sur de longues séquences, augmentant ainsi sa capacité à produire des prédictions précises.

Dans un modèle Bi-LSTMs, chaque séquence d'entrée est traitée par deux couches LSTMs distinctes : l'une dans le sens direct et l'autre dans le sens inverse. À chaque étape temporelle, les sorties des deux directions sont combinées en concaténant les résultats des LSTMs avant et arrière. Cela crée une représentation enrichie de la séquence, incorporant le contexte des deux sens.

Cette représentation bidirectionnelle peut ensuite être utilisée pour des tâches de classification, de régression, ou encore d'autres traitements nécessitant une compréhension approfondie de la séquence entière. Le Bi-LSTMs s'avère ainsi particulièrement efficace pour les applications nécessitant une prise en compte globale du contexte séquentiel.

3.5 Mécanismes d'attention

Les mécanismes d'attention sont des techniques largement utilisées dans le domaine de l'apprentissage profond, notamment pour les tâches de modélisation de séquences et de NLP. L'idée centrale de l'attention est de permettre aux modèles de se concentrer sélectivement sur les parties les plus pertinentes de la séquence d'entrée pour une tâche donnée.

Une fonction d'attention peut être décrite comme une correspondance entre une requête et un ensemble de paires clé-valeur pour produire une sortie, où la requête, les clés, les valeurs et la sortie sont tous des vecteurs. La sortie est calculée comme

une somme pondérée des valeurs, où le poids attribué à chaque valeur est déterminé par une fonction de compatibilité entre la requête et la clé correspondante [90]. Étant donné un ensemble de paires clé-valeur (K, V) et une requête Q, les poids d'attention α sont calculés comme suit :

$$\alpha_i = \frac{\exp(\text{score}(Q, K_i))}{\sum_j \exp(\text{score}(Q, K_j))} \tag{3.23}$$

Où $\text{score}(Q, K_i)$ est une fonction mesurant la compatibilité entre la requête et la clé. Les fonctions de score couramment utilisées incluent le produit scalaire et l'attention additive / concaténative [84]. La sortie de la couche d'attention est ensuite une somme pondérée des valeurs :

$$\text{Attention}(Q, K, V) = \sum_i \alpha_i V_i \tag{3.24}$$

3.6 Transformeurs

Les transformeurs [90] sont une architecture de réseau neuronal conçue pour gérer les tâches de séquence-à-séquence, telles que la traduction automatique et le NLP. Ils se distinguent par leur capacité à traiter simultanément toutes les entrées d'une séquence, grâce à l'utilisation intensive des mécanismes d'attention.

Au cœur des transformeurs se trouve le mécanisme d'attention, notamment l'attention multi-têtes (multi-head attention). Ce mécanisme permet au modèle de se concentrer sur différentes parties de la séquence d'entrée simultanément, en attribuant des poids aux éléments les plus pertinents pour chaque tâche spécifique. L'attention multi-têtes aide à capturer les relations complexes entre les mots d'une phrase, indépendamment de leur position relative. L'architecture des transformeurs est généralement divisée en deux parties principales : l'encodeur et le décodeur.

3.6.1 Encodeur

L'encodeur reçoit une séquence d'entrée et génère une représentation interne de celle-ci, souvent appelée vecteur d'encodage (embedding). Il est constitué de plusieurs couches d'attention multi-têtes et de couches entièrement connectées. L'objectif de l'encodeur est de capturer les caractéristiques contextuelles de la séquence d'entrée, sans tenir compte de la position des éléments dans la séquence.

3.6.2 Décodeur

Le décodeur utilise la sortie de l'encodeur pour générer la séquence de sortie. Il est également composé de couches d'attention multi-têtes et de couches entièrement connectées, mais intègre un mécanisme d'attention supplémentaire appelé "attention sur l'encodeur" (encoder-decoder attention). Cela permet au décodeur de se concentrer

TABLE 3.1 – Comparaison entre BERT-Base et BERT-Large.

Paramètre	**BERT-Base**	**BERT-Large**
Blocs de Transformeurs	12	24
Têtes d'Attention	12	16
Unités Cachées	768	1024
Total de Paramètres	~110 millions	~340 millions

sur différentes parties de la séquence d'entrée lors de la génération de la sortie. Le décodeur est souvent utilisé pour des tâches de traduction ou de génération de texte, où il doit produire une sortie mot par mot.

Les transformeurs sont devenus la base de nombreux modèles avancés en NLP, tels que BERT [29], GPT [72], et T5 [73], grâce à leur efficacité à capturer des dépendances à longue portée dans les données textuelles.

3.7 Représentations bidirectionnelles d'encodeurs à partir de transformeurs

Les Représentations bidirectionnelles d'encodeurs à partir de transformeurs (Bidirectional Encoder Representations from Transformers (BERT)) sont des représentations contextuelles pré-entraînées utilisant un modèle de Transformeurs bidirectionnel et un mécanisme d'attention multi-têtes [29]. BERT est pré-entraîné sur de vastes corpus de textes en s'appuyant sur deux tâches non supervisées principales. La première est la Modélisation de Langage Masqué (Masked LM ou MLM), où certains tokens d'entrée sont masqués aléatoirement, et le modèle doit prédire ces tokens masqués, ce qui lui permet de comprendre les sémantiques et les relations entre les mots. La seconde tâche est la Prédiction de la Prochaine Phrase (Next Sentence Prediction ou NSP), où le modèle apprend à prédire si deux phrases sont consécutives dans un texte donné, aidant ainsi à capturer les relations au niveau des phrases. En réalisant ces deux tâches, BERT intègre les forces de modèles tels que GPT [72] et ELMO [70] dans la compréhension du langage.

Depuis son introduction, BERT a inspiré de nombreux modèles pré-entraînés. Parmi ces modèles, les deux principales versions de BERT, à savoir BERT-Base et BERT-Large [29], diffèrent en taille et complexité, offrant ainsi une flexibilité en termes de demande computationnelle et de capacités de performance. BERT-Base, de taille relativement plus petite, est plus accessible pour des applications pratiques, tandis que BERT-Large, étant plus vaste, est adapté aux tâches nécessitant une compréhension linguistique plus approfondie. Les statistiques de BERT-Base et BERT-Large sont illustrées dans le tableau 3.1.

Les modèles BERT peuvent servir de vecteurs de caractéristiques de base pour les phrases d'entrée et peuvent être ajustés pour des tâches en aval telles que la Recon-

naissance d'Entités Nommées. L'affinage consiste à ajouter des couches spécifiques à la tâche, généralement focalisées sur les couches supérieures de l'encodeur de transformeurs au sein du modèle BERT. Ces couches supérieures sont particulièrement efficaces pour capturer des représentations textuelles de haut niveau et peuvent être adaptées aux nuances de la tâche grâce à un entraînement avec des données spécifiques.

BERT possède diverses variantes et modèles pré-entraînés, tels que RoBERTa [54], une version optimisée de manière robuste avec des techniques d'entraînement améliorées. ALBERT [47], une version allégée qui réduit la taille du modèle grâce au partage des paramètres et aux embeddings factorisés. DistilBERT [77], une version distillée qui conserve 97% des performances de BERT tout en étant 60% plus rapide et plus petite.

3.8 Apprentissage par renforcement

L'apprentissage par renforcement est un domaine de l'apprentissage automatique où un agent apprend à prendre des décisions optimales en interagissant directement avec un environnement. Contrairement à l'apprentissage supervisé, qui dépend d'exemples étiquetés, et à l'apprentissage non supervisé, qui cherche à identifier des structures cachées dans les données, l'apprentissage par renforcement se concentre sur l'amélioration progressive des décisions à partir d'expériences répétées.

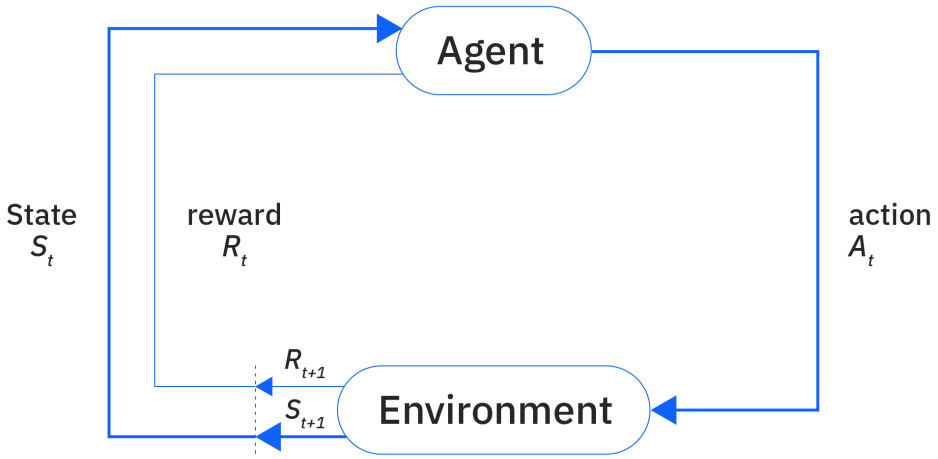


FIGURE 3.9 – Schéma de fonctionnement de l'apprentissage par renforcement. À chaque instant t, l'agent observe l'état S_t de l'environnement, choisit une action A_t et reçoit une récompense R_t. L'environnement évolue alors vers un nouvel état S_{t+1}, fournissant une nouvelle observation à l'agent. Ce cycle permet à l'agent d'apprendre une politique optimale au fil du temps.

Dans un système d'apprentissage par renforcement (Figure 3.9), plusieurs composantes fondamentales interviennent :

— **Agent :** c'est l'entité décisionnelle qui observe l'état de l'environnement et choisit des actions à réaliser.

— **Environnement :** il s'agit du contexte ou du système avec lequel l'agent interagit. L'environnement fournit des retours sous forme de nouveaux états et de récompenses après chaque action prise par l'agent.
— **Action :** une décision prise par l'agent en réponse à l'état actuel de l'environnement.
— **Récompense :** un retour numérique fourni par l'environnement indiquant à l'agent la qualité de l'action réalisée. Elle oriente l'apprentissage de l'agent vers des stratégies bénéfiques à long terme.

Le processus d'apprentissage repose sur l'interaction continue entre l'agent et l'environnement. À chaque étape, l'agent observe un état, effectue une action selon une politique de décision, reçoit une récompense et transite vers un nouvel état. L'objectif principal de l'agent est d'apprendre une politique optimale permettant de maximiser la somme cumulative des récompenses attendues au fil du temps.

3.9 Deep Q-Network (DQN)

Le Deep Q-Network, introduit par [62], constitue une avancée majeure combinant l'algorithme traditionnel Q-learning avec les réseaux de neurones profonds. Contrairement au Q-learning classique, qui utilise une table Q pour stocker les valeurs des états-actions, le DQN généralise cette idée en utilisant un réseau neuronal pour approximer la fonction Q :

$$Q(s, a) \approx Q(s, a; \theta) \tag{3.25}$$

où s désigne l'état courant, a l'action envisagée, et θ représente les paramètres ajustables du réseau neuronal.

Les composantes essentielles du DQN sont :

— **Réseau Q principal :** le réseau neuronal qui est entraîné continuellement pour prédire les valeurs Q des paires état-action.
— **Réseau Q cible (Target Q-Network) :** une copie périodiquement mise à jour du réseau principal, utilisée pour calculer les valeurs cibles durant l'apprentissage afin de stabiliser le processus d'entraînement.
— **Mémoire de rappel (Experience Replay) :** une mémoire qui conserve les expériences passées sous forme de tuples (s_t, a_t, r_t, s_{t+1}). L'agent prélève aléatoirement des mini-lots d'expériences de cette mémoire pour entraîner le réseau Q, réduisant ainsi la corrélation et améliorant la stabilité de l'apprentissage.

Le choix des actions par l'agent se fait généralement selon une stratégie dite ϵ-greedy, équilibrant exploration (choisir une action aléatoire avec probabilité ϵ) et exploitation (choisir la meilleure action connue avec probabilité $1 - \epsilon$).

Grâce à ces innovations, le DQN a prouvé sa capacité à maîtriser des tâches complexes à partir d'entrées sensorielles brutes, notamment en surpassant des performances humaines sur divers jeux Atari 2600.

3.9.1 Algorithme d'entraînement du DQN

L'entraînement du DQN se déroule en plusieurs épisodes, où chaque épisode consiste en une séquence d'interactions de l'agent avec l'environnement jusqu'à atteindre un état terminal. À chaque pas, l'expérience vécue est stockée dans la mémoire de rappel.

La fonction de perte utilisée pour l'entraînement du réseau principal est généralement l'erreur quadratique moyenne entre la valeur Q prédite et la cible calculée à partir du réseau Q cible :

$$\text{Loss}(\theta) = \mathbb{E}(s_t, a_t, r_t, st+1)\left[\left(r_t + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-) - Q(s_t, a_t; \theta)\right)^2\right] \tag{3.26}$$

avec γ représentant le facteur d'actualisation, et θ^- les paramètres du réseau cible. À chaque étape d'entraînement :

1. Un mini-lot d'expériences est prélevé aléatoirement depuis la mémoire.
2. La fonction de perte est calculée.
3. Les paramètres du réseau Q principal sont mis à jour par rétropropagation afin de minimiser cette perte.

Ce processus est répété à travers de nombreux épisodes, permettant progressivement à l'agent d'apprendre une politique optimale.

3.10 Conclusion

Ce chapitre a présenté les concepts fondamentaux de l'apprentissage apprentissage, en soulignant leur rôle crucial dans le développement de modèles capables d'apprendre à partir de données sans nécessiter de programmation explicite. Nous avons exploré les différentes formes d'apprentissage automatique, notamment l'apprentissage supervisé, tout en mettant l'accent sur l'apprentissage profond.

L'apprentissage profond a été illustré à travers l'étude des réseaux de neurones profonds, tels que les réseaux de neurones denses, convolutifs (CNNs) et récurrents (RNNs). De plus, nous avons examiné les mécanismes d'attention, les transformeurs, notamment le modèle pré-entraîné BERT, pour montrer comment ces approches permettent de traiter et d'analyser de vastes ensembles de données, en identifiant des motifs complexes et en améliorant la précision des prédictions.

Chapitre 4

Cyber threat intelligence et techniques d'apprentissage automatique : état de l'art

4.1 Introduction

La cybersécurité est devenue un enjeu majeur dans le monde numérique actuel, où les menaces évoluent rapidement en complexité et en sophistication. Les organisations sont confrontées à un paysage de menaces en constante mutation, ce qui rend indispensable l'acquisition et l'exploitation efficaces de la CTI. Dans le contexte de notre thèse, une compréhension approfondie des techniques de collecte, d'extraction, d'attribution et d'estimation de la gravité des cybermenaces est essentielle pour mettre en évidence les points forts et les faiblesses des travaux actuels par rapport à nos contributions.

Ce chapitre présente un état de l'art des recherches actuelles dans le domaine de la CTI. Nous commençons par une synthèse des articles clés, en mettant l'accent sur quatre domaines principaux :

1. Techniques de collecte des informations liées à la CTI : nous explorons les méthodes employées pour recueillir des données pertinentes à partir de diverses sources, telles que les réseaux sociaux, les forums, et les rapports de sécurité.
2. Techniques d'extraction des IoCs : nous analysons les approches utilisées pour extraire automatiquement les IoCs à partir de données non structurées, en abordant les défis liés au traitement du langage naturel et à la reconnaissance des entités nommées.
3. Techniques d'attribution des cyberattaques : nous examinons les méthodes visant à identifier les acteurs derrière les cyberattaques, en discutant des approches basées sur l'analyse des rapports textuels et des logiciels malveillants.
4. Techniques de prédiction de la gravité des cybermenaces : nous étudions les modèles et les algorithmes permettant d'évaluer la gravité des menaces, afin de prioriser efficacement les efforts de remédiation.

Après cette synthèse, nous proposons une analyse critique de ces articles, en soulignant les forces et les faiblesses de chaque approche. Cette analyse vise à identifier les lacunes existantes et à suggérer des pistes pour de futures recherches, contribuant ainsi à l'amélioration des stratégies de défense en cybersécurité.

4.2 Techniques de collection des informations liées à la CTI

Dans l'article [12], les auteurs présentent une nouvelle approche d'extraction d'informations textuelles utilisant l'apprentissage automatique non supervisé pour détecter les événements liés à la cybermenace sur Twitter. Les principales contributions de ce travail sont : la détection ainsi que le type ; la classification des événements de cybermenace en fonction d'un score d'importance obtenu en extrayant les termes de tweets caractérisés comme des entités, des mots-clés, ou les deux ; l'attribution d'un score pondéré aux expressions nominales en fonction de l'influence de l'utilisateur et du score de l'événement correspondant pour les entités et les mots-clés. Pour mettre en œuvre cette approche, les auteurs ont suivi les étapes suivantes : 1) collecte et annotation précoce des tweets, y compris le prétraitement et le nettoyage des tweets ; 2) mesure de l'impact des utilisateurs influents sur Twitter : le nombre de followers est directement proportionnel à l'influence de l'utilisateur ; 3) vectorisation des mots : les corps des tweets sont vectorisés en utilisant l'algorithme TF-IDF ; 4) regroupement des tweets : l'algorithme de regroupement DBSCAN est appliqué à la matrice des tweets vectorisés pour regrouper les tweets de sens similaire ; 5) détection et notation d'événements : le contenu textuel concaténé de chaque groupe est classé en utilisant l'algorithme TextRank. Chaque groupe est classé en tant qu'événement ou non et noté selon un ensemble de règles , puis leur score correspondant est utilisé pour classer les événements liés à la cybersécurité. Il convient de noter que l'algorithme de regroupement a été entraîné et validé sur des données provenant de Twitter.

L'article [28], traite le problème d'extraction d'informations CTI à partir de forums de hackers en ligne, qui contiennent des informations sur diverses menaces telles que, les serveurs proxy malveillants, les logiciels malveillants, en utilisant une source comme Darknet. L'approche applique un modèle hybride d'apprentissage automatique qui comprend : la recherche automatique à travers les forums de hackers, l'identification des messages les plus pertinents pour la cybersécurité, ensuite le regroupement de ces messages pertinents en attribuant des sujets (classes) que les hackers discutent. La première étape utilise les SVM pour la classification binaire des messages (pertinent, non pertinent), et la deuxième étape utilise l'algorithme LDA pour le regroupement de sujets. Cela permettra aux équipes de sécurité de se concentrer sur les messages les plus susceptibles de fournir des CTI. L'approche a été mise en œuvre en suivant les étapes suivantes : 1) Obtention des données à partir de forums de hackers ; 2) Prétraitement des données en supprimant les balises HTML, les mots vides, en convertissant le texte en minuscules, etc ; 3) Entraîner un classificateur SVM avec 16 000 messages répartis

entre pertinents et non pertinents ; 4) Les messages sélectionnés par le modèle (pertinents) sont ensuite analysés par l'algorithme LDA pour les regrouper sur 10 différents sujets.

L'article [51] se concentre sur l'évaluation des communautés et d'utilisateurs sur Twitter pour prioriser les informations CTI, en attribuant un score d'expertise à chaque utilisateur et communauté qui produit des tweets liés à la CTI. Ce framework se compose de quatre composants. Le premier composant est un connecteur de médias sociaux qui se connecte et recueille des données à partir de la plateforme Twitter. Le deuxième composant est un module pour identifier et étendre la liste des experts afin de trouver des sujets émergents. La contribution de poids et le calcul de la pertinence sont désignés comme le troisième composant pour expliquer le processus de chaque expert ciblé, ainsi que la manière d'extraire des informations de sécurité précieuses à partir des tweets publiés par les experts. Enfin, explorer puis exploiter les informations de la communauté d'experts qui permettent de reconnaître les menaces émergentes avec un algorithme de classification de sujets basé sur LDA. Cette méthode dépend fortement de l'identification des experts et de l'extraction d'informations auprès d'eux. Ainsi, cela peut conduire à des erreurs de notification si les experts ne sont pas réellement des experts et si les indications de menace ne sont pas suffisamment référencées par les experts.

Dans l'article [60], les auteurs exploitent le fait que Twitter est devenu une source vitale en cybersécurité, grâce à la nature des informations présentes, soit sous forme d'événements, soit de rapports d'incidents de sécurité publiés quotidiennement par les entreprises de cybersécurité, les utilisateurs individuels, souvent les hackers éthiques, etc. Dans leur travail, les auteurs proposent un outil appelé CyberTwitter, qui est un framework d'analyse des tweets qui permet d'émettre des alertes de menace en temps opportun aux analystes de sécurité. Les alertes générées par CyberTwitter peuvent ensuite servir divers autres systèmes de sécurité qui peuvent les utiliser afin de renforcer leur défense. Le framework commence par collecter les tweets pertinents en utilisant l'API de Twitter. Dans les tweets collectés, les auteurs identifient les mots clés liés à la cybersécurité tels que les moyens d'attaque, les conséquences d'une attaque, les logiciels, matériels et vendeurs affectés, etc. Chaque tweet est ensuite traité par l'outil SVCE, qui est basé sur le traitement du langage naturel pour extraire différents termes et concepts liés aux vulnérabilités de sécurité. Ces informations sont ensuite transmises au module RDF en utilisant l'ontologie UCO (Unified Cybersecurity Ontology), pour fournir au système des informations sur le domaine de la cybersécurité. La représentation RDF des données est stockée dans une base de connaissances de cybersécurité appelée "Cybersecurity Knowledge Base", permettant ainsi au système d'émettre une alerte pour informer l'utilisateur final d'une menace ou d'une vulnérabilité potentielle qui pourrait exister.

L'article [65] présente un système de collecte de CTI à partir des discussions sur les forums des hackers, qui proposent des produits et services axés sur les plateformes sociales, notamment les marketplaces sur darknet et sur deepnet. Les CTI identifiées comprennent des informations sur les nouveaux logiciels malveillants, qui ne sont pas

encore exploités pour faire des cyberattaques. La communication de ces informations aux professionnels de la cybersécurité peut s'avérer très utile pour soutenir leur planification stratégique de défense contre les attaques. Pour mettre en œuvre ce système, un problème de classification binaire est résolu pour séparer les produits pertinents sur marketplace du darknet/deepnet et les sujets pertinents sur les messages des forums contenant des communications liées au piratage malveillant. Les méthodes de classification supervisée utilisées comprennent les techniques bien connues telles que NB, RF et SVM. En raison de la quantité limitée de données étiquetées, des approches semi-supervisées sont utilisées pour annoter automatiquement les données non étiquetées. L'expérimentation de cette approche sur les sources de marketplace et des forums permet de collecter environ 305 cybermenaces chaque semaine.

Dans l'article [5], les auteurs développent une plateforme de surveillance basée sur Twitter, appelée SYNAPSE, pour générer des résumés sur les menaces liées à une infrastructure informatique donnée. Plus précisément, un outil d'apprentissage automatique est utilisé pour rassembler les tweets provenant de comptes liés à la cybersécurité. Selon les auteurs, Twitter a été choisi comme source d'intelligence pour deux raisons principales. Premièrement, Twitter est reconnu comme une source pertinente des événements en cours presque en temps réel. Deuxièmement, la taille limitée d'un tweet, le rend facile à traiter à travers les approches d'apprentissage automatique. Le pipeline complet de SYNAPSE est composé de : collection de données, filtrage, extraction de caractéristiques, classification binaire, agrégation, et enfin génération des CTI. Le module de collecte de données nécessite un ensemble de comptes, à partir desquels il capte chaque tweet posté en utilisant l'API de Twitter. Ces comptes peuvent être des analystes, des entreprises de sécurité, des hackers, des chercheurs, entre autres. Le module de filtrage est conçu pour supprimer les tweets qui ne sont pas liés à l'infrastructure surveillée par l'analyste des données collectées. Par conséquent, une deuxième entrée est requise : seuls les tweets qui incluent au moins un des mots-clés liés à la cybersécurité passeront le filtre. Dans le module d'extraction de caractéristiques, des opérations de prétraitement sont appliquées aux tweets sélectionnés par le filtre. Cette étape consiste à convertir le texte en minuscules, à supprimer les mots vides, les hyperliens, etc. Les caractéristiques sont ensuite extraites à l'aide de l'algorithme TF-IDF, qui calcule les poids des mots en fonction de leur fréquence d'occurrence. Pour la classification des tweets en fonction de leur pertinence en matière de sécurité, deux classificateurs ont été entraînés et testés (les SVM et un DNNs). Un réglage des hyperparamètres est effectué dans cette étape, afin de sélectionner le meilleur modèle avec un nombre élevé de vrais positifs et vrais négatifs qui classent les tweets en deux classes (pertinentes/non-pertinentes). Dans la phase d'agrégation, un algorithme de regroupement (k-means) est utilisé pour éliminer les informations en double et fournir un résumé concis des menaces actuelles. Après la phase de regroupement, les CTI de tweets seront transformés en format adapté pour être exploités par d'autres plateformes de cybersécurité.

L'article [78] présente DISCOVER, un framework qui exploite plusieurs sources de données en ligne pour créer des signaux potentiels permettant d'identifier une attaque imminente ou une nouvelle vulnérabilité cybernétique. Plus précisément, il s'agit d'un

système d'alerte précoce aux menaces, qui extrait les discussions en ligne des acteurs d'attaques afin d'identifier les mots qui pourraient être liés à une nouvelle attaque potentielle ou être le nom réel de la menace (nom d'un logiciel malveillant, d'un cheval de Troie, d'un exploit, etc.). Le principal avantage de ce framework est d'aider les organisations et les victimes à se préparer et à limiter leur vulnérabilité. DISCOVER surveille plusieurs canaux de discussion en ligne liés à la sécurité, y compris les blogs d'experts en sécurité, les white hat hackers, ainsi que les postes sur les réseaux sociaux, ensuite vérifie les co-occurrences de termes pour découvrir les menaces dans les discussions des acteurs malveillants sur les forums, marketplace du darkweb. DISCOVER traite les données de ces sources en utilisant des techniques d'exploitation de données pour identifier les termes nouveaux liés à une menace potentielle, qu'il retourne sous forme d'alerte. Pour l'implémentation, les auteurs commencent d'abord par : obtenir des données à partir de sources de données primaires telles que les médias sociaux (Twitter), des blogs rédigés par des experts en cybersécurité, des white hat hackers, qui contiennent des informations techniques riches sur les dernières exploitations, vulnérabilités logicielles, logiciels malveillants, et d'autres sujets liés à la cybersécurité. Deuxièmement, en raison de la nature différente des sources de données, chacune fournissant un type de signal unique. Le contenu de Twitter et des blogs est plus propre par rapport aux forums du darkweb. Pour résoudre ce problème, les auteurs appliquent une procédure de filtrage et de prétraitement de données en deux étapes sur les sources primaires Twitter et blogs. L'étape de filtrage élimine les termes dans le texte qui ne sont pas écrits en anglais. Après le filtrage, ils prétraitent les données en supprimant les URL, les symboles, les chiffres, etc., et en tokenisant le texte pour obtenir une liste unique de termes. Troisièmement, l'étape de prétraitement de données donne une grande liste de mots qui pourraient ne pas être pertinents pour les menaces cybernétiques. Les auteurs excluent les termes en les filtrant s'ils se produisent dans l'un des dictionnaires (anglais, stop words, vocabulaire de domaine et dictionnaire de menaces). Enfin, pour générer les alertes, deux règles sont introduites par les auteurs : 1) exclure les mots qui n'apparaissent qu'une seule fois dans tous les messages au cours de la période donnée. 2) s'assurer que le terme détecté est lié à un sujet de cybersécurité. Tout nouveau terme qui répond à ces exigences sera une alerte générée par le framework DISCOVER.

L'article [27] étudie l'utilisation des techniques d'apprentissage automatique pour parcourir les forums de hackers, ensuite extraire des informations CTI pertinentes. Les auteurs ont utilisé des données d'un forum appelé "Nulled.IO". Ce travail englobe deux grandes phases : phase de construction de l'ensemble de données et la phase de prétraitement de données. Au cours de la phase de construction de l'ensemble de données, deux sous-ensembles de données ont été produits à partir d'une base de données appelée "Nulled.IO". Le premier sous-ensemble de données est des données binomiales, partiellement divisé entre deux classes, une pertinente qui regroupe les mots-clés de sécurité, et une deuxième non-pertinente contenant d'autres mots qui répondent à certains critères définis par les auteurs. Le deuxième sous-ensemble de données est des données multinomiales. Pour les données provenant des publications non-pertinentes de la sous-ensemble de données multinomiales, il n'y a qu'une seule classe ou étiquette,

tandis qu'il y a de nombreuses classes ou étiquettes pour les données provenant des publications pertinentes. Les auteurs ont utilisé deux critères pour déterminer le nombre et le type de classes. La deuxième phase du travail est une phase de prétraitement, inclut l'étape de préparation, seuls les titres et le contenu de chaque publication ont été choisis pour un examen plus approfondi. Après le nettoyage des données pour éliminer les bruits, les auteurs ont utilisé deux méthodes de classification supervisée (méthodes d'apprentissage automatique traditionnelles et les CNNs), pour classer le contenu des publications en différentes catégories liées à la cybersécurité. Chaque contenu de publication séparé a été considéré comme un document séparé. Les résultats obtenus de la comparaison montrent que le classificateur SVM traditionnel fonctionne au moins aussi bien que le CNNs. Bien que ce dernier ait été considéré comme très performant dans la résolution de problèmes similaires.

L'article [49] propose un framework qui applique une approche non supervisée pour détecter et catégoriser les événements de cybersécurité à partir des tweets. Leur approche proposée repose sur un ensemble de mots-clés initiaux spécifiés pour chaque niveau de la taxonomie CTI. En conséquence, les auteurs présentent une méthode pour élargir l'ensemble de mots-clés initiaux en identifiant et en ajoutant de nouveaux mots avec des significations similaires dans le contexte des embeddings de mots en utilisant un seuil spécifié manuellement, dans la distance de similarité cosinus entre les vecteurs de mots. Ce framework considère les événements comme des regroupements de textes de tweets générés via la méthode TF-IDF. Cependant, cet algorithme est sujet à de forts taux de faux positifs en raison des effets de biais involontaires des mots-clés initiaux. De plus, un seuil fixe spécifié manuellement pour l'annexion de nouveaux mots s'avère inefficace dans la sélection efficace de nouveaux mots-clés pour un type d'événement CTI particulier.

4.3 Techniques d'extraction des Indicateurs de Compromission

Dans l'article [53], les auteurs présentent une nouvelle approche appelée iACE (IOC Automatic Extractor) pour extraire les IoCs à partir de rapports textuels non structurés. Les IoCs comprennent des informations telles que les URL, les domaines, les adresses IP et des empreintes MD5, qui sont essentielles dans les enquêtes en cybersécurité. Les auteurs abordent les limites des techniques existantes de NLP qui ne sont pas directement applicables aux IoCs. L'outil iACE utilise des techniques de NLP adaptées aux caractéristiques uniques de la découverte des IoCs. Le processus commence par la préparation des rapports, y compris la classification des termes thématiques pour identifier les sections pertinentes et la conversion des figures en texte à l'aide de la reconnaissance optique de caractères (OCR). L'outil utilise ensuite des expressions régulières (regex) et des termes de contexte communs extraits de iocterms pour localiser les phrases dans les rapports qui contiennent potentiellement des jetons IoCs. Dans chaque phrase, l'approche établit une relation entre le jeton IoCs et le terme de

contexte. Elle convertit la phrase en un graphe de dépendance (DG) pour capturer sa structure grammaticale et extrait les chemins les plus courts reliant chaque paire d'un jeton de contexte et du jeton IoCs présumé. Une technique d'exploration de graphes est utilisée pour analyser la relation entre les jetons, ce qui ne peut pas être géré par les techniques existantes d'extraction de relations. Cette analyse vise à déterminer si la phrase contient effectivement un IoCs et son contexte. Les auteurs décrivent trois étapes principales pour la mise en œuvre de leur approche proposée. Tout d'abord, le contenu pertinent est identifié en extrayant des pages web de sites de blogs, en prétraitant le contenu et en filtrant les informations non pertinentes. Les phrases qui portent potentiellement des IoCs sont identifiées à l'aide d'un classifieur à vecteurs de support SVM. Ensuite, un analyseur de dépendance transforme chaque phrase en un graphe de dépendance pour capturer les liens grammaticaux entre les jetons linguistiques. Un classifieur de régression logistique est entraîné pour analyser la relation entre un candidat IoCs et un terme de contexte au sein d'une phrase. Enfin, les IoCs extraits et leurs termes de contexte correspondants sont utilisés pour générer automatiquement des enregistrements OpenIOC, ce qui permet la conversion du contenu CTI à partir de blogs techniques.

Dans l'article [64], les auteurs proposent un Framework pour automatiser l'attribution des acteurs suspects des cybermenaces, en se basant sur leurs schémas d'attaque extraits a partir des rapports CTI. Ces rapports se présentent sous format non structurés et sont accessibles au public sous la forme de blogs et de bulletins d'information. L'approche utilise les IoCs de haut niveau (TTPs), les outils logiciels, les logiciels malveillants pour établir des profils des adversaires. Les schémas d'attaque sont présentés sous forme de descriptions textuelles compréhensibles par l'etre humain, dans les rapports CTI non structurés, qui ne peuvent pas être directement interprétés par les machines. Cependant, l'extraction manuelle des IoCs de haut niveau est un processus fastidieux, qui prend du temps et qui est source d'erreurs. Pour résoudre ce problème, les auteurs se concentrent sur la conception d'un mécanisme automatisé permettant d'extraire les IoCs de haut niveau d'un documents non structurés en utilisant un vocabulaire commun pour établir les profils, et utiliser ces profils pour attribuer à des auteurs de cybermenaces. Le Framework commence par une recherche sémantique des termes TTPs dans les rapports non structurés, en attribuant une classe à chaque terme en suivant le framework ATT&CK, afin obtenir des schémas d'attaque d'adversaires. Ensuite, un algorithme d'analyse sémantique latente (LSA) est utilisé pour indexer les rapports. L'objectif est de montrer comment l'acteur de la menace peut être déterminé en corrélant ces IoCs de haut niveau avec les menaces de nature similaire. L'ensemble de données qui en résulte est utilisé pour attribuer les cybermenaces à leurs adversaire en entraînant cinq classificateurs tels que : NB, KNN, DT, RF et DNNs, avec 327 rapports, ensuite choisir le meilleur modèle.

L'article [63] présente un outil de mappage d'informations appelé SECCMiner pour aider les professionnels de la cybersécurité à obtenir plus efficacement des CTI, à partir d'un ensemble de rapports APT non structurés. L'approche utilise diverses méthodes d'exploration de texte et de NLP afin de reconnaître automatiquement les techniques

et tactiques dans les rapport. Pour mettre en œuvre l'approche proposée, les auteurs commencent par convertir tous les rapport PDF d'entrée au format text. Ils effectuent également des opérations de post-traitement sur le texte obtenus afin de corriger plusieurs problèmes courants tels que les titres, le bruit, texte fragmenté, etc. Ensuite, SECCMiner extrait toutes les phrases nominales uniques apparaissant dans le corpus et appliquer l'algorithme TF-IDF pour chacune des phrases dans chaque document. Enfin, il enregistre les phrases nominales dont le score est supérieur à un seuil prédéfini en tant que concepts clés représentant chacun des documents d'entrée, ce qui permet aux professionnels d'extraire les techniques et tactiques utilisées par les attaquants. L'outil fournit également une analyse statistique des rapports qui peut aider les professionnels de la sécurité à concentrer leur attention sur les techniques adverses les plus courantes parmi les groupes APT. Il est à noter que l'outil développé a été entraîné sur 445 rapports techniques rédigés principalement par des analystes de fournisseurs réputés tels que ESET, FireEye, Kaspersky, McAfee, etc.

