RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
UNIVERSITÉ SAAD DAHLAB DE BLIDA 1
Faculté des Sciences
Département d'Informatique

Présentée pour l'obtention du grade de **Master**

En : Informatique

Speciality :Sécurité des Systèmes d'Informations

Présenté par :

ABDICHE Abdellah

THÈME

Vers des systèmes intelligents de détection d'attaques d'ingénierie sociale par chat

Soutenu publiquement le **07/07/2025**, devant un jury composé de :

Dr.	REZOUG Nachida	USDB	Présidente
Dr.	BOUMAHDI Fatima	USDB	Directrice de thèse
Prof.	AZZOUZ Imane	USDB	Examinatrice

Remerciements

Avec une profonde reconnaissance, nous rendons grâce à Dieu Tout-Puissant pour sa guidance et la force qu'Il m'a accordées tout au long de ce mémoire de Master. Nous lui suis sincèrement reconnaissant pour les bienfaits qui m'ont soutenu.

Nous exprimons nos gratitudes les plus sincères à nos encadrants, nous tenons particulièrement à remercier ma promotrice, Madame BOUMAHDI Fatima, pour son soutien indéfectible, son mentorat et son expertise inestimable. Son acceptation de diriger nos travaux a été déterminante pour leur aboutissement, sa supervision et son partage de connaissances ont été essentiels tout au long de ce processus.

Nous tenons à exprimer notre profonde gratitude à Madame REZOUG Nachida, soutien constant et son engagement sans faille, la qualité exceptionnelle de son mentorat, e partage de son expertise précieuse qui a été déterminante pour notre parcours, nous lui sommes infiniment reconnaissants pour les opportunités qu'elle nous a offertes et pour son accompagnement bienveillant tout au long de ce projet.

Nous souhaitons également témoigner notre reconnaissance envers mon co-promoteur, Monsieur REMMIDE Mohamed Abdelkarim, pour son précieux soutien indéfectible, ses conseils avisés, sa disponibilité, son mentorat, son expertise inestimable et sa contribution à nos travaux.

Nous souhaitons également témoigner notre reconnaissance envers notre directeur, Monsieur CHEBLI MAAROUF, pour sa confiance et son précieux soutien constant, pour son rôle de mentor et de leader, pour sa guidance qui a été essentielle à notre développement académique et professionnel, nous lui sommes particulièrement reconnaissants pour les opportunités qu'il nous a offertes et pour son accompagnement personnalisé qui ont grandement contribué à la réussite de ce projet.

Enfin, nous adresse ma profonde gratitude à ma mère (Que Dieu ait pitié de mon père, le Moudjahid, et de mon petit frère martyr), et mes fillettes, mon épouse, à mon unique frère et à toute ma famille pour leur amour inconditionnel, leur soutien et leurs encouragements constants, leur foi en nous et leur accompagnement permanent ont été une source inépuisable de force durant tout ce parcours.

À tous ceux qui ont participé ou soutenu de près ou de loin par leur contribution à nous travaux,

Nous remercions également les membres de notre jury pour avoir accepté la responsabilité d'évaluer et d'examiner notre travail.

ملخص

تشكل هجمات الهندسة الاجتماعية عبر منصات الدردشة CSE تهديداً إلكترونياًا متزايداً، حيث تستغل بشكل خفي العوامل النفسية البشرية بدلاً من نقاط الضعف التقنية، وتقترح هذه الأطروحة اسلوبا للكشف الاستباقي عن هذه الهجمات من خلال الجمع بين التحليل الدلالي العميق والتعلم الآلي، ويدمج هيكلنا الهجين مستويين: وحدة BERT دقيقة لالتقاط الفروق الدقيقة السياقية للمحادثات الضارة، خوارزمية التعلم الآلي المحسنة (XGBoost) للتصنيف القوي.

تتُظهر التجارب التي أجريت على مجموعة حقيقية أداءً كبيرًا بدرجة F1_score تبلغ ٥٨٪ (استدعاء ٨٨٪)، متجاوزة النماذج أحادية الواسطة، ويساهم هذا البحث في الأمن السيبراني الاستباقي من خلال تقديم حل قابل للتكيف مع التطورات في تكتيكات الهندسة الاجتماعية، لا سيما في البيئات المهنية حيث يجب تقليل النتائج الإيجابية الكاذبة (دقة ٢٨٪).

وتشير النتائج إلى تطبيقات واعدة لحماية رسائل الدردشة وتفتح آفاقاًا جديدة للتحليل السلوكي للتهديدات السيبرانية.

الكلمات المفتاحية: هجوم CSE، الهندسة الاجتماعية، الأمن السيبراني المعرفي، BERT، التعلم الآلي الهجين، الكشف الاستباقي، التحليل الدلالي، الدردشة الآمنة.

Abstract

Social engineering attacks via chat platforms CSE (CSE – Chat-based Social Engineering attacks) are a growing cyber threat, subtly exploiting human psychological factors rather than technical vulnerabilities, this thesis proposes an approach for proactive detection of these attacks by combining deep semantic analysis and machine learning, our hybrid architecture integrates tow levels: a fine-tuned BERT module to capture the contextual nuances of malicious conversations, a set of optimized machine learning algorithms (XGBoost) for robust classification.

Experiments on a real corpus demonstrate significant performance with an F1_score of 85% (88% recall), surpassing the unimodal models, this research contributes to proactive cybersecurity by offering a solution adaptable to evolutions in social engineering tactics, particularly in professional environments where false positives must be minimized (accuracy of 82%).

The results suggest promising applications for the protection of chat messaging and open new avenues for the behavioral analysis of cyber threats.

Keywords: Attack CSE, Social engineering, Cognitive cybersecurity, BERT, Hybrid machine learning, Proactive detection, Semantic analysis, Secure chat.

Résumé

Les attaques par ingénierie sociale via les plateformes de chat (CSE)(CSE – Chat-based Social Engineering attacks), constituent une menace cyber croissante, exploitant subtilement les facteurs psychologiques humaines plutôt que des vulnérabilités techniques, cette thèse propose une approche pour la détection proactive de ces attaques en combinant l'analyse sémantique profonde et l'apprentissage automatique, notre architecture hybride intègre deux niveaux : un module BERT fine-tuné pour capturer les nuances contextuelles des conversations malveillantes, un algorithmes de machine learning optimisés (XGBoost) pour la classification robuste.

Les expérimentations sur un corpus réel démontrent des performances significatives avec un F1_score de 85% (88% de rappel), surpassant les modèles unimodaux, cette recherche contribue à la cybersécurité proactive en offrant une solution adaptable aux évolutions des tactiques d'ingénierie sociale, particulièrement dans les environnements professionnels où les faux positifs doivent être minimisés (précision de 82%).

Les résultats suggèrent des applications prometteuses pour la protection des messageries chats et ouvrent de nouvelles pistes pour l'analyse comportementale des cybermenaces.

Mots Clée : Attaque (CSE), Ingénierie sociale, Cybersécurité cognitive, BERT, Apprentissage automatique hybride, Détection proactive, Analyse sémantique, Chat sécurisé.

Table des matières

Table des figures

Liste des tableaux

Nomenclature

Acronymes / Abréviations

BERT	Bidirectional Encoder Representations from Transformers
BT	Boosting Tree
CSE	Chat-based Social Engineering attacks
DL	Deep Learning
DT	Decision Trees
IA	Intelligence Artificielle
KG	Knowledge Graphs
ML	Machine Learning
NB	Naïve Bayes
NLP	Natural Language Processing
RF	Random Forest
SLR	Systematic Literature Review
SVM	Support Vector Machines

Introduction générale

L'essor des technologies numériques a profondément transformé les interactions sociales et professionnelles, mais a également ouvert la voie à de nouvelles formes de cybercriminalité, notamment les attaques d'ingénierie sociale dans les environnements conversationnels (Chat-based Social Engineering - CSE), ces attaques, qui exploitent la confiance et les biais psychologiques des utilisateurs, ont évolué en sophistication grâce aux avancées technologiques récentes, telles que l'intelligence artificielle IA (IA- Intelligence Artificielle), les chatbots malveillants et les deepfakes.

Problèmatique

Les attaques CSE représentent une menace majeure, tant pour les particuliers que pour les infrastructures critiques et les grandes organisations, notamment dans les secteurs de la santé, des entreprises et de l'éducation.

Les plateformes de messagerie instantanée et réseaux sociaux (Facebook, WhatsApp, Telegram, Discord... Ext) sont devenus des vecteurs privilégiés pour ces attaques, en raison de leur popularité et de leur accessibilité, selon les dernières statistiques, 62% des attaques d'ingénierie sociale passent désormais par des chats, avec des méthodes de plus en plus élaborées, comme le phishing conversationnel ou l'usurpation d'identité via IA (Check Point, 2024), par ailleurs, l'automatisation des attaques via des outils comme ChatGPT a entraîné une hausse de 300% des campagnes malveillantes en 2024 (Darktrace), rendant la détection plus complexe pour les utilisateurs et les systèmes de sécurité traditionnels.

Cette évolution pose des défis majeurs en cybersécurité, nécessitant des approches innovantes pour contrer ces menaces, les méthodes classiques de détection, basées sur des règles prédéfinies ou des algorithmes de Machine Learning simples, montrent leurs limites face à la rapidité et à l'adaptabilité des attaques modernes.

Dans ce contexte, une question cruciale se pose : face à l'influence croissante des avancées technologiques sur le développement des techniques d'ingénierie sociale dans les messageries instantanées, quelles sont les solutions émergentes pour y faire face ?

Objectifs

Pour combler cette lacune, nous proposons une approche hybride, combinant la force des modèles avancés de traitement du langage (comme BERT) et des techniques d'apprentissage automatique (comme XGBoost), émergent comme une solution prometteuse, et un système intelligent efficace capable d'identifier et de prévenir de manière proactive les attaques CSE avant qu'elles ne puissent causer des dommages substantiels.

Cette étude a le potentiel d'améliorer significativement la cybersécurité en fournissant aux particuliers et organisations un outil puissant pour la détection précoce et l'atténuation des attaques CSE.