Dans l'article [38], les auteurs proposent une approche de traitement du langage naturel pour extraire des indicateurs CTI de haut niveau à partir de sources non structurées tels que les blogs. La solution utilise une approche d'apprentissage supervisé et des rapport annotés contenant des données pertinentes provenant de plus de 20 sources différentes. Un service Web à été développé pour effectuer l'annotation et extraire les termes CTI, qui comprennent non seulement des indicateurs de bas niveau, mais aussi des informations de haut niveau, en particulier sur les techniques de l'adversaire. Cette solution transforme également un rapport de format non structurée, vers un format interprétante par les machine selon le standard STIX. Pour mettre en œuvre cette approche, les auteurs ont commencé par collecter des rapport provenant de blogs de sécurité bien connus et d'autres différentes sources telles que FireEye, Kaspersky Security Lab, etc. Ensuite, ils ont annoté les documents pour leur fournir des données supervisées contenant un texte continu étiqueté avec sept classes (Actor, Targeted Industry, Targeted Location, Intended Effect, Technique, Tool Used and Targeted Application). Ensuite, l'algorithme CFR est entraîné avec les données annotées pour faire la classification. Le modèle résultant est intégré au composant de traitement du langage naturel du système de production, qui est utilisé pour extraire les termes de CTI. Enfin, les données de sortie sont normalisées en format STIX afin d'envoyer les flux aux utilisateurs.

Dans l'article [7], les auteurs ont construit un framework offrant un système de bout en bout basé sur l'apprentissage automatique et les techniques de traitement du langage naturel pour extraire les actions de l'attaquant à partir de documents APT. Les auteurs ont construit un modèle pour la classification automatisée des rapports APT, qui peut généraliser sur les rapports provenant de différentes organisations. Ils ont entamé cette approche comme un problème de classification à plusieurs niveaux, classifiant d'abord les rapports APT en terme de tactiques, puis en terme techniques. Pour la mise en œuvre de la solution proposée, le framework commence par la collecte de rapports de menaces provenant de différentes organisations de sécurité, sous format HTML et PDF. Ensuite, les données sont transformées en texte brut. Les techniques de

traitement du langage naturel sont ensuite appliquées pour extraire les caractéristiques TF-IDF de chaque document. Après l'extraction des caractéristiques, les auteurs ont entraîné un classificateur SVM avec 169 documents étiquetés provenant du ATT&CK, 17600 documents provenant du Symantec et 488 documents provenant de sociétés de sécurité telles que FireEye, Mandiant, McAfee et d'autres compagnies d'antivirus.

L'article [32] propose un framework qui intègre la reconnaissance d'entités nommées NER, et des techniques basées sur l'ontologie pour classer les tweets en tant qu'événements pertinente ou non-pertinente pour le CTI. Si un tweet est classé comme pertinent, le framework effectue une détection de sujet via la mise en correspondance croisée des résultats de la NER avec des bases de connaissances externes telles que "DBPedia". De plus, ce travail produit un ensemble de données annotées de tweets CTI, et de types d'événements à l'aide du portail des événements actuels de Wikipedia, ainsi que de l'entrée humaine collectée via Amazon Mechanical Turk. Avec cet ensemble de données annotées, les auteurs étudient les performances de diverses approches d'apprentissage automatique telles que NB, les SVM et les LSTMs, et signalent que l'architecture LSTMs avec la représentation de caractéristiques par incorporation de mots produit les meilleurs résultats. Ils démontrent également que la catégorie générique de la NER, est utile dans la classification binaire de pertinence, tandis que des catégories spécifiques de la NER, sont utiles dans la classification des types et des catégories d'événements. Pour l'identification des sujets, ce travail adopte l'algorithme "PageRank" pour identifier le sujet le plus proche dans le graphe relationnel du concept de tweet.

L'article [55] propose une méthodologie pour la reconnaissance des entités nommées NER dans le domaine de la cybersécurité. Les auteurs combinent le réseau de neurones récurrents à mémoire à court terme bidirectionnel Bi-LSTMs avec les champs aléatoires conditionnels CFR pour améliorer la précision de l'identification des entités nommées liées à la cybersécurité. Les auteurs commencent par introduire l'importance de la NER en cybersécurité, en mettant en évidence les défis liés à l'extraction d'informations significatives à partir de données textuelles non structurées. Ils expliquent que les approches traditionnelles basées sur des règles sont limitées dans la capture de motifs complexes et de dépendances contextuelles. Pour remédier à ces limitations, les auteurs proposent l'utilisation de techniques d'apprentissage profond, notamment le Bi-LSTMs et le CFR. Le modèle proposé se compose d'une couche LSTM bidirectionnelle suivie d'une couche CRF. Le LSTM bidirectionnel capture les informations contextuelles en traitant le texte d'entrée dans les deux sens, avant et arrière. La couche CRF considère ensuite les dépendances entre les étiquettes consécutives et les utilise pour effectuer des prédictions plus précises. Les auteurs évaluent leur approche sur un ensemble de données et la comparent à plusieurs modèles de référence. Ils mesurent les performances en utilisant la précision, le rappel et le score F1. Les résultats expérimentaux démontrent que le modèle LSTM bidirectionnel-CRF proposé surpasse les modèles de référence, atteignant une précision plus élevée dans l'identification des entités nommées liées à la cybersécurité. Bien que l'article présente une approche prometteuse pour la NER en cybersécurité, il y a quelques limitations à prendre en compte. Une limitation est le

manque de comparaison approfondie avec les méthodes existantes de pointe. Il aurait été bénéfique d'inclure une évaluation plus étendue par rapport à d'autres techniques de NER pour démontrer la supériorité de l'approche proposée. De plus, l'article ne traite pas des défis spécifiques et des caractéristiques du domaine de la cybersécurité qui pourraient avoir un impact sur la NER.

L'article [95] présente une approche novatrice pour la reconnaissance des entités nommées NER dans le domaine de la cybersécurité. Les auteurs proposent d'incorporer plusieurs modalités, y compris des données textuelles, des images et des journaux de trafic réseau, afin améliorer la précision. Les auteurs commencent par souligner l'importance de la NER en cybersécurité et les défis liés à l'extraction d'informations significatives à partir de sources de données diverses. Ils soutiennent que l'utilisation de plusieurs modalités peut fournir des informations complémentaires et améliorer les performances des systèmes de NER. L'approche proposée se compose de trois étapes principales : l'extraction de caractéristiques, les modèles spécifiques à chaque modalité et l'apprentissage par ensemble. Dans l'étape d'extraction de caractéristiques, des caractéristiques sont extraites de chaque modalité, telles que des plongements de mots (word embeddings) à partir des données textuelles et des caractéristiques visuelles à partir des images. Des modèles spécifiques à chaque modalité, comprenant des CNNs et des réseaux neuronaux récurrents RNNs, sont ensuite appliqués à chaque modalité pour apprendre des représentations spécifiques à chaque modalité. Enfin, une technique d'apprentissage par ensemble est utilisée pour combiner les prédictions de chaque modèle spécifique à chaque modalité et générer la sortie finale. Pour évaluer l'efficacité de leur approche, les auteurs réalisent des expériences sur un ensemble de données. Ils comparent les performances de l'approche d'apprentissage par ensemble multi-modal avec des approches mono-modales. Les mesures d'évaluation utilisées comprennent la précision, le rappel et le score F1. Les résultats expérimentaux démontrent que l'approche proposée atteint une précision plus élevée dans l'identification des entités nommées liées à la cybersécurité par rapport aux approches mono-modales. L'article présente une contribution notable en incorporant plusieurs modalités dans le processus de NER et en démontrant son efficacité dans le domaine de la cybersécurité.

L'article [85] se concentre sur l'utilisation des word embeddings pour améliorer la reconnaissance des entités nommées NER dans le domaine de la cybersécurité. Les auteurs explorent différentes techniques deword embeddings et évaluent leur impact sur les performances des modèles de NER en cybersécurité. Les auteurs commencent par souligner l'importance de la NER en cybersécurité et les défis associés à l'extraction d'informations pertinentes à partir de données textuelles non structurées. Ils soutiennent que les word embeddings, qui représentent les mots sous forme de vecteurs denses dans un espace sémantique continu, ont le potentiel de capturer des relations sémantiques et d'améliorer les performances des systèmes de NER. L'étude compare plusieurs techniques de word embeddings, notamment Word2Vec, GloVe et FastText, et évalue leur efficacité dans l'amélioration de la NER en cybersécurité. Les auteurs utilisent un ensemble de données de référence spécifiquement conçu pour les tâches de NER en cybersécurité. Ils mesurent les performances des modèles à l'aide de métriques

d'évaluation standard telles que la précision, le rappel et le score F1. Les résultats expérimentaux démontrent que l'incorporation de word embeddings améliore les performances des modèles de NER par rapport aux représentations traditionnelles de sacs de mots. Plus précisément, les auteurs constatent que les embeddings FastText surpassent les embeddings Word2Vec et GloVe en termes de précision, de rappel et de score F1. Cependant, une limitation de l'article est le manque de comparaison exhaustive avec d'autres techniques de pointe de NER dans le domaine de la cybersécurité. Il aurait été bénéfique d'inclure une analyse comparative pour établir la supériorité des word embeddings par rapport à d'autres approches.

L'article [97] présente une approche de reconnaissance des entités nommées NER qui utilise BERT avec un masquage de mots entiers spécifiquement conçu pour le domaine de la cybersécurité. Les auteurs commencent par souligner l'importance de la NER en cybersécurité et les défis posés par la nature complexe et évolutive des données textuelles liées à la cybersécurité. Ils soutiennent que BERT, un modèle de langage de pointe, a montré des performances remarquables dans diverses tâches de traitement du langage naturel et a le potentiel d'améliorer la NER dans le domaine de la cybersécurité. L'approche proposée intègre un masquage de mots entiers, une modification du modèle BERT d'origine, pour résoudre le problème de la tokenisation. En considérant des mots entiers plutôt que des unités sous-mot, les auteurs visent à capturer les informations contextuelles et à améliorer la précision de la NER. Pour évaluer l'efficacité de leur approche, les auteurs mènent des expériences sur un ensemble de données de cybersécurité et comparent les performances de leur modèle basé sur BERT avec des modèles d'apprentissage automatique traditionnels et d'autres modèles d'apprentissage profond. Ils mesurent les performances à l'aide de métriques d'évaluation standard telles que la précision, le rappel et le score F1. Les résultats expérimentaux démontrent que l'approche proposée en utilisant BERT avec un masquage de mots entiers atteint des performances supérieures par rapport aux autres modèles. Elle surpasse les modèles d'apprentissage automatique traditionnels et les autres modèles d'apprentissage profond en termes de précision, de rappel et de score F1 pour les tâches de NER en cybersécurité. Cependant, une limitation de l'article est la discussion relativement brève des techniques spécifiques utilisées pour le masquage de mots entiers et de la façon dont elles abordent les défis propres au domaine de la cybersécurité. De plus, une comparaison plus complète avec d'autres méthodes de pointe de NER aurait renforcé l'évaluation de l'approche proposée.

L'article [37] se concentre sur une approche axée sur les données et les connaissances pour la reconnaissance des entités nommées NER dans le domaine de la cybersécurité. Les auteurs proposent une méthodologie qui combine des techniques axées sur les données, telles que l'apprentissage profond, avec des connaissances spécifiques au domaine pour améliorer la précision de la NER. . Ils soutiennent que les approches traditionnelles de NER ne capturent peut-être pas efficacement les caractéristiques spécifiques au domaine et les informations contextuelles nécessaires pour une NER plus précise. L'approche proposée comprend deux composants principaux : un modèle axé sur les données et un modèle axé sur les connaissances. Le modèle axé sur les données utilise

des techniques d'apprentissage profond, telles que les CNNs et les réseaux de neurones récurrents RNNs, pour apprendre des représentations à partir des données textuelles d'entrée. Le modèle axé sur les connaissances intègre des connaissances spécifiques au domaine, telles que des ontologies ou des lexiques de cybersécurité, pour améliorer le processus de NER. Pour évaluer l'efficacité de leur approche, les auteurs mènent des expériences sur un ensemble de données et comparent les performances de leur modèle axé sur les données et les connaissances à d'autres modèles de pointe de NER. Ils mesurent les performances à l'aide de métriques d'évaluation standard telles que la précision, le rappel et le score F1. Les résultats expérimentaux démontrent que l'approche proposée axée sur les données et les connaissances atteint une précision supérieure dans l'identification des entités nommées par rapport à d'autres modèles. En tirant parti à la fois des techniques axées sur les données et des connaissances spécifiques au domaine, le modèle est capable de capturer à la fois les informations contextuelles générales et les caractéristiques spécifiques du domaine, ce qui conduit à des performances améliorées. Cependant, une limitation de l'article est le manque de discussion détaillée sur les sources de connaissances spécifiques et sur la façon dont elles sont intégrées dans le processus de NER. Fournir davantage d'informations sur l'aspect axé sur les connaissances du modèle aurait amélioré la compréhension de son efficacité.

4.4 Techniques d'attribution des cyberattaques

L'article [69] se concentre sur l'attribution des cyberattaques à partir de rapports de renseignement sur les menaces. Les auteurs soulignent que les méthodes traditionnelles, axées sur l'analyse des codes malveillants, sont souvent insuffisantes pour identifier les acteurs des menaces, car le même malware peut être utilisé par différents groupes ou modifié pour brouiller l'identité de son auteur. NO-DOUBT transforme ce problème en une tâche de classification textuelle en exploitant les informations contenues dans les rapports textuels non structurés. La méthodologie repose sur un modèle de représentation vectorielle appelé SMOBI, conçu pour capturer le contexte des mots tout en réduisant la nécessité d'un prétraitement intensif. Deux ensembles de données sont utilisés : un ensemble de rapports étiquetés attribuant des attaques à des acteurs spécifiques et un corpus plus large de documents non étiquetés couvrant des thèmes généraux de cybersécurité. L'approche bénéficie d'une combinaison de données étiquetées limitées et de connaissances contextuelles tirées des données non étiquetées pour améliorer les performances du modèle. Les auteurs mettent également à disposition leurs jeux de données pour encourager la recherche future dans ce domaine. Une des forces de cette méthode est sa capacité à identifier de nouveaux acteurs de menace inconnus tout en enrichissant les profils des acteurs existants. Cependant, l'article ne discute pas en détail des défis liés à l'hétérogénéité et la qualité des rapports utilisés, ce qui pourrait affecter l'efficacité du modèle dans des contextes réels variés.

L'article [91] introduit une méthode novatrice pour l'attribution des groupes APT à l'aide d'une analyse de similarité binaire. L'objectif principal est d'améliorer la pré-

cision de l'attribution en utilisant des caractéristiques locales extraites de graphes de flux de contrôle attribués (ACFG). L'approche repose sur plusieurs étapes : l'extraction initiale des ACFG, le filtrage des fonctions via des appels API pour éliminer les chemins inutiles, et la génération de "shapelets" à partir des séquences de blocs de base. Ces shapelets, des segments représentatifs de chemins critiques, permettent une classification interprétable des malwares. Contrairement aux approches globales basées sur les caractéristiques fonctionnelles, ce modèle se concentre sur des sous-sequences locales pour réduire les bruits et améliorer la robustesse des classifications inter-plateformes. L'utilisation des shapelets rend le processus plus efficace en limitant les comparaisons nécessaires lors de la phase d'exécution, ce qui réduit considérablement le temps de calcul. Cependant, bien que cette méthode soit prometteuse pour la classification précise des malwares APT, elle dépend fortement de la qualité des fonctions filtrées, et son applicabilité pourrait être limitée dans des contextes où les données sont incomplètes ou fortement déséquilibrées.

L'article [40] propose un modèle de clustering consensuel flou multi-vue, appelé MVFCC, pour l'attribution des menaces. Les auteurs partent du constat que les techniques traditionnelles d'attribution, basées sur une seule source de données (statique ou dynamique), ne suffisent pas à capturer la complexité des tactiques, techniques et procédures (TTP) des acteurs de menace. MVFCC combine plusieurs perspectives (vues), telles que les caractéristiques statiques, les flux d'exécution dynamiques, et les comportements réseau, pour créer un ensemble robuste de classificateurs flous. Chaque vue est traitée indépendamment, puis les résultats sont agrégés via un mécanisme de consensus pour prendre des décisions globales. L'originalité de l'approche repose sur son utilisation d'arbres de motifs flous, qui offrent une flexibilité accrue dans l'identification des similarités entre différentes campagnes malveillantes. Ce modèle excelle dans la gestion des attaques multi-étapes en fournissant des estimations précises de l'origine des outils utilisés à différentes étapes de l'attaque. Une des limites identifiées est le coût computationnel élevé associé à la gestion de multiples perspectives, en particulier lorsque des jeux de données massifs sont impliqués.

L'article [41] introduit un framework APTMalInsight de détection et de classification des logiciels malveillants APT, combinant des analyses dynamiques et une modélisation ontologique. Les auteurs extraient des séquences d'appels système dynamiques pour caractériser les comportements malveillants et construisent un modèle ontologique pour organiser ces connaissances. L'approche propose trois contributions majeures : (1) une méthode de détection basée sur les caractéristiques comportementales dynamiques des APT, (2) une classification précise des malwares dans leurs familles respectives, et (3) un modèle ontologique permettant une représentation sémantique systématique des comportements malveillants. APTMalInsight permet de contourner les limitations des méthodes traditionnelles basées sur l'analyse du trafic réseau ou des journaux de sécurité, qui peuvent être difficiles à obtenir pour des raisons de confidentialité. En offrant une perspective plus compréhensive des comportements des malwares APT, le modèle permet une compréhension approfondie des intentions des attaquants. Toutefois, l'article ne fournit pas suffisamment de détails sur la scalabilité du cadre ontologique pour

des environnements en temps réel ou à grande échelle.

L'article [42] propose une approche d'attribution des acteurs de menace basée sur l'apprentissage profond, exploitant des rapports de renseignement sur les menaces. Les auteurs transforment l'attribution en un problème de classification multi-classe, où chaque acteur est représenté comme une classe distincte. Le modèle utilise des embeddings de mots spécifiques au domaine générés par Word2Vec pour représenter les rapports textuels dans un espace vectoriel dense. La structure du réseau neuronal inclut plusieurs couches denses, avec des mécanismes de régularisation tels que le dropout pour limiter le surapprentissage. Une attention particulière est accordée au prétraitement des rapports pour éliminer toute mention explicite des acteurs afin de renforcer la capacité du modèle à généraliser. L'approche surpasse les techniques traditionnelles, offrant une meilleure compréhension contextuelle des relations entre les TTPs des différents groupes APT. Cependant, une limite notable est que les performances du modèle dépendent fortement de la qualité et de la diversité des rapports utilisés pour l'entraînement, ce qui pourrait poser problème dans des environnements réels où les données sont souvent limitées ou biaisées.

L'article [75] explore l'attribution des acteurs de menace à partir de rapports de renseignement sur les menaces en utilisant une architecture d'apprentissage profond. Contrairement aux approches traditionnelles basées sur l'analyse de code malveillant ou les métadonnées techniques, cette étude utilise des rapports textuels contenant des informations contextuelles riches, telles que les tactiques, techniques et procédures (TTP) des attaques. L'objectif est de surmonter les limites des méthodes de classification existantes qui ne parviennent pas à identifier précisément les acteurs derrière des attaques spécifiques. Les auteurs introduisent un modèle de réseau de neurones dense qui intègre des représentations vectorielles basées sur des embeddings spécifiques au domaine de la cybersécurité, générés à partir de Word2Vec. Ils présentent également une nouvelle technique, SIMVER (SIMilarity based VEctor Representation), pour améliorer la représentation des données textuelles en exploitant les relations sémantiques. SIMVER utilise des vecteurs contextuels pour associer des mots à leurs voisins les plus proches en fonction de leur similarité, renforçant ainsi la capacité du modèle à capturer des indices subtils dans les rapports. Le modèle est entraîné sur un ensemble de données comprenant 238 rapports étiquetés, associés à 12 groupes APT distincts, tirés de sources telles que MITRE ATT&CK. Les rapports sont prétraités pour éliminer les informations explicites sur les acteurs, rendant la tâche d'apprentissage plus réaliste et complexe. Un ensemble de données non étiqueté est également utilisé pour former les embeddings spécifiques au domaine. L'approche est évaluée par une validation croisée en k-plis et en utilisant des métriques standard telles que la précision, le rappel et le score F1. Les auteurs discutent des défis liés à l'équilibrage des classes et à la gestion des biais induits par des classes surreprésentées dans les données. Cette recherche représente une avancée dans l'attribution des acteurs de menace en exploitant des rapports textuels, et les auteurs proposent d'explorer des réseaux neuronaux siamois pour améliorer davantage la détection des similarités à l'avenir.

4.5 Techniques de prédiction de niveau de gravité des cybermenaces

L'article [21] examine les incohérences entre différentes sources de données dans l'évaluation de la gravité des vulnérabilités logicielles (SV). L'objectif est de comprendre comment ces variations impactent les processus de priorisation des correctifs et les prédictions de gravité. Les auteurs analysent des vulnérabilités provenant de trois sources principales : Bugzilla, Mozilla Security Advisory, et la National Vulnerability Database (NVD). Ils identifient six facteurs influençant les incohérences, notamment les différences dans les méthodologies d'évaluation, les niveaux d'expertise des évaluateurs, et les délais de divulgation. L'étude utilise des modèles statistiques pour mesurer l'impact de ces incohérences sur les décisions critiques, en mettant un accent particulier sur les scores CVSS (Common Vulnerability Scoring System). Les résultats montrent que des divergences importantes existent entre les scores assignés par les différentes sources, ce qui affecte la capacité des modèles de machine learning à prédire correctement la gravité des vulnérabilités. Par exemple, des scores incohérents peuvent réduire les performances prédictives des modèles de 77%, ce qui entraîne des priorisations incorrectes et des risques accrus pour les systèmes affectés. Les auteurs proposent des recommandations pour améliorer la standardisation des méthodologies d'évaluation de gravité, en suggérant l'intégration de nouvelles métriques et l'utilisation de bases de données unifiées pour améliorer la cohérence dans les évaluations.

L'article [9] propose un cadre pour prédire l'exploitabilité des vulnérabilités logicielles, en s'appuyant sur des techniques d'apprentissage automatique supervisé. L'objectif est de permettre aux responsables de sécurité de prioriser efficacement les correctifs en identifiant les vulnérabilités les plus susceptibles d'être exploitées. L'étude exploite des bases de données telles que la NVD, Exploit-DB, et Zero Day Initiative (ZDI) pour collecter des informations sur les vulnérabilités et leurs exploitations passées. Les auteurs utilisent plusieurs algorithmes d'apprentissage supervisé, y compris Random Forest, Support Vector Machine (SVM), Naive Bayes, et Logistic Regression, pour classifier les vulnérabilités en exploitables ou non exploitables. Les caractéristiques prises en compte incluent des métriques CVSS (comme le score de base, d'impact et d'exploitabilité), des types de vulnérabilités (par exemple, injection SQL, XSS), et des informations sur les configurations logicielles. Le cadre proposé est structuré en trois étapes : (1) collecte des données à partir des bases disponibles, (2) extraction des caractéristiques pertinentes, et (3) entraînement des modèles de classification. Les résultats expérimentaux montrent que ces algorithmes peuvent atteindre des niveaux de précision supérieurs à 85% pour la prédiction de l'exploitabilité. Les auteurs soulignent également que leur approche peut être intégrée dans les processus de gestion des risques pour prioriser efficacement les vulnérabilités en fonction de leur potentiel d'exploitation.

Dans l'article [46], les auteurs proposent une méthode hybride combinant des NLP et des algorithmes d'apprentissage automatique pour estimer automatiquement la gra-

vité des vulnérabilités logicielles. L'objectif est de réduire les délais associés à l'évaluation manuelle des vulnérabilités et d'améliorer la précision de la classification des vulnérabilités en différentes catégories de gravité. Les auteurs utilisent plusieurs méthodes d'extraction de caractéristiques, y compris des approches statistiques comme TF-IDF (Term Frequency-Inverse Document Frequency) et des modèles d'embedding tels que Word2Vec, Doc2Vec, et FastText. Une fois les caractéristiques extraites, des algorithmes de classification tels que Naive Bayes, Decision Tree, K-Nearest Neighbors (KNN), Multi-Layer Perceptron (MLP), et Random Forest sont appliqués pour effectuer une classification multi-classes. La base de données utilisée est le NVD (National Vulnerability Database), contenant plus de 150 000 enregistrements de vulnérabilités, classées selon les versions CVSS 2.0 et 3.1. L'étude divise les données en ensembles d'entraînement et de test, en veillant à maintenir des distributions de classes homogènes. L'approche hybride proposée par les auteurs vise à combiner les forces des différentes méthodes d'extraction et de classification, offrant ainsi un modèle robuste pour l'estimation automatique de la gravité des vulnérabilités. Cette méthode réduit les erreurs humaines tout en accélérant le processus d'évaluation

L'article [82] concentre sur la priorisation des vulnérabilités logicielles en utilisant leurs descriptions textuelles. L'objectif est de fournir un mécanisme efficace pour classer les vulnérabilités en fonction de leur gravité potentielle, permettant aux équipes de sécurité de se concentrer sur les problèmes critiques. Contrairement aux méthodes traditionnelles basées sur les scores CVSS, cette étude propose d'utiliser des modèles de traitement du langage naturel et des réseaux neuronaux convolutionnels pour extraire des informations directement à partir des descriptions textuelles. L'approche commence par une phase d'extraction de caractéristiques où les descriptions textuelles sont transformées en vecteurs de mots à l'aide de l'algorithme GloVe (Global Vectors for Word Representation). Ces vecteurs sont ensuite introduits dans un CNN pour identifier automatiquement les vulnérabilités à haut, moyen et faible risque. Les auteurs testent leur modèle sur des ensembles de données provenant de différents fournisseurs, notamment Linux, Microsoft, et Google. Chaque ensemble contient 2000 descriptions de vulnérabilités, équilibrées pour éviter les biais. Cette approche permet de capturer les relations contextuelles et les mots-clés discriminants pour une classification précise. En réduisant la dépendance aux scores CVSS, cette méthode accélère le processus de priorisation et offre un support décisionnel amélioré pour les équipes de sécurité sous contrainte de ressources.

4.6 Analyse critique

4.6.1 Techniques de collection des informations liées à la CTI

Après avoir présenté les différentes études liées aux techniques de collecte des informations liées à la CTI dans la section 4.2, nous procédons maintenant à une analyse approfondie de ces recherches en nous concentrant sur les tâches accomplies. En effet,

les applications du NLP dans la collecte des informations liées à la CTI peuvent être résumées en trois tâches principales :

Classification

Les approches basées sur des algorithmes d'apprentissage automatique traditionnels tels que les SVM, NB et RF ont été largement utilisées. Ces méthodes, bien qu'efficaces pour des tâches spécifiques, présentent certaines limites de généralisation, notamment lorsqu'elles sont appliquées à des ensembles de données hétérogènes ou à des données provenant de sources variées comme les réseaux sociaux. De plus, elles nécessitent souvent une ingénierie des caractéristiques approfondie, ce qui peut être chronophage et dépendre fortement de l'expertise du domaine.

Les approches basées sur des règles ou des ontologies, comme dans [60] et [78], requièrent une mise à jour continue pour rester pertinentes face à l'évolution rapide des menaces et des techniques d'attaque. Elles dépendent également fortement de l'expertise humaine pour la création et le maintien des règles ou des dictionnaires, ce qui peut être une contrainte en termes de ressources.

Les méthodes utilisant l'apprentissage profond, comme les CNNs dans [27], offrent une meilleure capacité de modélisation des données complexes mais nécessitent des ensembles de données volumineux pour un entraînement efficace. Cela peut poser des problèmes lorsqu'il y a un manque de données étiquetées ou lorsque les données sont coûteuses à annoter.

Regroupement

Les méthodes de regroupement non supervisées, telles que DBSCAN [12] et KMeans [5], permettent de découvrir des structures sous-jacentes dans les données sans nécessiter de données étiquetées. Cependant, elles peuvent être sensibles aux paramètres initiaux (par exemple, le choix du nombre de clusters ou des seuils de densité) et peuvent produire des résultats incohérents si les données sont bruitées ou de faible qualité.

L'approche de l'article [49] utilise des encodeurs de mots pour étendre un ensemble initial de mots-clés. Cette méthode peut introduire un biais si les mots-clés initiaux ne sont pas représentatifs ou si l'expansion ne capture pas correctement le contexte spécifique de la cybersécurité.

Notation

Les systèmes de notation visent à attribuer un score d'importance ou de pertinence aux informations, ce qui aide à la priorisation des alertes. Dans [51], le score est basé sur l'expertise des utilisateurs identifiés, tandis que [12] attribue des scores en fonction de l'influence de l'utilisateur et de la fréquence des termes.

Ces approches peuvent être limitées par la dépendance à des métriques externes (comme le nombre de followers sur Twitter pour mesurer l'influence), qui ne reflètent

pas nécessairement la qualité ou la fiabilité des informations en matière de cybersécurité. De plus, elles peuvent nécessiter une mise à jour continue des critères de notation pour rester pertinentes face à l'évolution des menaces et des acteurs.

Le tableau 4.1 résume les différentes techniques de collecte des informations liées à la CTI.

TABLE 4.1 – Techniques de collecte des informations liées à la CTI.

Article	**Approche**	**Source de données**	**Algorithme**	**Force(+)/Faiblesse(-)**
[12]	Regroupement ; Notation	Twitter	DBSCAN ; TextRank	+ Ne nécessite pas de données étiquetées. - Précision limitée due à l'approche non supervisée.
[28]	Classification ; Regroupement	Darknet	SVM ; LDA	+ Exploite des sources du darknet. + Combine classification et regroupement. - Données étiquetées limitées. - Généralisation limitée des modèles ML simples.
[51]	Classification ; Notation.	Twitter	LDA	+ Identification des experts pour prioriser l'information. - Dépendance sur l'identification initiale des experts.- Nécessite mise à jour continue.
[60]	Classification	Twitter	SVCE ; UCO	+ Intègre des connaissances via des ontologies. - Ontologies nécessitent mise à jour continue. - Dépendance à des règles.
[65]	Classification	Darknet	NB ; RF ; SVM	+ Exploite des sources du darknet. - Données étiquetées limitées. - Généralisation limitée des modèles ML simples.
[5]	Classification ; Regroupement	Twitter	TF-IDF ; SVM ; DNNs ; KMeans	+ Pipeline complet utilisant ML et DL. - Dépendance sur les mots-clés sélectionnés. - Deep learning nécessite de grands jeux de données.
[78]	Classification	Twitter Blogs Darknet	Règles	+ Sources multiples. - Approche basée sur des règles nécessite expertise. - Mise à jour continue des règles.

Table 4.1 – suite

Article	Approche	Source de données	Algorithme	Force(+)/Faiblesse(-)
[27]	Classification	Forums	SVM ; CNNs	+ Compare ML simple et DL. - Données limitées affectent les performances du DL. - Généralisation limitée.
[49]	Regroupement	Twitter	TF-IDF ; CS	+ Approche non supervisée (pas besoin de données étiquetées). - Biais des mots-clés initiaux. - Nécessite mise à jour des mots-clés.

4.6.2 Techniques d'extraction des Indicateurs de Compromission

Après avoir présenté les différentes études liées à l'extraction des IoCs dans la section 4.3, nous procédons maintenant à une analyse approfondie de ces recherches en nous concentrant sur les approches employées, qui peuvent être résumées en cinq catégories :

Approches basées sur les règles

Les approches basées sur les règles utilisent des ensembles de règles prédéfinies pour extraire des IoCs à partir de textes non structurés. Ces règles sont souvent des expressions régulières ou des motifs syntaxiques conçus manuellement pour correspondre à des formats spécifiques d'IoCs, tels que les adresses IP, les URLs ou les hachages.

Dans l'article [53], les auteurs ont utilisé des expressions régulières pour identifier les IoCs dans les rapports textuels. Cette méthode offre une précision élevée sur les motifs connus et est relativement simple et rapide à implémenter. Cependant, elle présente des limites significatives : elle nécessite une expertise pour la création et la maintenance des règles, est sensible au bruit dans les données textuelles, et manque de flexibilité pour détecter de nouveaux types d'IoCs ou des variations inattendues dans les formats existants. De plus, l'approche peut être laborieuse à maintenir face à l'évolution rapide des techniques utilisées par les attaquants.

Approches basées sur les ontologies

Ces approches s'appuient sur des ressources lexicales ou des structures de connaissances préétablies pour extraire les IoCs. Les dictionnaires sont des listes de termes spécifiques au domaine, tandis que les ontologies sont des représentations formelles des connaissances, incluant des concepts, des relations et des règles au sein d'un domaine particulier.

L'article [64] utilise le framework MITRE ATT&CK comme ontologie pour annoter les TTPs dans les rapports de cybermenaces. De même, [32] intègre des ontologies et

des bases de connaissances externes comme DBPedia pour améliorer la classification et l'identification des entités dans les tweets. L'article [37] combine l'apprentissage profond avec des connaissances spécifiques au domaine en intégrant des ontologies pour améliorer la reconnaissance des entités nommées.

Ces approches offrent une simplicité et rapidité d'implémentation et une précision sur les motifs connus grâce à l'utilisation de ressources spécialisées. Toutefois, elles nécessitent une mise à jour continue pour rester pertinentes face à l'évolution des menaces et sont sensibles au bruit dans les données textuelles. De plus, leur efficacité dépend fortement de la qualité et de l'exhaustivité des ressources utilisées, et elles peuvent manquer de détecter des IoCs non présents dans les dictionnaires ou ontologies.

Approches basées sur l'apprentissage automatique simple

Ces approches utilisent des algorithmes classiques d'apprentissage automatique, tels que les SVM, les classifieurs bayésiens naïfs ou les forêts aléatoires, pour apprendre à partir de données étiquetées et extraire des IoCs. Elles nécessitent généralement une étape d'extraction des caractéristiques, où des fonctionnalités pertinentes sont sélectionnées pour représenter les données d'entrée.

Les articles [53], [64], [38], [7] et [85] utilisent ces approches pour extraire des IoCs à partir de diverses sources textuelles. Ces méthodes sont robustes au bruit et peuvent être efficaces pour des tâches spécifiques. Cependant, elles nécessitent des algorithmes d'extraction des caractéristiques, ce qui peut être laborieux et dépendre de l'expertise du domaine. De plus, elles requièrent de larges ensembles de données étiquetées pour un entraînement efficace, ce qui peut être coûteux en termes de temps et de ressources.

Approches basées sur l'apprentissage profond

Les approches d'apprentissage profond exploitent des réseaux de neurones profonds, tels que les réseaux neuronaux convolutifs ou les réseaux de neurones récurrents, pour apprendre automatiquement des représentations à plusieurs niveaux des données d'entrée sans nécessiter d'un algorithme d'extraction des caractéristiques.