Cette recherche relève le défi majeur que représente la détection des attaques d'ingénierie sociale CSE, nous proposons une approche hybride combinant la puissance des modèles de langage BERT avec un algorithme classiques de Machine Learning (XGBoost) pour une détection plus robuste, notre méthode exploite simultanément les capacités de traitement contextuel des transformers pour analyser les vecteurs d'attaque (messages chat) et la force discriminative des modèles ML (ML- Machine Learning) traditionnels pour identifier les schémas d'attaque, l'objectif est de développer un système capable de détecter précocement divers types d'attaques CSE (hameçonnage, pretexting) tout en anticipant les nouvelles variantes grâce à l'inférence comportementale.

Organisation de la thèse

Notre thèse s'organise en trois chapitres thématiques permettant une progression logique de la réflexion.

Chapitre 1 : État de l'art Pose les fondements théoriques en définissant précisément le concept d'attaques CSE, il présente une taxonomie détaillée de ces attaques, analyse leur cycle de vie complet, et établit une typologie rigoureuse des différents types d'attaques, finalement une revue approfondie de la littérature existante.

Chapitre 2 : Solution proposée Ce chapitre présente une analyse statistique détaillée des datasets utilisés, décrit l'architecture globale de notre système incluant les différentes étapes de traitement des données (nettoyage et gestion du déséquilibre des classes), en dernier étape la solution proposée, nous avons présenté nos trois approches principales : une méthode basée sur le machine learning classique (implémentant DT, , AdaBoost, XGBoost, SVM et TinyML), une approche exploitant le modèle BERT pour l'analyse contextuelle, ainsi que notre solution hybride intégrant les avantages des deux premières méthodes.

Chapitre 3 : Tests et Résultats Présente les résultats des différentes approches

proposées, les métriques d'évaluation utilisées, la configuration des hyperparamètres et une discussion approfondie sur les performances des modèles, les conclusions et les perspectives de recherches futures sont également abordées.

Chapitre 1

Etat de l art

1.1 Introduction

Les attaques d'ingénierie sociale basées sur le chat CSE (CSE – Chat-based Social Engineering attacks) représentent une menace de sécurité persistante et en constante mutation, Face à cette réalité, une vigilance accrue et une adaptation proactive des stratégies de défense sont indispensables.

Ce chapitre initie une exploration de ce domaine crucial, en posant les bases conceptuelles des attaques CSE et en offrant un aperçu des efforts scientifiques et technologiques déployés pour contrer ces attaques insidieuses.

1.2 Définition

Le terme « Ingénierie Sociale » désigne une tactique utilisée par les attaquants pour manipuler des individus afin qu'ils divulguent des informations sensibles ou qu'ils effectuent des actions susceptibles de compromettre la cybersécurité [Gehl and Lawson, 2022], il est également utilisée en sécurité de l'information pour désigner un type d'attaque dans lequel un attaquant manipule des individus afin de compromettre la confidentialité, l'intégrité et la disponibilité des données et des processus en exploitant les vulnérabilités humaines [Wang et al., 2020].

Les attaques d'ingénierie sociale peuvent prendre diverses formes, notamment :

— Le phishing (hameçonnage)
— Le prétexte
— L'appât
— Le quid pro quo

Ces attaques peuvent être menées par divers canaux ou surfaces d'attaque tels que :

— Le chat
— Le courrier électronique
— Le téléphone
— Les interactions en personne [Tsinganos, 2023]

1.3 Taxonomie

Plusieurs chercheurs ont proposé différentes taxonomies pour classifier les attaques d'ingénierie sociale, chaque taxonomie se concentre sur des aspects ou des dimensions spécifiques de ces attaques.

L'ingénierie sociale est traditionnellement divisée en deux catégories principales :

1. Humaine (basée sur l'interaction humaine)
2. Technologique (basée sur la technologie) [Zaoui et al., 2024]

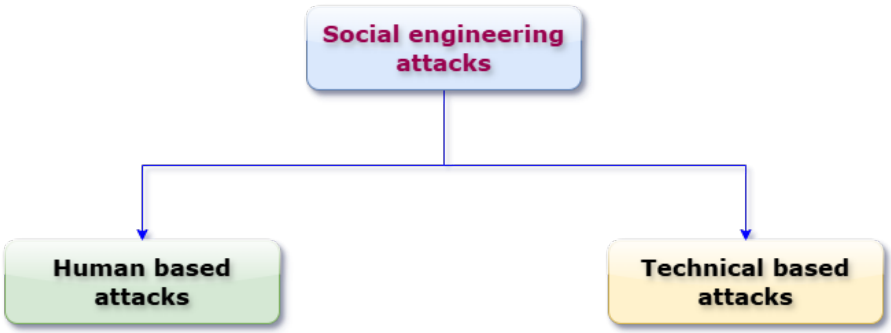

FIGURE 1.1 – Les catégories principales de l'ingénierie sociale [Zaoui et al., 2024].

1.4 Le cycle d'attaque d'ingénierie sociale

Parmi les nombreuses taxonomies disponibles, le cycle d'attaque d'ingénierie sociale de Kevin Mitnick [Mitnick and Simon, 2003], est le plus largement reconnu.

Comme illustré dans la Figure 1.2, ce modèle décrit les quatre phases qui se déroulent avant et pendant une attaque d'ingénierie sociale.

Premièrement, l'attaquant cible, généralement une personne vulnérable et facilement manipulable [Gururaj et al., 2024].

L'étape de reconnaissance permet de recueillir des renseignements sur la cible, ses potentielles vulnérabilités et tout vecteur d'attaque pertinent pour les phases ultérieures.

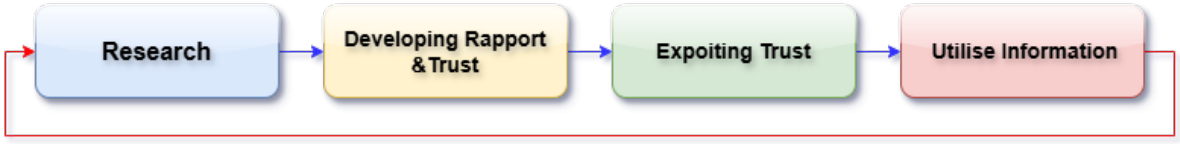

FIGURE 1.2 – Le cycle d'attaque d'ingénierie sociale de Kevin Mitnick [Zaoui et al., 2024]

Établir un lien de confiance constitue la deuxième phase de l'attaque, où l'attaquant cherche à gagner la confiance de la cible - une confiance qui sera exploitée lors de la troisième phase.

Exploiter la confiance. durant cette étape, l'attaquant extrait des informations de la cible, Le manipule, ou simplement lui donne des instructions pour obtenir les connaissances ou actions désirées.

Exploiter les informations est la quatrième et dernière phase du modèle, elle représente l'acte final de l'attaque, où l'attaquant met en pratique les informations et ressources acquises durant les trois premières étapes pour atteindre son objectif [Ghafir et al., 2018].

1.5 Les types d'attaques d'ingénierie sociale

L'ingénierie sociale regroupe diverses techniques visant à manipuler les individus pour qu'ils divulguent des informations confidentielles ou exécutent des actions compromettant la sécurité, chaque méthode exploite la psychologie et les comportements humains, ce qui les rend particulièrement efficaces dans les cyberattaques.

Phishing : Il s'agit de la forme la plus répandue, où les attaquants se font passer pour des entités légitimes afin de voler des informations sensibles.

C'est une manœuvre frauduleuse par courrier électronique, SMS ou communication téléphonique destinée à tromper le destinataire pour qu'il installe des programmes nuisibles (par l'intermédiaire d'un lien ou d'un fichier joint), divulgue des données sensibles, effectue un transfert monétaire à des cybercriminels ou entreprenne d'autres actes nuisibles à sa protection ou à celle de son organisation.

Les variantes incluent :

— Vishing : Le mot vient de la combinaison entre les termes « voice » et « phishing » qui se traduit par hameçonnage vocal en français, désignant une forme d'hameçonnage vocal.

Contrairement au phishing qui utilise les e-mails et SMS, le vishing exploite les appels téléphoniques.

— Smishing : C'est est la contraction de « SMS » et « phishing », Cette technique consiste pour les cybercriminels à usurper l'identité d'un tiers de confiance par SMS afin de tromper leurs victimes.
— Spear phishing : Consiste à envoyer des emails d'apparence légitime pour inciter le destinataire à partager des informations confidentielles avec l'attaquant, l'objectif principal est souvent le vol d'identifiants ou de données bancaires, certaines attaques visent l'infection des appareils par des logiciels malveillants.
— Whaling : Le whaling fait référence à une attaque de phishing spécifiquement orientée vers un cadre ou un représentant de haut rang.

    Le pirate a souvent tendance à se faire passer pour un collègue au sein de l'entité visée, ou pour un associé ou un collaborateur d'un autre organisme ayant un rang équivalent ou supérieur [Leonov et al., 2021].

Pretexting : les cybercriminels inventent de faux scénarios mettent en place des faux scénarios crédibles afin de gagner la confiance de leur victime et l'inciter à divulguer des informations sensibles ou à entreprendre des actions que l'agresseur pourra par la suite exploiter à son profit.

Pour y parvenir, les pirates informatiques adoptent souvent une identité fictive et se présentent comme quelqu'un en qui la cible a confiance [Kamruzzaman et al., 2023].

Baiting : Les cybercriminels exploitent la curiosité, l'envie (Exp :le désir des gens d'obtenir des choses gratuitement), voire l'altruisme pour piéger leurs victimes et les amener à utiliser une clé USB compromise ou à cliquer sur un lien malveillant, dans le but de voler des données, compromettre des systèmes ou extorquer de l'argent.

L'appât, souvent émotionnellement attrayant, est conçu pour faire agir la victime sans réfléchir, et c'est précisément ce qui rend ce type de cyberattaque redoutable [Kamruzzaman et al., 2023].

Quid Pro Quo : Cette expression latine signifie littéralement "ceci pour cela" et décrit un échange de quelque chose contre des biens ou des services.

Les cybercriminels veulent des données, et un Quid Pro Quo est un moyen d'obtenir ces données [Mahanta and Maringanti, 2023].

Tailgating : C'est l'une des fraudes par ingénierie sociale les plus anciennes.

Elle peut également se manifester sous forme numérique, souvent perçue comme une escroquerie matérielle.

Dans le monde réel, le tailgating est exemplifié par un individu malintentionné qui se glisse discrètement dans les locaux d'une entreprise sans être détecté.

Les imposteurs sont généralement très adroits pour rester discrets, ce qui leur donne l'occasion d'entrer dans un bâtiment sans permission ou de se présenter comme une personne autorisée à accéder [Kamruzzaman et al., 2023].

Bien que ces techniques soient répandues, il est essentiel de reconnaître que l'efficacité de l'ingénierie sociale repose fortement sur l'erreur humaine et le manque de sensibilisation aux menaces de cybersécurité.

Inversement, certains soutiennent que les avancées technologiques et les protocoles de sécurité robustes peuvent atténuer considérablement les risques associés à ces attaques, soulignant l'importance d'une approche équilibrée de la cybersécurité [Örberg Kätterer, 2023].

1.6 Travaux connexes

Plusieurs études de recherche ont été menées sur la reconnaissance des attaques d'ingénierie sociale basées sur le chat CSE.

Par conséquent, des informations précieuses peuvent être trouvées dans les études suivantes :

L'ouvrage séminal de [Mitnick and Simon, 2003] offre un cadre d'analyse complet des cyberattaques par ingénierie sociale, mettant en lumière, les biais psychologiques exploités chez les victimes, l'évolution des tactiques d'attaque, l'efficacité différentielle des parades technologiques (dont les systèmes machine learning) et organisationnelles.

[Tsinganos et al., 2018] dresse un état complet des systèmes existants et examine en profondeur les sous-systèmes d'architecture de détection, notamment pour les aspects d'influence, tromperie, personnalité, actes de parole et historique comportemental.

[Lansley et al., 2019] propose par cette étude une approche combinant NLP et réseaux de neurones pour identifier les attaques d'ingénierie sociale, après analyse et correction grammaticale du texte, un ANN classe les échanges comme malveillants, testée sur un dataset réel (147 attaques) et un dataset semi-synthétique (600 tweets de support client), la méthode a atteint une précision élevée 92.5%.

[Salahdine and Kaabouch, 2019] Synthétise l'état de l'art sur l'ingénierie sociale : classifications des attaques, méthodes de détection (automatisées et humaines) et protocoles de prévention.

Cette étude de [Lansley et al., 2020], évalue les performances d'un Multi-Layer Perceptron et de classifieurs ensemblistes dans la détection d'attaques d'ingénierie sociale, en proposant un nouveau modèle SEADER++, le vote pondéré (soft voting), avec une AUC de 72,2 %, s'est avéré légèrement plus efficace que le vote majoritaire (hard voting)

[Wang et al., 2020] Ce document tente de combler ces lacunes conceptuelles en proposant une définition plus compatible et plus précise de l'ingénierie sociale dans le domaine de la cybersécurité SEiCS.

[Alsufyani and Alzahrani, 2021] applique Word2Vec, CBOW et divers modèles ML (ML- Machine Learning) à 20k URLs pour détecter le phishing textuel, le CNN obtient 99,2% de précision, surpassant KNN, Naïve Bayes et AdaBoost.

Révolutionnant la représentation sémantique, [Basiri et al., 2021] a conçu l'ABCDM, un modèle hybride CNN-RNN à attention pour l'analyse de sentiments, alliant LSTM/GRU bidirectionnels et embeddings GloVe pré-entraînés, il obtient 99,04% de précision.

Une taxonomie des approches d'apprentissage profond DL (DL- Deep Learning) pour la détection de phishing via une revue systématique de la littérature SLR (SLR- Systematic Literature Review) est établie par [Do et al., 2022], l'analyse de 81 publications scientifiques permet de catégoriser et d'évaluer les méthodes existantes, tout en dressant un état de l'art des techniques de DL appliquées à cette problématique.

[Tsinganos et al., 2022] propose un CNN enrichi de techniques NLP (NLP- Natural Language Processing) pour détecter la persuasion dans les conversations d'ingénierie sociale, offrant une défense proactive contre les attaques, il obtient 71.6% de accuracy.

L'étude [Wang et al., 2022] prouve qu'un simple arbre de décision (4 features) détecte efficacement le phishing/email (précision=89,6%, Rappel=85,5%, F1=87,6%), complétant les graphes de connaissances KG (KG- Knowledge Graphs).

[Abobor and Josyula, 2023] présente SOCIALBERT, un modèle BERT fine-tuné pour détecter l'ingénierie sociale dans les SMS, entraîné sur 9 754 messages annotés, il atteint 97,55% de précision.

Afin d'informer les individus sur les techniques actuelles d'attaques d'ingénierie sociale, [Alemayehu, 2023] dans son article propose un modèle de détection d'attaques d'ingénierie sociale personnalisées TSEADM, basé sur un modèle antérieur SEADM v2.

Pour simplifier la détection, il recourt à un arbre de décision qui s'appuie sur des règles prédéfinies.

Trois modèles d'apprentissage profond (CNN, LSTM et un hybride LSTM-CNN) sont proposés par [Alshingiti et al., 2023], pour la détection de sites de phishing.

Les résultats expérimentaux démontrent que le CNN est le plus performant, atteignant une précision de 99,28 %, supérieure aux autres approches.

[Benavides-Astudillo et al., 2023] Aborde une nouvelle approche de détection de phishing par analyse textuelle NLP/DL, utilisant des réseaux LSTM/BiLSTM/GRU/BiGRU avec embedding GloVe, le modèle BiGRU obtient 97,39% de précision, surpassant l'analyse traditionnelle par URLs.

Dans cette étude [Falade, 2023] explore l'utilisation de l'IA générative dans les attaques d'ingénierie sociale, elle se concentre sur l'extraction d'informations à partir de blogs pour comprendre comment l'IA facilite de nouvelles tactiques d'ingénierie sociale.

L'étude [Jamal and Wimmer, 2023] présente IPSDM, un BERT fine-tuné détectant

spam/phishing avec 97,5% de précision, validé sur un corpus déséquilibré (747 spams/189 phishing/4825 ham).

Dans leur publication, [Kamruzzaman et al., 2023] analysent systématiquement les méthodes de prévention contre les attaques d'ingénierie sociale, mettant en lumière les approches les plus performantes pour contrer ces menaces.

S'appuyant sur ChatGPT [Koide et al., 2023] propose une approche innovante de détection de phishing.

Un web crawler analyse les sites web, distinguant les pages frauduleuses des sites légitimes à l'aide d'un dataset équilibré (1 000 phishing / 1 000 non-phishing), les testes avec GPT-4 ont atteint 98,3 % en précision et 98,4 % en rappel, confirmant son efficacité.

[Mahanta and Maringanti, 2023] analyse les attaques d'ingénierie sociale (phishing, pretexting...), leurs mécanismes psychologiques et les moyens de s'en prémunir.

SG-CSE-BERT, un modèle IA détectant les CSE dans les dialogues en ligne via une ontologie spécialisée et une annotation du SG-CSE Corpus, est proposé par [Tsinganos et al., 2023], le modèle, validé expérimentalement, identifie en temps réel les intentions malveillantes lors d'échanges humains directs (excluant les attaques automatisées), il obtien 89.6% F1-score.

Les travaux de [Tsinganos, 2023] et [Tsinganos et al., 2024] proposent CSE-ARS, un système intelligent de détection d'attaques d'ingénierie sociale par chat CSE, la première version 2023 utilise quatre modèles spécialisés en deep learning et NLP pour analyser différents aspects des attaques : fuites d'informations CRINL-R, traits de personnalité PERST-R, techniques de persuasion PERSU-R et persistance de l'attaquant PERSI-R, ce dernier étant le plus performant de 76.8% accuracy, la version améliorée (2024) introduit un cinquième modèle DIACT-R pour reconnaître les actes de dialogue et adopte une fusion tardive, atteignant une précision globale de 79,96%.

[Wang and Ghaleb, 2023] ont développé une architecture novatrice pour la détection d'intrusions, basé sur un CNN avec mécanisme d'attention, accompagné de méthodes de génération d'images utilisant le dataset CSE-CIC-IDS2018, les résultats expérimentaux démontrent que le modèle proposé atteint une précision de détection élevée de 96%.

[Ai et al., 2024] propose ConvoSentinel, un système modulaire combinant LLM et un nouveau dataset SEConvo pour détecter les attaques CSE (message + conversation), il obtient 75% de F1-score.

Le modèle Llama2-7B SE obtient un F1=75% sur des scénarios académiques/recrutement.

[Egigogo et al., 2024] compare CNN, LSTM et l'hybride ODAE-WPDC pour la détection de phishing, ce dernier atteignant 99,28% de précision.

[First et al., 2024] présente une architecture hybride CNN-LSTM novatrice, enrichie d'un mécanisme d'attention dédié à l'identification de commentaires haineux en tamoul,

l'approche proposée atteint 75,98% de précision, surpassant les méthodes conventionnelles.

L'étude de [Zaoui et al., 2024] présente une taxonomie exhaustive des attaques d'ingénierie sociale, structurée en trois niveaux (environnement, approches et moyens).

De plus, elle propose une classification des contre-mesures d'ingénierie sociale, incluant les aspects techniques et non techniques.

La recherche de [Rathod et al., 2025] a développé une nouvelle taxonomie anti-ingénierie sociale et un système IA-blockchain analysant 3,9 millions d'URLs via différents algorithmes (NB , DT, SVM, BT). Les URLs sûres sont stockées en blockchain, avec une précision de 76,87% pour Naïve Bayes.