Les articles [32], [55] et [95] appliquent l'apprentissage profond à la reconnaissance des entités nommées. Ces méthodes offrent une généralisation relativement bonne et sont capables de capturer des motifs complexes dans les données. Néanmoins, elles nécessitent de larges ensembles de données étiquetées pour l'entraînement, ce qui peut être une limitation majeure. De plus, elles sont complexes sur le plan computationnel et peuvent être difficiles à interpréter.

Approches basées sur les modèles pré-entraînés

Ces approches tirent parti de modèles de langage pré-entraînés sur de grands corpus de données, tels que BERT, en les ajustant (fine-tuning) pour des tâches spécifiques comme l'extraction des IoCs.

L'article [97] utilise BERT pour améliorer la reconnaissance des entités nommées liées à la cybersécurité. Cette méthode ne nécessite pas de larges ensembles de données étiquetées pour le fine-tuning et offre une bonne généralisation grâce à la richesse des représentations contextuelles apprises. Cependant, ces modèles peuvent avoir une mauvaise détection des entités spécifiques à la cybersécurité si le pré-entraînement n'a pas suffisamment couvert le vocabulaire du domaine. De plus, ils peuvent être complexes à ajuster et exigeants en ressources computationnelles.

Le tableau 4.2 résume les différentes techniques d'extraction des Indicateurs de Compromission.

TABLE 4.2 – Techniques d'extraction des Indicateurs de Compromission.

Article	**Type de IoC**	**Approche**	**Source de données**	**Algorithme**	**Force(+)/Faiblesse(-)**
[53]	Atomiques Calculés	ML Règles	Rapports non structurés	SVM ; LR ; DG	+ Précision sur motifs connus. - Nécessite expertise pour la création des règles. - Nécessite de données étiquetées.
[64]	TTPs	Ontologies ML	Rapports non structurés	LSA ; NB ; KNN ; DT ; RF ; DNNs	+ Précision sur motifs connus. - Sensible au bruit. - Nécessitent mise à jour continue. - Besoins des algorithmes d'extraction des caractéristiques.
[63]	TTPs	Règles	Rapports APT	TF-IDF	+ Précision sur motifs connus. - Sensible au bruit. - Nécessite expertise pour la création des règles.
[38]	Atomiques TTPs	ML	Blogs	CFR	+ Robuste au bruit. - Nécessite de larges données étiquetées.
[7]	TTPs	ML	Rapports APT	SVM ; TF-IDF	+ Robuste au bruit. - Besoins des algorithmes d'extraction des caractéristiques. - Nécessite de larges données étiquetées.
[32]	Atomiques TTPs	Ontologies ML DL	Twitter	NB ; SVM ; LSTMs	+ Généralisation relativement bonne. - Nécessite de larges données étiquetées.

Table 4.2 – suite

Article	Type de IoC	Approche	Source de données	Algorithme	Force(+)/Faiblesse(-)
[55]	Entités en cybersécurité	DL	Rapports non structurés	Bi-LSTMs ; CFR	+ Généralisation relativement bonne. - Nécessite de larges données étiquetées.
[95]	Entités en cybersécurité	DL	Rapports non structurés	CNNs ; RNNs	+ Généralisation relativement bonne. - Nécessite de larges données étiquetées.
[85]	Entités en cybersécurité	DL	Rapports non structurés	Word2Vec ; GloVe ; FastText	+ Robuste au bruit. - Besoins des algorithmes d'extraction des caractéristiques.
[97]	Entités en cybersécurité	Modèles pré-entraînés	Rapports non structurés	BERT	+ Ne nécessite pas une large donnée étiquetée. - Détection d'entités liées à la cybersécurité relativement mauvaise
[37]	Entités en cybersécurité	Ontologies ML DL	Rapports non structurés	CNNs ; RNNs	+ Généralisation relativement bonne. - Nécessite de larges données étiquetées. - Nécessitent mise à jour continue.

4.6.3 Techniques d'attribution des cyberattaques

Après avoir présenté les différentes études liées aux techniques d'attribution des cyberattaques dans la section 4.4, nous procédons maintenant à une analyse approfondie de ces recherches en nous concentrant sur les approches employées. Ces approches peuvent être classées en deux grandes catégories :

Attribution basée sur des rapports de texte non structurés

Cette approche exploite les informations contenues dans les rapports d'incidents des menaces, qui sont généralement des documents textuels non structurés rédigés par des experts en cybersécurité. L'objectif est d'analyser ces rapports pour extraire des indices permettant d'attribuer une cyberattaque à un acteur spécifique.

[69] propose une méthode transformant l'attribution en une tâche de classification textuelle en utilisant un modèle de représentation vectorielle. Cette approche se distingue par sa capacité à identifier de nouveaux acteurs de menace et à enrichir les

profils des acteurs existants. Elle combine des données étiquetées limitées avec un corpus plus large de documents non étiquetés pour améliorer les performances du modèle. Cependant, l'article ne discute pas des défis liés à l'hétérogénéité et à la qualité des rapports, ce qui peut affecter l'efficacité du modèle dans des contextes réels variés. [42] développe une approche d'attribution basée sur l'apprentissage profond, utilisant des encodeurs de mots spécifiques au domaine générés par l'algorithme Word2Vec. Le modèle comprend plusieurs couches denses avec des mécanismes de régularisation comme le Dropout. Cette méthode surpasse les techniques traditionnelles et offre une meilleure compréhension contextuelle des relations entre les TTPs des différents groupes APT. Néanmoins, ses performances dépendent fortement de la qualité et de la diversité des rapports utilisés pour l'entraînement, ce qui peut poser des problèmes lorsque les données sont limitées ou biaisées. [75] introduit un réseau de neurones dense utilisant une technique pour améliorer la représentation des données textuelles. En exploitant les relations sémantiques dans les rapports, le modèle est capable de capturer des indices subtils pour une attribution plus précise. Toutefois, l'étude souligne les défis liés à l'équilibrage des classes et à la gestion des biais induits par des classes surreprésentées, ce qui peut affecter la généralisation du modèle.

Les approches basées sur l'analyse de textes non structurés offrent l'avantage de tirer parti d'informations contextuelles riches présentes dans les rapports de renseignement. Elles sont adaptables et peuvent identifier des acteurs même lorsque les indices techniques sont insuffisants. Cependant, elles sont fortement dépendantes de la qualité et de la diversité des données textuelles disponibles. Les défis incluent la gestion de l'hétérogénéité des rapports, la nécessité de prétraitements sophistiqués pour extraire des informations pertinentes, et la sensibilité aux biais dans les données d'entraînement.

Attribution basée sur l'analyse des logiciels malveillants

Cette approche se concentre sur l'analyse directe des logiciels malveillants impliqués dans les cyberattaques. Elle vise à extraire des caractéristiques techniques du malware, soit par des méthodes statiques (analyse du code binaire sans exécution), soit par des méthodes dynamiques (observation du comportement du malware lors de son exécution dans un environnement contrôlé).

[91] présente une méthode d'attribution en utilisant une analyse de similarité binaire basée sur des graphes de flux de contrôle attribués. L'approche utilise des shapelets pour une classification interprétable des malwares, améliorant la robustesse des classifications inter-plateformes. Cependant, elle dépend fortement de la qualité des fonctions filtrées, et son applicabilité peut être limitée en présence de données incomplètes ou fortement déséquilibrées. [40] propose le modèle MVFCC, un clustering consensuel flou multi-vue qui combine plusieurs perspectives d'analyse (statique, dynamique). Cette méthode gère efficacement les attaques multi-étapes et fournit des estimations précises de l'origine des outils utilisés. Néanmoins, le coût computationnel élevé et la complexité associée à la gestion de multiples perspectives peuvent être des obstacles à son déploiement à grande échelle. [41] introduit le framework APTMalInsight, qui combine

des analyses dynamiques des séquences d'appels système avec une modélisation ontologique pour la détection et la classification des malwares APT. Cette approche offre une compréhension approfondie des comportements des malwares et une représentation sémantique systématique. Toutefois, l'article ne fournit pas suffisamment de détails sur la scalabilité du cadre ontologique pour des environnements en temps réel ou à grande échelle.

Les approches basées sur l'analyse des logiciels malveillants permettent une attribution précise en se basant sur des caractéristiques techniques spécifiques du malware. Elles sont particulièrement efficaces pour identifier des signatures uniques ou des patterns comportementaux propres à certains groupes d'attaquants. Cependant, ces méthodes peuvent être contournées par des techniques d'obfuscation ou de polymorphisme utilisées par les attaquants pour masquer le code. De plus, l'analyse dynamique peut être lente et coûteuse en ressources, et nécessite des environnements sécurisés pour exécuter les malwares sans risque.

Le tableau 4.3 résume les différentes techniques d'attribution des cyberattaques.

TABLE 4.3 – Techniques d'attribution des cyberattaques.

Article	**Type de données**	**Type d'analyse**	**Approche**	**Forces(+)/Faiblesses(-)**
[69]	Rapports de texte non structurés	NLP	Classification	+Exploite le NLP pour traiter des informations contextuelles riches. -Problèmes de scalabilité avec un grand nombre de classes. -Incapacité à identifier de nouveaux acteurs de menace. -Performances dépendent de la qualité des rapports.
[91]	Logiciels malveillants	Statique	Classification	+Précision élevée dans la classification des malwares. -Risque que le même malware soit utilisé par différents acteurs. -Incapacité à identifier de nouveaux acteurs de menace. -Problèmes de scalabilité avec un grand nombre de classes.

Table 4.3 – suite

Article	Type de données	Type d'analyse	Approche	Forces(+)/Faiblesses(-)
[40]	Logiciels malveillants	Statique Dynamique	Regroupement	+Capacité à identifier de nouveaux acteurs de menace grâce au regroupement. +Précision élevée en combinant analyses statique et dynamique. -Complexité dans l'interprétation des clusters en acteurs spécifiques. -Risque que les malwares soient utilisés par plusieurs acteurs.
[41]	Logiciels malveillants	Dynamique	Classification	+Précision élevée grâce à l'analyse dynamique. -Risque que le même malware soit utilisé par différents acteurs. -Problèmes de scalabilité avec un grand nombre de classes. -Incapacité à identifier de nouveaux acteurs de menace.
[42]	Rapports de texte non structurés	NLP	Classification	+Traite des informations riches à partir des rapports. -Problèmes de scalabilité avec un grand nombre de classes. -Incapacité à identifier de nouveaux acteurs de menace. -Performances dépendent de la qualité des rapports.
[75]	Rapports de texte non structurés	NLP	Classification	+Nouvelle technique (SIMVER) pour améliorer la représentation textuelle. -Problèmes de scalabilité avec un grand nombre de classes. -Incapacité à identifier de nouveaux acteurs de menace. -Défis liés à l'équilibrage des classes.

4.6.4 Techniques de prédiction de niveau de gravité des cybermenaces

Après avoir présenté les différentes études liées aux techniques d'estimation de la gravité des cybermenaces dans la section 4.5, nous procédons à une analyse approfondie en nous concentrant sur les approches employées. Ces approches peuvent être classées en trois catégories principales :

Approches basées sur l'analyse statistique

L'article [21] examine les incohérences entre différentes sources de données dans l'évaluation de la gravité des vulnérabilités logicielles. En analysant des vulnérabilités provenant de Bugzilla, Mozilla Security Advisory et la NVD, les auteurs identifient six facteurs influençant ces incohérences, tels que les différences méthodologiques et les niveaux d'expertise des évaluateurs. Ils utilisent des modèles statistiques pour mesurer l'impact de ces incohérences sur les décisions critiques, en se concentrant particulièrement sur un système commun de notation des vulnérabilités.

Une force de cette approche réside dans sa capacité à révéler les facteurs systémiques qui conduisent à des évaluations incohérentes, ce qui est essentiel pour améliorer la fiabilité des processus de priorisation des vulnérabilités. En mettant en évidence que ces incohérences peuvent réduire les performances prédictives de modèles d'apprentissage automatique jusqu'à 77%, l'étude souligne l'importance de la qualité et de la cohérence des données pour le développement de modèles prédictifs efficaces. Cependant, une faiblesse de cette étude est qu'elle ne propose pas de solutions automatisées pour corriger ou harmoniser ces incohérences entre les sources de données. L'analyse est principalement descriptive et ne fournit pas de méthodologie pour intégrer de manière cohérente des données hétérogènes dans un modèle unifié d'estimation de la gravité. De plus, la focalisation sur un nombre limité de sources peut limiter la généralisation des conclusions à l'ensemble du paysage des vulnérabilités.

Approches basées sur l'apprentissage automatique pour la prédiction de l'exploitabilité

Dans [9], les auteurs proposent un cadre pour prédire l'exploitabilité des vulnérabilités en utilisant des techniques d'apprentissage automatique supervisé. Ils exploitent des bases de données telles que la NVD, Exploit-DB et ZDI pour collecter des caractéristiques comme les métriques CVSS, les types de vulnérabilités et les configurations logicielles. Des algorithmes tels que Random Forest, SVM, Naïve Bayes et régression logistique sont utilisés pour classifier les vulnérabilités en exploitables ou non.

Cette approche utilise multiples algorithmes de classification et de diverses caractéristiques, ce qui peut améliorer la robustesse et la précision de la prédiction. L'inclusion de métriques CVSS et de types de vulnérabilités permet au modèle de capturer à la fois des aspects quantitatifs et qualitatifs des vulnérabilités. Cependant,la dépendance

aux données historiques d'exploitations connues limite la capacité du modèle à généraliser aux vulnérabilités zéro-day ou aux nouvelles techniques d'exploitation qui n'ont pas encore été observées. De plus, les caractéristiques utilisées sont basées sur les métriques CVSS, qui peuvent elles-mêmes souffrir d'incohérences et de subjectivité, comme le souligne [21]. Cette dépendance à des données potentiellement inconsistantes peut affecter la fiabilité des prédictions du modèle.

Approches basées sur le NLP pour l'estimation de la gravité

Les articles [46] et [82] proposent des méthodes qui combinent le NLP avec des algorithmes d'apprentissage automatique et d'apprentissage profond pour estimer automatiquement la gravité des vulnérabilités en se basant sur leurs descriptions textuelles.

Dans [46], les auteurs utilisent des techniques d'extraction de caractéristiques telles que TF-IDF et des modèles d'embedding comme Word2Vec, Doc2Vec et FastText pour représenter les descriptions textuelles des vulnérabilités. Ils appliquent ensuite des algorithmes de classification tels que Naïve Bayes, Decision Tree, KNN, MLP et Random Forest pour effectuer une classification multi-classes de la gravité.

Cette approche a la capacité à automatiser le processus d'estimation de la gravité, réduisant ainsi les délais et les erreurs humaines. En combinant différentes méthodes d'extraction de caractéristiques et de classification, le modèle vise à capitaliser sur les forces de chaque technique pour améliorer les performances globales. Cependant, les modèles d'embedding traditionnels utilisés (comme Word2Vec et Doc2Vec) ne prennent pas en compte le contexte global et la sémantique à l'échelle de la phrase ou du document. Ces modèles génèrent des embeddings basés sur des contextes locaux limités, ce qui peut entraîner une perte d'informations contextuelles essentielles pour une estimation précise de la gravité. Par exemple, l'ambiguïté lexicale et les relations sémantiques complexes dans les descriptions de vulnérabilités peuvent ne pas être correctement capturées, ce qui affecte la précision du modèle.

Dans [82], les auteurs utilisent l'algorithme d'embeddings GloVe pour représenter les mots des descriptions textuelles et appliquent des réseaux neuronaux convolutionnels pour classer les vulnérabilités en risques élevés, moyens ou faibles. L'utilisation des CNNs permet de capturer des motifs locaux dans les données textuelles. La force de cette approche réside dans l'application de l'apprentissage profond pour extraire automatiquement des caractéristiques complexes à partir des données textuelles, ce qui peut potentiellement améliorer la précision de la classification. Néanmoins, les CNNs sont limités dans leur capacité à capturer les dépendances à longue portée et la structure sémantique globale du texte. De plus, les embeddings GloVe sont des vecteurs statiques qui n'adaptent pas les représentations des mots en fonction du contexte, ce qui peut entraîner une mauvaise interprétation des mots polysémiques ou des expressions contextuelles critiques pour l'évaluation de la gravité.

Le tableau 4.4 résume les différentes techniques d'estimation de la gravité pour la gestion des risques.

TABLE 4.4 – Techniques de prédiction de la gravité des cybermenaces.

Article	**Source de données**	**Méthode d'embedding**	**Algorithmes**	**Forces (+)/Faiblesses (-)**
[21]	Bugzilla Mozilla Security NVD	Pas d'embedding	Méthodes statistiques	- Dépendance aux sources multiples. - Ne fournit pas de solutions automatisées.
[9]	NVD Exploit-DB ZDI	Pas d'embedding	RF ; SVM ; NB	+ Précision élevée dans la prédiction de l'exploitabilité. + Utilisation de multiples algorithmes de classification - Dépendance aux données historiques d'exploitations connues. - Nécessite des données étiquetées exploitables/non exploitables.
[46]	ML DL	Word2Vec FastText Doc2Vec	NB ; DT ; RF ; DNNs	+ Automatisation du processus d'estimation de la gravité. + Combine différentes méthodes d'extraction de caractéristiques - Les modèles d'embedding utilisés ne capturent pas le contexte et la sémantique du texte.
[82]	CVE	GloVe	CNNs	+ Utilisation de l'apprentissage profond pour extraire automatiquement des caractéristiques. - Les modèles d'embedding utilisés ne capturent pas le contexte et la sémantique du texte.

Les limites identifiées dans l'état de l'art pour les quatre volets de notre thèse ne sont pas simplement théoriques : elles reflètent des enjeux concrets pour les utilisateurs finaux, qu'il s'agisse d'analystes en cybersécurité, d'opérateurs SOC ou de décideurs techniques. En particulier, la capacité des modèles à réduire efficacement les faux positifs et les faux négatifs demeure un défi central, car elle conditionne la fiabilité des systèmes d'aide à la décision. Ces lacunes compromettent la réactivité face aux menaces réelles et peuvent entraîner une surcharge opérationnelle ou, à l'inverse, une sous-estimation des risques. Ces constats motivent directement les contributions de notre travail, qui vise à proposer des approches à la fois robustes, adaptées au contexte

cyber et exploitables dans des environnements opérationnels.

4.7 Conclusion

L'examen approfondi des différentes techniques présentées dans ce chapitre met en évidence l'évolution constante des méthodes de CTI et les défis auxquels sont confrontées les organisations en matière de cybersécurité. Les approches de collecte d'informations jouent un rôle crucial en fournissant les données nécessaires pour détecter et anticiper les menaces émergentes. Cependant, la diversité des sources et la qualité variable des informations recueillies posent des problèmes de fiabilité et de pertinence.

Les techniques d'extraction des IoCs ont montré des progrès significatifs, notamment grâce à l'intégration de modèles d'apprentissage automatique et profond. Néanmoins, des défis subsistent quant à la nécessité de grandes quantités de données annotées et à la capacité des modèles à capturer le contexte et la sémantique spécifiques au domaine de la cybersécurité.

En ce qui concerne l'attribution des cyberattaques, les méthodes actuelles offrent des perspectives intéressantes pour identifier les acteurs malveillants. Les approches basées sur l'analyse des rapports textuels apportent une compréhension contextuelle riche, tandis que l'analyse des logiciels malveillants fournit des preuves techniques tangibles. Toutefois, la scalabilité et la capacité à identifier de nouveaux acteurs restent des défis majeurs.

Enfin, les techniques d'estimation de la gravité sont essentielles pour la gestion efficace des risques. Les modèles actuels, qu'ils soient basés sur l'apprentissage automatique supervisé ou sur le traitement du langage naturel, contribuent à automatiser et à améliorer le processus de priorisation. Cependant, ils sont limités par la qualité des données d'entrée, les incohérences dans les évaluations et les contraintes computationnelles.

Chapitre 5

Détection en temps réel d'informations sur les cybermenaces à partir des réseaux sociaux

5.1 Introduction

La CTI est devenue un composant crucial des stratégies modernes de cybersécurité, permettant aux organisations d'identifier et de mitiger proactivement les menaces potentielles en collectant et en analysant des données provenant de diverses sources. La CTI implique la transformation d'informations brutes en intelligences exploitables à travers des techniques rigoureuses et structurées menées par des experts en cybersécurité [14]. Ces informations traitées incluent des détails sur les cyberattaques provenant des journaux de pare-feu, des systèmes de détection d'intrusion, des malwares, des courriels de phishing, et d'autres artefacts connexes.

L'OSINT désigne le processus de collecte d'informations sur les menaces disponibles publiquement à partir de diverses sources telles que des articles de presse, des réseaux sociaux, des blogs, des forums, et des sites web [16]. Twitter, en tant que plateforme sociale majeure, sert de source riche d'OSINT à des fins de cybersécurité, avec des milliers de professionnels de la cybersécurité, d'entreprises, et du grand public rapportant en temps réel des informations sur de nouvelles vulnérabilités, des exploits, et des cyberattaques en cours sous forme de tweets. Ces informations partagées peuvent être collectées et analysées pour détecter et surveiller des événements, aidant à comprendre les acteurs de la menace, leurs motivations, et leurs TTPs [87].

Les tweets sont souvent rédigés en langage naturel avec un format de texte non structuré. Pour exploiter le grand nombre de tweets publiés quotidiennement sur Twitter, plusieurs travaux ont été proposés pour automatiser la tâche d'analyse des tweets, et de nombreuses approches utilisent le NLP pour la détection en temps réel des informations sur les cybermenaces. Les techniques NLP traditionnelles échouent souvent à identifier et classifier précisément les tweets en raison de leur dépendance à des règles statiques

et de l'évolution continue des nouvelles cyberattaques. De plus, diverses architectures de réseaux neuronaux, telles que les réseaux de neurones convolutifs, les mémoires à long terme unidirectionnelles/bidirectionnelles, ou des combinaisons de celles-ci, ont été utilisées pour extraire des entités pertinentes des tweets et identifier des événements de cybersécurité. Cependant, ces types de modèles nécessitent une grande quantité de données pour l'entraînement et ne sont pas efficaces pour capturer la sémantique du texte, ce qui implique une mauvaise généralisation sur de nouvelles données.

Les LLMs, tels que BERT [29], ont révolutionné le NLP en fournissant des capacités avancées pour comprendre et générer le langage humain. BERT, qui est un encodeur basé sur les Transformeurs [90], est entraîné sur des quantités massives de données, pouvant être affiné pour des tâches spécifiques avec des jeux de données plus petits, telles que la reconnaissance d'entités nommées, l'analyse de sentiment, et la classification de texte, conduisant à de meilleures performances et une convergence plus rapide.

Dans ce chapitre, nous proposons une approche basée sur l'apprentissage par renforcement pour l'analyse en temps réel et la détection des informations pertinentes sur les cybermenaces au sein des tweets. Un tweet est identifié comme pertinent s'il contient des informations liées à la CTI telles que des vulnérabilités, des exploits, des malwares, etc.

Notre approche combine le modèle pré-entraîné BERT pour l'encodage des tweets, générant des embeddings, avec un Deep Q-Network (DQN) optimisé, spécifiquement conçu pour prendre des décisions basées sur ces embeddings. La principale justification de l'utilisation de cette combinaison est que la capacité de BERT à capturer à la fois l'information sémantique et contextuelle d'un tweet le rend particulièrement adapté pour comprendre les nuances du contenu des tweets. D'autre part, le DQN offre un équilibre entre l'exploration et l'exploitation, contrairement aux modèles d'apprentissage supervisé standard, qui sont plus exploitants et dépendent fortement des données préexistantes. Cela rend le DQN plus adaptable pour la détection en temps réel sur Twitter, où les tweets sont continuellement partagés et de nouveaux schémas peuvent émerger. En outre, la capacité du DQN à apprendre au fil du temps grâce au mécanisme de relecture d'expérience lui permet d'évoluer sans nécessiter une nouvelle phase d'entraînement, contrairement aux modèles supervisés, qui nécessitent une phase d'entraînement supplémentaire pour intégrer de nouvelles données.

Le modèle proposé commence par la réception en temps réel de tweets à l'aide de l'API Twitter. Chaque tweet collecté est encodé en utilisant un modèle pré-entraîné BERT, le transformant en une représentation vectorielle dense. Le tweet encodé est ensuite passé à travers l'agent DQN avec une politique entraînée pour décider si le tweet représente ou non une information pertinente sur une cybermenace.

Les principales contributions de ce travail proposé sont les suivantes :

— Nous introduisons un modèle conçu pour l'analyse et la détection des informations sur les cybermenaces au sein des tweets.

— Nous optimisons l'algorithme DQN spécifiquement adapté pour la détection des informations sur les menaces cybernétiques dans les données de tweets non structurés, en intégrant des mécanismes d'encodage de texte avancés tels que

BERT.

— Nous étudions et comparons les effets de diverses méthodes d'embedding de texte, telles que BERT, RoBERTa, ALBERT, et DistilBERT, sur les capacités de prise de décision de l'agent DQN pour la détection des informations sur les cybermenaces.

Le reste de ce chapitre est organisé comme suit. L'architecture du modèle proposé est expliquée dans la section 5.2, suivie des résultats expérimentaux dans les sections 5.3 et 5.4. Enfin, la conclusion est tirée dans la section 5.5.

5.2 Modèle proposé

5.2.1 Conception du modèle pour la détection en temps réel

Dans cette section, nous présentons la conception du système intelligent pour la détection en temps réel. Le modèle, illustré dans la Figure 5.1, comprend quatre modules principaux : l'API Twitter, le Fournisseur de Tweets, l'Encodeur de Tweets, et l'agent DQN entraîné.

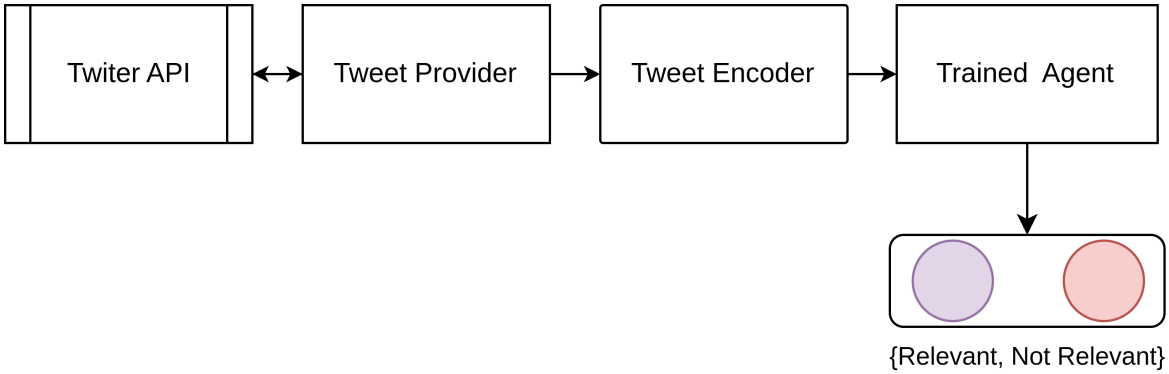


FIGURE 5.1 – Conception du modèle proposé pour la détection en temps réel des informations liées aux cybermenaces dans les tweets.

Le module API Twitter "Twitter API" sert de point d'entrée pour notre système, permettant la récupération en temps réel des tweets liés à la cybersécurité. Diverses approches de récupération de données sont disponibles via l'API Twitter [39]. Dans notre travail, nous adoptons une méthode de recherche par mots-clés. Cette approche capture les tweets et retweets contenant des mots-clés spécifiés, garantissant que notre modèle surveille et collecte en continu des données pertinentes pour une analyse en temps réel.

Le module Fournisseur de Tweets "Tweet Provider" agit comme un intermédiaire entre l'API Twitter et le reste du système. Il est chargé de filtrer les tweets en double. Chaque tweet capturé passe par un pipeline de prétraitement NLP, qui consiste à supprimer les signes de ponctuation (par ex., ".", ",", ";", "?", "!") pour réduire

le bruit et à standardiser la casse des lettres en minuscules. De plus, il effectue une tokenisation, convertissant le texte pré-traité en tokens.

L'Encodeur de Tweets "Tweet Encoder" est le module qui analyse le tweet pré-traité et le convertit en une représentation numérique. Dans notre approche, nous utilisons un modèle de langage de grande taille pré-entraîné (LLMs), en particulier le modèle BERT, qui prend en entrée un tweet tokenisé donné et génère un vecteur d'embedding qui encapsule la représentation sémantique et contextuelle du tweet.

L'agent DQN "Trained Agent" observe un état en tant qu'entrée, qui est l'embedding BERT d'un tweet, le passe à travers le réseau profond Q avec une politique entraînée, puis produit une action correspondant à la décision de classer le tweet comme indiquant ou non une information de cybermenace.

5.2.2 Modélisation du système pour l'entraînement de l'agent DQN

L'apprentissage par renforcement profond est un domaine de l'intelligence artificielle utilisé pour résoudre des problèmes de prise de décision [66]. Cette méthode implique un agent interagissant avec son environnement, recevant des retours sous forme de récompenses, et utilisant ces retours pour améliorer sa politique au fil du temps. Le DQN, qui est un type d'algorithme d'apprentissage par renforcement, utilise un réseau de neurones pour approximer la valeur Q [62], contrairement à l'approche Q-learning traditionnelle qui repose sur une approche tabulaire ou une approximation par fonction linéaire. Cette approche permet à DQN d'apprendre une politique π à partir d'espaces d'état de haute dimension [61].

Cette section présente la modélisation de notre système basée sur les composantes fondamentales de la discipline de l'apprentissage par renforcement, telles que l'Environnement et l'Agent, comme illustré dans la Figure 5.2.

Environnement

Dans notre modèle, l'environnement représente le cadre externe avec lequel l'agent DQN interagit. Il fournit les données d'entrée nécessaires, permettant à l'agent d'observer, de traiter et d'agir sur les informations provenant des tweets. L'environnement se compose de plusieurs composantes clés, y compris le Fournisseur de Tweets, l'Encodeur de Tweets, l'État, l'Espace d'Actions, et la Fonction de Récompense, qui seront expliqués en détail ci-dessous.

Le premier module de l'environnement, le Fournisseur de Tweets, génère des tweets aléatoires pour l'analyse. Il est modélisé comme un ensemble de données annotées $\mathcal{D} = \{(T_i, c_i)\}_{i=1}^{N}$, où T_i représente un tweet et c_i est une étiquette binaire indiquant si le tweet est lié à une information de cybermenace ($c_i = 1$) ou non ($c_i = 0$). Cet ensemble de données sert à entraîner l'agent DQN en fournissant des exemples de tweets pertinents et non pertinents.

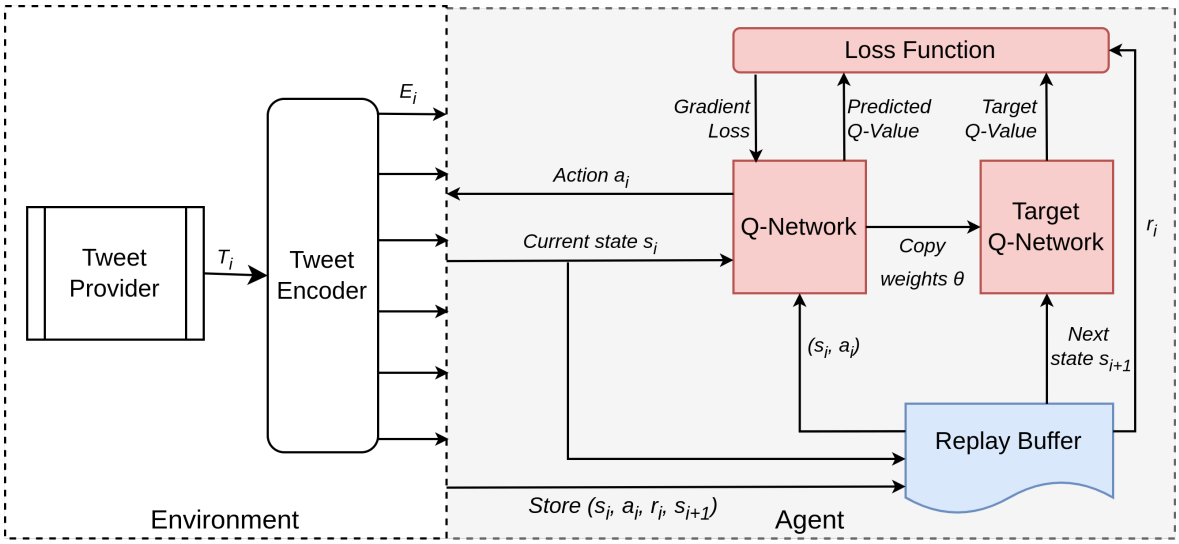


FIGURE 5.2 – Modélisation du système basé sur l'apprentissage par renforcement pour la détection des informations sur les cybermenaces dans les tweets.

L'Encodeur de Tweets convertit les données brutes des tweets en un format numérique significatif que l'agent peut traiter. À chaque itération d'entraînement, l'Encodeur de Tweets reçoit un tweet aléatoire T_i du Fournisseur de Tweets et utilise un modèle BERT pré-entraîné pour le transformer en une représentation vectorielle E_i. Ce vecteur encapsule la signification sémantique et contextuelle du tweet, représentée comme $E_i = \text{BERT}(T_i)$.

L'État s_i dans notre système d'apprentissage par renforcement est l'embedding BERT d'un tweet, tel que généré par le module Fournisseur de Tweets. L'état $s_i = E_i$ représente l'entrée actuelle que l'agent DQN utilise pour améliorer sa politique de décision. Cet état fournit à l'agent une représentation détaillée du contenu du tweet, lui permettant d'évaluer si le tweet pourrait indiquer une information de menace.

L'Espace d'Actions A définit l'ensemble des actions possibles que l'agent DQN peut entreprendre en réponse à l'état actuel. Dans notre cas, l'espace d'actions se compose de deux actions discrètes $A = \{a_0, a_1\}$, où a_0, a_1 correspondent à la classification du tweet comme indicatif ou non indicatif d'une information de cybermenace. L'agent DQN sélectionne une action $a_i \in A$ en fonction de l'état actuel s_i, dans le but de maximiser la récompense cumulative attendue R.

La Fonction de Récompense r_i fournit un retour d'information à l'agent DQN concernant la justesse de ses actions, guidant l'agent pour améliorer son processus de prise de décision au fil du temps. La fonction de récompense est définie comme $r_i = \{R_{\text{positive}}, R_{\text{negative}}\}$, où R_{positive} est accordée lorsque l'agent classe correctement un tweet, et R_{negative} est accordée lorsque la classification est incorrecte. L'agent vise à apprendre une politique $\pi(s)$ qui maximise la récompense cumulative attendue R, calculée avec la formule 5.1, où $\gamma \in [0, 1]$ est le facteur de réduction qui détermine

l'importance des récompenses futures, et T est le nombre d'étapes par épisode.

$$R = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)\right] \tag{5.1}$$

Agent

Dans notre modèle d'apprentissage par renforcement, l'agent est représenté par un agent DQN, responsable de prendre des décisions basées sur l'état actuel fourni par l'environnement. L'agent interagit avec l'environnement en sélectionnant des actions dans l'espace d'actions et en mettant à jour sa politique de décision grâce à l'entraînement. Les composantes clés de l'agent incluent le Réseau-Q, le Réseau-Q Cible, la mémoire de Relecture, et la Fonction de Perte.

Le Réseau-Q est un réseau de neurones profond qui approxime la fonction de valeur d'action $Q(s_i, a_i; \theta)$, où s_i est l'état actuel, a_i est l'action prise, et θ représente les poids du réseau. Le Réseau-Q reçoit l'état actuel s_i, qui est l'embedding BERT du tweet, et produit les valeurs Q pour toutes les actions possibles dans l'espace d'actions. Sur la base de ces valeurs Q, l'agent sélectionne l'action qui maximise la récompense cumulative attendue R.

Le Réseau-Q Cible est un réseau séparé utilisé pour stabiliser le processus d'entraînement du Réseau-Q. Alors que le Réseau-Q est continuellement mis à jour pendant l'entraînement, les poids du Réseau-Q Cible, θ', sont mis à jour moins fréquemment en copiant les poids du Réseau-Q. Le Réseau-Q Cible fournit les valeurs Q-cibles, $Q'(s_{i+1}, a_{i+1}; \theta')$, qui sont utilisées pour calculer l'erreur de différence temporelle lors du calcul de la perte.

La Mémoire de relecture $\mathcal{M}$ est une composante qui stocke les expériences de l'agent (s_i, a_i, r_i, s_{i+1}) au fil du temps. Ces expériences sont des tuples de l'état actuel, de l'action prise, de la récompense reçue, et de l'état suivant. Au lieu d'apprendre directement à partir d'expériences consécutives, l'agent DQN échantillonne aléatoirement des mini-lots d'expériences de la Mémoire de relecture.

La Fonction de Perte est utilisée pour minimiser l'erreur entre les valeurs Q prédites par le Réseau-Q et les valeurs Q-cibles du Réseau-Q Cible. La perte est calculée comme l'erreur quadratique moyenne entre la valeur Q prédite $Q(s_i, a_i; \theta)$ et la valeur cible $r_i + \gamma \max Q'(s_{i+1}, a_{i+1}; \theta')$, où r_i est la récompense reçue, et γ est le facteur d'actualisation. Le gradient de la perte est utilisé pour mettre à jour les poids θ du Réseau-Q via la rétropropagation.

5.2.3 Algorithme d'entraînement

Le processus d'entraînement de l'agent DQN avec la relecture d'expérience, comme indiqué dans l'Algorithme 1, implique l'initialisation de la mémoire de relecture $\mathcal{M}$, du Réseau-Q, et du Réseau-Q Cible avec des poids θ et θ', ainsi que l'initialisation du

Fournisseur de Tweets modélisé par l'ensemble de données étiquetées $\mathcal{D}$, et la définition de la taille de lot.

Chaque épisode parmi les M épisodes commence par l'encodage d'un tweet aléatoire pour initialiser l'état s_i. Une politique ϵ-greedy est utilisée pour sélectionner les actions $a_i \in A$. L'agent DQN exécute l'action sélectionnée et reçoit un retour de l'environnement sous forme de récompense $r_i \in \{R_{\text{positive}}, R_{\text{negative}}\}$, en fonction de la justesse de l'action, puis observe l'état suivant s_{i+1}. Les transitions (s_i, a_i, r_i, s_{i+1}) sont stockées dans $\mathcal{M}$ pour être utilisées lors de la relecture de l'expérience.

Au cours de chaque épisode, ce processus se poursuit pendant T étapes. Lorsque la taille de $\mathcal{M}$ dépasse la taille de lot, un mini-lot de transitions est échantillonné pour entraîner le Réseau-Q. L'entraînement consiste à minimiser la perte entre les valeurs Q prédites et les valeurs Q-cibles, calculée à l'aide du Réseau-Q Cible. Périodiquement, le Réseau-Q Cible est mis à jour pour correspondre au Réseau-Q afin d'assurer un entraînement stable.

Le nombre d'épisodes M et d'étapes T est crucial pour garantir une exploration et un apprentissage adéquats, permettant au modèle d'améliorer ses performances au fil du temps en mettant à jour de manière itérative la politique $\pi(s)$ pour mieux prédire les récompenses associées aux différentes actions.

5.3 Expériences

5.3.1 Jeu de données

Le jeu de données utilisé pour entraîner notre agent basé sur l'apprentissage par renforcement est référencé dans [31]. Initialement, les auteurs ont utilisé un ensemble de comptes Twitter sélectionnés manuellement pour collecter des données en utilisant le streaming de l'API de Twitter. Les tweets ont été collectés sur deux périodes distinctes : du 21 novembre 2016 au 27 mars 2017, et du 1er juin 2018 au 1er septembre 2018. Pour garantir la pertinence, les auteurs ont utilisé un ensemble de mots-clés définis par l'utilisateur pour filtrer les tweets collectés. De plus, le jeu de données a été annoté manuellement pour deux tâches principales : la reconnaissance d'entités nommées pour extraire des entités liées à la cybersécurité, et la classification binaire pour indiquer si un tweet mentionne ou non une information de cybermenace. Des exemples de tweets annotés sont illustrés dans le Tableau 5.1.

Dans notre expérience, nous avons partitionné le jeu de données en deux sous-ensembles : 85% des données ont été intégrées dans l'environnement d'apprentissage par renforcement pour entraîner l'agent DQN, comme décrit dans la Section 5.2.2, et 15% ont été réservés pour tester l'agent DQN. Le Tableau 5.2 présente les statistiques des données expérimentales.

Algorithm 1 Le processus d'entraînement de l'agent DQN avec relecture d'expérience pour la détection des informations sur les cybermenaces dans les tweets.

Initialiser la mémoire de relecture $\mathcal{M}$ avec une capacité N
Initialiser la fonction de valeur d'action Q avec des poids aléatoires θ
Initialiser la fonction de valeur d'action cible Q' avec des poids $\theta' = \theta$
Initialiser le Fournisseur de Tweets $\mathcal{D}$
Définir la taille du lot
for épisode = 1 à M **do**
    Initialiser l'état s_i en encodant un tweet aléatoire T_i du Fournisseur de Tweets $\mathcal{D}$
    for t = 1 à T **do**
        Avec une probabilité ϵ, sélectionner une action aléatoire $a_i \in A$
        Sinon, sélectionner $a_i = \arg\max Q(s_i, a_i; \theta)$
        Exécuter l'action a_i et observer la récompense r_i et le tweet suivant T_{i+1}
        Encoder T_{i+1} pour produire s_{i+1}
        Stocker la transition (s_i, a_i, r_i, s_{i+1}) dans $\mathcal{M}$
        $s_i \leftarrow s_{i+1}$
        if taille de $\mathcal{M} \geq$ taille_du_lot **then**
            Échantillonner un mini-lot de transitions (s_j, a_j, r_j, s_{j+1}) de $\mathcal{M}$
            for chaque (s_j, a_j, r_j, s_{j+1}) **do**
                $y_j \leftarrow r_j$ si s_{j+1} est terminal, sinon $y_j \leftarrow r_j + \gamma \arg\max Q'(s_{j+1}, a_{j+1}; \theta')$
                Mettre à jour le Réseau-Q en minimisant $(y_j - Q(s_j, a_j; \theta))^2$
            end for
        end if
        Toutes les C étapes, mettre à jour le Réseau-Q Cible $\theta \leftarrow \theta'$
    end for
end for

5.3.2 Configuration du modèle

Dans cette étude, nous avons expérimenté trois architectures différentes pour le Réseau-Q et le Réseau-Q Cible : un perceptron multicouche (MLP), un réseau de neurones convolutifs (CNN), et un réseau de mémoire à long court terme bidirectionnelle (BiLSTM). Pour l'encodage des tweets, nous avons utilisé le modèle BERT comme notre modèle de base tout en explorant également RoBERTa, ALBERT, et DistilBERT. Les architectures du Réseau-Q et du Réseau-Q Cible sont décrites dans le Tableau 5.3.

5.3.3 Paramètres expérimentaux

Nos expériences ont été réalisées en utilisant un GPU NVIDIA GeForce RTX 2070 avec 16GB de mémoire, et toutes les phases du modèle proposé ont été implémentées

Tweet	**Étiquette**
Astuce de cybersécurité pour Windows 10 : gardez votre compte Microsoft sécurisé avec une authentification à deux facteurs. #infosec	Non-Pertinent
Une vulnérabilité exploitable de type XML external entity CVE-2017-16349 existe dans la fonctionnalité de reporting de SAP BPC.	Pertinent
Une vulnérabilité d'escalade de privilèges locale CVE-2016-7255 existe dans le noyau Microsoft Windows, spécifiquement dans le fichier win32k.sys.	Pertinent
Une vulnérabilité de débordement de tampon local CVE-2016-7425 existe dans le pilote SCSI arcmsr du noyau Linux.	Pertinent

TABLE 5.1 – Exemple de tweets annotés.

Jeu de données	**Tweets pertinents**	**Tweets non pertinents**	**Total**
Entraînement	11,073	20,208	31,281
Test	2,164	4,093	6,257
Total	13,237	24,301	37,538

TABLE 5.2 – Statistiques des données expérimentales.

Réseau	**Type de couche**	**Nombre**	**Paramètres**
MLP	Complètement Connectée	5	1024, 2048, 2048, 512, 32 unités
	Activation	5	ReLU
	Dropout	1	0.3
CNN	Convolutionnelle	3	32, 64, 128 filtres
	Complètement Connectée	3	24*32*128, 256, 128 unités
	Activation	6	ReLU
BiLSTM	LSTM	2	256 unités cachées, bidirectionnel
	Complètement Connectée	3	128, 64 unités
	Activation	3	ReLU

TABLE 5.3 – Architectures du Réseau-Q et du Réseau-Q Cible.

avec PyTorch. L'agent DQN a été entraîné pour $M = 3000$ épisodes, chacun consistant en $T = 50$ étapes, avec une mémoire de relecture de $N = 10000$, à travers toutes les configurations. Les hyperparamètres d'entraînement incluaient : l'optimiseur Adam, une taille de lot de 64, un taux d'apprentissage de 0.001, et un facteur d'actualisation $\gamma = 0.99$. La structure des récompenses était cohérente dans toutes les configurations, avec $R_{positive} = +1$ et $R_{negative} = -1$. La fonction de perte utilisée était l'Erreur Quadratique Moyenne entre les valeurs Q prédites par le Réseau-Q et les valeurs Q-cibles qui maximisent la récompense cumulative attendue.

5.3.4 Métriques d'évaluation

L'agent DQN entraîné a été évalué en utilisant les métriques de précision, de rappel, de score F1, et d'exactitude.

— **Accuracy** : mesure la proportion de prédictions correctes parmi le nombre total de cas.

$$\text{Accuracy} = \frac{\text{Vrais Positifs} + \text{Vrais Négatifs}}{\text{Population Totale}}$$

— **Précision** : mesure la précision des classes correctement identifiées parmi toutes les classes prédites par le modèle.

$$\text{Précision} = \frac{\text{Vrais Positifs}}{\text{Vrais Positifs} + \text{Faux Positifs}}$$

— **Rappel** : évalue la capacité du modèle à identifier toutes les classes réelles au sein du jeu de données de test.

$$\text{Rappel} = \frac{\text{Vrais Positifs}}{\text{Vrais Positifs} + \text{Faux Négatifs}}$$

— **Score F1** : la moyenne harmonique de la précision et du rappel, fournissant une mesure équilibrée de la performance du modèle en termes de précision et de rappel.

$$\text{Score F1} = \frac{2 \cdot \text{Précision} \cdot \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

5.4 Analyse des résultats

5.4.1 Performances du modèle

La Figure 5.3 présente les récompenses d'entraînement sur 3000 épisodes pour l'agent DQN utilisant différentes architectures de Réseau-Q et de Réseau-Q Cible : MLP, CNN, et BiLSTM, comme expliqué dans la Section 5.3.2. Cette analyse utilise le modèle d'embedding BERT comme référence. Les récompenses cumulées sont normalisées entre 0 et 1, en considérant que les récompenses cumulées maximales qu'un agent DQN peut atteindre en un épisode sont égales à $T \times R_{positive} = 50$.

Les récompenses cumulées de l'agent DQN commencent à se stabiliser autour de l'épisode 1000. Cela indique la progression de l'apprentissage de l'agent et sa convergence vers une politique optimale au fur et à mesure de l'entraînement. En comparant les architectures, les architectures CNN et BiLSTM permettent à l'agent DQN d'atteindre environ 90% des récompenses cumulées totales. L'architecture BiLSTM montre une légère amélioration par rapport à la CNN, indiquant de meilleures capacités de généralisation. En revanche, l'architecture MLP présente une performance significativement inférieure comparée à CNN et BiLSTM, l'agent DQN perdant plus de 40% des récompenses en un seul épisode.

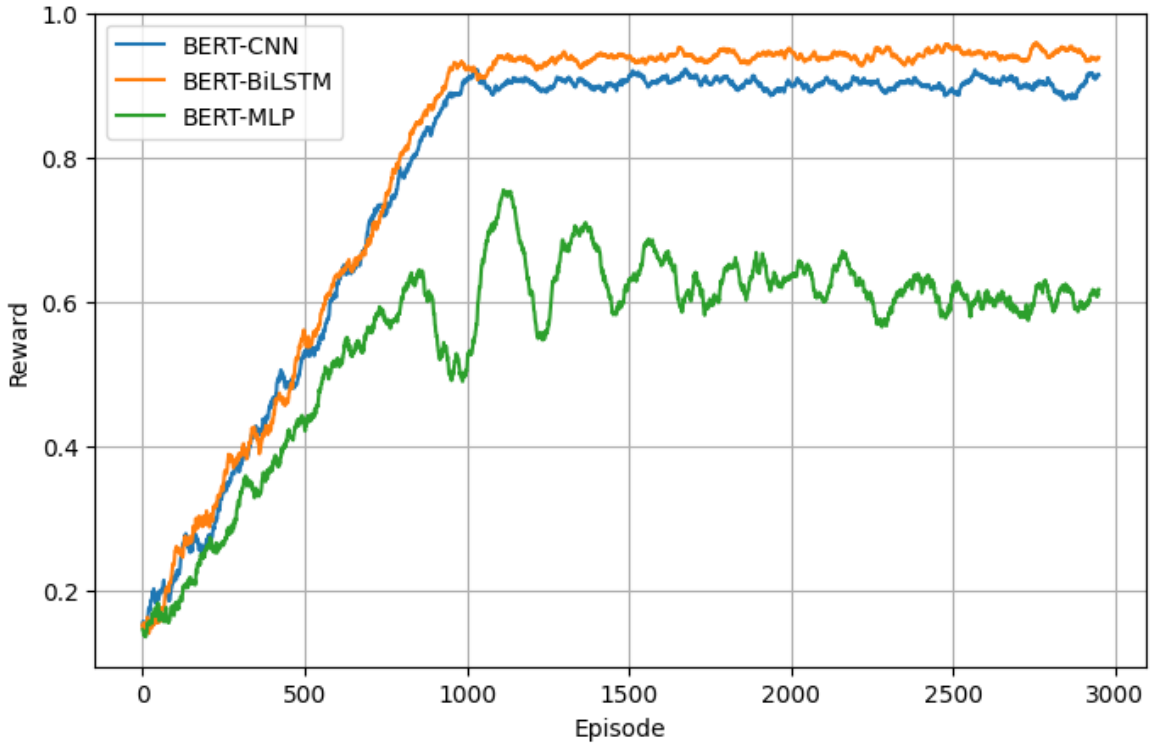


FIGURE 5.3 – Récompenses d'entraînement de l'agent DQN avec différentes architectures de Réseau-Q et de Réseau-Q Cible utilisant le modèle d'embedding BERT.

L'agent DQN entraîné est ensuite évalué avec le jeu de données de test décrit dans la Section 5.3.1 pour mesurer ses capacités à détecter les informations pertinentes sur les cybermenaces. Les résultats montrent que l'agent performe mieux lorsqu'il utilise BiLSTM comme Réseau-Q et Réseau-Q Cible, atteignant 97,6% d'exactitude et 96,5% de score F1. Le Tableau 5.4 illustre les métriques de performance sur le jeu de données de test.

Réseau Q/Cible Architecture	Métriques de Performance %			
	Accuracy	**Précision**	**Rappel**	**Score F1**
MLP	96.3	**95.9**	93.2	94.5
CNN	93.6	88.5	93.7	91
BiLSTM	**97.6**	95.5	**97.6**	**96.5**

TABLE 5.4 – Métriques de performance de l'agent DQN avec différentes architectures de Réseau-Q et de Réseau-Q Cible utilisant le modèle d'embedding BERT.

Modèle d'embedding	Réseau Q/Cible Network	Accuracy	Précision	Rappel	Score F1
BERT	MLP	96.3	95.9	93.2	94.5
	CNN	93.6	88.5	93.7	91
	BiLSTM	97.6	95.5	**97.6**	96.5
RoBERTa	MLP	95.7	95.6	92	93.8
	CNN	95.2	91.4	93.4	92.9
	BiLSTM	**98.2**	**97.5**	97.2	**97.4**
ALBERT	MLP	93.6	92.4	89.9	91.1
	CNN	94.5	90.8	93.6	92.2
	BiLSTM	96.9	94.1	97.1	95.6
DistilBERT	MLP	96	92.6	96.2	94.4
	CNN	95.9	96.1	92	94
	BiLSTM	97.4	95.6	96.8	96.2

TABLE 5.5 – Performances de l'agent DQN avec différentes configurations de Réseau-Q, Réseau-Q Cible et divers modèles d'embedding.

5.4.2 Effets des techniques d'embedding

Les performances de l'agent DQN avec différentes techniques d'embedding et configurations de Réseau-Q et de Réseau-Q Cible sont résumées dans le Tableau 5.5. En utilisant le modèle BERT comme référence, nous observons que tous les modèles d'embedding atteignent leurs meilleures performances avec l'architecture BiLSTM, soulignant sa capacité supérieure à capturer les dépendances temporelles. Les embeddings de RoBERTa obtiennent la meilleure performance globale, avec une précision de 98,2% et un score F1 de 97,4%. ALBERT montre également des résultats compétitifs, maintenant un rappel élevé de 97,1% et un solide score F1 de 95,6%. DistilBERT, une version allégée de BERT, performe bien, avec un score F1 de 96,2% et une précision de 97,4%.

La Figure 5.4 illustre la durée d'entraînement de l'agent DQN avec différents modèles d'embedding (BERT, RoBERTa, ALBERT, et DistilBERT) et architectures de Réseau Q, Réseau-Q Cible (MLP, CNN, et BiLSTM). Il est évident que BERT nécessite le temps d'entraînement le plus long pour toutes les architectures, BiLSTM prenant le

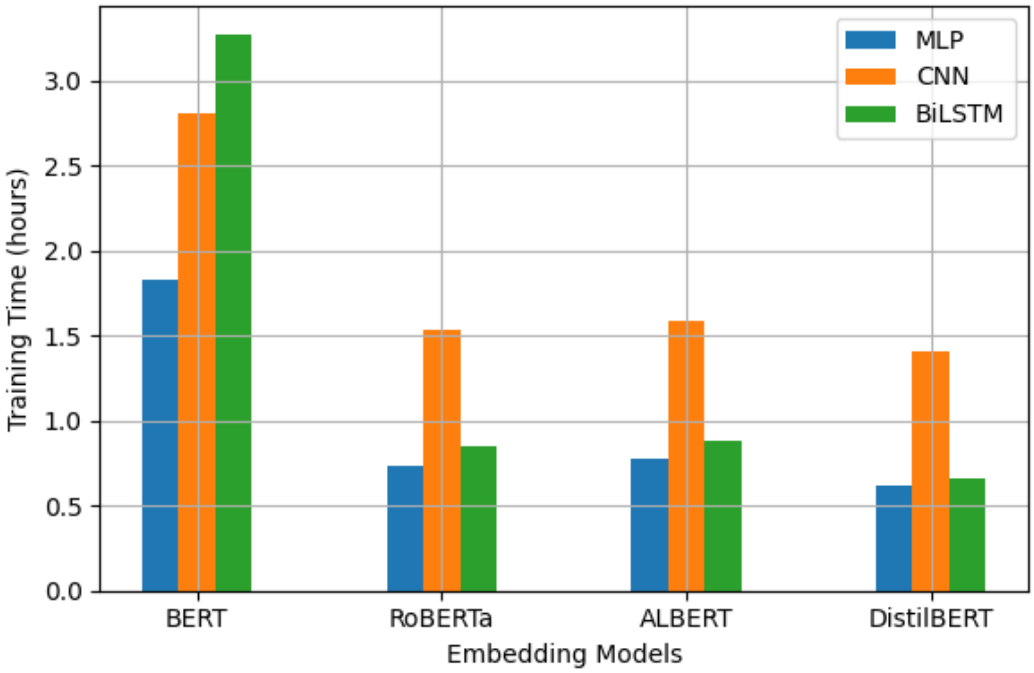


FIGURE 5.4 – Temps d'entraînement des différentes configurations de modèle.

plus de temps avec plus de 3 heures. ALBERT, RoBERTa, et DistilBERT montrent des durées d'entraînement significativement réduites, toutes les configurations se terminant en environ 1,5 heure.

Le choix du meilleur modèle dépend donc des exigences spécifiques de l'application, équilibrant le besoin de haute précision et de performance robuste avec l'efficacité computationnelle et la durée d'entraînement.

5.4.3 Comparaison avec les travaux connexes

Pour comparer nos résultats avec les approches existantes, nous avons évalué les performances de notre modèle par rapport à une étude qui a utilisé le même jeu de données pour la classification des tweets. Dans les travaux connexes [31], les auteurs ont appliqué diverses architectures d'apprentissage profond telles que CharCNN, CharRNN, WordCNN, et WordRNN pour la classification binaire des tweets en fonction de leur pertinence par rapport aux événements de cybersécurité. Le Tableau 5.6 montre la comparaison des performances entre notre approche et les travaux connexes, en mettant en évidence les améliorations des scores F1 obtenus par notre modèle.

Article	**Modèle**	**Validation%**	**Test%**
[31]	WordCNN + CharCNN	94.5	91.5
	WordRNN + CharCNN	95.1	91.8
	WordCNN + CharRNN	94.9	92.1
	WordRNN + CharRNN	95.1	92.2
Notre travail	BERT-based + DQN	N/A	**97.4**

TABLE 5.6 – Comparaison des scores F1 entre les travaux connexes et notre approche proposée sur l'ensemble de test pour la détection des informations sur les menaces dans les tweets.

5.5 Conclusion

Dans ce chapitre, nous avons présenté un modèle intelligent basé sur l'apprentissage par renforcement et le modèle pré-entraîné BERT, optimisé pour l'analyse et la détection en temps réel d'événements au sein des tweets. Le modèle proposé automatise et accélère de manière significative le processus d'analyse et de détection des tweets. Les résultats expérimentaux démontrent l'efficacité de BERT dans la représentation des aspects sémantiques et contextuels des tweets liés à la CTI. Cette représentation a facilité une bonne convergence de notre agent DQN dans le processus de prise de décision pour la détection d'événements, en particulier lors de l'utilisation de BiLSTM comme Réseau-Q et Réseau-Q Cible, atteignant une précision de 97,6% et un score F1 de 96,5% sur le jeu de données de test.

Nous avons également étudié l'impact de variantes des méthodes d'embedding de BERT, telles que RoBERTa, ALBERT et DistilBERT, sur les capacités de prise de décision de l'agent DQN pour la détection d'événements. L'étude a montré de meilleurs résultats lors de l'utilisation de RoBERTa avec BiLSTM, atteignant une précision de 98,2% et un score F1 de 97,4%, tout en optimisant le temps d'entraînement de plus de 50%.

Dans les recherches futures, notre objectif est d'approfondir l'extraction des entités associées et les relations entre ces entités afin de mieux comprendre les schémas de menace. Cette analyse approfondie aidera à identifier des scénarios de menace complexes et à améliorer l'efficacité globale de notre approche de CTI.

Chapitre 6

Extraction des entités liées à la CTI à partir des rapports d'incidents

6.1 Introduction

Aujourd'hui, les cybercriminels posent des défis significatifs pour les organisations chargées de défendre leurs données et systèmes contre les cyberattaques. Ils utilisent une gamme de méthodes, allant du vol d'identité basique à des attaques complexes sur les infrastructures critiques. Malgré la diversité des techniques d'attaque, les cyberattaques suivent généralement un cycle de vie similaire, commençant par la reconnaissance sur la machine de la victime et se terminant par l'exécution de code malveillant [52]. De plus, le volume croissant et la complexité des cyberattaques rendent l'analyse de la cybersécurité plus complexe, soulignant les limites des méthodes traditionnelles semi-manuelles pour la détection et la réponse aux incidents. Pour surmonter cela, il est urgent de développer une nouvelle approche de CTI qui automatise entièrement ce processus. Traditionnellement, la CTI a été un processus manuel, aidant les professionnels de la cybersécurité à identifier les (IoCs) et à recueillir des détails sur les attaques [15]. L'analyse de la CTI produit des informations stratégiques sur les menaces cybernétiques, extraites des rapports d'incidents et analysées manuellement par des experts, souvent représentées par des IoCs comportementaux tels que les TTPs utilisées par les attaquants [86].

Contrairement aux IoCs atomiques et calculés, tels que les adresses IP, les URL, et les empreintes MD5, qui ont une durée de vie plus courte [6], les IoCs comportementaux persistent plus longtemps et sont plus difficiles à modifier pour les attaquants [81]. De plus, les données comportementales peuvent être agrégées pour construire des profils détaillés des attaquants, aidant les analystes cyber à prédire les futures attaques et à renforcer la résilience de la détection des menaces. La nature structurée des TTPs permet aux analystes de repérer des actions spécifiques menées par des adversaires et de les relier à des techniques et tactiques particulières [86]. Cela aide les analystes à comprendre les objectifs des adversaires et à élaborer des stratégies défensives plus

efficaces.

Les rapports d'incidents sont une source courante d'IoCs comportementaux, qui sont généralement extraits par analyse textuelle manuelle. Cependant, avec le volume croissant de rapports publiés quotidiennement, cette tâche devient de plus en plus chronophage pour les humains. Pour relever ce défi, plusieurs recherches ont tenté d'automatiser le processus en utilisant des techniques de NLP. Divers modèles de réseaux neuronaux, tels que les CNNs, les LSTMs, et leurs combinaisons, ont été employés pour extraire des caractéristiques textuelles, reconnaître des entités, et capturer leurs relations pour classifier les rapports d'incidents. Malgré ces développements, ces modèles ont souvent du mal à capturer la sémantique du texte et montrent une généralisation limitée lorsqu'ils sont appliqués à des domaines spécialisés comme les documents de cybersécurité.

L'émergence des Transformeurs a transformé le paysage du NLP, offrant des représentations linguistiques puissantes essentielles pour une large gamme de tâches NLP [90]. Les Transformeurs pré-entraînés, tels que BERT, sont entraînés sur de vastes quantités de données textuelles et peuvent être affinés sur des tâches spécifiques en utilisant des jeux de données plus petits, tels que l'analyse de sentiment, la reconnaissance d'entités nommées, et la génération de texte, conduisant à une meilleure performance et une convergence plus rapide. Ces avantages peuvent être exploités pour relever les défis NLP en cybersécurité, notamment le manque de données annotées et le format unique des rapports d'incidents.

Dans ce contexte, nous proposons un modèle hybride pour la reconnaissance automatisée d'entités nommées afin d'identifier et de reconnaître les IoCs comportementaux dans les rapports d'incidents. Ces rapports, rédigés en langage naturel et dans des formats non structurés, sont issus de plateformes telles que MITRE ATT&CK, FireEye, Crowdstrike, et divers blogs de sécurité (ce travail se concentre sur les rapports en anglais). Le modèle proposé se compose de cinq couches : couche d'entrée, couche d'embedding, couche d'attention, couche Bi-LSTM, et couche de classification d'entités. La couche d'embedding, qui constitue le cœur de notre contribution, combine le modèle d'embedding de mots BERT avec un modèle d'embedding spécifique au domaine. Cette approche gère efficacement le vocabulaire spécialisé et les termes fréquemment rencontrés dans les rapports d'incidents de cybersécurité, tout en exploitant la compréhension contextuelle de BERT. Pour les phases d'entraînement et de test, nous proposons également un pipeline de données qui traite les documents non structurés et fournit des informations structurées aux analystes. Le pipeline comprend quatre étapes : agrégation de données, nettoyage de données, prétraitement des données, et classification des entités, qui sont décrites en détail ci-dessous.

Pour résumer, les principales contributions de ce travail sont les suivantes :

— NER-IoCs : un modèle hybride combinant un modèle d'encodage BERT, un modèle d'encodage spécifiques au domaine de la cybersécurité, et un mécanisme d'attention pour les tâches de reconnaissance d'entités nommées en Cyber Threat Intelligence.

— Un pipeline de données efficace pour préparer les rapports d'incidents de sécurité

aux tâches NLP.
— Une étude de diverses configurations de modèles pour analyser l'impact de différentes combinaisons de techniques d'embedding sur l'extraction et l'identification des IoCs comportementaux.

Le reste de ce chapitre est organisé comme suit. L'architecture du modèle est expliquée dans la Section 6.2. Le pipeline de données de bout en bout est décrit dans la Section 6.3, suivi des résultats expérimentaux dans les Sections 6.4 et 6.5. La conclusion est tirée dans la Section 6.6.

6.2 Architecture du modèle

Dans cette section, l'architecture du modèle est expliquée en détail. Cette architecture définit cinq couches (entrée, embedding, attention, Bi-LSTM et classification d'entités) pour réaliser une reconnaissance d'entités nommées comme IoCs (NER-IoCs). L'architecture de ce modèle est illustrée dans la Figure 6.1.

6.2.1 Couche d'entrée

La couche d'entrée de notre modèle fonctionne sur un ensemble de phrases extraites des rapports d'incidents, représentées par $S = \{s_1, s_2, .., s_n\}$, où n est le nombre total de phrases. Chaque phrase s_i est tokenisée en une séquence de tokens, notée $T_i = \{w_1, w_2, .., w_m\}$, avec m étant le nombre de tokens dans la phrase. Pour répondre aux exigences de BERT, chaque séquence tokenisée est ensuite formatée avec des tokens spéciaux, aboutissant à $T_i' = \{\text{'CLS'}, w_1, w_2, .., w_m, \text{'SEP'}\}$. De plus, pour assurer que toutes les séquences de tokens maintiennent une longueur uniforme, les séquences plus courtes que la longueur maximale L_{max} sont remplies de tokens 'PAD', ce qui donne la représentation finale illustrée en 6.1, où le nombre de tokens 'PAD' ajoutés est égal à $L_{max} - m - 2$.

$$T_i'' = \{\text{'CLS'}, w_1, w_2, ..., w_m, \text{'SEP'}, \text{'PAD'}, ..., \text{'PAD'}\} \tag{6.1}$$

6.2.2 Couche d'embedding

Dans la couche d'embedding, nous adoptons une méthode hybride qui inclut un modèle BERT combiné avec un modèle d'embedding spécifique au domaine. Cette approche peut capturer les caractéristiques linguistiques uniques inhérentes aux textes de cybersécurité, tout en bénéficiant de la compréhension contextuelle de BERT.

Embedding avec le modèle BERT

Lors de l'utilisation d'un modèle d'embedding pré-entraîné BERT, un format d'entrée spécifique est requis. Initialement, les tokens sont mappés à des identifiants uniques

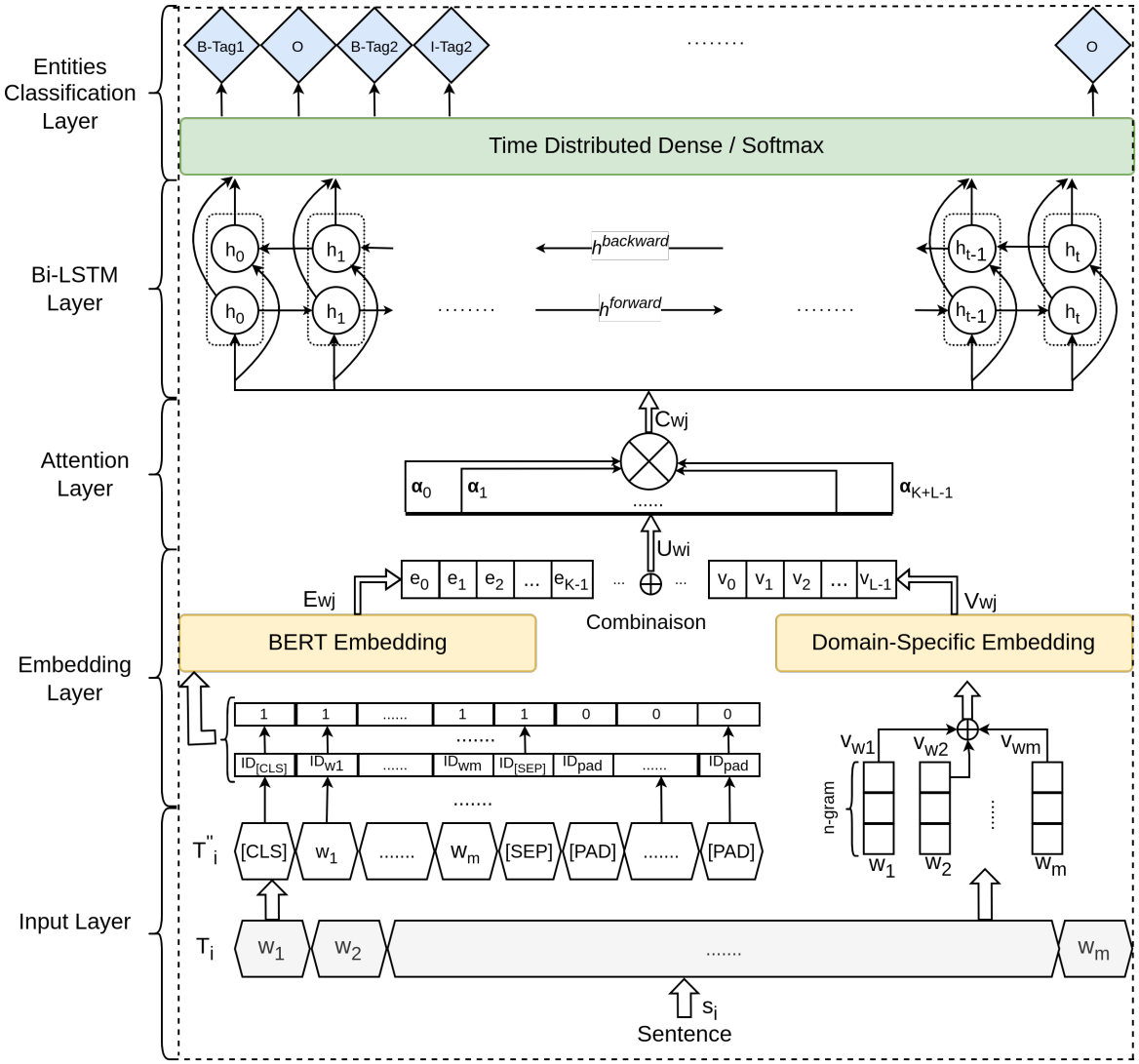


FIGURE 6.1 – NER-IoCs : architecture du modèle proposé.

selon le vocabulaire de BERT en utilisant une fonction de mappage $M : w_j \rightarrow id$. Ensuite, un masque d'attention est généré pour la séquence afin d'aider le modèle BERT à distinguer les tokens réels des tokens de remplissage. Ce masque est créé à l'aide de la fonction $A : w_j \rightarrow a$ illustrée en 6.2, attribuant une valeur de '1' pour les tokens réels et '0' pour les tokens de remplissage.

$$A(w_j) = \begin{cases} 1 & \text{si } w_j \text{ est un token réel} \\ 0 & \text{si } w_j = \text{'PAD'} \end{cases} \tag{6.2}$$

Une fois les identifiants mappés et le masque d'attention créé, pour tout token dans une séquence donnée, la sortie de BERT (que nous notons O_{BERT}) après traitement de la séquence est un vecteur d'embedding qui encapsule le contexte de la séquence

entière. Nous pouvons représenter la fonction de sortie d'embedding de BERT comme :

$$O_{BERT} : (T_i'', id) \rightarrow E_{w_j} \tag{6.3}$$

La fonction 6.3 signifie que BERT ne considère pas simplement le token w_j isolément. Il prend en compte la séquence entière T_i'' pour produire la sortie d'embedding E_{w_j} de longueur K pour ce token, le rendant contextuellement conscient.