Auteurs	**Approches**	**Techniques**	**Datasets**	**Résultats**
[Lansley et al., 2019]	NLP + Réseaux de neurones	Deep Learning	147 attaques + 600 tweets	précision=92.5%
[Lansley et al., 2020]	SEADER++	Machine Learning	147 entrées + 600 tweets	AUC=72.2%
[Alsufyani and Alzahrani, 2021]	Détection de phishing textuel	Deep Learning (Word2Vec)	20 000 URLs	précision=99.2%
[Basiri et al., 2021]	Modèle hybride CNN-RNN	Deep Learning	T4SA	précision=99.04%
[Tsinganos et al., 2022]	CNN + Word Embeddings	Machine Learning	CSE-RSE Corpus (56 dialogues)	accuracy=71.6%
[Wang et al., 2022]	Arbre de décision	Machine Learning	14 types d'attaques	F1=87.6%
[Alshingiti et al., 2023]	CNN/LSTM	Deep Learning	ISCX-URL2016	précision=99.28%
[Benavides-Astudillo et al., 2023]	BiGRU + GloVe	Deep Learning	Phishload (10 373 URLs)	précision=97.39%
[Jamal and Wimmer, 2023]	BERT fine-tuné	Deep Learning	747 spams/189 phishing	précision=97.5%
[Koide et al., 2023]	ChatGPT (GPT-4)	Deep Learning	1000 phishing/non-phishing	précision=98.3%
[Tsinganos, 2023]	CSE-ARS	Deep Learning	56 dialogues	accuracy=76.8%
[Tsinganos et al., 2023]	SG-CSE-BERT	Deep Learning	SG-CSE Corpus	F1=89.6%
[Wang and Ghaleb, 2023]	CNN à attention	Machine Learning	CSE-CIC-IDS2018	précision=96%
[Abobor and Josyula, 2023]	SOCIALBERT	Deep Learning BERT	9 754 SMS	précision=97.55%
[Ai et al., 2024]	ConvoSentinel (Llama2-7B)	Deep Learning	SEConvo	F1=75%
[Egigogo et al., 2024]	ODAE-WPDC	Deep Learning	/	précision=99.28%
[First et al., 2024]	CNN-LSTM à attention	Deep Learning	Tamil ACL 2022	précision=75.98%
[Tsinganos et al., 2024]	CSE-ARS (fusion tardive)	Deep Learning	180 dialogues	précision=79.96%
[Rathod et al., 2025]	IA + Blockchain	Machine Learning	3.9M URLs	précision=76.87%

TABLEAU 1.1 – Table des travaux connexes.

1.7 Synthèse

Comme le montre le tableau 1.1, les études menées sur la détection des attaques en ingénierie sociale ces dernières années montrent une forte prépondérance des techniques de deep learning (74% des études(14 parmi 19 etudes)), avec des modèles avancés (CNN-RNN, BERT, GPT-4) atteignant des performances élevées (moyenne de 97,3% en précision).

Les techniques hybrides et à attention se distinguent, comme le montre [Egigogo et al., 2024] avec 99,28% ou [Koide et al., 2023] utilisant GPT-4 (98,3%).

Cependant, les LLMs (ex. Llama2-7B) peinent sur petits datasets (F1=75%), tandis que les méthodes classiques (arbres de décision) gardent l'avantage de l'interprétabilité (F1=87,6%).

L'évolution va des embeddings (Word2Vec) vers les transformers et LLMs, soulignant le besoin d'optimiser ces derniers pour des données limitées et d'améliorer la fusion multimodale, un équilibre entre performance et coût computationnel reste à trouver.

1.8 Conclusion

Ce chapitre a présenté les bases théoriques des attaques CSE, le cycle d'attaque et ses types, enfin nous avons analysé les approches de détection proposées et les solutions existantes trouvées dans la littérature.

Dans le chapitre suivant, nous détaillerons notre solution proposée.

Chapitre 2

Solution proposée

2.1 Introduction

Il n'y a pas si longtemps, la détection des Attaques CSE, présentait une réelle difficulté, Les récentes avancées en traitement automatique du langage NLP, en apprentissage automatique et en apprentissage profond ont conduit au développement des techniques plus sophistiquées pour l'identification de ces attaques.

Ces méthodes, employées dans des travaux antérieurs, ont démontré un grand potentiel pour améliorer les résultats de cette tâche.

Dans ce chapitre, nous présenterons en détail l'architecture que nous proposons pour aborder le problème de détection des attaques CSE dans les discussions en ligne.

Nous décrirons également les datasets utilisé ainsi que les différentes étapes nécessaires à la préparation de nos données.

2.2 Architecture générale

Pour répondre au défi de détection des attaques d'ingénierie sociale par chat CSE, notre solution s'appuie sur une architecture modulaire organisée en trois étapes clés Figure 2.1.

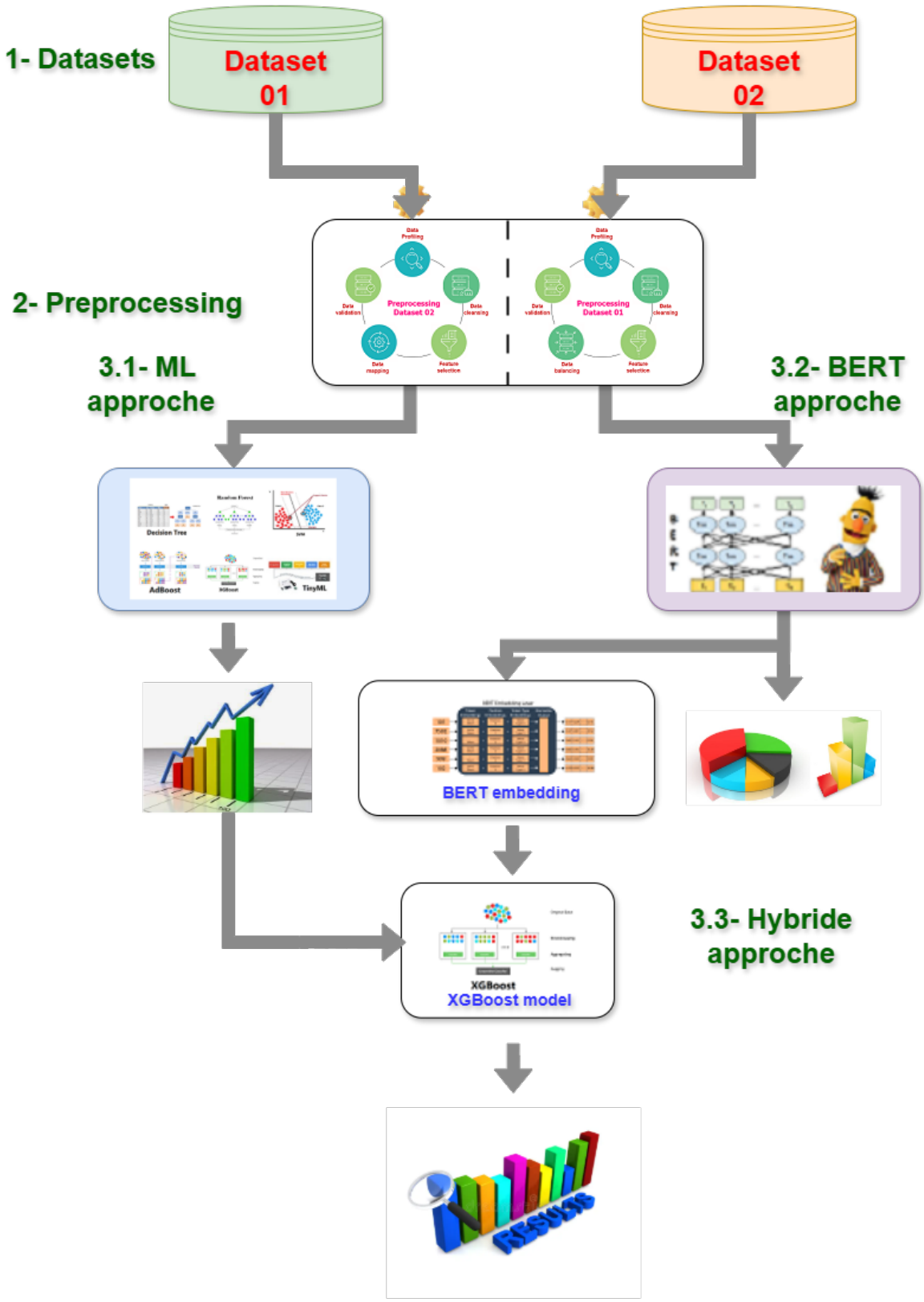

FIGURE 2.1 – Architecture générale du système proposé

Prétraitement des données

— Dans une première phase expérimentale, nous avons évalué six algorithmes de machine learning reconnus pour leur puissance d'analyse (DT, RF, AdaBoost, XGBoost, SVM et TinyML).

Notre dataset initial (compound_dataset.csv) présentant un déséquilibre important entre les classes, nous avons appliqué la technique SMOTE (SMOTE-Synthetic Minority Over-sampling Technique) pour rééquilibrer la distribution des échantillons.

Finalement, une analyse approfondie des résultats avec et sans SMOTE a été réalisée, permettant d'identifier objectivement les classificateurs présentant les meilleures métriques de performance en vue de les intégrer dans notre architecture globale.

— La deuxième phase a consacré BERTBASE comme modèle de référence, sélectionné pour son excellence en traitement du langage naturel NLP et sa parfaite adéquation avec nos données textuelles, son architecture transformer, capable de saisir les subtilités linguistiques des dialogues, a été fine-tune sur notre corpus spécifique (colonne 'dialog' du deuxième dataset (dialogues_dataset_formatted.csv), permettant une spécialisation optimale pour notre tâche de détection des attaques CSE.

— Notre architecture finale intègre de manière synergique le meilleur classificateur ML (identifiés lors de la première phase) avec BERTBASE fine-tuné, créant ainsi un système hybride optimisé pour la détection des attaques CSE.

Cette combinaison intelligente capitalise sur les forces complémentaires des deux approches : la puissance d'analyse des modèles traditionnels sur les données structurées, combinée à la capacité avancée de compréhension contextuelle des transformers.