Embedding avec le modèle spécifique pour la cybersécurité

L'Embedding Spécifique au Domaine fonctionne en parallèle avec la couche d'embedding BERT. Nous utilisons le modèle d'embedding FastText pour apprendre les représentations des mots basées sur leurs sous-unités constituantes [11]. Le modèle FastText représente chaque mot w_j comme un ensemble de n-grams de caractères G_{w_j}. Le vecteur final du mot V_{w_j} de longueur L est calculé comme la somme de ces vecteurs n-gram. Mathématiquement, cela peut être représenté comme :

$$V_{w_j} = \sum_{g \in G_{w_j}} z_g \tag{6.4}$$

où z_g représente le vecteur de représentation du n-gram du mot. La longueur de chaque n-gram est limitée par des longueurs minimum et maximum prédéfinies, $minlen$ et $maxlen$, respectivement. L'embedding final pour la phrase s_i est une agrégation des embeddings de ses mots constitutifs. Nous définissons la fonction d'embedding $E : T_i \rightarrow V_{s_i}$ comme :

$$V_{s_i} = \frac{1}{|T_i|} \sum_{w \in w_j} v_w \tag{6.5}$$

où $|T_i|$ représente le nombre de mots dans la phrase s_i, et V_{s_i} représente le vecteur agrégé de la phrase. Cette approche assure que chaque phrase encodée capture les nuances du langage spécifique à la cybersécurité.

Dans notre travail, nous avons utilisé un modèle FastText pré-entraîné ajusté pour le corpus de cybersécurité. Ce modèle, que nous avons obtenu à partir d'un projet open source disponible sur GitHub à [19], est entraîné sur un ensemble de données complet adapté à la CTI. Les données utilisées pour entraîner le modèle d'embedding FastText, qui représente pour nous l'embedding spécifique au domaine, comprennent une vaste gamme de documents de cybersécurité, y compris des rapports de menaces, des articles et des bulletins, fournissant un riche vocabulaire de terminologie spécifique au domaine.

Combinaison des modèles

La combinaison de l'embedding spécifique au domaine avec les embeddings BERT implique la concaténation des deux vecteurs d'embedding. Pour un token donné w_j,

soit E_{w_j} le vecteur d'embedding BERT et V_{w_j} le vecteur d'embedding spécifique au domaine. Le vecteur intégré U_{w_j} est obtenu en concaténant ceux-ci, comme illustré en 6.6. Ce vecteur concaténé U_{w_j} encapsule à la fois l'information contextuelle de BERT et la compréhension spécialisée du domaine de la cybersécurité.

$$U_{w_i} = [E_{w_i}; V_{w_i}] \tag{6.6}$$

6.2.3 Couche d'attention

L'idée principale derrière le mécanisme d'attention est d'assigner un score de pertinence à chaque partie de l'entrée, qui dans notre cas sont les sorties combinées d'embedding de la couche précédente. Ces scores déterminent combien de focus le modèle doit accorder à chaque partie de l'entrée lorsqu'il fait des prédictions. Étant donné un token w_j avec son vecteur d'embedding intégré U_{w_j} (tel que défini dans l'équation 6.6), le mécanisme d'attention calcule un ensemble de poids α_{w_j}, qui représentent l'importance de chaque élément dans le vecteur U_{w_j}. Les poids d'attention pour le token w_j sont calculés en utilisant une couche d'attention entraînable, qui consiste en un vecteur de poids W_{att} et un biais b_{att}. Le score d'attention pour chaque élément dans U_{w_j} est calculé comme suit :

$$A_{w_j} = \tanh(W_{att}^T U_{w_j} + b_{att}) \tag{6.7}$$

où tanh est la fonction tangente hyperbolique, fournissant une non-linéarité aux scores d'attention. Ensuite, nous normalisons ces scores en utilisant la fonction softmax pour s'assurer qu'ils s'additionnent à 1 :

$$\alpha_{w_j} = \frac{\exp(A_{w_j})}{\sum_n \exp(A_{w_n})} \tag{6.8}$$

L'étape finale dans le mécanisme d'attention est de calculer une somme pondérée du vecteur d'embedding, en utilisant les poids d'attention α_{w_j}. Cela aboutit à une représentation contextuellement enrichie du token, se concentrant davantage sur les parties pertinentes :

$$C_{w_j} = \sum_n \alpha_{w_j}(n) \cdot U_{w_j}(n) \tag{6.9}$$

où C_{w_j} est la représentation pondérée contextuellement de l'embedding combiné U_{w_j}. Cette représentation est ensuite transmise à la couche Bi-LSTM pour un traitement supplémentaire.

6.2.4 Couche Bi-LSTM

La sortie de la couche des Mécanismes d'Attention, représentée par C_{w_j} pour chaque token, sert d'entrée à la couche Bi-LSTM, passant par un ensemble d'équations illustré

en 6.10, qui décrivent les mécanismes internes d'une cellule LSTM à une étape temporelle donnée [23]. Cela implique plusieurs composants clés : la porte d'entrée i_t, la porte d'oubli f_t, la porte de sortie o_t, et les mises à jour de l'état de la cellule c_t. La porte d'entrée contrôle l'étendue à laquelle de nouvelles informations sont stockées dans l'état de la cellule, la porte d'oubli décide combien de l'état de la cellule précédent doit être conservé, et la porte de sortie détermine quelle partie de l'état de la cellule doit être sortie. Ces portes utilisent des fonctions sigmoïdes σ pour s'assurer que les valeurs sont comprises entre 0 et 1. L'état de la cellule c_t est mis à jour en fonction de ces portes, impliquant une combinaison de l'état précédent et de nouvelles valeurs candidates $\tilde{c}_t$, qui sont générées en utilisant une fonction *tanh* pour la normalisation. L'état caché h_t, sorti à chaque étape temporelle, est dérivé de la porte de sortie et de l'état de la cellule mis à jour, fournissant la sortie de l'LSTM qui sera passée à la couche suivante ou utilisée pour les prédictions.

$$\begin{cases} i_t & = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i) \\ f_t & = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f) \\ o_t & = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o) \\ \tilde{c}t & = \tanh(W_{cx}x_t + W_{ch}h_{t-1} + b_c) \\ c_t & = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \\ h_t & = o_t \circ \tanh(c_t) \end{cases} \tag{6.10}$$

Enfin, le LSTM traite l'embedding combiné C_{w_j} dans les directions avant et arrière pour produire une représentation bidirectionnelle.

$$H_t = [h_t^{forward}; h_t^{backward}] \tag{6.11}$$

6.2.5 Couche de classification d'entités

La couche de classification d'entités est l'étape finale de notre modèle, conçue pour attribuer des étiquettes d'entités à chaque token dans la séquence, en se basant sur les caractéristiques extraites par la couche Bi-LSTM précédente. Cette couche fonctionne sur une base distribuée temporelle [1], traitant chaque étape temporelle indépendamment.

Une couche dense distribuée temporellement applique une opération de réseau neural complètement connecté à chaque étape temporelle de la sortie Bi-LSTM indépendamment. Soit H_t la sortie de la Bi-LSTM à l'étape temporelle t. La couche dense transforme H_t en un nouveau vecteur D_t en utilisant une matrice de poids apprise W_d et un biais b_d :

$$D_t = \text{ReLU}(W_d H_t + b_d) \tag{6.12}$$

Ici, ReLU (Rectified Linear Unit) est utilisée comme fonction d'activation pour introduire de la non-linéarité. Ensuite, chaque vecteur D_t est passé à travers une couche

de classification pour attribuer une étiquette à chaque token. Cela est réalisé en appliquant une autre matrice de poids W_c et un biais b_c :

$$L_t = W_c D_t + b_c \tag{6.13}$$

L_t représente les logits correspondant à chaque classe pour le token à l'étape temporelle t. Enfin, une fonction d'activation softmax est appliquée à ces logits pour calculer la distribution de probabilité sur les classes d'entités (tags) pour chaque token :

$$P_t(i) = \frac{\exp(L_t(i))}{\sum_j \exp(L_t(j))} \tag{6.14}$$

où $P_t(i)$ est la probabilité que le token à l'étape temporelle t appartienne à la classe i, et la somme dans le dénominateur couvre toutes les classes d'entités possibles.

6.3 Pipeline de données

Cette section présente le pipeline de données développé pour traiter les rapports d'incidents de sécurité, à la fois pour les phases d'entraînement et de test. La structure du pipeline proposé est illustrée dans la Figure 6.2.

6.3.1 Agrégation des données

La première phase de notre pipeline se concentre sur l'agrégation des données, comprenant deux étapes clés : l'identification des sources de CTI et la collecte des rapports de CTI. Ces étapes constituent la base de la construction d'un corpus diversifié et représentatif de rapports d'incidents de sécurité, un facteur crucial pour le succès de notre modèle. Le processus commence par l'identification de sources de CTI fiables. Les principales sources de cybersécurité intégrées dans notre pipeline incluent MITRE ATT&CK [86], FireEye [35], CrowdStrike [22], ainsi que plusieurs sources moins utilisées. Celles-ci ont été sélectionnées pour leur crédibilité, leur couverture complète et la diversité de leur intelligence sur les menaces, ce qui soutient une meilleure généralisation du modèle d'IA lors de la phase de production. Après avoir identifié ces sources, nous avons collecté les rapports d'incidents de sécurité en interrogeant une API qui donne accès à leurs dépôts. Un total de 1,776 rapports ont été recueillis aux formats HTML et PDF.

6.3.2 Nettoyage des données

Dans la deuxième phase, l'accent est mis sur la préparation et l'affinage des rapports d'incidents collectés lors de la phase 1. Nous appliquons une série d'étapes de traitement pour transformer les rapports bruts en un format adapté à l'analyse, l'annotation et l'extraction de caractéristiques. Tout d'abord, nous convertissons les rapports, initialement au format HTML et PDF, en texte brut standardisé. Cette étape est cruciale

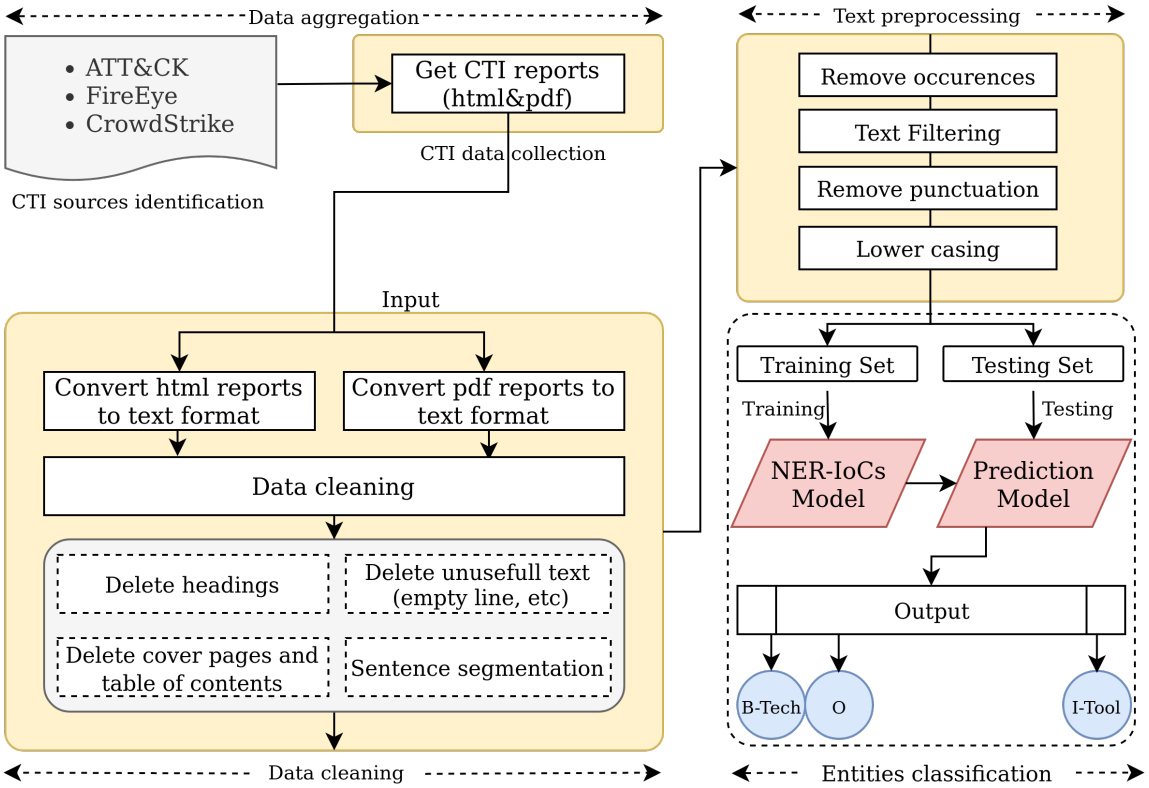


FIGURE 6.2 – Pipeline de préparation de données.

car ces formats contiennent souvent des éléments structurels complexes et du contenu non textuel qui entravent le traitement. La standardisation des données par la conversion en texte brut assure une gestion uniforme. De plus, nous éliminons les éléments inutiles tels que les pages de couverture, les tables des matières, les en-têtes et les sections introductives qui ne contribuent pas au contenu principal des rapports d'incidents de sécurité. Cela aide à rationaliser le pipeline en réduisant le bruit et en améliorant l'efficacité du traitement. Nous supprimons également les lignes superflues, telles que les lignes vides et les lignes d'un seul mot, qui peuvent interférer avec l'analyse. Ces améliorations augmentent la clarté et la cohérence des rapports, garantissant un focus sur les descriptions substantielles des incidents de sécurité. Enfin, nous segmentons le texte en phrases individuelles pour préparer les tâches de NLP, assurant la granularité nécessaire à une reconnaissance précise des entités nommées (NER).

6.3.3 Prétraitement du texte

Cette phase englobe une série d'opérations de prétraitement essentielles visant à optimiser les rapports d'incidents pour la tâche de reconnaissance des entités nom-

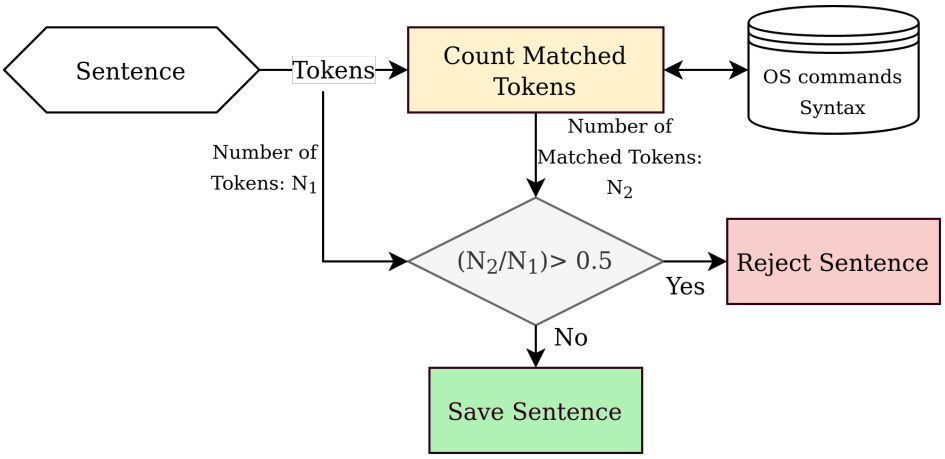


FIGURE 6.3 – Filtrage des phrases.

mées (NER). Tout d'abord, nous supprimons les phrases récurrentes qui incluent généralement des informations générales de contexte, des avertissements ou du contenu générique répété dans plusieurs rapports, assurant que le focus reste sur les détails pertinents et spécifiques des incidents. Ensuite, un processus de filtrage du texte (illustré à la Figure 6.3) est appliqué pour éliminer les phrases composées entièrement d'éléments de lexique prédéfini, tels que les commandes du système d'exploitation ou la syntaxe des langages de programmation. Après une analyse préliminaire, nous avons fixé le seuil de filtrage à 0.5, ce qui a permis d'équilibrer efficacement la suppression du contenu non pertinent tout en conservant les informations pertinentes. Ce seuil marquait un point d'inflexion où l'augmentation de la valeur ne menait plus à des améliorations significatives de la performance du cadre. De plus, nous supprimons les signes de ponctuation tels que ".", ",", ";", " ?" et " !" pour réduire le bruit et simplifier l'analyse textuelle. Enfin, nous uniformisons la casse des lettres, traitant "Malware" et "malware" comme la même entité pour améliorer la précision dans la reconnaissance des entités.

6.3.4 Classification des entités

Les données prétraitées de la phase précédente sont divisées en jeux de données d'entraînement et de test. Le modèle décrit dans la Section 6.2 est ensuite entraîné en utilisant les données d'entraînement annotées pour prédire les tags pertinents pour chaque token. Ces tags représentent différentes catégories d'IoCs comportementaux.

6.4 Expérimentations

6.4.1 Jeu de données

Le Tableau 6.1 illustre le nombre de rapports obtenus de chaque source, comme expliqué dans la Section 6.3. Après l'exécution des différentes phases du pipeline sur ces

TABLE 6.1 – Nombre de rapports d'incidents obtenus de chaque source identifiée.

Source	PDF	HTML	Total
MITRE ATT&CK	55	373	428
FireEye	35	324	359
Crowdstrike	27	206	233
Autres	110	646	756

TABLE 6.2 – Statistiques du jeu de données.

Jeu de données	Phrase	Token	Entité
Ensemble d'entraînement	12,059	585,811	19,124
Ensemble de test	2,128	103,379	3,375
Total	14,187	689,190	22,499

documents, les données pré-traitées sont transformées en un DataFrame pour obtenir un fichier dans un format d'entrée standardisé pour la tâche de reconnaissance d'entités nommées (NER). Le jeu de données est divisé en deux sous-ensembles : 85% pour l'entraînement (y compris la validation) et 15% pour le test. Le Tableau 6.2 présente les statistiques des données expérimentales.

6.4.2 Annotation des données

L'annotation des séquences pour les tâches de NER implique de labelliser des entités spécifiques avec leurs classes correspondantes. Pour notre expérience, nous définissons cinq IoCs comportementaux, y compris des entités de tactiques, techniques, sous-techniques, outils et groupes d'adversaires, qui sont labellisées en deux étapes. La première partie de nos données (environ 20%) a été labellisée en utilisant une méthode basée sur un dictionnaire. Ce dictionnaire est construit en exploitant des rapports précédemment annotés par des experts en cybersécurité, en particulier ceux alignés avec le framework MITRE ATT&CK. Ensuite, une annotation manuelle a été utilisée pour la seconde partie des données en raison des limitations de la méthode basée sur le dictionnaire pour reconnaître de nouvelles entités. Notre équipe a examiné chaque phrase pour taguer les entités qui n'étaient pas dans le dictionnaire. La distribution des différentes entités dans le jeu de données est illustrée dans le Tableau 6.3. De plus, suivant la méthode universelle d'annotation NER, le mot initial d'une entité est marqué comme 'B-TagName', les mots à l'intérieur de l'entité sont marqués comme 'I-TagName', et tout mot en dehors des entités est étiqueté 'O' pour 'Outside'. Un exemple de phrase annotée est montré dans la Figure 6.4.

TABLE 6.3 – Distribution des entités dans le jeu de données.

Jeu de données	Tactique	Tech	Sous-Tech	Outil	Groupe
Ensemble d'entraînement	4,103	561	355	10,458	3,648
Ensemble de test	725	99	63	1,845	644
Total	4,828	660	418	12,303	4,292

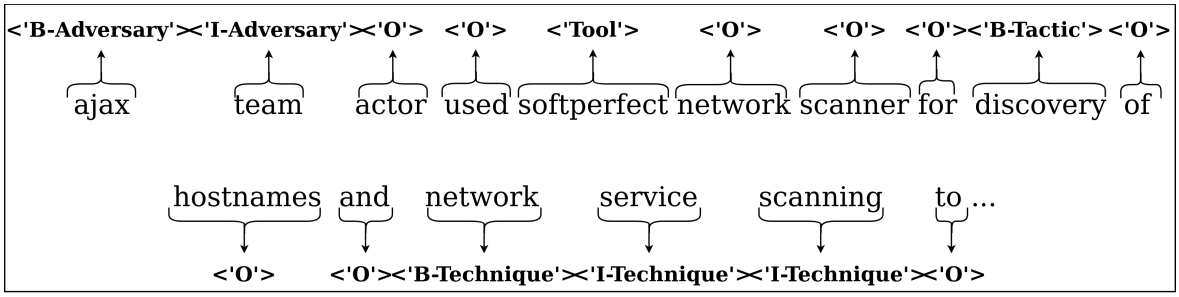


FIGURE 6.4 – Exemple de phrase annotée (Source : FireEye).

6.4.3 Paramètres expérimentaux

Dans notre configuration expérimentale, nous utilisons un GPU NVIDIA GeForce RTX 2070 avec 16 Go de mémoire. Nous avons utilisé PyTorch pour implémenter les différentes phases du modèle proposé. Pour l'entraînement, tous les modèles ont été entraînés pendant 25 époques, avec les hyperparamètres suivants : optimiseur = Adam, taille de lot = 32, taux d'apprentissage = 0.001. La fonction de perte adoptée pour l'entraînement était l'entropie croisée catégorielle.

6.4.4 Métriques d'évaluation

Le modèle NER-IoCs a été évalué à l'aide des métriques de Précision, Rappel et F1-score.

— Précision dans les tâches de NER mesure l'exactitude des entités correctement identifiées parmi toutes les entités prédites par le modèle.

$$\text{Précision} = \frac{\text{Vrais Positifs}}{\text{Vrais Positifs} + \text{Faux Positifs}}$$

— Rappel évalue la capacité d'un modèle à identifier toutes les entités pertinentes dans le texte.

$$\text{Rappel} = \frac{\text{Vrais Positifs}}{\text{Vrais Positifs} + \text{Faux Négatifs}}$$

— F1-score est la moyenne harmonique de la précision et du rappel, offrant une mesure équilibrée de la performance du modèle en termes de précision et de rappel.

$$\text{F1-score} = \frac{2 \cdot \text{Précision} \cdot \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

6.4.5 Configuration du modèle

Notre modèle NER-IoCs a été entraîné et testé avec le jeu de données CTI comprenant cinq classes distinctes (expliquées en 6.4.2). L'analyse comprend l'étude de différents modèles d'embedding à travers quatre configurations distinctes, chacune explorant différentes stratégies d'embedding pour évaluer leur impact sur l'identification et l'extraction précises des IoCs comportementaux à partir des rapports d'incidents. La première configuration se concentre sur l'entraînement du modèle exclusivement avec l'embedding spécifique au domaine (noté D-S), servant de base pour comprendre les capacités intrinsèques de la connaissance du domaine dans la reconnaissance des IoCs. La deuxième configuration examine l'efficacité du modèle lorsqu'il est entraîné uniquement sur l'embedding BERT, évaluant l'utilité d'un modèle de langage généraliste dans la reconnaissance d'entités liées à la sécurité. Dans la troisième configuration, le modèle est entraîné en utilisant une combinaison ($\oplus$) des embeddings spécifiques au domaine et BERT, visant à évaluer comment cette approche synergique améliore les performances. Enfin, la quatrième configuration s'appuie sur la configuration précédente en incorporant un mécanisme d'attention, fournissant des perspectives sur la façon dont des couches supplémentaires peuvent affiner les performances. De plus, dans les deuxième, troisième et quatrième configurations, nous explorons les deux variantes de BERT : BERT Base et BERT Large, respectivement notées BERT-B et BERT-L.

6.5 Analyse des résultats

6.5.1 Courbe d'apprentissage

La Figure 6.5 illustre la perte et le F1-score de l'entraînement et de la validation de la quatrième configuration (BERT-L$\oplus$D-S+Mécanisme d'attention) au fil des époques. En particulier, notre modèle montre un schéma de convergence distinct, avec une amélioration notable du F1-score après la 15e époque.

6.5.2 Performances du modèle

Les résultats expérimentaux, comme décrit dans le Tableau 6.4, offrent des observations perspicaces sur les performances des différents modèles. Initialement, notre modèle de base utilisant l'embedding spécifique au domaine dans la première configuration montre des résultats modestes avec une précision de 66,2%, un rappel de 49,9%, et un F1-score de 55,9%. Cependant, une amélioration notable est observée dans la deuxième configuration, où l'adoption de l'embedding BERT donne des résultats nettement meilleurs : précision, rappel, et F1-score atteignant tous 89,7%. L'intégration de BERT avec l'embedding spécifique au domaine dans la troisième configuration améliore encore l'efficacité du modèle, particulièrement évidente en utilisant la version large de BERT (BERT-L). Cette combinaison atteint une précision impressionnante de 93,5%, un rappel de 89,8%, et un F1-score de 91,2%. Le sommet de performance est atteint

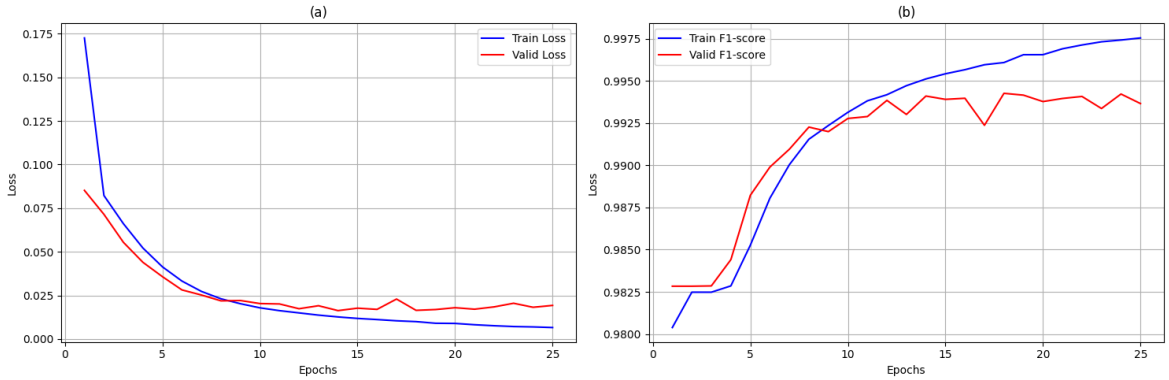


FIGURE 6.5 – (a) Perte d'entraînement et de validation du modèle BERT-L⊕D-S+Mécanisme d'attention, (b) F1-score d'entraînement et de validation du modèle BERT-L⊕D-S+Mécanisme d'attention.

dans la quatrième configuration, qui intègre un mécanisme d'attention, aboutissant à la précision la plus élevée de 94,5%, un rappel de 93,6%, et un F1-score de 94,2%. Ces résultats soulignent l'impact substantiel de la combinaison de BERT avec l'embedding spécifique au domaine et les mécanismes d'attention sur l'amélioration de l'exactitude et de la fiabilité du modèle NER-IoCs.

La matrice de confusion représentée dans la Figure 6.6 montre que notre modèle est très précis pour classifier les entités 'Tactique' et 'Adversaire', avec respectivement 99,5% et 95% de prédictions correctes. Le modèle performe également bien sur les entités 'Technique' et 'Outil', avec une précision de 92,5% et 93,5%, respectivement. La catégorie 'Sous-Technique' a un taux de prédiction correcte plus bas, mais néanmoins élevé, de 87%.

6.5.3 Effets des techniques d'embedding

Embedding spécifique au domaine

Dans la première configuration, notre modèle de base utilisant l'embedding spécifique au domaine a montré des performances sous-optimales pour l'identification et la reconnaissance des différentes entités liées aux IoCs comportementaux. Cette limitation résulte principalement de deux facteurs : d'une part, le modèle d'embedding spécifique au domaine manque d'une représentation sémantique complète des tokens, ce qui limite sa capacité à contextualiser et à interpréter avec précision les textes complexes liés à la cybersécurité. D'autre part, l'embedding spécifique au domaine est entraîné sur une quantité de données insuffisante, ce qui explique ses faibles capacités de généralisation lorsqu'il est appliqué à notre jeu de test.

Configuration	Modèle	Métrique	Technique	Sous-Tech	Tactique	Outil	Groupe	Moyenne
(1)	D-S	Précision	0,17	0,16	**1**	**0,98**	**1**	0,662
		Rappel	0,15	0,14	0,99	0,735	0,48	0,499
		F1-score	0,16	0,15	**0,995**	0,835	0,655	0,559
(2)	BERT-B	Précision	0,79	0,86	0,87	0,87	0,915	0,861
		Rappel	0,82	0,81	0,85	0,83	0,88	0,838
		F1-score	0,81	0,83	0,86	0,85	0,895	0,849
	BERT-L	Précision	0,82	**0,97**	0,94	0,89	0,865	0,897
		Rappel	0,88	**0,97**	0,92	0,855	0,86	0,897
		F1-score	0,85	**0,97**	0,93	0,87	0,86	0,896
(3)	BERT-B⊕D-S	Précision	0,83	0,93	**1**	0,95	0,955	0,933
		Rappel	0,9	0,79	0,99	0,85	0,89	0,884
		F1-score	0,86	0,85	**0,995**	0,945	0,92	0,914
	BERT-L⊕D-S	Précision	0,85	0,93	0,994	0,95	0,95	0,935
		Rappel	0,91	0,8	**0,995**	0,93	0,855	0,898
		F1-score	0,878	0,83	**0,995**	0,94	0,916	0,912
(4)	(BERT-B⊕D-S)+ATT	Précision	0,86	0,91	0,995	0,975	0,97	0,942
		Rappel	0,9	0,9	0,985	**0,95**	0,945	**0,936**
		F1-score	0,879	0,91	0,99	**0,96**	0,96	0,939
	(BERT-L⊕D-S)+ATT	Précision	**0,87**	0,94	0,995	0,95	0,97	**0,945**
		Rappel	**0,925**	0,87	**0,995**	0,935	**0,95**	0,935
		F1-score	**0,897**	0,904	**0,995**	0,945	**0,962**	**0,942**

TABLE 6.4 – Performances du modèle proposé avec différentes configurations sur l'ensemble de test.

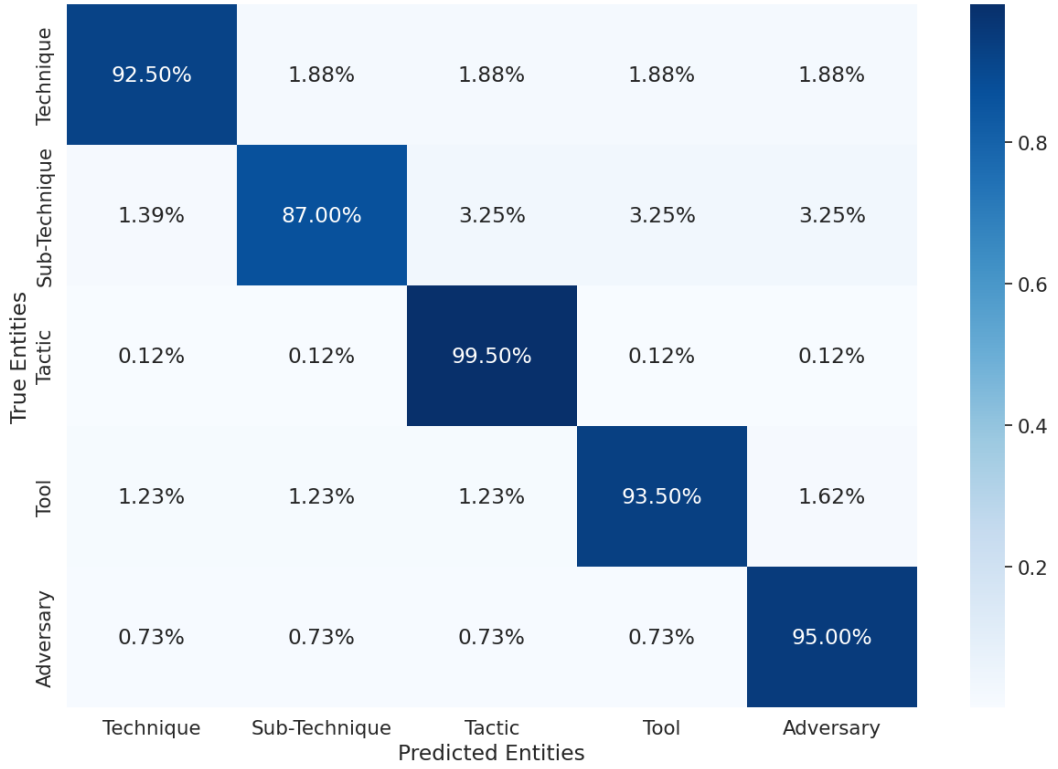


FIGURE 6.6 – Matrice de confusion pour les diverses entités du modèle BERT-L⊕D-S+Mécanisme d'attention sur l'ensemble de test.

Embedding BERT

Dans la deuxième configuration, qui intègre l'embedding BERT, nous avons observé une amélioration remarquable, avec une augmentation du F1-score de 29% et 33,7% pour les versions de base et large de BERT, respectivement, par rapport à la première configuration. Cette amélioration significative est attribuée à la représentation contextuelle profonde de BERT, qui facilite une extraction plus nuancée et complète des IoCs comportementaux à partir de textes complexes. Cependant, malgré ses forces, l'embedding BERT rencontre encore des défis pour identifier et reconnaître avec précision les diverses techniques adverses et entités dans le texte.

Combinaison de BERT et d'embedding spécifique au domaine

Dans la troisième configuration, où BERT est combiné avec l'embedding spécifique au domaine, nous avons atteint la meilleure performance avec un F1-score de 91,2%, représentant une augmentation de 1,6% par rapport à la deuxième configuration. Cette amélioration illustre l'effet synergique de la fusion des perspectives contextuelles larges de BERT avec l'expertise spécialisée de l'embedding spécifique au domaine. Cette combinaison compense efficacement les limitations contextuelles inhérentes aux modèles spécifiques au domaine, car BERT ajoute une compréhension plus profonde et nuancée du texte, améliorant ainsi la précision globale dans l'identification des IoCs comportementaux.

6.5.4 Effets du mécanisme d'attention

Dans la quatrième configuration, l'intégration d'un mécanisme d'attention a abouti à un F1-score de 94,2%, marquant une amélioration de 3% par rapport à la troisième configuration. Cette augmentation peut être attribuée à la capacité du mécanisme d'attention à se concentrer sélectivement sur des segments spécifiques du vecteur d'embedding combiné, accordant plus de poids aux aspects les plus significatifs parmi les deux représentations, ce qui conduit à une identification plus précise des IoCs comportementaux dans les rapports d'incidents de cybersécurité.