L'intégration de ces technologies permet d'obtenir un système robuste, précis et adapté aux spécificités de notre problématique.

On ce qui suit, nous détaillons méthodiquement les étapes de prétraitement des données et la modélisation adoptée pour développer notre système de détection d'attaques CSE.

2.3 Détails des datasets CSE

Nous avons utilisé deux datasets annotés [Lansley et al., 2020], de volumes différents pour la détection d'attaques CSE, Leur différence principale réside dans

le nombre d'enregistrements et le niveau de prétraitement : le premier dataset (compound_dataset.csv), semi-synthétique, intégralement numérique et prétraité, présente l'avantage d'être immédiatement exploitable mais souffre d'un déséquilibre marqué entre les classes, Le second corpus, constitué de données réelles (dialogues_dataset_formatted.csv), qui a un dééquilibre acceptable [Lamari et al., 2021] et concédée comme 'équilibré, comporte des données hétérogènes incluant des éléments textuels non structurés, nécessitant par conséquent des opérations supplémentaires de nettoyage et de normalisation.

2.3.1 Structure du premier dataset

Ce dataset nommé "compound_dataset.csv", constitue un corpus semi-synthétique composé de 748 échanges conversationnels annotés pour la détection d'attaques d'ingénierie sociale par chat CSE, il comprend 650 conversations légitimes (négatives, 89.6%) et 78 attaques CSE (positives, 10.4%).

Les caractéristiques détaillées de ce dataset sont présentées dans le tableau 2.1 ci-dessous.

Champ	Description
intent	Représente le but de l'interlocuteur
spelling	Numéro de téléphone de l'appelant (émetteur)
link	Numéro de téléphone du récepteur
is_attack	Étiquette binaire indiquant si l'échange est une attaque (1) ou légitime (0)

Tableau 2.1 – Métadonnées du premier dataset (compound_dataset.csv)

Confrontés à un déséquilibre marqué (78 attaques vs 650 échanges normaux), nous avons implémenté la technique de rééquilibrage SMOTE (Synthetic Minority Over-sampling Technique) qu'est une méthode utilisée pour traiter les datasets déséquilibrés dans les problèmes de classification, il génère des échantillons synthétiques pour la classe minoritaire en se basant sur le nombre d'échantillons de cette classe, le taux de suréchantillonnage appliqué et le nombre de plus proches voisins considérés [Lamari et al., 2021].

2.3.2 Structure du deuxième dataset

Nommé "dialogues_dataset_formatted.csv", sa particularité distinctive réside dans l'inclusion du contenu textuel brut des messages réales (la colonne "dialog") , offrant un accès direct à la sémantique conversationnelle.

Constitue un corpus réel annoté composé de 147 échanges conversationnels pour la détection d'attaques d'ingénierie sociale par chat CSE, il comprend 50 conversations légitimes (négatives, 47.6%) et 77 attaques CSE (positives, 52.4%), ce qui fait un taux de déséquilibre de 4.8%, qui est un déséquilibre acceptable selon [Koudri, 2021].

Les caractéristiques détaillées de ce dataset sont présentées dans le tableau 2.2 ci-dessous.

Champ	**Description**
voice_id	Identifiant unique de la conversation
num_caller	Numéro de téléphone de l'appelant (émetteur)
num_receiver	Numéro de téléphone du récepteur
date	Date de la communication (format JJ/MM/AAAA)
time	Heure de la communication (format HH :MM)
is_attack	Indicateur binaire d'attaque (1 : attaque, 0 : légitime)
dialog	Contenu textuel de l'échange entre l'émetteur et le récepteur

Tableau 2.2 – Métadonnées du deuxième dataset (dialogues_dataset_formatted.csv)

2.4 Prétraitement des données

Comme les performances d'un modèle de ML dépendent fortement de la qualité des données d'entraînement, la détection et la correction des erreurs dans les datasets constituent un enjeu majeur [Côté et al., 2024], ainsi, une phase de nettoyage du dataset s'avère très essentielle, en suivant des méthodes adaptées pour garantir des résultats fiables.

le nettoyage (la suppression des données superflues telles que les identifiants d'utilisateur), la vérification, l'analyse et leur conversion dans des formats lisibles.

Le traitement des données peut être effectué selon deux méthodes : manuelle, automatique.

2.4.1 Le premier dataset

Voici les techniques de prétraitement appliquées à notre premier dataset.

2.4.1.1 Équilibrage des données

On remarque que notre premier datast contient 78 permis 748 enregistrements, soit 10,4% (78/748) correspondant à des attaques CSE, contre 650 enregistrements

(soit 89,6%, 650/748) représentant des communications saines, Cela révèle un déséquilibre significatif dans la répartition des données.

La figure 2.2 présente la distribution du nombre de safe et de vraies attaques avant le processus de nettoyage des données.

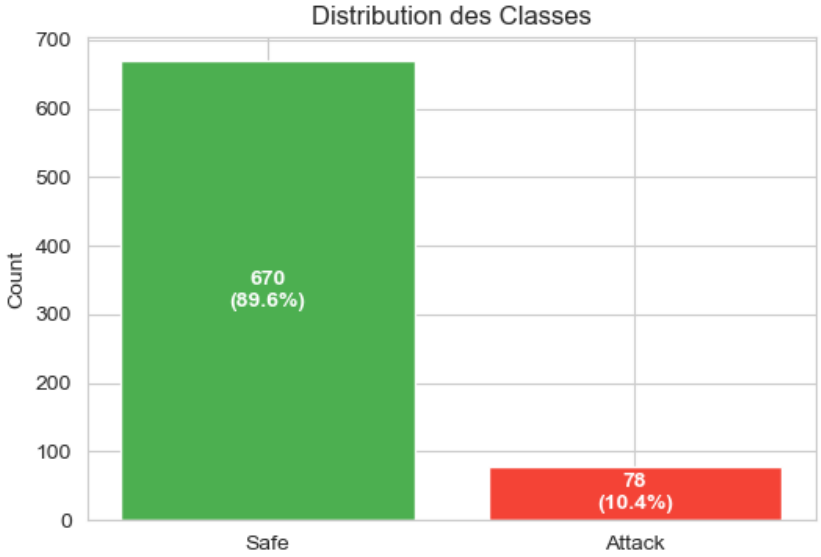

FIGURE 2.2 – Distribution des classes (Premier dataset) avant le prétraitement

Selon [Koudri, 2021], un déséquilibre signifie que le nombre de points de données disponibles pour différentes classes est différent : s'il y a deux classes, alors des données équilibrées signifieraient 50% de points pour chacune des classes.

Pour la plupart des techniques d'apprentissage automatique, peu de déséquilibre n'est pas un problème, ainsi, s'il y a 60% de points pour une classe et 40% pour l'autre classe, cela ne devrait pas entraîner de dégradation significative des performances, ce n'est que lorsque le déséquilibre de classe est élevé, par exemple 90% de points pour une classe et 10% pour l'autre, que les critères d'optimisation standard ou les mesures de performance peuvent ne pas être aussi efficaces et nécessiter des modifications [Koudri, 2021].

Face à ce déséquilibre de notre dataset, nous avons choisi d'utiliser SMOTE.

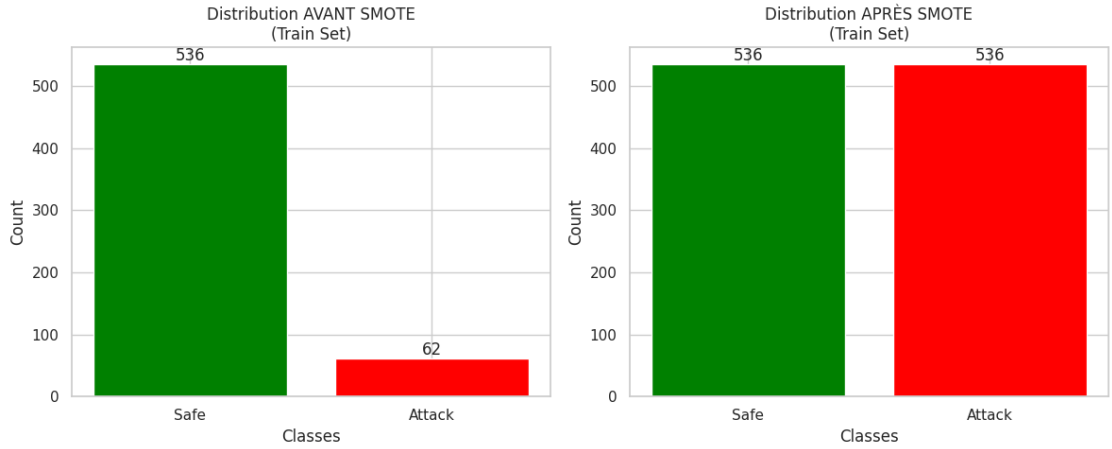

FIGURE 2.3 – Distribution des classes (Premier dataset) avant et après SMOTE

2.4.1.2 Sélection des features

L'objectif principal est d'identifier les features les plus pertinentes, non redondantes et cohérentes pour la construction d'un modèle [Chouaib, 2011], Cette réduction méthodique de la taille du dataset devient essentielle face à la croissance continue du volume et de la variété des données disponibles.

Pour atteindre cet objectif, nous avons procédé comme suit.

— **Élimination des features de faible importance**

Parmi les techniques de prétraitement des données, nous avons appliqué l'élimination des features de faible importance [Guyon and Elisseeff, 2003].

Cette décision s'appuie sur les résultats des tests d'importance des variables, qui ont révélé les contributions relatives suivantes : [intent :77.54%, spelling :19.90%, link :2.56%]. Ainsi, nous avons supprimé la colonne (link) dont l'impact sur la performance du modèle était négligeable.

2.4.2 Le deuxième dataset

Voici les techniques de prétraitement appliquées à notre deuxième dataset.

2.4.2.1 Le nettoyage des données

— **Filtrage des doublons**

Consiste à éliminer les entrées redondantes dans le dataset, afin d'éviter les biais dans l'apprentissage du modèle et améliorer la qualité des données.