6.5.5 Durée d'entraînement

La durée d'entraînement illustrée dans la Figure 6.7 montre une augmentation progressive du temps nécessaire pour entraîner les différentes configurations de notre modèle. À partir des configurations initiales qui nécessitent le moins de temps d'entraînement, chaque amélioration supplémentaire, en particulier l'intégration des mécanismes d'attention, prolonge substantiellement la période d'entraînement. Cette augmentation est due à l'augmentation du nombre de paramètres entraînables dans chaque configuration. Par conséquent, le choix du meilleur modèle pour une application pratique doit prendre en compte les ressources computationnelles disponibles et les besoins spécifiques de la tâche.

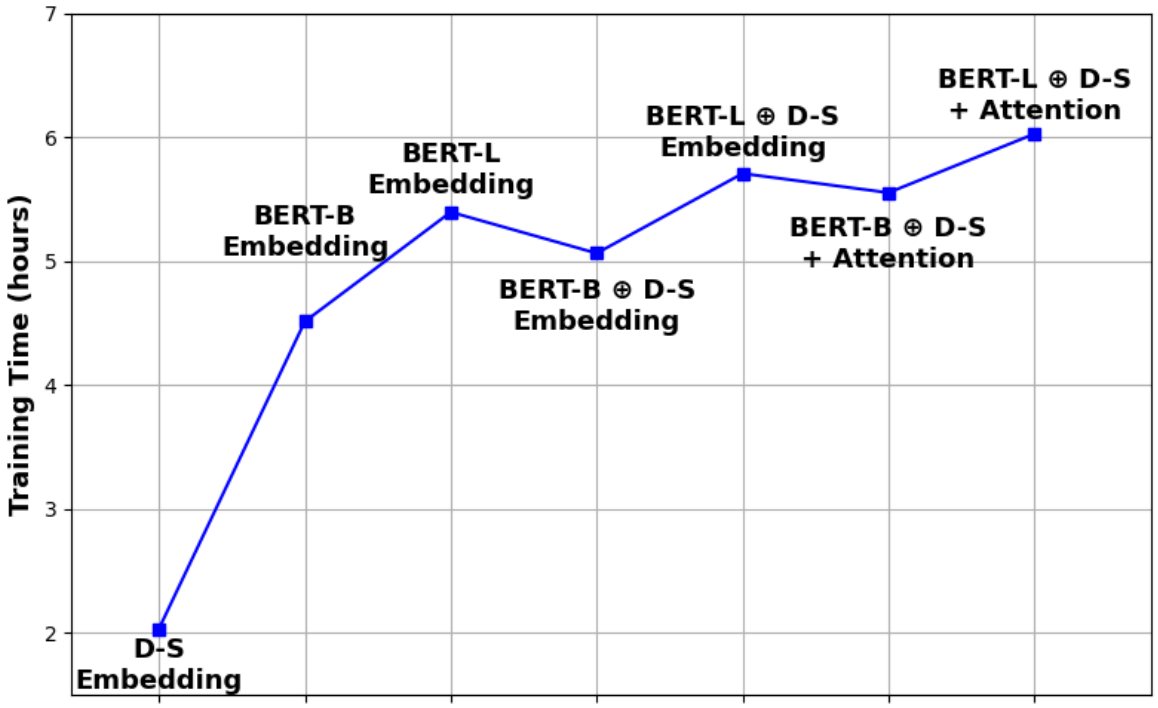


FIGURE 6.7 – Temps d'entraînement des différentes configurations sur une NVIDIA GeForce RTX 2070 GPU.

6.5.6 Comparaison des performances

Dans notre étude, en raison de l'absence de recherches utilisant des jeux de données identiques aux nôtres, nous avons implémenté deux architectures décrites dans [55] et [43] en utilisant le framework PyTorch. Cette implémentation adhère strictement aux méthodologies détaillées par les auteurs dans leurs publications, garantissant que notre adaptation de ces modèles à notre jeu de données est précise et fidèle aux conceptions originales. Ces modèles ont été évalués sur la base des moyennes de Précision, Rappel et F1-score. Le Tableau 6.5 démontre des améliorations significatives des performances par rapport à d'autres approches.

6.6 Conclusion

Dans ce chapitre, nous avons présenté NER-IoCs, un modèle hybride pour les tâches de reconnaissance d'entités nommées dans le domaine de la CTI. Le modèle apporte une contribution significative à l'automatisation et à l'accélération du processus d'analyse CTI. Les résultats expérimentaux soulignent l'efficacité de la combinaison du modèle d'embedding BERT avec le modèle d'embedding spécifique au domaine, avec un F1-score moyen de 94,2%. De plus, notre modèle est entraîné et testé sur diverses sources de rapports d'incidents, ce qui lui confère des capacités accrues et une robustesse en

TABLE 6.5 – Comparaison des résultats d'entraînement : comparaison de notre travail avec d'autres architectures utilisant notre jeu de données.

Auteurs	**Modèle Architecture**	**Embedding Technique**	**Moy. des métriques** (%)
[55]	BiLSTM-CRF	Couche entraînable	P = 79,7 R = 65,4 F1 = 63,3
[43]	CNN-BiLSTM Attention	Word2Vec	P = 68,7 R = 52,7 F1 = 55,3
Proposé Modèle	Attention-BiLSTM TimeDistributed	BERT-L⊕D-S	P = 94,5 R = 93,5 F1 = 94,2

termes de généralisation. Nous avons également introduit un pipeline de préparation des données efficace, spécifiquement conçu pour les tâches de NLP en cybersécurité, garantissant un traitement de données de haute qualité pour les rapports d'incidents. En outre, notre étude sur diverses configurations de modèles d'embedding a révélé que l'intégration d'un mécanisme d'attention avec la combinaison de BERT large et de l'embedding spécifique au domaine améliore considérablement les capacités de généralisation du modèle à travers différentes sources de rapports d'incidents.

Chapitre 7

Attribution des cyberattaques

7.1 Introduction

Les Menaces Persistantes Avancées (Advanced Persistent Threats ou APTs) se caractérisent par des attaques sophistiquées et continues, orchestrées par des entités organisées motivées par des objectifs politiques, économiques ou stratégiques. Ces attaquants cherchent généralement à compromettre des informations sensibles, perturber des infrastructures critiques ou exercer une influence sur des organisations ou des nations ciblées. L'un des défis majeurs en cybersécurité est l'attribution des APTs à des acteurs spécifiques, un processus appelé attribution des cyberattaques. Ce processus consiste à identifier l'attaquant ou un intermédiaire en déterminant son identité ou sa localisation. Une identité peut inclure le nom d'une personne, un compte, un alias ou d'autres informations d'identification, tandis qu'une localisation peut désigner une adresse physique ou un emplacement virtuel, tel qu'une adresse IP ou un domaine réseau [92]. Cependant, l'attribution est compliquée par les techniques avancées d'évasion utilisées par les adversaires. Ces derniers emploient souvent des méthodes telles que l'obfuscation, les faux drapeaux et les outils partagés pour masquer leur véritable identité, rendant difficile la distinction entre les acteurs et le traçage précis de l'origine d'une attaque.

Les IoCs traditionnels, tels que les adresses IP, les URL et les empreintes MD5, ont souvent une durée de vie limitée, car les attaquants peuvent facilement les modifier pour échapper à la détection [8]. Cette volatilité rend difficile de se fier uniquement à ces IoCs atomiques et calculés pour identifier avec précision les acteurs de la menace. Toutefois, les logiciels malveillants jouent un rôle crucial dans le processus d'attribution, car ils contiennent fréquemment des motifs uniques, des styles de codage et des TTPs plus difficiles à altérer et pouvant être associés à des groupes ou individus spécifiques. Ces TTPs englobent les méthodes et techniques opérationnelles spécifiques utilisées par les adversaires, notamment la manière dont ils développent, déploient et modifient les logiciels malveillants au fil du temps pour s'adapter à de nouveaux environnements et échapper à la détection. En analysant à la fois les caractéristiques statiques et

comportementales, il est possible de créer des signatures distinctes pour les acteurs de la menace, permettant une attribution plus précise.

Plusieurs travaux de recherche ont proposé des approches pour l'attribution des cyberattaques en utilisant des algorithmes d'apprentissage supervisé, traitant le problème comme une tâche de classification. Ces approches exploitent généralement les caractéristiques des logiciels malveillants — qu'elles soient statiques, dynamiques ou une combinaison des deux, afin d'entraîner des modèles de prédiction. Pour atteindre cet objectif, diverses architectures de réseaux neuronaux, telles que les réseaux de neurones convolutifs et les réseaux de mémoire à long terme unidirectionnels/bidirectionnels, ont été utilisées pour apprendre des motifs à partir des caractéristiques extraites des logiciels malveillants. Cependant, les modèles de classification présentent des limitations significatives lorsqu'ils sont appliqués à l'attribution des cyberattaques. Un problème majeur est celui de la scalabilité : à mesure que le nombre de classes d'acteurs de la menace augmente, la précision de ces modèles tend à diminuer, rendant difficile l'attribution des attaques dans un ensemble large et diversifié d'acteurs. De plus, ces modèles sont intrinsèquement limités par leurs données d'entraînement. Si une nouvelle attaque provient d'un acteur inconnu, le modèle tentera tout de même de la classer dans une catégorie connue, ce qui pourrait entraîner une attribution incorrecte. Cela s'explique par le fait que les modèles de classification ne sont pas conçus pour gérer des classes nouvelles qui n'étaient pas incluses dans la phase d'entraînement, limitant ainsi leur capacité à s'adapter à l'évolution des acteurs de la cybermenace.

Dans ce contexte, nous proposons un framework permettant d'attribuer un acteur de menace à une attaque en se basant sur une analyse statique et comportementale des logiciels malveillants. Le framework comprend trois composants principaux. Premièrement, un réseau siamois, qui constitue le cœur de notre approche, est utilisé pour prédire un score de similarité entre deux échantillons de logiciels malveillants. Ce réseau implémente deux sous-réseaux, abrégés CLA-Net dans ce chapitre, qui combinent des (CNNs), des (LSTMs) et un mécanisme d'attention. Cette architecture permet au modèle de capturer des motifs locaux dans les échantillons de logiciels malveillants, d'apprendre des traits comportementaux séquentiels et de se concentrer sur les caractéristiques les plus pertinentes lors de la prédiction.

Deuxièmement, des profils d'acteurs de cybermenace sont construits à l'aide d'un ensemble de données étiquetées d'échantillons de logiciels malveillants, chacun étant associé à un acteur de la menace connu. Le CLA-Net entraîné pour générer des embeddings pour chaque famille de logiciels malveillants, ces embeddings servant de signatures uniques caractérisant les logiciels malveillants, et forment collectivement un profil distinct pour chaque acteur de cybermenace, fournissant une référence pour le processus d'attribution.

Enfin, lors de la phase de production, le framework attribue de nouveaux échantillons de logiciels malveillants en calculant leurs embeddings, et en les comparant aux profils existants d'acteurs de cybermenace en fonction du score de similarité prédit par le réseau siamois. Le profil avec le score de similarité le plus élevé indique l'acteur de la cybermenace le plus probable responsable de l'attaque. Si un échantillon de logi-

ciel malveillant ne correspond étroitement à aucun profil existant, un mécanisme de détection d'anomalies est activé pour déterminer si l'échantillon représente un nouvel acteur de cybermenace. Dans de tels cas, un nouveau profil est créé, et les échantillons similaires suivants sont progressivement ajoutés pour affiner ce profil au fil du temps.

En résumé, les principales contributions de ce travail sont les suivantes :

— **Framework d'attribution des cyberattaques** : nous introduisons une nouvelle approche pour attribuer les cyberattaques à des acteurs de la menace en exploitant à la fois des caractéristiques statiques et comportementales des logiciels malveillants via un réseau siamois. Ce framework est conçu pour relever les défis de scalabilité rencontrés dans les modèles de classification traditionnels.

— **Conception flexible avec détection de nouveauté** : le framework intègre une conception flexible qui inclut un mécanisme de détection d'anomalies pour identifier de potentiels nouveaux acteurs de cybermenace, permettant une adaptation aux menaces évolutives.

— **Architecture efficace pour l'analyse des caractéristiques** : nous proposons une architecture performante intégrant des CNNs, LSTMs et mécanismes d'attention pour l'analyse des caractéristiques statiques et comportementales des logiciels malveillants.

Le reste de ce chapitre est organisé comme suit : la section 7.2 décrit les concepts fondamentaux nécessaires à la compréhension de notre travail. Le framework proposé est expliqué dans la section 7.3, suivi des résultats expérimentaux dans les sections 7.4 et 7.5. La conclusion est présentée dans la section 7.6.

7.2 Concepts théoriques

Cette section présente des concepts essentiels, notamment les réseaux siamois, les caractéristiques des logiciels malveillants et la similarité cosinus, qui constituent la base de notre framework d'attribution des cyberattaques.

7.2.1 Réseau siamois

Un réseau siamois est une architecture neuronale conçue pour apprendre la similarité entre deux échantillons d'entrée en utilisant deux sous-réseaux identiques partageant les mêmes poids. Il permet de distinguer les entrées similaires des entrées dissemblables [13]. Les réseaux siamois sont largement utilisés dans des tâches nécessitant des comparaisons, telles que la correspondance d'images [57], où le réseau identifie si deux images représentent le même objet ou la même scène ; la reconnaissance faciale [93], qui vise à déterminer si deux images faciales appartiennent à la même personne ; et la vérification de signatures [30], où il distingue les signatures authentiques des signatures falsifiées.

Le processus d'entraînement d'un réseau siamois commence par l'utilisation de paires d'échantillons d'entrée, tels que des images, des fichiers ou des logiciels mal-

veillants, qui sont étiquetés comme similaires ('1') ou dissemblables ('0'). Ensuite, le sous-réseau partagé, par exemple un réseau de neurones convolutif pour des paires d'images, extrait les caractéristiques de chaque entrée afin de générer des embeddings qui encapsulent les informations pertinentes. Le réseau siamois calcule ensuite la similarité entre ces embeddings à l'aide d'une métrique telle que la similarité cosinus ou la distance euclidienne, produisant un score qui reflète le degré de ressemblance entre les entrées. Enfin, ce score de similarité est utilisé pour classer les entrées comme étant correspondantes ou non, en fonction d'un seuil prédéfini.

7.2.2 Caractéristiques des logiciels malveillants

Caractéristiques statiques

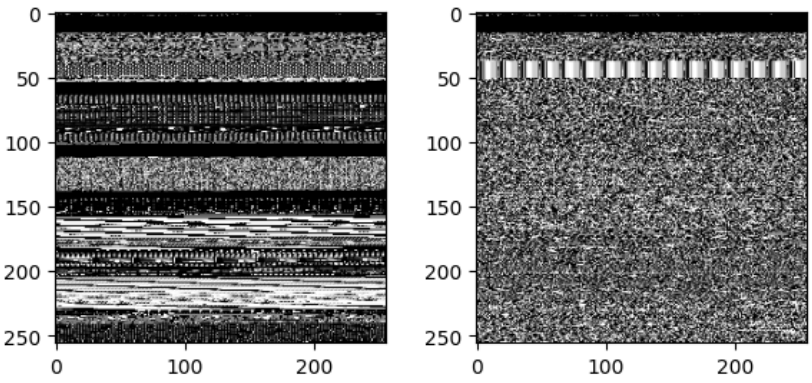
Les caractéristiques statiques désignent les attributs intrinsèques des logiciels malveillants qui peuvent être extraits sans exécuter l'échantillon. Ces caractéristiques offrent des informations précieuses sur la structure et le contenu des logiciels malveillants. Parmi les caractéristiques statiques courantes, on trouve les opcodes, les bytecodes et les en-têtes. Le bytecode, en particulier, représente un ensemble d'instructions lisibles par machine exécutées par le logiciel malveillant. Contrairement à d'autres caractéristiques statiques, les séquences de bytecode peuvent être converties en un format d'image bidimensionnelle (2D) en mappant les valeurs aux intensités des pixels. Cette approche est largement adoptée dans la recherche sur la détection et la classification des logiciels malveillants [94, 45, 44]. La conversion du bytecode en format image permet l'application de techniques avancées de vision par ordinateur, améliorant ainsi l'extraction des caractéristiques pour une attribution des menaces plus précise.

La Figure 7.1 ci-dessous illustre les représentations de bytecode d'échantillons de logiciels malveillants attribués à deux acteurs différents. Chaque image capture des motifs distincts dans le bytecode, caractéristiques de l'acteur respectif.

Dans notre étude, nous nous concentrons sur l'analyse des bytecodes, car elle a montré des résultats plus prometteurs, tandis que d'autres caractéristiques statiques, telles que les opcodes et les en-têtes, n'ont pas démontré d'améliorations significatives en termes de précision de l'attribution.

Caractéristiques comportementales (TTPs)

Contrairement aux caractéristiques statiques, les caractéristiques comportementales sont collectées en exécutant le logiciel malveillant dans un environnement contrôlé, tel qu'un "sandbox", afin d'observer ses actions et ses interactions avec le système. Cette exécution révèle les caractéristiques opérationnelles du logiciel malveillant, permettant d'identifier et de cartographier des comportements spécifiques tels que la manipulation de fichiers, les modifications du registre ou les communications réseau. Ces comportements sont souvent représentés sous forme de TTPs, qui encapsulent les méthodes et stratégies distinctives de l'acteur, établissant ainsi un lien entre le comportement observé et l'identité de l'attaquant.



(a) APT 10. (b) Winnti.

FIGURE 7.1 – Représentations de bytecode d'échantillons de logiciels malveillants – (a) Image de bytecode représentant un logiciel malveillant attribué à APT 10. (b) Image de bytecode représentant un logiciel malveillant attribué à Winnti.

Les tactiques décrivent les objectifs globaux ou stratégiques qui guident une attaque, tels que le maintien de la persistance au sein d'un système, l'escalade des privilèges ou l'exfiltration de données. Les techniques représentent les méthodes spécifiques qu'un logiciel malveillant ou un attaquant utilise pour atteindre ces objectifs, lesquelles peuvent être systématiquement cartographiées en exécutant le logiciel malveillant afin de capturer ses interactions et impacts réels sur le système cible. Par exemple, les attaquants peuvent utiliser l'injection de code pour l'escalade des privilèges ou les protocoles de communication HTTP pour exfiltrer des données. Les procédures font référence aux étapes détaillées, outils ou configurations qu'un attaquant utilise pour exécuter une technique de manière efficace, ces éléments variant souvent d'un acteur à un autre [86].

Les techniques, telles que définies dans le framework ATT&CK, sont particulièrement difficiles à modifier pour les attaquants sans nuire à l'efficacité de leurs opérations et représentent souvent des aspects fondamentaux de la méthodologie de l'attaquant [8]. Par conséquent, les techniques constituent une dimension stable et distinctive au sein de notre framework, permettant une attribution plus cohérente et fiable.

7.2.3 Similarité cosinus

La similarité cosinus est une métrique utilisée pour mesurer la similarité entre deux vecteurs dans un espace multidimensionnel, basée sur le cosinus de l'angle formé entre eux. Cette mesure reflète leur orientation plutôt que leur magnitude. Dans les réseaux siamois, elle est employée pour comparer les embeddings des caractéristiques de deux échantillons d'entrée, en produisant un score indiquant leur degré de similarité. La similarité cosinus entre deux vecteurs A et B est donnée par la formule 7.1, où $A \cdot B$ représente le produit scalaire et $\|A\|, \|B\|$ leurs magnitudes respectives. Cette métrique varie de 0 (peu similaire) à 1 (très similaire), ce qui en fait une mesure précieuse pour

comparer les similarités des embeddings générés par les réseaux siamois.

$$\text{cosine\_similarity}(A, B) = \frac{A \cdot B}{\|A\|\|B\|} \tag{7.1}$$

7.3 Framework proposé

Notre framework pour l'attribution des cyberattaques se compose de trois phases principales. Tout d'abord, nous utilisons un modèle de réseau siamois pour prédire des scores de similarité entre des paires d'échantillons de logiciels malveillants. Dans la deuxième phase, nous créons des profils d'acteurs de la menace en agrégeant les embeddings générés pour des échantillons de logiciels malveillants connus, associés à chaque acteur. Enfin, lors de la phase d'attribution des attaques, de nouveaux échantillons de logiciels malveillants sont comparés à ces profils établis, permettant au framework d'attribuer les attaques à des acteurs spécifiques en fonction des scores de similarité.

7.3.1 Modèle de réseau siamois

L'architecture proposée de réseau siamois, illustrée dans la Figure 7.2, vise à estimer la similarité entre deux échantillons de logiciels malveillants. Cette architecture comprend quatre étapes clés : entrée "input", extraction des caractéristiques "Features Extraction", encodage des caractéristiques "Features Encoding", et Sortie "Output". Chaque étape est expliquée en détail ci-dessous.

Entrée

Le réseau siamois commence par prendre en entrée une paire d'échantillons de logiciels malveillants. Ces paires sont étiquetées '1' si elles appartiennent au même acteur de la menace et '0' si elles appartiennent à des acteurs différents. Cet étiquetage fournit les vérités terrain nécessaires pour que le réseau apprenne les motifs de similarité associés aux différents acteurs de la menace.

Extraction des caractéristiques

Pour chaque paire de logiciels malveillants, nous extrayons à la fois des caractéristiques statiques, représentées par les bytecodes, et des caractéristiques comportementales, représentées par les TTPs, comme suit :

La représentation des bytecodes sous forme d'images permet au modèle de capturer des caractéristiques spatiales et des structures de code récurrentes qui peuvent être associées à des acteurs de la menace spécifiques. Pour ce faire, nous extrayons les données de bytecode de chaque échantillon de logiciel malveillant et les transformons en une image bidimensionnelle (2D) avec une résolution standard de 512x512 pixels. Dans cette représentation, chaque pixel encode un niveau d'intensité de 8 bits correspondant

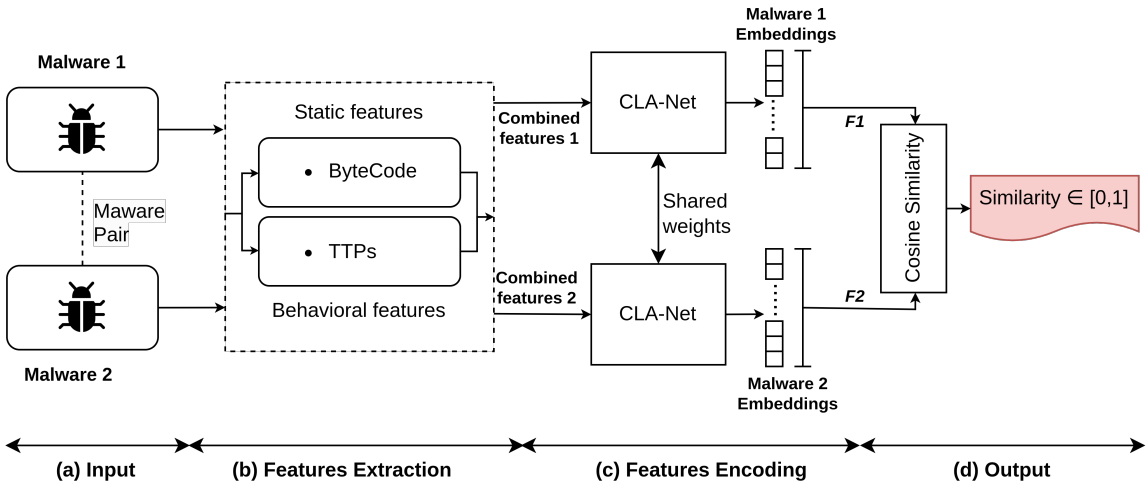


FIGURE 7.2 – Architecture proposée du réseau siamois pour l'estimation de la similarité des logiciels malveillants. (a) Entrée : une paire d'échantillons de logiciels malveillants, étiquetée '1' si les échantillons appartiennent au même acteur de la menace, et '0' sinon. (b) Extraction des caractéristiques : extraction des caractéristiques statiques (ByteCode) et comportementales (TTPs) de chaque échantillon. (c) Encodage des caractéristiques : passage des caractéristiques extraites à travers un réseau partagé CNN+LSTM+Attention (CLA-Net), produisant des embeddings capturant les motifs locaux et les dépendances comportementales séquentielles. (d) Sortie : calcul de la similarité entre les deux représentations d'embeddings à l'aide de la similarité cosinus.

à une valeur de byte dans la séquence de bytecode du logiciel malveillant (comme décrit dans la section 7.2). Lorsque la séquence de bytecode est trop courte pour remplir une grille de 512x512, l'image est complétée avec des valeurs nulles pour garantir des dimensions cohérentes. Cette technique de remplissage préserve la structure originale du logiciel malveillant sans altérer ses caractéristiques clés.

Pour capturer les caractéristiques comportementales, nous analysons les TTPs via l'exécution dynamique. Ce processus consiste à exécuter chaque échantillon dans un environnement contrôlé afin d'observer ses actions sans compromettre la sécurité du système. Pendant cette exécution, nous surveillons et enregistrons les actions du logiciel malveillant, telles que les appels système, les manipulations de fichiers et les communications réseau. Cela nous permet d'identifier les techniques spécifiques employées par le logiciel malveillant et la séquence de ces actions. Après avoir collecté les TTPs, nous construisons un dictionnaire de toutes les techniques observées, aligné avec le framework MITRE ATT&CK. Dans notre ensemble de données (décrit dans la section 7.4.1), nous avons identifié un ensemble de 182 techniques uniques. Pour chaque échantillon, nous créons un vecteur binaire de caractéristiques comportementales en appliquant un encodage "one-hot-encoding" [76]. Si un échantillon utilise une technique particulière, la valeur '1' est attribuée à la position correspondante dans le vecteur ; sinon, une valeur '0' est assignée. Cela donne un vecteur binaire de dimension 182, capturant le

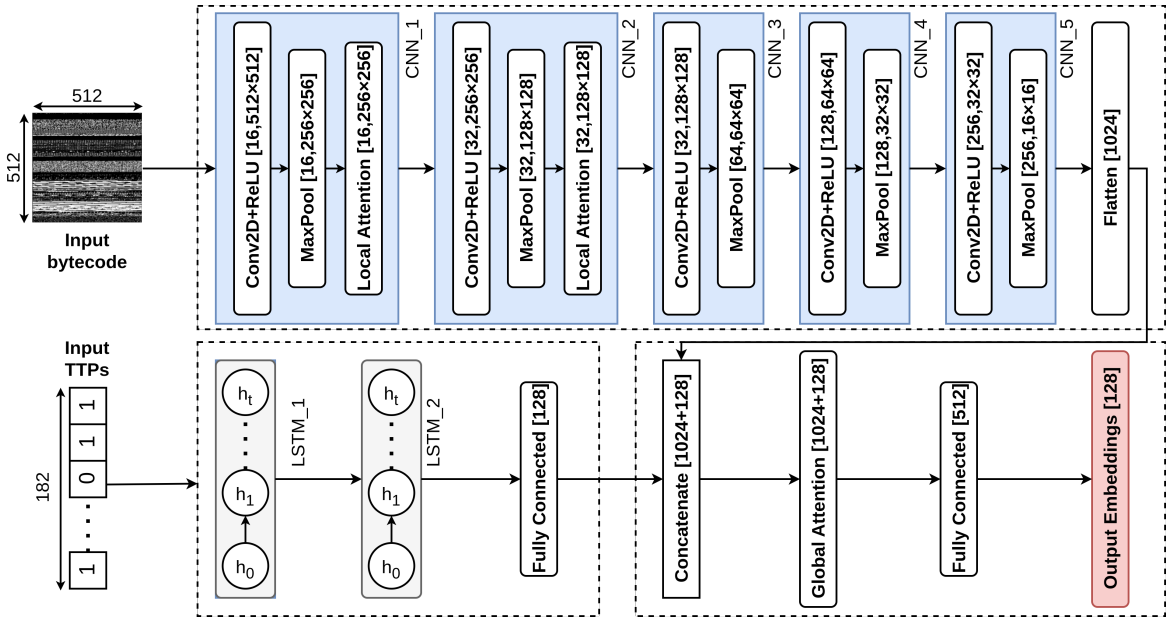


FIGURE 7.3 – Architecture CLA-Net pour l'encodage des caractéristiques des logiciels malveillants : un réseau hybride CNN+LSTM+Attention qui encode les caractéristiques statiques et comportementales en un embedding unifié.

comportement de chaque échantillon.

Encodage des caractéristiques

L'architecture CLA-Net, illustrée à la figure 7.3, est conçue pour encoder les caractéristiques statiques et comportementales des échantillons de logiciels malveillants en combinant des couches convolutives, séquentielles et basées sur l'attention. Ce réseau hybride se compose de trois composants principaux : un module CNNs pour les images de bytecode, un module LSTMs pour les TTPs comportementaux, et une couche d'attention globale intégrée qui fusionne les sorties des deux modules pour créer un embedding unifié.

Le module CNNs traite les images de bytecode représentant les caractéristiques statiques. Il comprend cinq couches convolutives, capturant des motifs spatiaux complexes dans les images. Chaque couche applique des convolutions 2D suivies d'une activation ReLU et d'un max-pooling pour réduire les dimensions spatiales. Les deux premières couches convolutives intègrent des blocs d'attention locale, permettant au modèle de se concentrer sur des régions spécifiques des images associées à des acteurs de la menace particuliers. La sortie finale est aplatit en un vecteur 1D de 1024 dimensions.

Le module LSTMs traite les TTPs comportementaux sous forme séquentielle. Il comprend deux couches LSTMs empilées, capturant les dépendances temporelles dans les données comportementales. La sortie finale est transformée en un vecteur de 128 dimensions.

Les vecteurs CNNs (1024 dimensions) et LSTMs (128 dimensions) sont concaténés, formant un vecteur combiné de 1152 dimensions, puis passés dans une couche d'attention globale et des couches entièrement connectées, produisant un embedding final de 128 dimensions.

Sortie

Après encodage des échantillons, la similarité cosinus est appliquée aux embeddings pour mesurer leur similarité, produisant un score entre 0 et 1. Ce score reflète la probabilité que les échantillons proviennent du même acteur de cybermenace. Durant l'entraînement, les poids de CLA-Net sont optimisés pour maximiser la similarité entre les échantillons du même acteur et la minimiser pour les acteurs différents.

7.3.2 Création des profils des acteurs de la menace

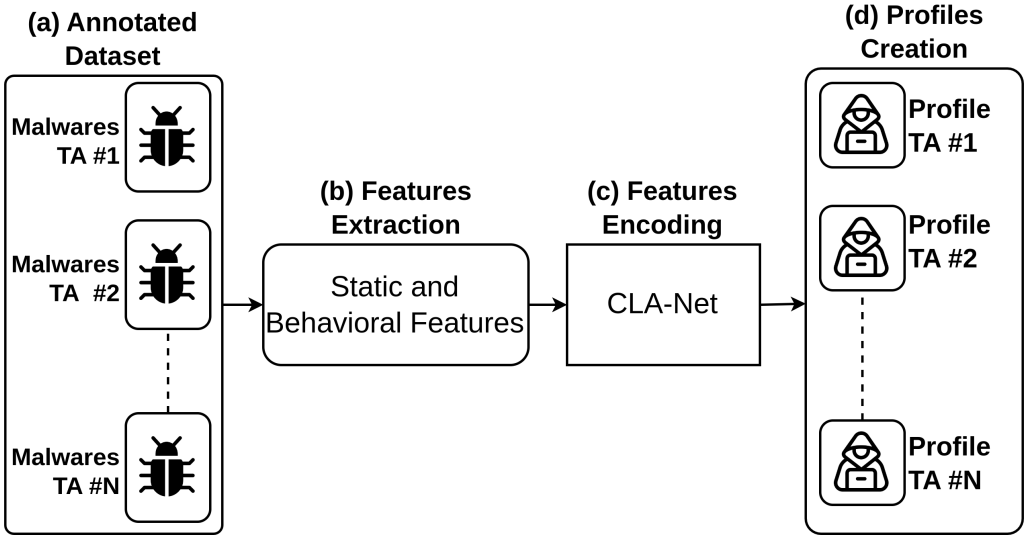


FIGURE 7.4 – Processus de création des profils des acteurs de la menace (TA). (a) Jeu de données annoté : contient une liste d'échantillons de logiciels malveillants étiquetés avec leur TA correspondant. (b) Extraction des caractéristiques : extraction des caractéristiques statiques et comportementales de chaque échantillon. (c) Encodage des caractéristiques : calcul des embeddings pour chaque échantillon à l'aide du CLA-Net entraîné. (d) Création de profils : ajout de chaque embedding calculé à son profil TA correspondant.

La création des profils des acteurs de la menace (TA), illustrée dans la Figure 7.4, consiste à encoder systématiquement et à regrouper les échantillons de logiciels malveillants associés à des TA spécifiques afin de constituer une référence pour les tâches d'attribution.

Le processus commence par un jeu de données annoté contenant des échantillons de logiciels malveillants étiquetés avec leur TA correspondant. Chaque échantillon est attribué à un TA connu sur la base de connaissances préalables, fournissant ainsi la vérité terrain nécessaire pour créer des profils distincts. Pour chaque échantillon de logiciel malveillant, les caractéristiques statiques et comportementales sont extraites, comme expliqué dans la section 7.3.1.

Les caractéristiques extraites sont ensuite passées à travers un réseau pré-entraîné (correspondant à CLA-Net du réseau siamois entraîné) pour calculer un embedding (e) pour chaque échantillon. Pour chaque TA ayant un ensemble d'échantillons associés $\{m_1, m_2, \ldots, m_n\}$, l'embedding de chaque échantillon m_i est calculé et ajouté au profil TA correspondant, représenté par $P_{TA} = \{e_1, e_2, \ldots, e_n\}$, où $e_i = \text{CLA-Net}(m_i)$ désigne l'embedding de l'échantillon m_i obtenu à partir du réseau CLA-Net.

Les profils créés $\{P_{TA}\}$ servent de base de référence et seront utilisés lors de la phase de production pour comparer de nouveaux échantillons de logiciels malveillants et déterminer leur attribution la plus probable en calculant des scores de similarité avec chaque profil.

7.3.3 Attribution des attaques

La phase d'attribution des cyberattaques, illustrée dans la Figure 7.5, s'appuie sur des profils d'acteurs de la menace (TA) préalablement calculés et se compose de cinq étapes principales :

Le processus débute avec un nouvel échantillon de logiciel malveillant nécessitant une attribution. Ensuite, les caractéristiques statiques et comportementales de cet échantillon sont extraites en utilisant la même méthodologie que celle employée pour l'entraînement du réseau siamois. Les caractéristiques extraites sont ensuite passées à travers le CLA-Net entraîné pour calculer l'embedding (e_{new}) du nouvel échantillon.

Cet embedding est comparé à chaque profil TA précalculé. Pour chaque profil TA, nous calculons la similarité cosinus moyenne entre l'embedding du nouvel échantillon et les embeddings présents dans le profil TA. Mathématiquement, la similarité moyenne pour un profil d'acteur T contenant n embeddings de logiciels malveillants $\{e_1, e_2, \ldots, e_n\}$ est donnée par :

$$\text{Similarité moyenne} = \frac{1}{N} \sum_{i=1}^{N} \cos(e_{\text{new}}, e_i)$$

où (e_{new}) est l'embedding du nouvel échantillon, et $\cos(e_{\text{new}}, e_i)$ représente la similarité cosinus entre le nouvel échantillon et le i-ème embedding dans le profil TA.