— **Le prétraitement textuel** Standardise les données par convertissant tout en minuscules, supprimant les caractères non alphanumériques (ponctuation,

symboles), et éliminant les éléments superflus (stop words, espaces multiples, fautes de frappe) pour obtenir un corpus normalisé prêt pour l'analyse.

— **Algorithme de Prétraitement des données** L'algorithme résume les étapes de prétraitement des données de notre dataset :

Algorithme 1 Data preparation

Algorithme 1 : CSE_Data_Preprocess

```
1 : procedure CSE_Preprocess(CSE_dataset)
2 :    Input : CSV file path
3 :    Output : CSE_Clean_dataset
4 :    CSE_Dataset ← load_csv('dataset.csv')
5 :    Drop NaN values in 'is_attack' column
6 :    for each text x in 'is_attack' do
7 :       x ← Remove_doublons(x)
8 :       x ← Remove_mentions(x)
9 :       x ← Remove_Hashtags(x)
10 :      x ← Remove_special_characters(x)
11 :      x ← Remove_consecutive_spaces(x)
12 :      x ← Lowercase(x)
13 :   end for
14 :   return CSE_Clean_dataset
15 : end procedure
```

— **Exemple d'application :** En ce qui suit, on a les deux conversations extraites du deuxième dataset, colonne "dialog", comme des exemples d'application des traitements effectués sur les données.

Dialog1 : "This is mike smith from orange. I am a sales rep. would you like to buy some software.".

Dialog2 :"I am john doe from apple computers where I am a sales rep. could you read me your passcode to make sure our records are up to date. thanks. bye.

donc on consider que Dialog1 contient 03 phrase separes et Dialog2 une seul phrase comme suite :

Dialog1 = ["This is mike smith from orange.", "I am a sales rep.", "Would you like to buy some software."].

Dialog2 = ["I am john doe from apple computers where I am a sales rep. could you read me your passcode to make sure our records are up to date thanks bye."].

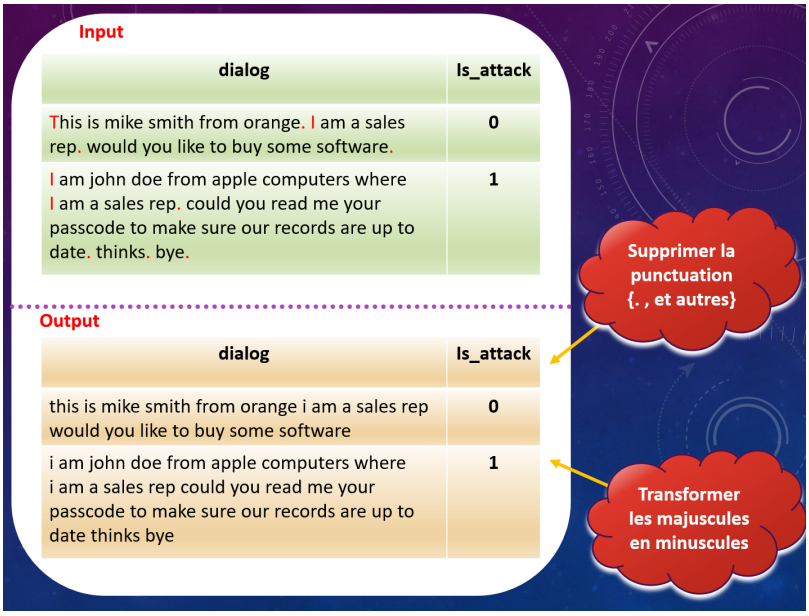

FIGURE 2.4 – Prétraitement des données : Nettoyage des données

2.4.2.2 Le mappage des données

Cette étape consiste à convertir les étiquettes textuelles de la colonne "is_attack" (True et False) en valeurs numériques (‘1’ pour ‘True’ et ‘0’ pour ‘False’).

dialog	**is_attack**	**is_attack mapped**
This is mike smith from orange. I am a sales rep. would you like to buy some software.	False	0
I am john doe from apple computers where I am a sales rep. could you read me your passcode to make sure our records are up to date. thanks. bye.	True	1
Hi this is mike jones from orange. I am the sales rep. would you like a firewall. thanks bye	False	0

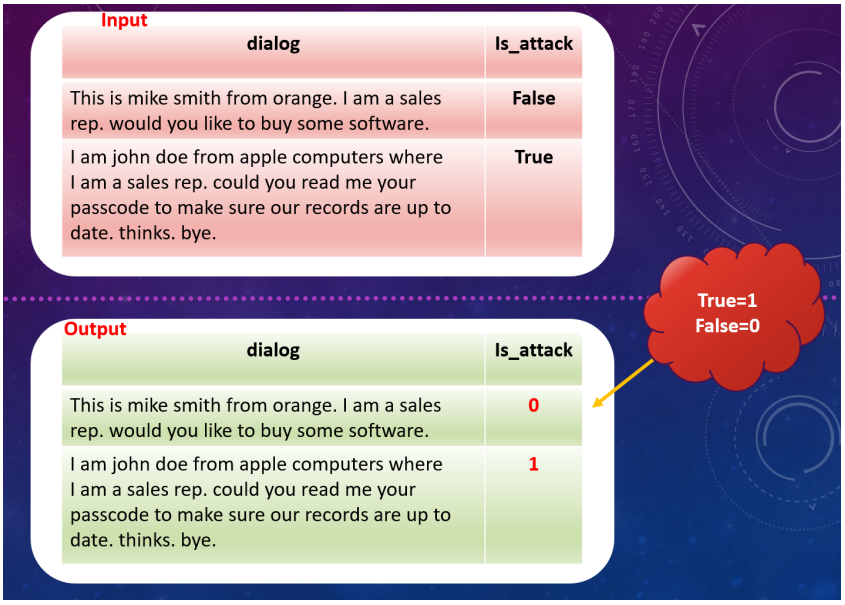

FIGURE 2.5 – Prétraitement des données : Mappage des données

2.4.2.3 Sélection des features

Dans le cadre du prétraitement de nos données, nous allons éliminer certaines features non pertinentes du second dataset, notamment les identifiants uniques (voice_id, num_caller, num_receiver) ainsi que les données temporelles brutes (date, time) qui n'apportent pas de valeur analytique dans leur format actuel, ainsi que le feature "is_attack" qui devient redondante après le mappage, ainsi, notre dataset conservera les autre features pertinentes (dialog, is_attack_mapped).

2.5 Tokenisation

C'est le processus de division du texte en unités plus petites, communément appelées tokens, qui peuvent être des mots, des expressions, voire des caractères [Grefenstette, 1999].

C'est une étape importante à réaliser avant la vectorisation, car elle facilite le traitement et l'extraction d'informations précieuses.

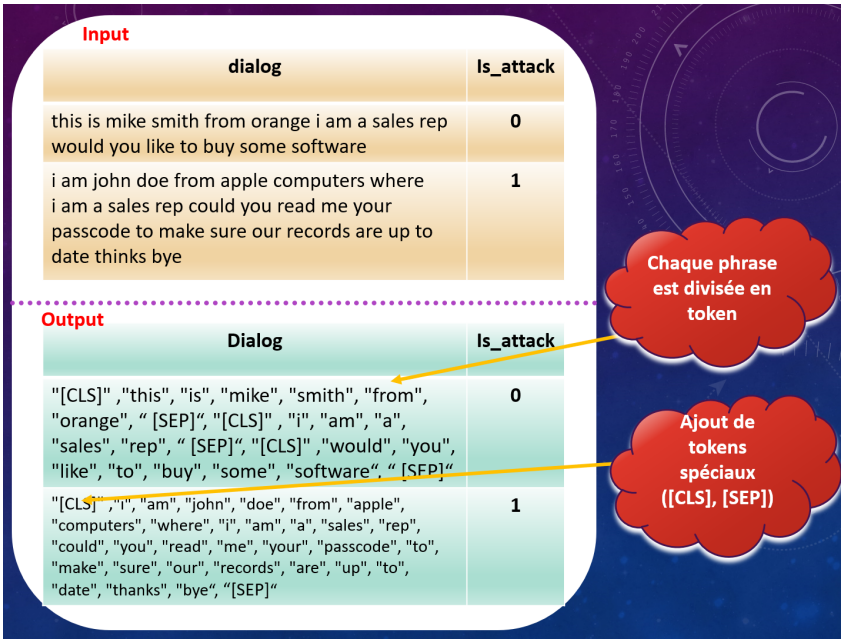

FIGURE 2.6 – Prétraitement des données : Tokenisation

2.6 Vectorisation

La vectorisation consiste à convertir des données textuelles en vecteurs numériques, un format directement compréhensible par l'ordinateur [Držík and Šteflovič, 2023], C'est une étape cruciale, car elle transforme des données non structurées en une représentation qui capture le sens sémantique, les relations et les motifs présents dans le texte.

Cependant, il est impératif de passer au préalable par l'étape de tokenisation, suivie du padding (rembourrage), ces deux étapes cruciales seront détaillées dans les sections suivantes.

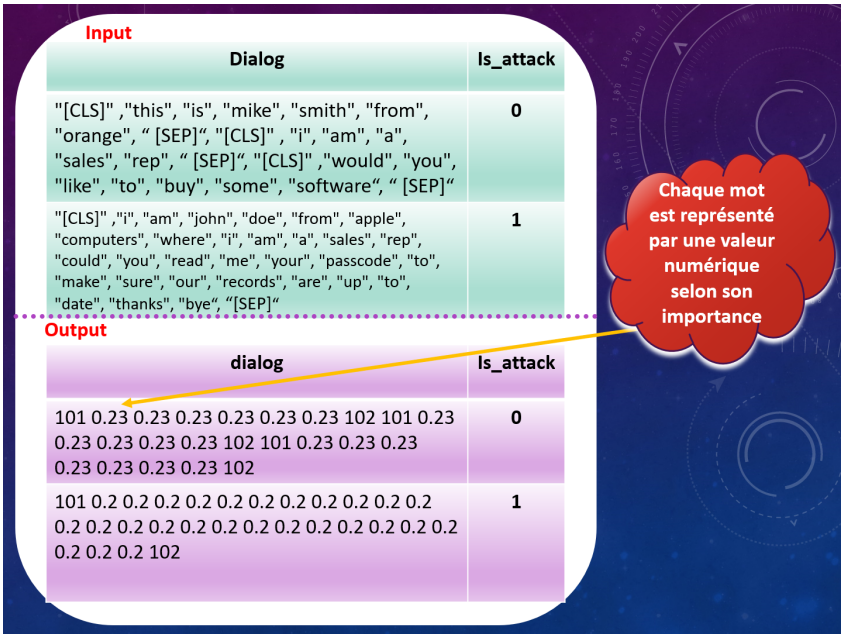

FIGURE 2.7 – Prétraitement des données : Vectorisation

2.7 Padding

Consiste à ajouter des valeurs supplémentaires au début ou à la fin de séquences de données afin d'assurer qu'elles aient toutes la même longueur [Dang et al., 2022]. Cette opération est essentielle pour permettre le traitement de ces séquences en lots (batches), notamment dans les réseaux de neurones où les dimensions doivent être cohérentes.

Concrètement, les séquences les plus courtes sont étendues pour atteindre la longueur de la séquence la plus longue du dataset, en ajoutant une valeur spécifique de remplissage, souvent le zéro.

Il existe principalement deux types de padding :

— **Pré-padding** : ajout de valeurs au début des séquences ;

— **Post-padding** : ajout de valeurs à la fin des séquences.

En appliquant ce principe à notre exemple applicatif (cf. exemple 2.4.2.1), les séquences sont alignées sur la phrase la plus longue du premier dialogue (**Dialog1**), ce qui donne la représentation suivante des trois phrases incluses :

```
[101, 2023, 2003, 7932, 3722, 2091, 4864, 1012, 102, 0, 0, 0, 0, 0]
[101, 1045, 2572, 1037, 2965, 2372, 1012, 102, 0, 0, 0, 0, 0, 0]
[101, 2058, 2017, 2066, 2000, 2984, 2075, 7987, 1012, 102, 0, 0, 0, 0]
```

Le masque d'attention (*Attention Mask*) associé, qui indique quelles positions correspondent à des tokens réels (valeur 1) et lesquelles sont du padding (valeur

0), est alors :

```
[1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0]
[1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0]
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0]
```

2.8 Word Embedding

C'est une méthode utilisée pour projeter des mots ou des expressions textuelles dans un espace continu de faible dimension, cette projection permet aux mots similaires d'avoir des encodages similaires [Ghannay et al., 2016].

Word Embedding est un outil puissant largement utilisé dans diverses tâches modernes de traitement automatique du langage naturel (TALN), notamment la classification de textes, le clustering de documents et la classification de sentiments.

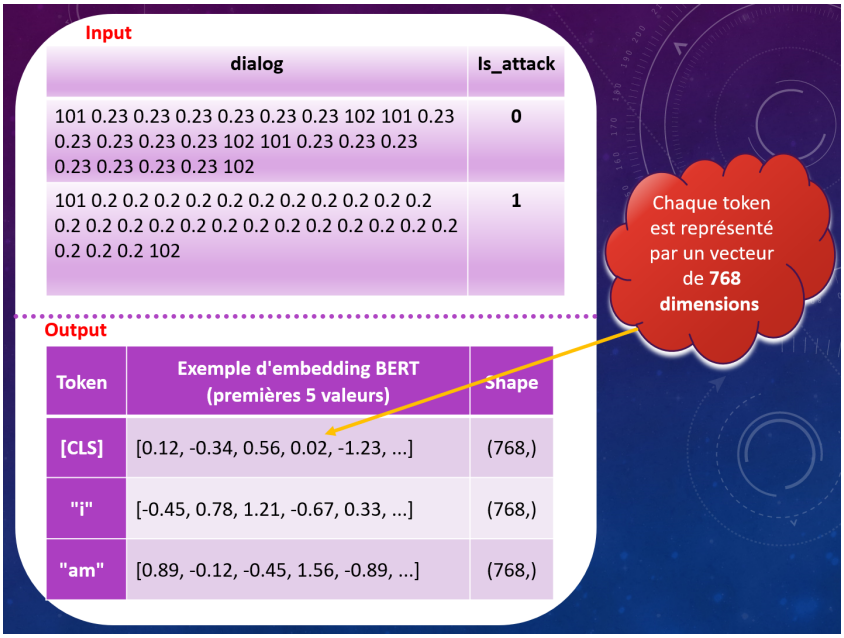

FIGURE 2.8 – Prétraitement des données : Word Embedding

2.9 Modèles proposés

À ce stade, nous avons choisi trois approches :

2.9.1 Approche machine learning traditionnel ML

Nous avons proposé six algorithmes de classification (DT, RF, AdaBoost, XGBoost, SVM, TinyML), couvrant divers paradigmes algorithmiques, ce qui nous permet une sélection objective du modèle le plus performant pour la tâche considérée.

2.9.1.1 Descriptions des modèles choisis

1. **DT** Les arbres de décision sont des algorithmes d'apprentissage supervisé utilisés pour la classification et la régression, leur principe repose sur la division récursive de l'espace des données selon des règles simples (basées sur les caractéristiques(features)), formant une structure arborescente où chaque nœud interne représente un test sur une variable, chaque branche un résultat du test, et chaque feuille une décision ou prédiction.

FIGURE 2.9 – Présentation de l'algorithme des arbres de décision (DT)

2. **RF** Introduites par Breiman (2001), sont une méthode d'apprentissage ensembliste qui combine plusieurs arbres de décision pour améliorer les performances prédictives et réduire le surapprentissage, chaque arbre est entraîné sur un sous-ensemble aléatoire des données (échantillonnage avec remise, ou bootstrap) et un sous-ensemble aléatoire des caractéristiques (feature bagging), ce qui augmente la diversité des modèles [Hastie et al., 2009].

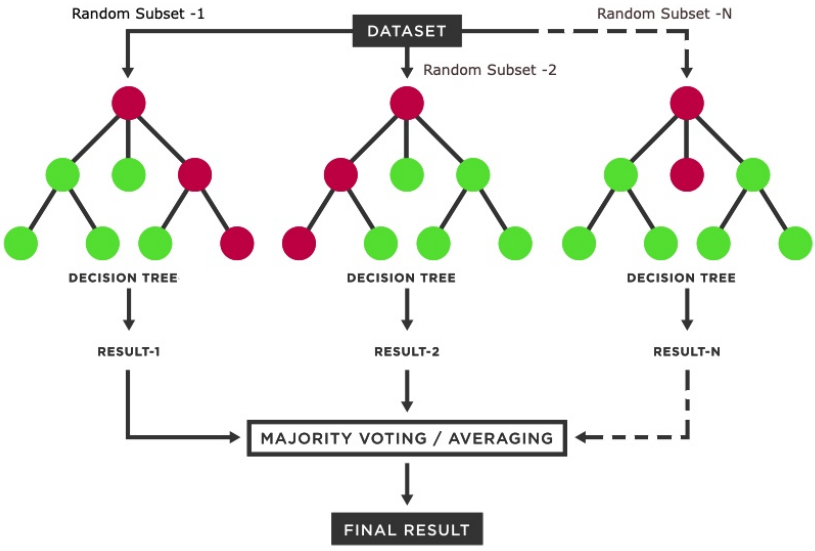

FIGURE 2.10 – Présentation de l'algorithme des forêts aléatoires (RF)

3. **AdaBoost** AdaBoost (Adaptive Boosting), proposé par Freund et Schapire (1997), est un algorithme de boosting qui combine plusieurs classifieurs faibles (généralement des arbres de décision simples) pour former un modèle fort, le principe clé repose sur l'affectation itérative de poids aux observations, en augmentant l'importance des instances mal classées à chaque itération, ce qui permet aux classifieurs suivants de se concentrer sur les cas difficiles [Freund and Schapire, 1997] [Hastie, 2009].

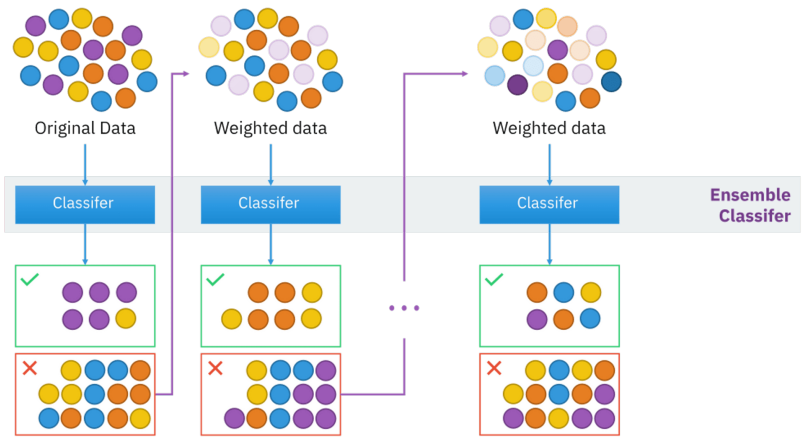

FIGURE 2.11 – Présentation Présentation AdaBoost

4. **XGBoost** XGBoost (eXtreme Gradient Boosting), développé par Chen et Guestrin (2016), est une méthode avancée de boosting basée sur des arbres de décision qui se distingue par son efficacité et ses performances élevées, Contrairement aux algorithmes de boosting traditionnels, XGBoost intègre une régularisation L1/L2

pour prévenir le surapprentissage, gère efficacement les valeurs manquantes, et utilise des techniques de parallélisation pour accélérer l'entraînement [Chen and Guestrin, 2016], [Friedman, 2001].