L'attaque est attribuée au profil TA avec le score de similarité moyenne le plus élevé. Si le score de similarité maximal pour tous les profils est inférieur à un seuil prédéfini, le nouvel échantillon est signalé comme appartenant potentiellement à un acteur de la menace inconnu ou nouveau. Ce mécanisme permet au framework de s'adapter aux

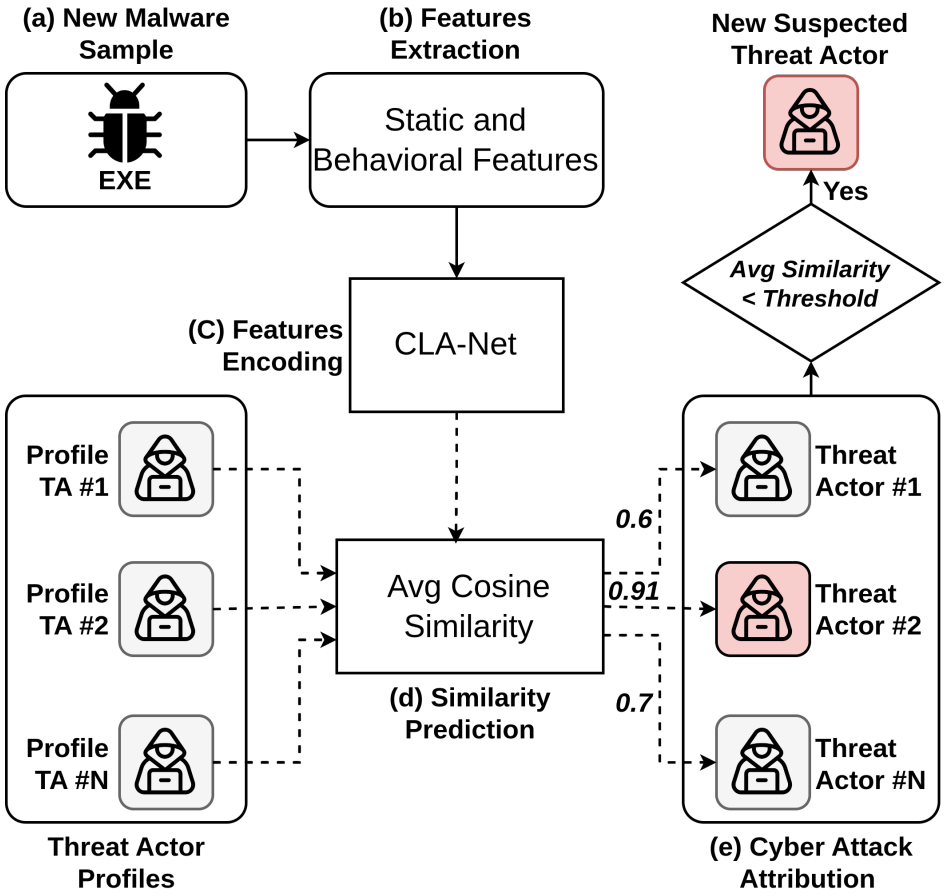


FIGURE 7.5 – Phase de production du framework d'attribution des cyberattaques. (a) Nouveau logiciel malveillant : un échantillon récemment observé. (b) extraction des caractéristiques : extraction des caractéristiques statiques et comportementales de l'échantillon. (c) Encodage des caractéristiques : calcul des embeddings du logiciel malveillant à l'aide d'un réseau pré-entraîné (CLA-Net du réseau siamois). (d) Prédiction de similarité : prédiction des similarités entre les profils TA existants et le nouvel échantillon. (e) Attribution de l'attaque : attribution de l'attaque à l'acteur de la menace avec la plus grande similarité ou signalement en tant que nouvelle menace si la similarité est en dessous d'un seuil prédéfini.

menaces émergentes en identifiant des acteurs inconnus ne correspondant à aucun profil existant.

7.4 Expérimentations

7.4.1 Jeu de données

Le jeu de données utilisé dans cette étude, provenant du dépôt APTMalware [24], comprend plus de 3 500 échantillons de logiciels malveillants associés à 12 acteurs de la menace distincts. Chaque échantillon est étiqueté et regroupé par l'acteur de la menace correspondant. Les échantillons sont nommés à l'aide de valeurs de hachage SHA-256, qui servent d'identifiants uniques.

Les échantillons de logiciels malveillants ont été obtenus via l'API de VirusTotal, qui permet un accès sécurisé aux fichiers binaires nécessaires pour l'extraction des caractéristiques statiques. En outre, VirusTotal offre la possibilité de télécharger des informations comportementales détaillées, y compris les TTPs, au format JSON. Cette fonctionnalité permet de collecter des caractéristiques comportementales dynamiques sans avoir besoin d'exécuter les logiciels malveillants dans un environnement contrôlé.

Une fois tous les échantillons récupérés, nous avons divisé le jeu de données en quatre sous-ensembles principaux, chacun conçu pour remplir un rôle spécifique dans notre framework. Tout d'abord, le jeu de données a été scindé en deux divisions principales. La première division comprend des échantillons provenant de six acteurs de la menace (APT 1, APT 29, Gorgon Group, Equation Group, Winnti, et Energetic Bear), chacun avec un nombre représentatif d'échantillons. Ces échantillons ont été alloués pour l'entraînement, la validation et les tests de notre framework d'attribution des cyberattaques. La deuxième division contient des échantillons des six autres acteurs de la menace (APT 10, APT 19, APT 21, APT 28, APT 30, et Dark Hotel), chacun avec un nombre minimal d'échantillons. Cette division (Subset 1) a été réservée exclusivement pour évaluer la capacité du framework à détecter les acteurs émergents, permettant ainsi une évaluation de son adaptabilité à des acteurs nouveaux et inconnus.

La première division du jeu de données a été ensuite séparée en trois sous-ensembles pour répondre à des tâches distinctes. Subset 2 comprend 30% des échantillons de chaque classe d'acteurs de la menace et a été alloué à l'entraînement et à la validation du réseau siamois, garantissant une distribution équilibrée pour éviter les biais durant l'entraînement. Subset 3, contenant 35% des échantillons de chaque classe, a été destiné à la création des profils d'acteurs de la menace. Ces profils servent de base de référence pour l'attribution des attaques dans la phase de production du framework. Subset 4, qui inclut les 35% restants des échantillons de chaque classe, a été réservé pour évaluer les performances de la tâche d'attribution des attaques. Ce sous-ensemble nous permet d'évaluer la capacité du modèle à attribuer avec précision les cyberattaques à des acteurs de la menace connus en se basant sur la similarité des échantillons de logiciels malveillants. Le tableau 7.1 présente la distribution des données pour les différentes tâches de l'évaluation expérimentale.

Acteur de la menace	Nombre	Total	Acteur de la menace	Nombre	Total
Entraînement/Validation du réseau siamois			Test d'attribution des attaques		
APT 1	121		APT 1	203	
Gorgon Group	288		Gorgon Group	481	
Equation Group	119	768	Equation Group	198	1281
Winnti	116		Winnti	193	
APT 29	84		APT 29	140	
Energetic Bear	40		Energetic Bear	66	
Création des profils des acteurs de la menace			Détection de nouveauté		
APT 1	202		APT 10	244	
Gorgon Group	480		APT 19	32	
Equation Group	197	1280	APT 21	106	1033
Winnti	194		APT 28	514	
APT 29	141		Dark Hotel	273	
Energetic Bear	66		APT 30	164	

TABLE 7.1 – Distribution des échantillons de logiciels malveillants pour différentes tâches, y compris l'entraînement/validation du réseau siamois, la création des profils des acteurs de la menace, les tests d'attribution des attaques et la détection de nouveauté. Chaque sous-ensemble est organisé par acteur de la menace et nombre d'échantillons.

7.4.2 Étiquetage des données pour le réseau siamois

Les réseaux siamois nécessitent un format spécifique pour l'entraînement et la validation, sous forme de paires étiquetées '1' pour des entrées similaires et '0' pour des entrées dissemblables. Dans notre cas, chaque paire d'échantillons de logiciels malveillants doit être étiquetée pour indiquer leur similarité en fonction de leur acteur de la menace. Les paires représentant le même acteur sont étiquetées '1' (similaires), tandis que celles provenant d'acteurs différents sont étiquetées '0' (dissemblables).

Pour faciliter le processus d'annotation, nous avons créé un script qui génère automatiquement cinq paires similaires et cinq paires dissemblables pour chaque échantillon de logiciel malveillant, basé sur sa valeur de hachage SHA-256. Ce processus de génération automatique garantit une représentation équilibrée, permettant au réseau siamois d'apprendre efficacement les distinctions entre les échantillons liés au même acteur de la menace ou à des acteurs différents. Le tableau 7.2 présente la distribution des paires de données étiquetées pour l'entraînement et la validation.

Données	Paires similaires '1'	Paires dissemblables '0'	Total
Entraînement	3264	3264	6528
Validation	576	576	1152
Total	3840	3840	7680

TABLE 7.2 – Distribution des paires de données étiquetées pour l'entraînement et la validation du réseau siamois.

7.4.3 Paramètres expérimentaux

Pour fournir une évaluation complète de notre framework proposé, nous avons mené des expériences selon trois configurations différentes : en utilisant uniquement les caractéristiques statiques, uniquement les caractéristiques comportementales, et une combinaison des deux types de caractéristiques. Cette approche nous a permis d'évaluer l'efficacité de chaque type de caractéristique individuellement et en combinaison.

Nos expériences ont été exécutées sur un GPU NVIDIA GeForce RTX 2070 avec 16 Go de mémoire. Le framework a été implémenté en utilisant la bibliothèque PyTorch. Tous les modèles ont été entraînés sur 25 époques avec un ensemble cohérent d'hyperparamètres. Plus précisément, nous avons utilisé l'optimiseur RMSprop avec une taille de lot de 32 et un taux d'apprentissage de 0,001. Le processus d'entraînement a utilisé la fonction de perte Binary Cross-Entropy with Logits Loss (BCEWithLogitsLoss). Ces paramètres ont été choisis pour équilibrer la vitesse de convergence et les performances des modèles dans les différentes configurations de caractéristiques.

7.4.4 Métriques d'évaluation

Pour évaluer les performances de notre framework, nous avons utilisé un ensemble de métriques couramment employées dans les tâches de classification : la matrice de confusion, la précision (accuracy), la précision positive (precision), le rappel (recall) et le F1-score. Dans notre évaluation, une prédiction est considérée comme correcte si le score de similarité prédit par le réseau siamois est supérieur à 0,5, indiquant que la paire d'échantillons de logiciels malveillants est probablement associée au même acteur de la menace. À l'inverse, si le score de similarité est inférieur ou égal à 0,5, la prédiction est jugée incorrecte, suggérant une dissimilarité entre les échantillons.

— **Matrice de confusion** : utilisée pour visualiser les performances du modèle, elle affiche les comptes des vrais positifs (VP), vrais négatifs (VN), faux positifs (FP) et faux négatifs (FN).

— **Accuracy** : calculée comme le ratio des échantillons correctement prédits sur le nombre total d'échantillons.

$$\text{Accuracy} = \frac{\text{VP} + \text{VN}}{\text{VP}+\text{VN}+\text{FP}+\text{FN}}$$

— **Précision (Precision)** : mesure la proportion des échantillons positifs correctement identifiés parmi tous les échantillons prédits comme positifs. Elle reflète la capacité du modèle à éviter les faux positifs.

$$\text{Precision} = \frac{\text{VP}}{\text{VP} + \text{FP}}$$

— **Rappel (Recall)** : ratio des échantillons positifs correctement identifiés par rapport au nombre total d'échantillons positifs réels. Cette métrique indique l'efficacité du modèle à capturer les cas positifs réels.

$$\text{Recall} = \frac{\text{VP}}{\text{VP} + \text{FN}}$$

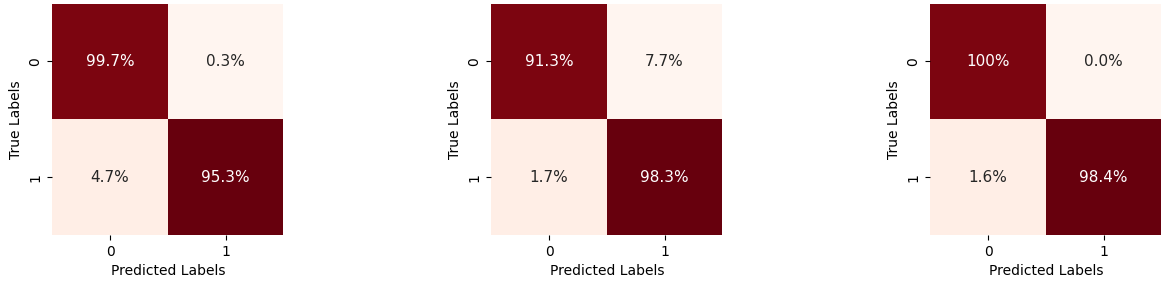
— **F1-score** : moyenne harmonique de la précision positive et du rappel, offrant une métrique unique équilibrant les deux.

$$\text{F1-score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

7.5 Analyse des résultats

7.5.1 Entraînement du réseau siamois

Les performances du réseau siamois ont été évaluées selon trois configurations : en utilisant uniquement les caractéristiques statiques, uniquement les caractéristiques comportementales, et une combinaison des deux types de caractéristiques. La Figure 7.6 illustre les matrices de confusion pour chaque configuration.



(a) Caractéristiques statiques.

(b) Caractéristiques comportementales.

(c) Caractéristiques combinées.

FIGURE 7.6 – Matrices de confusion pour l'entraînement du réseau siamois selon trois configurations : (a) caractéristiques statiques, (b) caractéristiques comportementales, et (c) caractéristiques combinées.

Dans la configuration basée sur les caractéristiques statiques (Figure 7.6a), le modèle affiche une capacité élevée à minimiser les faux négatifs, avec un taux de seulement 0,3% et un taux de vrais positifs de 99,7%. Cela indique que les caractéristiques statiques (bytecode) permettent d'identifier efficacement les paires associées au même acteur de la menace, réduisant ainsi le risque de détections manquées. Cependant, un léger taux de faux positifs de 4,7% est observé, suggérant que les caractéristiques statiques seules peuvent parfois classifier à tort des paires non liées comme similaires.

Dans la configuration basée sur les caractéristiques comportementales (Figure 7.6b), le modèle réduit davantage les faux positifs à 1,7%, reflétant la capacité des caractéristiques comportementales à différencier précisément les acteurs de la menace grâce à leurs TTPs distincts. Cependant, le taux de faux négatifs est légèrement plus élevé (7,7%) comparé au modèle basé sur les caractéristiques statiques, indiquant que les données comportementales seules peuvent parfois ne pas détecter certaines similarités.

La configuration combinant les deux types de caractéristiques (Figure 7.6c) atteint un équilibre parfait, avec un taux de vrais positifs de 100% et un faible taux de faux négatifs de 1,6%. En intégrant les caractéristiques statiques et comportementales, cette configuration gère efficacement les cas de faux positifs et faux négatifs.

Les métriques mesurées pour chaque configuration, notamment la précision (accuracy), la précision positive (precision), le rappel (recall) et le F1-score, sont présentées dans le tableau 7.3.

Caractéristiques utilisées	**Métriques de performance %**			
	Accuracy	**Précision**	**Rappel**	**F1-score**
Statique	97.6	99.8	95.3	97.5
Comportementale	95.3	92.6	**98.5**	95.5
Combinée	**99.2**	**100**	98.3	**99.2**

TABLE 7.3 – Métriques de performance du réseau siamois selon différentes configurations de caractéristiques.

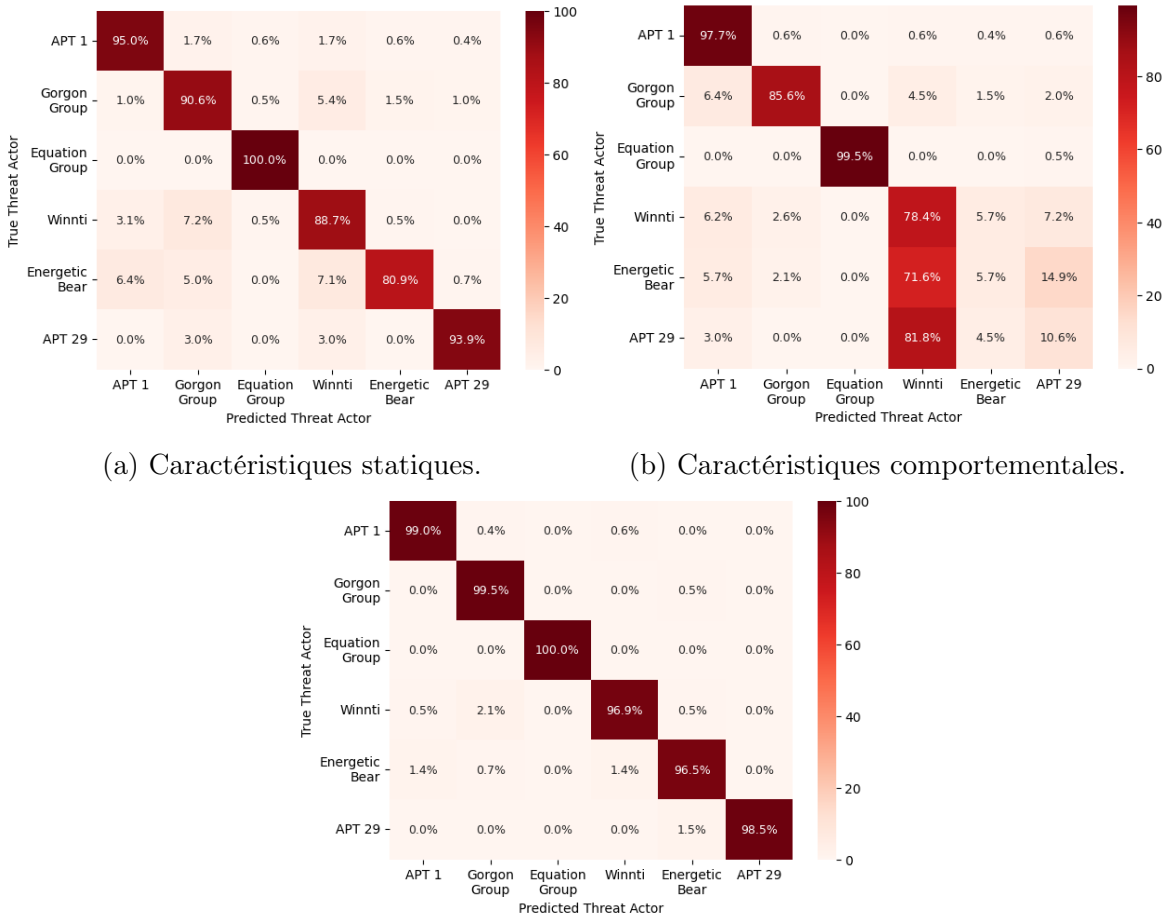
7.5.2 Attribution des attaques

Les performances de notre framework dans l'attribution des cyberattaques ont été évaluées à l'aide de matrices de confusion pour les six acteurs de la menace sélectionnés, comme illustré dans la Figure 7.7. L'évaluation a été réalisée selon trois configurations : caractéristiques statiques, caractéristiques comportementales, et une combinaison des deux.

Notre approche s'appuie sur les scores de similarité du réseau siamois, compris entre 0 et 1. Un score de similarité supérieur à 0,5 est considéré comme une prédiction correcte, indiquant que l'échantillon de logiciel malveillant appartient probablement au même acteur de la menace. Pour chaque échantillon, les scores de similarité ont été calculés par rapport aux profils de tous les acteurs de la menace, et le profil avec la plus haute similarité moyenne a été sélectionné comme prédiction de l'acteur de la menace.

Dans la configuration basée sur les caractéristiques statiques (Figure 7.7a), le framework montre une forte performance d'attribution pour la plupart des acteurs de la menace, avec une précision particulièrement élevée pour Equation Group (100%) et APT 1 (95%). Cependant, des erreurs de classification sont notées, notamment pour Energetic Bear et Winnti, indiquant que les caractéristiques statiques seules ne suffisent pas toujours à capturer les modèles comportementaux uniques de ces acteurs.

La configuration basée sur les caractéristiques comportementales (Figure 7.7b) révèle des résultats différents, avec une haute précision pour des acteurs comme Equation Group (99,5%) et APT 1 (97,7%). Cependant, on observe une baisse notable dans l'attribution correcte des échantillons d'Energetic Bear (5,7%) et d'APT 29 (10,6%), dont une grande partie est souvent mal classée comme appartenant à Winnti. Cela suggère que ces acteurs présentent des similarités comportementales avec Winnti, rendant difficile leur distinction sur la base des caractéristiques comportementales seules.



(a) Caractéristiques statiques.

(b) Caractéristiques comportementales.

(c) Caractéristiques combinées.

FIGURE 7.7 – Matrices de confusion pour l'attribution des attaques parmi six acteurs de la menace utilisant (a) caractéristiques statiques, (b) caractéristiques comportementales, et (c) caractéristiques combinées.

En revanche, la configuration combinée (Figure 7.7c) offre le meilleur équilibre global, avec une précision élevée pour tous les acteurs, y compris une amélioration significative pour Energetic Bear (96,5%) et Winnti (96,9%). L'intégration des caractéristiques statiques et comportementales permet au modèle de tirer parti des forces des deux types de caractéristiques, offrant une représentation plus complète de chaque acteur de la menace et réduisant ainsi les erreurs de classification.

7.5.3 Détection de nouveauté

Pour évaluer la capacité du framework à identifier de potentiels nouveaux acteurs de la menace, nous avons examiné ses performances dans la distinction des menaces émergentes par rapport aux profils d'acteurs de la menace existants.

Dans cette évaluation, nous avons mesuré le score moyen de similarité entre chacun des six nouveaux acteurs de la menace sélectionnés et les profils des acteurs existants. Les résultats, illustrés dans la Figure 7.8, montrent que le framework maintient avec succès des scores de similarité moyens faibles (inférieurs à 0,5) pour les nouveaux acteurs de la menace, les distinguant ainsi des entités connues. Ce seuil moyen de similarité faible agit efficacement comme une limite, permettant au modèle d'identifier de nouveaux ou émergents acteurs de la menace avec un chevauchement minimal avec les profils existants.

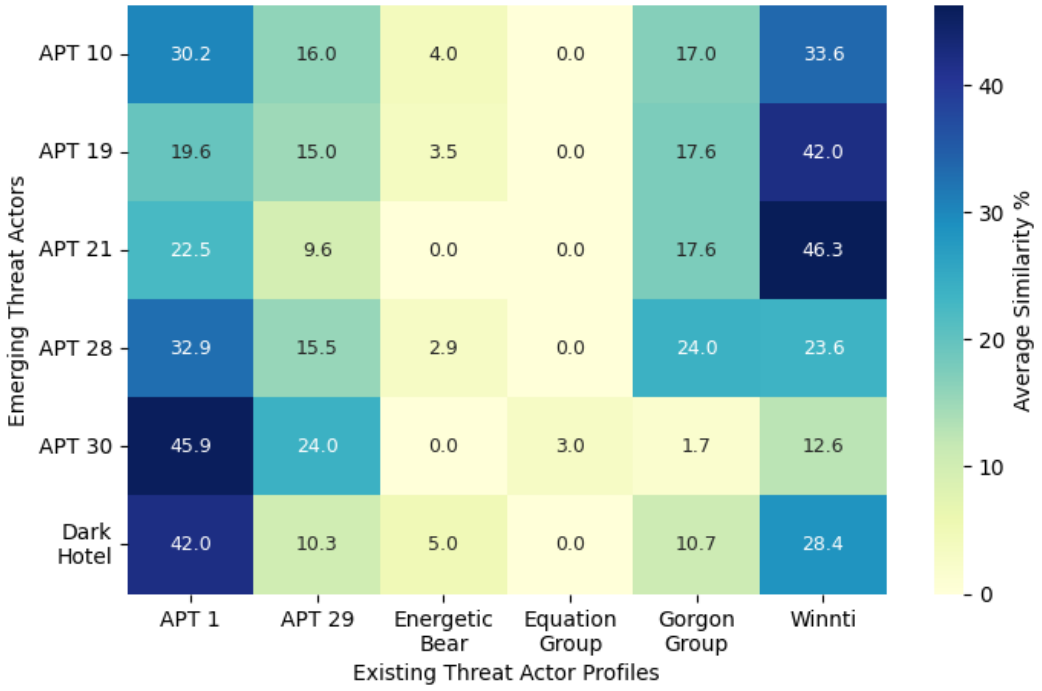


FIGURE 7.8 – Scores moyens de similarité entre les nouveaux acteurs de la menace et les profils existants.

7.5.4 Comparaison des performances

L'une des principales limitations des modèles de classification traditionnels en matière d'attribution des cybermenaces réside dans leur évolutivité et leur adaptabilité. À mesure que le nombre de classes d'acteurs de la menace augmente, ces modèles peinent à maintenir leur précision, souvent en raison de la rigidité des frontières de classification. Notre framework aborde ce défi en comparant les nouveaux échantillons de logiciels malveillants à plusieurs signatures au sein de chaque profil d'acteur de la menace, permettant ainsi un processus d'attribution plus précis. En calculant un score de similarité entre le nouvel échantillon et les signatures existantes, notre méthode affine l'attribution pour identifier l'acteur de la menace le plus probable.

Une autre limitation majeure des modèles de classification est leur incapacité à gérer les nouveaux acteurs émergents, car ils sont conçus pour classer les échantillons dans des catégories prédéfinies. Notre framework surmonte cette limitation en identifiant de potentiels nouveaux acteurs : si un échantillon ne correspond étroitement à aucun profil connu, il est signalé comme une menace potentiellement nouvelle, déclenchant la création d'un nouveau profil.

La comparaison des performances de notre framework avec les études citées est présentée dans le tableau 7.4. Il est important de noter que chaque étude a utilisé des jeux de données différents, donc cette comparaison est basée sur les métriques de performance rapportées dans les articles respectifs. Les résultats indiquent que notre framework surpasse les autres approches en termes de précision, de rappel et de F1-score sur un ensemble de données plus large.

Étude	**Entraînement/ Validation**	**Accuracy**	**Precision**	**Rappel**	**F1-score**
[69]	172	58.4	55.0	52.4	N/A
[64]	327	94.0	90.0	89.0	89.0
[42]	238	86.5	83.3	85.4	87.9
[41]	864	**98.8**	97.5	98.0	97.8
[40]	1200	95.2	94.4	97.4	95.9
Notre travail	**3329**	98.4	**100**	**98.4**	**99.2**

TABLE 7.4 – Comparaison des performances de notre framework avec d'autres approches d'attribution des cyberattaques en termes de précision, précision positive, rappel et F1-score.

7.6 Conclusion

Cette recherche propose un framework complet pour l'attribution des cyberattaques, répondant aux limitations des modèles de classification traditionnels en intégrant des caractéristiques statiques et comportementales via une approche basée sur un réseau siamois. Les principales contributions de notre travail incluent une meilleure scalabilité pour gérer un grand nombre d'acteurs de la menace, une adaptabilité permettant d'identifier de nouvelles menaces émergentes, et la capacité d'exploiter à la fois les informations statiques et dynamiques des logiciels malveillants.

Nos résultats expérimentaux, avec un score F1 impressionnant de 99,2% et une précision de 98,4% en utilisant des caractéristiques combinées, valident l'efficacité de notre framework. Grâce à son adaptabilité, notre framework est bien adapté aux scénarios réels, où différents types de données (statiques, dynamiques ou combinées) sont disponibles. Cette polyvalence garantit que notre approche peut être déployée dans divers environnements de cybersécurité, en faisant un outil robuste pour des paysages de menaces dynamiques.

Chapitre 8

Prédiction de niveau de gravité des cybermenaces

8.1 Introduction

La CTI joue un rôle crucial dans la cybersécurité, permettant aux organisations de collecter, analyser et interpréter des données sur les menaces pour anticiper les attaques. L'un des aspects essentiels de cette démarche est la prédiction de la gravité des incidents de sécurité. Cette étape est fondamentale pour prioriser les réponses et optimiser les ressources disponibles. En classant les menaces en fonction de leur gravité, les organisations peuvent adopter une défense proactive contre les menaces les plus dangereuses, limitant ainsi leur impact potentiel sur les systèmes critiques.

Traditionnellement, la prédiction de la gravité des menaces repose sur plusieurs approches, telles que : l'analyse des malwares, où les caractéristiques statiques et dynamiques sont évaluées pour estimer leur potentiel destructeur ; l'analyse des vulnérabilités, basée sur les scores CVSS des identifiants CVE pour évaluer leur sévérité ; et l'exploitation des rapports d'incidents, dont les descriptions textuelles fournissent des indices précieux sur la gravité des attaques. Toutefois, ces méthodes reposent fortement sur l'intervention humaine, notamment pour l'interprétation des données et la classification des incidents, ce qui les rend non seulement coûteuses, mais également sujettes à des erreurs et des incohérences.

Pour automatiser cette tâche, plusieurs travaux qui utilisent les techniques d'apprentissage automatique ont été adoptés. Celles-ci exploitent le NLP pour analyser les descriptions textuelles des rapports d'incidents. Cependant, ces modèles utilisent des techniques d'embedding traditionnelles comme Word2Vec [59] et GloVe [68], qui présentent des limitations importantes. Ces modèles génèrent des représentations vectorielles statiques des mots, incapables de capturer le contexte sémantique global ou les dépendances à longue portée dans les textes. En conséquence, leur capacité à interpréter correctement les descriptions complexes des menaces est limitée, entraînant des performances sous-optimales.

L'introduction de BERT [29] a marqué une avancée majeure dans le traitement du langage naturel. Contrairement aux techniques d'embedding traditionnelles, BERT permet d'encoder le contexte bidirectionnel des textes, offrant des représentations plus riches et précises. Ses capacités d'attention améliorent considérablement la compréhension des textes complexes, comme les rapports d'incidents en cybersécurité. De plus, des variantes spécialisées de BERT ont vu le jour, telles que SecureBERT et SecureBERT+ [2], optimisées pour les tâches spécifiques à la cybersécurité. Ces modèles adaptent BERT pour mieux traiter les descriptions de menaces, en exploitant des corpus de données spécialisés, ce qui améliore la performance sur des tâches telles que la classification de la gravité.

Dans cette étude, nous proposons une approche innovante basée sur des encodeurs LLMs tels que BERT pour prédire la gravité des cybermenaces à partir de rapports d'incidents. Le modèle prend en entrée un rapport d'incident, encode les descriptions textuelles en utilisant BERT, puis applique un mécanisme d'attention suivi par des couches LSTMs pour capturer les dépendances à longue portée et les motifs récurrents dans les données textuelles. Enfin, le modèle classe les incidents en quatre catégories de gravité : "Low", "Medium", "High", et "Critical". Cette approche vise à surmonter les limitations des méthodes existantes en améliorant la précision et la cohérence des prédictions, tout en réduisant la dépendance à l'intervention humaine.

En résumé, les principales contributions de ce travail sont les suivantes :

— Développement d'un modèle de prédiction de la gravité des incidents à partir des rapports d'incidents de sécurité. Ce modèle exploite les encodeurs LLMs tels que BERT, intègre des mécanismes d'attention et des couches LSTMs, afin de classifier la gravité des incidents en quatre catégories : "Low", "Medium", "High", et "Critical".

— Analyse de l'impact des modèles d'embedding, notamment BERT et SecureBERT+, sur la précision des prédictions.

Le reste de ce document est organisé comme suit : la section 8.2 présente le modèle proposé. Les résultats expérimentaux sont discutés dans les sections 8.3 et 8.4. Enfin, la section 8.5 conclut ce travail.

8.2 Modèle proposé

Le modèle proposé illustré dans la figure 8.1 est conçu pour prédire la gravité des incidents de sécurité en exploitant les rapports d'incidents rédigés en langage naturel. Il repose sur une architecture composée de cinq couches principales, chacune jouant un rôle crucial dans le processus de classification.

8.2.1 Couche d'entrée

La première étape du modèle consiste à recevoir un rapport d'incident en texte brut. Étant donné la nature variée et parfois désorganisée des données textuelles, un

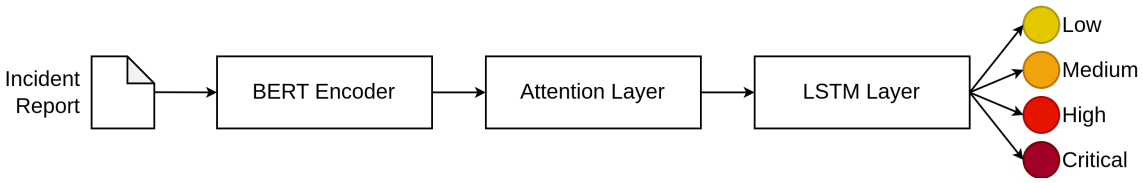


FIGURE 8.1 – Modèle proposé pour la prédiction de la gravité des cybermenaces.

prétraitement rigoureux est essentiel pour garantir la qualité des données d'entrée. Les opérations de prétraitement incluent : suppression des lignes vides : élimine les espaces inutiles qui pourraient introduire du bruit. Double espaces et mise en minuscules : Assure la standardisation du texte. Tokenisation : divise le texte en unités lexicales ou tokens. Padding : harmonise la longueur des séquences pour un traitement par lots. Ces opérations permettent de normaliser les données et de les rendre exploitables par l'encodeur BERT, tout en préservant la sémantique du texte.

8.2.2 BERT Encoder

Après le prétraitement, le texte passe par un encodeur BERT. Chaque token est transformé en un vecteur de dimension 768, encapsulant des informations sémantiques et contextuelles. BERT utilise des transformeurs bidirectionnels pour comprendre le contexte des mots à partir de leurs relations avec les mots précédents et suivants. Cela permet de capturer des informations contextuelles riches, ce qui est efficace pour comprendre les nuances des rapports d'incidents en cybersécurité, où des termes spécifiques peuvent avoir des significations différentes selon leur contexte.

8.2.3 Couche d'attention

Les vecteurs générés par BERT sont ensuite passés dans une couche d'attention. Cette couche attribue des poids aux tokens en fonction de leur pertinence pour la prédiction de la gravité. Dans un rapport d'incident, tous les tokens n'ont pas le même poids pour déterminer la gravité. Par exemple, des termes comme "exploit" ou "Critical" peuvent être plus significatifs que des mots communs. La couche d'attention permet de mettre l'accent sur les tokens pertinents, améliorant ainsi la capacité du modèle à identifier les indices critiques dans le texte.

8.2.4 Couche LSTM

Les vecteurs pondérés par la couche d'attention sont ensuite traités par une couche LSTMs. LSTMs est particulièrement efficace pour capturer les dépendances séquentielles et les dépendances à longue portée dans les données textuelles. Les rapports d'incidents contiennent souvent des informations critiques réparties dans différentes

parties du texte. La couche LSTMs aide à modéliser ces relations temporelles ou séquentielles. Elle affine les poids attribués par la couche d'attention, améliorant ainsi la cohérence et la pertinence des prédictions finales.

8.2.5 Couche de sortie

Enfin, les sorties de la couche LSTMs sont acheminées vers une couche entièrement connectée qui effectue la classification finale. Le modèle prédit la gravité du rapport d'incident en l'une des quatre catégories : "Low", "Medium", "High", ou "Critical".

8.3 Expérimentations

8.3.1 Dataset

Pour les expérimentations, nous avons utilisé un jeu de données issu de la base des CVE, accessible via le site officiel CVE [20]. Cette base de données constitue une référence essentielle pour la communauté de la cybersécurité, centralisant des informations détaillées sur les vulnérabilités connues. Les données extraites sont au format JSON, chaque fichier contenant des informations variées, notamment la description en langage naturel des menaces, et son score CVSS. Ce score, compris entre 0 et 10, reflète la gravité de la vulnérabilité, avec 0 correspondant à une gravité négligeable et 10 indiquant une vulnérabilité critique.

Pour garantir que le modèle soit évalué sur des données représentatives des menaces actuelles, nous avons sélectionné les descriptions textuelles publiées entre 2022 et 2024. Cette période récente permet de prendre en compte les tendances émergentes et les nouvelles formes de menaces.

8.3.2 Annotation des données

Chaque description textuelle extraite des fichiers JSON est annotée en utilisant le score CVSS correspondant. Ce score, qui évalue la gravité de la vulnérabilité sur une échelle de 0 à 10, a été transformé en quatre catégories distinctes de gravité pour les besoins de la classification. Les catégories définies sont les suivantes : "Low" (score entre 0 et 3.9), "Medium" (score entre 4.0 et 6.9), "High" (score entre 7.0 et 8.9), et "Critical" (score entre 9.0 et 10). Cette catégorisation permet de simplifier l'évaluation de la gravité tout en conservant les informations nécessaires pour prioriser efficacement les menaces.

Le tableau 8.1 représente la distribution des classes sur l'ensemble d'entraînement et de test :

TABLE 8.1 – Distribution des classes dans le dataset.

Dataset	Low	Medium	High	Critical	Total
Train set	1600	1600	1600	1600	6400
Test set	400	400	400	400	1600
Total	2000	2000	2000	2000	8000

8.3.3 Paramètres expérimentaux

Pour évaluer de manière exhaustive le modèle proposé, nous avons mené des expériences selon deux configurations différentes de la couche d'encodage : l'une utilisant BERT et l'autre SecureBERT+. Les expériences ont été exécutées sur un GPU NVIDIA GeForce RTX 2070 équipé de 16 Go de mémoire. Le modèle a été implémenté à l'aide de la bibliothèque PyTorch.

Chaque modèle a été entraîné sur 100 époques, avec un ensemble d'hyperparamètres. Nous avons utilisé l'optimiseur RMSprop, une taille de lot de 32, et un taux d'apprentissage fixé à 0,001. Le processus d'entraînement a été supervisé par la fonction de perte CrossEntropyLoss. Ces paramètres ont été soigneusement sélectionnés pour trouver un équilibre entre la vitesse de convergence et les performances globales des modèles dans différentes configurations.

8.4 Analyse des résultats d'entraînement

Les performances des modèles BERT et SecureBERT+ ont été évaluées à l'aide des métriques de précision, rappel et F1-score pour quatre catégories de gravité des incidents : "Critical", "High", "Medium" et "Low". Les résultats sont synthétisés dans le tableau 8.2.

TABLE 8.2 – Comparaison des performances des modèles BERT et SecureBERT+.

Classe	BERT			SecureBERT+		
	Précision	**Rappel**	**F1-score**	**Précision**	**Rappel**	**F1-score**
Critical	**0.95**	0.83	0.88	0.92	**0.99**	**0.95**
High	0.83	0.77	0.80	**0.85**	**0.90**	**0.88**
Medium	0.77	0.79	0.78	**0.90**	**0.86**	**0.88**
Low	0.81	**0.95**	0.87	**0.96**	0.88	**0.92**
Moyenne	0.84	0.835	0.833	**0.908**	**0.908**	**0.908**

Dans la classe "Critical", SecureBERT+ surpasse BERT avec un rappel de 0.99 contre 0.83, indiquant une meilleure capacité à détecter presque toutes les vulnérabilités critiques. Cela est particulièrement crucial dans un contexte de cybersécurité, où des incidents critiques nécessitent une attention immédiate. En revanche, BERT obtient une précision légèrement supérieure (0.95 contre 0.92), montrant qu'il fait moins de

fausses alertes pour cette classe. Cependant, le F1-score global de SecureBERT+ (0.95) reste nettement meilleur, démontrant un équilibre supérieur entre précision et rappel.

Pour la classe "High", SecureBERT+ affiche également de meilleures performances dans toutes les métriques. Son F1-score de 0.88 dépasse significativement celui de BERT (0.80), soulignant une amélioration notable dans la classification des vulnérabilités de gravité élevée.

La classe "Medium" révèle une amélioration encore plus marquée avec SecureBERT+. Ce dernier atteint une précision de 0.90, un rappel de 0.86, et un F1-score de 0.88, tandis que BERT reste en deçà avec un F1-score de 0.78. Cela démontre que SecureBERT+ est plus efficace dans la détection et la classification des vulnérabilités de gravité moyenne.

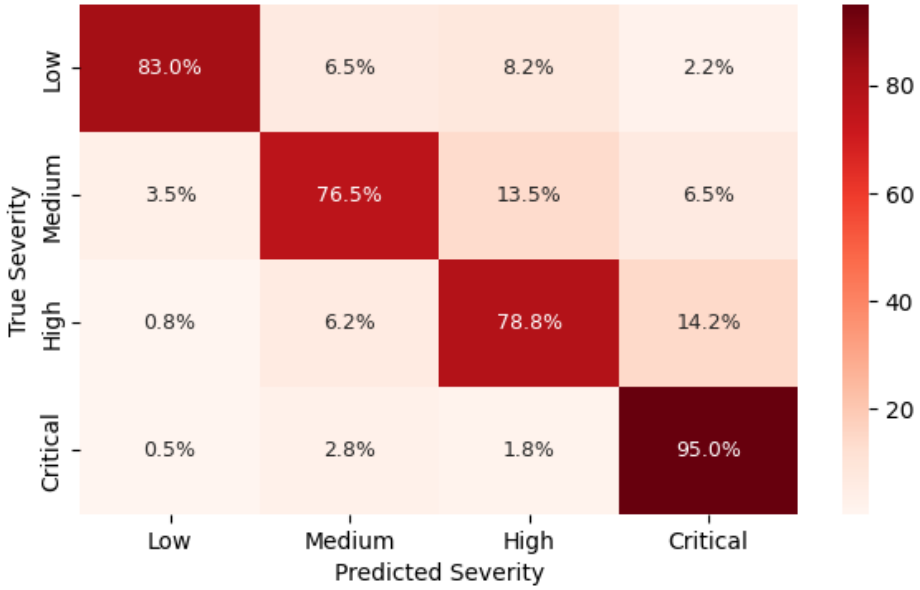
Enfin, pour la classe "Low", SecureBERT+ affiche une précision de 0.96, ce qui surpasse de loin celle de BERT (0.81). Toutefois, BERT présente un rappel légèrement supérieur (0.95 contre 0.88), ce qui indique qu'il identifie mieux les vulnérabilités de faible gravité. Malgré cela, le F1-score de SecureBERT+ (0.92) demeure supérieur, confirmant son efficacité globale.

8.4.1 Analyse des matrices de confusion

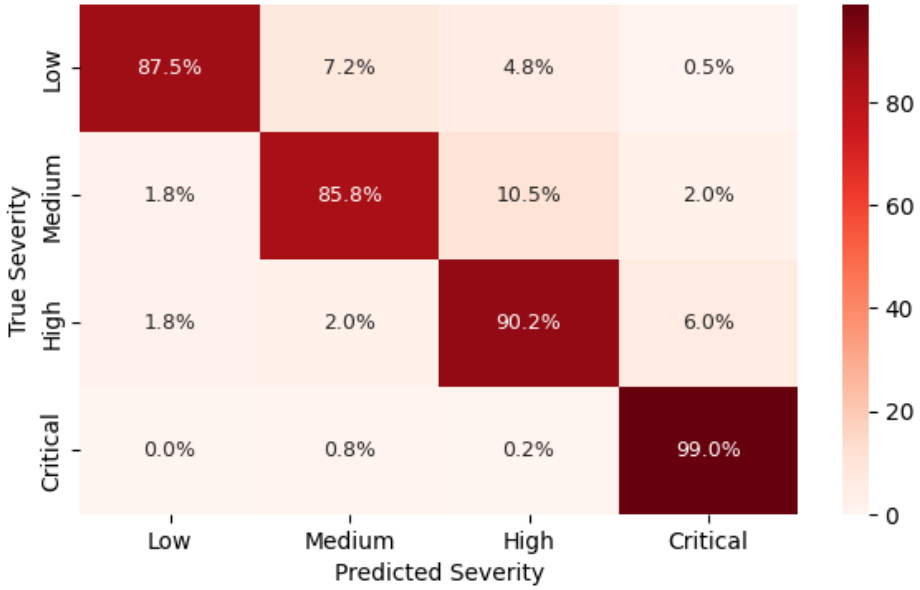
Les matrices de confusion des modèles BERT et SecureBERT+, illustrées dans la figure 8.2, montrent les performances de classification pour chaque niveau de gravité : "Low", "Medium", "High", et "Critical". Ces figures mettent en évidence les confusions entre les classes adjacentes, ce qui peut être attribué à la proximité des scores de gravité associés. En effet, des incidents dont les scores se situent près des seuils de changement de catégorie peuvent être classés à tort dans une classe voisine.

Dans le cas de BERT, on observe une précision correcte pour les classes "Low" et "Critical", avec respectivement 83% et 95% des rapports bien classés. Cependant, des confusions significatives sont apparues entre les classes adjacentes, notamment entre "Medium" et "High", où 13.5% des échantillons de "Medium" ont été classés à tort comme "High". De même, 14.2% des incidents de gravité "High" ont été confondus avec la classe "Critical". Ces erreurs sont dues à la proximité des scores utilisés pour catégoriser les incidents. Par exemple, un score de 8.8 (catégorie "High") est proche du seuil "Critical" (9.0), rendant difficile la distinction.

Avec SecureBERT+, on observe une amélioration significative dans toutes les classes. La précision pour la classe "Low" est passée à 87.5%, tandis que la classe "Critical" atteint une performance quasi parfaite avec 99% des échantillons correctement classés. Les confusions entre les classes "High" et "Critical" sont réduites, avec seulement 6% des incidents "High" mal classés comme "Critical". SecureBERT+ montre également une meilleure performance pour la classe "Medium", avec une précision de 85.8%, réduisant les erreurs par rapport à BERT.



(a) Modèle utilisant BERT.



(b) Modèle utilisant SecureBERT+.

FIGURE 8.2 – Matrices de confusion pour la prédiction niveau de gravité des cybermenaces.

8.5 Conclusion

Dans ce chapitre, nous avons proposé un modèle pour la prédiction du niveau de gravité des incidents de sécurité à partir des descriptions textuelles des rapports d'in-

cidents. Ce modèle s'appuie sur des LLMs tels que BERT et SecureBERT+, intégrant des mécanismes d'attention et des couches LSTMs pour capturer les relations contextuelles et séquentielles dans les données. Nos contributions principales incluent : le développement d'une architecture capable de classifier automatiquement la gravité des incidents en quatre catégories : "Low", "Medium", "High", et "Critical". Une analyse comparative entre BERT et SecureBERT+, démontrant que SecureBERT+ surpasse BERT sur toutes les métriques d'évaluation.

Le meilleur score obtenu est de 90.8% en terme de score F1 avec SecureBERT+, confirmant sa supériorité dans la classification des incidents de sécurité par rapport à BERT.

Ce travail est particulièrement pertinent pour la CTI, un domaine essentiel de la cybersécurité visant à collecter, analyser et interpréter des données sur les menaces émergentes. Une classification précise de la gravité des incidents permet aux organisations de prioriser leurs ressources et de se concentrer sur les menaces les plus critiques.

Chapitre 9

Conclusion générale

Dans un monde où les cybermenaces évoluent à un rythme sans précédent, cette thèse s'est attachée à répondre aux défis posés par la Cyber Threat Intelligence. Les approches traditionnelles de la CTI, bien qu'essentielles, peinent à suivre l'évolution fulgurante en complexité des menaces modernes. Ce constat a motivé le recours à des techniques d'apprentissage automatique afin d'améliorer l'efficacité des processus de CTI.

Quatre contributions majeures caractérisent le présent travail de recherche. Tout d'abord, la première contribution consiste en la proposition d'un nouveau modèle de détection en temps réel des informations liées aux cybermenaces dans les tweets. Ce modèle a été principalement conçu pour résoudre deux problèmes majeurs rencontrés dans les travaux existants sur la détection des menaces à travers les tweets. Le premier problème concerne la dépendance excessive de ces modèles à la quantité de données d'entraînement, ce qui limite leur capacité de généralisation en phase opérationnelle. Pour y remédier, nous avons adopté le modèle BERT. Grâce à son apprentissage sur un vaste ensemble de données, ce modèles offre une meilleure représentation contextuelle et sémantique, ce qui améliore la capacité de généralisation avec un volume de données réduit. Ensuite, le deuxième problème réside dans l'évolution constante de la nature des données publiées sur Twitter. Cette variabilité rend les modèles de classification traditionnels inefficaces face aux changements, entraînant des problèmes tels que le "data drift", où le modèle commence à voir ses performances se dégrader au fil du temps et nécessite un entraînement périodique pour être maintenu à jour. Pour surmonter cette limitation, nous avons adopté une approche basée sur l'apprentissage par renforcement, qui permet au modèle d'apprendre continuellement grâce à un mécanisme de relecture d'expérience. Ainsi, il peut évoluer sans nécessiter de nouvelle phase d'entraînement, contrairement aux modèles de classification traditionnels. Enfin, les résultats expérimentaux ont démontré que notre approche surpasse les méthodes existantes en termes de performances.

Dans la deuxième contribution, nous avons introduit un modèle hybride pour l'extraction et la reconnaissance des IoCs critiques à partir de rapports d'incidents de sécurité, écrits en langage naturel et sous un format non-structuré. Le modèle proposé

combine les encodeurs LLMs, tels que le modèle BERT, avec un modèle d'encodage spécifique au domaine de la cybersécurité. Cette combinaison a amélioré la capacité du modèle à comprendre aussi bien les termes généraux que ceux liés à la CTI. Dans cette perspective, les résultats expérimentaux ont démontré une amélioration remarquable en comparaison avec d'autres travaux.

La troisième contribution propose une nouvelle méthode pour l'attribution des cyberattaques à des acteurs des cybermenaces, en se basant sur l'analyse des caractéristiques statiques et dynamiques des logiciels malveillants. Cette méthode est conçue pour dépasser les limites des modèles de classification, qui sont généralement contraints par un nombre fixe de classes (acteurs d'attaque) et qui les rend ainsi moins adaptés à la nature du problème. De plus, les modèles de classification ont du mal à maintenir une précision élevée lorsque le nombre d'acteurs (classes) augmente. Par ailleurs, ces modèles ne possèdent pas la capacité de détecter de nouveaux acteurs jamais rencontrés lors de la phase d'entraînement. Dans ce contexte, notre approche, basée sur un réseau siamois, les réseaux de neurones convolutifs et des mécanismes d'attention, s'est révélée efficace en matière de précision d'attribution des acteurs malveillants, tout en permettant la détection de nouveaux acteurs de menaces émergents.

Enfin, dans la dernière contribution, nous avons introduit un modèle de prédiction de la gravité des cybermenaces à partir de rapports d'incidents écrits en langage naturel. Ce modèle a été conçu pour améliorer la précision des méthodes existantes qui utilisent le NLP. Dans un premier temps, nous avons exploité le modèle BERT afin de générer une représentation contextuelle et sémantique du texte à partir de rapports d'incidents. Ensuite, nous avons utilisé ces représentations pour entraîner un modèle de classification capable de catégoriser les rapports en fonction de la gravité de la cybermenace selon quatre classes : Low, Medium, High et Critical. Cependant, l'utilisation de BERT peine à s'adapter au contexte particulier de la cybersécurité. Pour pallier cette limitation, nous avons également exploré le modèle SecureBERT+, un variant du modèle BERT, fine-tuné sur un vaste ensemble de données spécifiques à la cybersécurité, permettant une amélioration significative de la précision du modèle proposé.

Chacune des quatre contributions présentées dans cette thèse a été explicitement conçue en tenant compte du facteur temps, élément crucial dans le domaine de la cybersécurité. La première contribution permet une détection proactive et en temps réel des menaces publiées sur les réseaux sociaux, réduisant drastiquement le délai entre l'apparition d'une menace et sa détection effective. En intégrant un apprentissage continu, elle élimine la nécessité d'une ré-entraînement fréquent, économisant ainsi un temps précieux. La deuxième contribution accélère considérablement l'extraction automatique des indicateurs critiques depuis les rapports d'incidents, en évitant la lenteur inhérente aux méthodes manuelles traditionnelles. La troisième contribution apporte une rapidité accrue à l'attribution des cyberattaques en automatisant le processus d'identification des acteurs malveillants, tout en maintenant une précision élevée même avec des acteurs inconnus, évitant ainsi les retards liés aux approches conventionnelles. Enfin, la quatrième contribution optimise le temps nécessaire pour évaluer la gravité d'une menace, permettant une priorisation rapide des incidents selon leur criticité, ce qui

améliore les temps de réponse opérationnels. Globalement, ces avancées renforcent substantiellement la réactivité de la CTI, permettant une réponse rapide et efficace face à l'évolution rapide et constante des cybermenaces.

Dans des travaux futurs, l'amélioration des méthodes proposées reste une perspective valable afin qu'elles puissent être adoptées comme des solutions fiables pour renforcer la sécurité des systèmes. La première piste de recherche consiste à déployer le modèle de collecte des informations liées aux cybermenaces à partir des réseaux sociaux dans des conditions réelles, ce qui peut engendrer un problème de scalabilité en raison du volume massif de données publiées quotidiennement. Pour répondre à ce défi, des solutions basées sur le traitement distribué et l'optimisation des algorithmes de filtrage et de priorisation des informations pourraient être envisagées. Une autre piste de recherche consiste à étudier les relations entre les différentes entités extraites à partir des rapports d'incidents, afin de mieux comprendre les schémas de menace. Cette approche permettra d'identifier des scénarios de menace complexes et d'améliorer les stratégies de défense proactive.

Bibliographie

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow : Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Ehsan Aghaei, Xi Niu, Waseem Shadid, and Ehab Al-Shaer. Securebert : A domain-specific language model for cybersecurity. In *International Conference on Security and Privacy in Communication Systems*, pages 39–56, 2022.

[3] Yacine Akrour. *Prévision de la demande d'électricité par régression linéaire et réseaux de neurones artificiels : application au réseau de la ville de Baie-Comeau.* PhD thesis, Université du Québec à Trois-Rivières, 2022.

[4] Otis Alexander, Misha Belisle, and Jacob Steele. Mitre att&ck for industrial control systems : Design and philosophy. *The MITRE Corporation : Bedford, MA, USA*, 29, 2020.

[5] Fernando Alves, Pedro Miguel Ferreira, and Alysson Bessani. Design of a classification model for a twitter-based streaming threat monitor. In *2019 49th annual IEEE/IFIP international conference on dependable systems and networks workshops (DSN-W)*, pages 9–14. IEEE, 2019.

[6] Mohammed Asiri, Neetesh Saxena, Rigel Gjomemo, and Pete Burnap. Understanding indicators of compromise against cyber-attacks in industrial control systems : a security perspective. *ACM transactions on cyber-physical systems*, 7(2) :1–33, 2023.

[7] Gbadebo Ayoade, Swarup Chandra, Latifur Khan, Kevin Hamlen, and Bhavani Thuraisingham. Automated threat report classification over multi-source data. In *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, pages 236–245. IEEE, 2018.

[8] Mohamed El Amine Bekhouche and Kamel Adi. A bert-based framework for automated extraction of behavioral indicators of compromise from security incident reports. In *International Symposium on Foundations and Practice of Security*, pages 219–232. Springer, 2023.

[9] Navneet Bhatt, Adarsh Anand, and Venkata SS Yadavalli. Exploitability prediction of software vulnerabilities. *Quality and Reliability Engineering International*, 37(2) :648–663, 2021.

[10] David Bianco. The pyramid of pain. *Enterprise Detection & Response*, 2013.

[11] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5 :135–146, 2017.

[12] Avishek Bose, Vahid Behzadan, Carlos Aguirre, and William H Hsu. A novel approach for detection and ranking of trendy and emerging cyber threat events in twitter streams. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 871–878, 2019.

[13] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*, 6, 1993.

[14] Rebekah Brown and Robert M Lee. The evolution of cyber threat intelligence (cti) : 2019 sans cti survey. *SANS Institute. Available online : https ://www. sans. org/white-papers/38790/(accessed on 12 July 2021)*, 2019.

[15] Sarah Brown, Joep Gommers, and Oscar Serrano. From cyber security information sharing to threat management. In *Proceedings of the 2nd ACM workshop on information sharing and collaborative security*, pages 43–49, 2015.

[16] Thomas Oakley Browne, Mohammad Abedin, and Mohammad Jabed Morshed Chowdhury. A systematic review on research utilising artificial intelligence for open source intelligence (osint) applications. *International Journal of Information Security*, pages 1–28, 2024.

[17] AI but Simple. Modern convolutional neural network architectures, 2025. Accessed : 2025-01-17.

[18] Thanasis Chantzios, Paris Koloveas, Spiros Skiadopoulos, Nicholas Kolokotronis, Christos Tryfonopoulos, Vasiliki-Georgia Bilali, and Dimitris Kavallieros. The quest for the appropriate cyber-threat intelligence sharing platform. In *DATA*, pages 369–376, 2019.

[19] Hannah Chen and Arvis Su. sec2vec : An embedding method for cyber threat intelligence. `https://github.com/0xyd/sec2vec`, 2018. Accessed : 2025-06-08.

[20] MITRE Corporation. Common vulnerabilities and exposures (cve), 2024.

[21] Roland Croft, M Ali Babar, and Li Li. An investigation into inconsistency of software vulnerability severity across data sources. In *2022 IEEE International*

Conference on Software Analysis, Evolution and Reengineering (SANER), pages 338–348. IEEE, 2022.

[22] CrowdStrike, Inc. CrowdStrike, Inc.

[23] Zhiyong Cui, Ruimin Ke, Ziyuan Pu, and Yinhai Wang. Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction. *arXiv preprint arXiv :1801.02143*, 2018.

[24] Cyber Research. APTMalware : APT Malware Samples for Cybersecurity Research. `https://github.com/cyber-research/APTMalware`, 2023. Accessed : 2024-10-29.

[25] Cyble. What is operational threat intelligence ?, 2025. Accessed : 2025-01-17.

[26] Ricardo M Czekster, Roberto Metere, and Charles Morisset. cyberactive : a stix-based tool for cyber threat intelligence in complex models. *arXiv preprint arXiv :2204.03676*, 2022.

[27] Isuf Deliu. Extracting cyber threat intelligence from hacker forums. Master's thesis, NTNU, 2017.

[28] Isuf Deliu, Carl Leichter, and Katrin Franke. Collecting cyber threat intelligence from hacker forums via a two-stage, hybrid process using support vector machines and latent dirichlet allocation. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 5008–5013. IEEE, 2018.

[29] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert : Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv :1810.04805*, 2018.

[30] Sounak Dey, Anjan Dutta, J Ignacio Toledo, Suman K Ghosh, Josep Lladós, and Umapada Pal. Signet : Convolutional siamese network for writer independent offline signature verification. *arXiv preprint arXiv :1707.02131*, 2017.

[31] Nuno Dionísio, Fernando Alves, Pedro M Ferreira, and Alysson Bessani. Towards end-to-end cyberthreat detection from twitter using multi-task learning. In *2020 international joint conference on neural networks (IJCNN)*, pages 1–8. IEEE, 2020.

[32] Amosse Edouard. *Event detection and analysis on short text messages*. PhD thesis, Université Côte D'Azur, 2017.

[33] ESET. Services de threat intelligence, 2023. Consulté le 2 novembre 2024.

[34] Caroline Etienne. *Apprentissage profond appliqué à la reconnaissance des émotions dans la voix*. PhD thesis, Université Paris Saclay (COmUE), 2019.

[35] FireEye, Inc. FireEye, Inc.

[36] Fr0gger. Cyber threat intelligence analysis, 2023. Accessed : 2025-01-17.

[37] Chen Gao, Xuan Zhang, and Hui Liu. Data and knowledge-driven named entity recognition for cyber security. *Cybersecurity*, 4(1) :9, 2021.

[38] Yumna Ghazi, Zahid Anwar, Rafia Mumtaz, Shahzad Saleem, and Ali Tahir. A supervised machine learning based approach for automatically extracting high-level threat intelligence from unstructured sources. In *2018 International Conference on Frontiers of Information Technology (FIT)*, pages 129–134. IEEE, 2018.

[39] Vibhuti Gupta and Rattikorn Hewett. Real-time tweet analytics using hybrid hashtags on twitter big data streams. *Information*, 11(7) :341, 2020.

[40] Hamed Haddadpajouh, Amin Azmoodeh, Ali Dehghantanha, and Reza M Parizi. Mvfcc : A multi-view fuzzy consensus clustering model for malware threat attribution. *IEEE Access*, 8 :139188–139198, 2020.

[41] Weijie Han, Jingfeng Xue, Yong Wang, Fuquan Zhang, and Xianwei Gao. Aptmalinsight : Identify and cognize apt malware based on system call information and ontology knowledge framework. *Information Sciences*, 546 :633–664, 2021.

[42] Ehtsham Irshad and Abdul Basit Siddiqui. Cyber threat attribution using unstructured reports in cyber threat intelligence. *Egyptian Informatics Journal*, 24(1) :43–59, 2023.

[43] Beakcheol Jang, Myeonghwi Kim, Gaspard Harerimana, Sang-ug Kang, and Jong Wook Kim. Bi-lstm model to increase accuracy in text classification : Combining word2vec cnn and attention mechanism. *Applied Sciences*, 10(17) :5841, 2020.

[44] Yifei Jian, Hongbo Kuang, Chenglong Ren, Zicheng Ma, and Haizhou Wang. A novel framework for image-based malware detection with a deep neural network. *Computers Security*, 109 :102400, 2021.

[45] Byungho Jung, Taeguen Kim, and Eul Gyu Im. Malware classification using byte sequence information. In *Proceedings of the 2018 Conference on Research in Adaptive and Convergent Systems*, pages 143–148, 2018.

[46] HAKAN Kekül, Burhan ERGEN, and Halil ARSLAN. A multiclass approach to estimating software vulnerability severity rating with statistical and word embedding methods. *Int J Comput Netw Inf Secur*, 12(4) :27, 2022.

[47] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert : A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv :1909.11942*, 2019.

[48] Hugo Larochelle, Yoshua Bengio, Jérôme Louradour, and Pascal Lamblin. Exploring strategies for training deep neural networks. *Journal of machine learning research*, 10(1), 2009.

[49] Quentin Le Sceller, ElMouatez Billah Karbab, Mourad Debbabi, and Farkhund Iqbal. Sonar : Automatic detection of cyber security events over the twitter stream. In *Proceedings of the 12th International Conference on Availability, Reliability and Security*, pages 1–11, 2017.

[50] Inside Machine Learning. Fonction d'activation : comment ça marche ? une explication simple, 2025. Accessed : 2025-01-17.

[51] Kuo-Chan Lee, Chih-Hung Hsieh, Li-Jia Wei, Ching-Hao Mao, Jyun-Han Dai, and Yu-Ting Kuang. Sec-buzzer : cyber security emerging topic mining with open threat intelligence retrieval and timeline event annotation. *Soft Computing*, 21 :2883–2896, 2017.

[52] Martti Lehto. Apt cyber-attack modelling : Building a general model. In *International Conference on Cyber Warfare and Security*, pages 121–129. Academic Conferences International Limited, 2022.

[53] Xiaojing Liao, Kan Yuan, XiaoFeng Wang, Zhou Li, Luyi Xing, and Raheem Beyah. Acing the ioc game : Toward automatic discovery and analysis of open-source cyber threat intelligence. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 755–766, 2016.

[54] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta : A robustly optimized bert pretraining approach. *arXiv preprint arXiv :1907.11692*, 2019.

[55] Pingchuan Ma, Bo Jiang, Zhigang Lu, Ning Li, and Zhengwei Jiang. Cybersecurity named entity recognition using bidirectional long short-term memory with conditional random fields. *Tsinghua Science and Technology*, 26(3) :259–265, 2020.

[56] Claudia Maguito Lontchi. Contribution à la prédiction des pertes de puissance sur un réseau électrique par un modèle à base de réseau de neurones. *Culminating Projects in Information Assurance*, 2021.

[57] Iaroslav Melekhov, Juho Kannala, and Esa Rahtu. Siamese network features for image matching. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 378–383, 2016.

[58] Dirk Meyer. *Situationsabhängige Bekleidungsmodellierung mit Hilfe von Machine Learning für die Erstellung von Avataren.* PhD thesis, Universitätsbibliothek Johann Christian Senckenberg, 2022.

[59] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, arxiv13013781 cs.(2013), 2020.

[60] Sudip Mittal, Prajit Kumar Das, Varish Mulwad, Anupam Joshi, and Tim Finin. Cybertwitter : Using twitter to generate alerts for cybersecurity threats and vulnerabilities. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 860–867. IEEE, 2016.

[61] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv :1312.5602*, 2013.

[62] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540) :529–533, 2015.

[63] Amirreza Niakanlahiji, Jinpeng Wei, and Bei-Tseng Chu. A natural language processing based trend analysis of advanced persistent threat techniques. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 2995–3000. IEEE, 2018.

[64] Umara Noor, Zahid Anwar, Tehmina Amjad, and Kim-Kwang Raymond Choo. A machine learning-based fintech cyber threat attribution framework using high-level indicators of compromise. *Future Generation Computer Systems*, 96 :227–242, 2019.

[65] Eric Nunes, Ahmad Diab, Andrew Gunn, Ericsson Marin, Vineet Mishra, Vivin Paliath, John Robertson, Jana Shakarian, Amanda Thart, and Paulo Shakarian. Darknet and deepnet mining for proactive cybersecurity threat intelligence. In *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*, pages 7–12. IEEE, 2016.

[66] John P O'Doherty, Jeffrey Cockburn, and Wolfgang M Pauli. Learning, reward, and decision making. *Annual review of psychology*, 68 :73–100, 2017.

[67] Keiron O'shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv :1511.08458*, 2015.

[68] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove : Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[69] Lior Perry, Bracha Shapira, and Rami Puzis. No-doubt : Attack attribution based on threat intelligence reports. In *2019 IEEE International Conference on Intelligence and Security Informatics (ISI)*, pages 80–85. IEEE, 2019.

[70] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies (NAACL-HLT), Volume 1 (Long Papers)*, 2018.

[71] Hugo Pompougnac. *Spécification et compilation de réseaux de neurones embarqués*. PhD thesis, Sorbonne Université, 2022.

[72] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *OpenAI preprint*, 2018.

[73] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 2020.

[74] Md Rayhanur Rahman and Laurie Williams. From threat reports to continuous threat intelligence : A comparison of attack technique extraction methods from textual artifacts. *arXiv preprint arXiv :2210.02601*, 2022.

[75] Naveen S, Rami Puzis, and Kumaresan Angappan. Deep learning for threat actor attribution from threat reports. In *2020 4th International Conference on Computer, Communication and Signal Processing (ICCCSP)*, pages 1–6, 2020.

[76] Jamell Samuels. One-hot encoding and two-hot encoding : An introduction, 01 2024.

[77] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert : smaller, faster, cheaper and lighter. *arXiv preprint arXiv :1910.01108*, 2019.

[78] Anna Sapienza, Sindhu Kiranmai Ernala, Alessandro Bessi, Kristina Lerman, and Emilio Ferrara. Discover : Mining online chatter for emerging cyber threats. In *Companion Proceedings of the The Web Conference 2018*, pages 983–990, 2018.

[79] Clemens Sauerwein, Daniel Fischer, Milena Rubsamen, Guido Rosenberger, Dirk Stelzer, and Ruth Breu. From threat data to actionable intelligence : an exploratory analysis of the intelligence cycle implementation in cyber threat intelligence sharing platforms. In *Proceedings of the 16th International Conference on Availability, Reliability and Security*, pages 1–9, 2021.

[80] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11) :2673–2681, 1997.

[81] Mohammad Ashraful Huq Shahi. Tactics, techniques and procedures (ttps) to augment cyber threat intelligence (cti) : A comprehensive study. *Advances in neural information processing systems*, 2018.

[82] Ruchi Sharma, Ritu Sibal, and Sangeeta Sabharwal. Software vulnerability prioritization using vulnerability description. *International Journal of System Assurance Engineering and Management*, 12 :58–64, 2021.

[83] Iryna Sopilko. Cyber threat intelligence as a new phenomenon : legal aspect. *J. Int'l Legal Commc'n*, 4 :8, 2022.

[84] Derya Soydaner. Attention mechanism in neural networks : where it comes and where it goes. *Neural Computing and Applications*, 34(16) :13371–13385, 2022.

[85] Smita Srivastava, Biswajit Paul, and Deepa Gupta. Study of word embeddings for enhanced cyber security named entity recognition. *Procedia Computer Science*, 218 :449–460, 2023.

[86] Blake E Strom, Andy Applebaum, Doug P Miller, Kathryn C Nickels, Adam G Pennington, and Cody B Thomas. Mitre att&ck : Design and philosophy. In *Technical report*. The MITRE Corporation, 2018.

[87] Nan Sun, Ming Ding, Jiaojiao Jiang, Weikang Xu, Xiaoxing Mo, Yonghang Tai, and Jun Zhang. Cyber threat intelligence mining for proactive cybersecurity defense : A survey and new perspectives. *IEEE Communications Surveys & Tutorials*, 2023.

[88] Wiem Tounsi and Helmi Rais. A survey on technical threat intelligence in the age of sophisticated cyber attacks. *Computers & security*, 72 :212–233, 2018.

[89] Roumen Trifonov, Ognyan Nakov, and Valeri Mladenov. Artificial intelligence in cyber threats intelligence. In *2018 international conference on intelligent and innovative computing applications (ICONIC)*, pages 1–4. IEEE, 2018.

[90] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[91] Qinqin Wang, Hanbing Yan, Chang Zhao, Rui Mei, Zhihui Han, and Yu Zhou. Apt attribution for malware based on time series shapelets. In *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 769–777, 2022.

[92] David A Wheeler and Gregory N Larsen. Techniques for cyber attack attribution. *Institute for Defense Analysis*, 2, 2003.

[93] Haoran Wu, Zhiyong Xu, Jianlin Zhang, Wei Yan, and Xiao Ma. Face recognition based on convolution siamese networks. In *2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pages 1–5, 2017.

[94] Mao Xiao, Chun Guo, Guowei Shen, Yunhe Cui, and Chaohui Jiang. Image-based malware classification using section distribution information. *Computers & Security*, 110 :102420, 2021.

[95] Feng Yi, Bo Jiang, Lu Wang, and Jianjun Wu. Cybersecurity named entity recognition using multi-modal ensemble learning. *IEEE Access*, 8 :63214–63224, 2020.

[96] A. Zenebe. Cyber threat intelligence discovery using machine learning from the dark web. *Communications of the IIMA*, 2023.

[97] Shieheng Zhou, Jingju Liu, Xiaofeng Zhong, and Wendian Zhao. Named entity recognition using bert with whole world masking in cybersecurity domain. In *2021 IEEE 6th International Conference on Big Data Analytics (ICBDA)*, pages 316–320. IEEE, 2021.