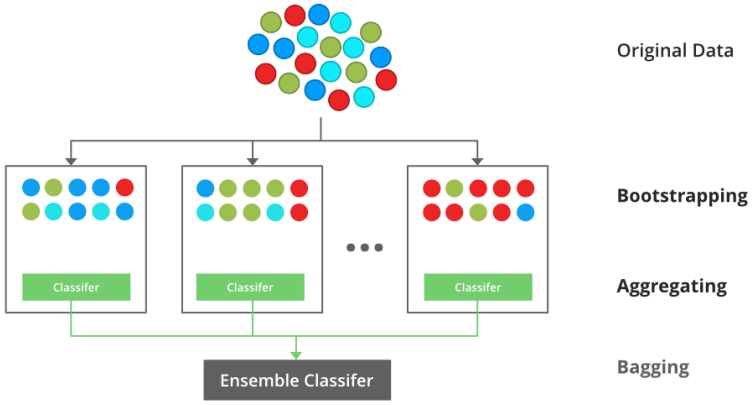

FIGURE 2.12 – Présentation de l'algorithme XGBoost

5. **Support Vector Machines SVM** Les SVM (Support Vector Machines) développée initialement par Vapnik et Chervonenkis dans les années 1960, puis popularisée dans les années 1990 [Cortes and Vapnik, 1995], sont des algorithmes d'apprentissage supervisé utilisés pour la classification et la régression, Leur objectif est de trouver la frontière optimale (hyperplan) qui sépare les classes en maximisant la marge entre elles, Grâce aux fonctions noyau (kernel trick), les SVM peuvent traiter des problèmes complexes non linéaires [Boser et al., 1992], Efficaces en haute dimension et résistants au surapprentissage, ils sont cependant sensibles aux hyperparamètres et coûteux en calcul pour de grands datasets [Pedregosa et al., 2011].

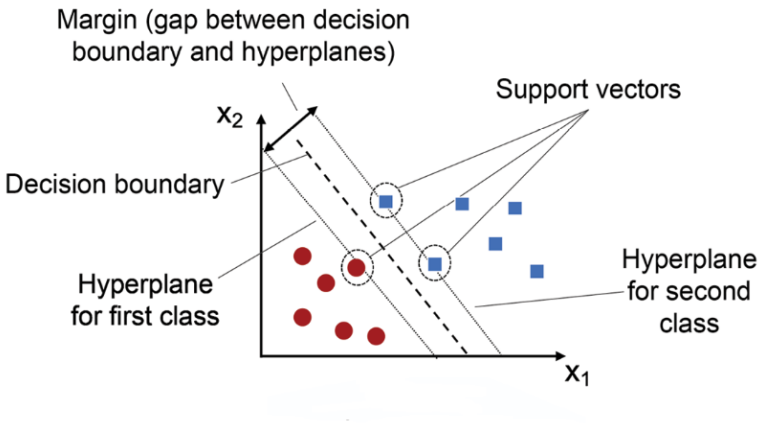

FIGURE 2.13 – Présentation de l'algorithme Support Vector Machines SVM

6. **TinyML** TinyML (Tiny Machine Learning) popularisée par Warden et Situnayake (2019), désigne l'ensemble des techniques permettant d'exécuter des modèles de

machine learning sur des dispositifs embarqués à très faible puissance, comme les microcontrôleurs (ex. : Arduino, ESP32). Les applications typiques incluent la reconnaissance vocale locale, la maintenance prédictive ou l'agriculture intelligente [David et al., 2021].

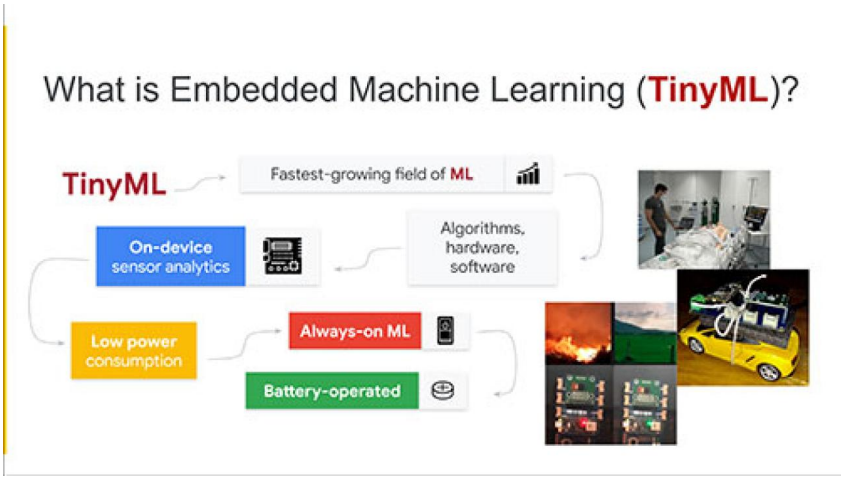

FIGURE 2.14 – Présentation de l'algorithme TinyML

2.9.1.2 Méthode de sélection de l'algorithme ML le plus performant

Elle consiste à sélectionner, parmi les six modèles d'apprentissage automatique proposés, celui qui donne les meilleurs résultats à partir des données du premier dataset, après une série de tests effectués sur chacun des algorithmes choisis, la figure **??** illustre la méthode utilisée pour sélectionner l'algorithme de machine learning le plus performant.

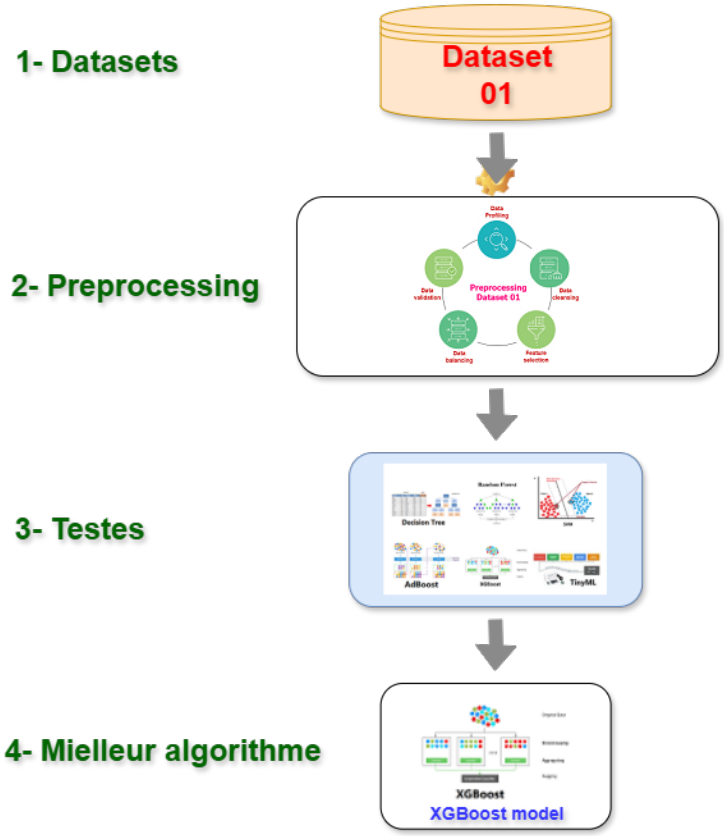

FIGURE 2.15 – Processus de sélection du meilleur algorithme d'apprentissage automatique

2.9.2 Approche par transformer BERT

Il convient de souligner que le modèle BERT a été retenu comme référence dans notre cadre expérimental.

BERT (BERT- Bidirectional Encoder Representations from Transformers) est un modèle de représentation contextuelle du langage naturel développé par Google [Devlin et al., 2019], fondé sur l'architecture transformer, il utilise des mécanismes d'attention multi-têtes pour traiter efficacement les séquences textuelles, sa particularité réside dans son approche bidirectionnelle profonde, qui capture simultanément le contexte gauche et droit de chaque token, permettant ainsi une modélisation fine des dépendances linguistiques [Mezhoud, 2024].

Le pré-entraînement de BERT s'effectue de manière auto-supervisée sur des importants corpus non annotés (Wikipedia, BookCorpus).

L'architecture de BERT repose sur un encodeur transformer bidirectionnel avec plusieurs couches [Vaswani et al., 2017], Deux configurations principales, BERTBASE et BERTLARGE, sont définies par le nombre de couches (L), la taille cachée (H), et le nombre de têtes d'auto-attention (A).

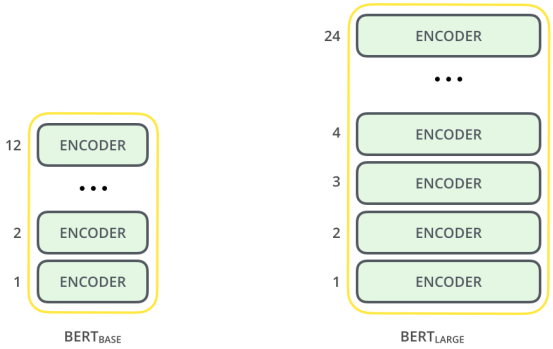

FIGURE 2.16 – Architecture de BERT

Chaque encodeur transforme une séquence de 512 tockens en une représentation vectorielle, la sortie de chaque encodeur peut être considérée comme une représentation contextuelle de la séquence d'entrée, obtenue grâce au mécanisme d'attention.

La nature textuelle des dialogues (la colonne "dialog") dans notre second dataset justifie l'emploi du modèle BERT, nous avons opté pour BERTBASE qui offre un bon compromis entre performance et coût computationnel, faisant de lui le choix le plus adapté à notre étude.

2.9.3 Approche hybrid BERT et machine learning traditionnel

Cette dualité méthodologique permet d'exploiter simultanément la rapidité des algorithmes classiques (XGBoost qui a donné les meilleurs résultats dans notre cas.) et la puissance des modèles de langue modernes BERT, offrant ainsi une vision exhaustive des capacités prédictives sur nos données, cette approche est illustrée dans la figure 2.17 suivante.

Nous exploitons le modèle BERT pour générer des embeddings à partir des données du second dataset CSE, ces embeddings sont ensuite directement utilisés pour classè les attaques CSE par le modèle de Machine Learning (XGBoost) préalablement sélectionnés pour ses performances optimales, ensuite de l'évaluer (précision, rappel, F1, etc.).

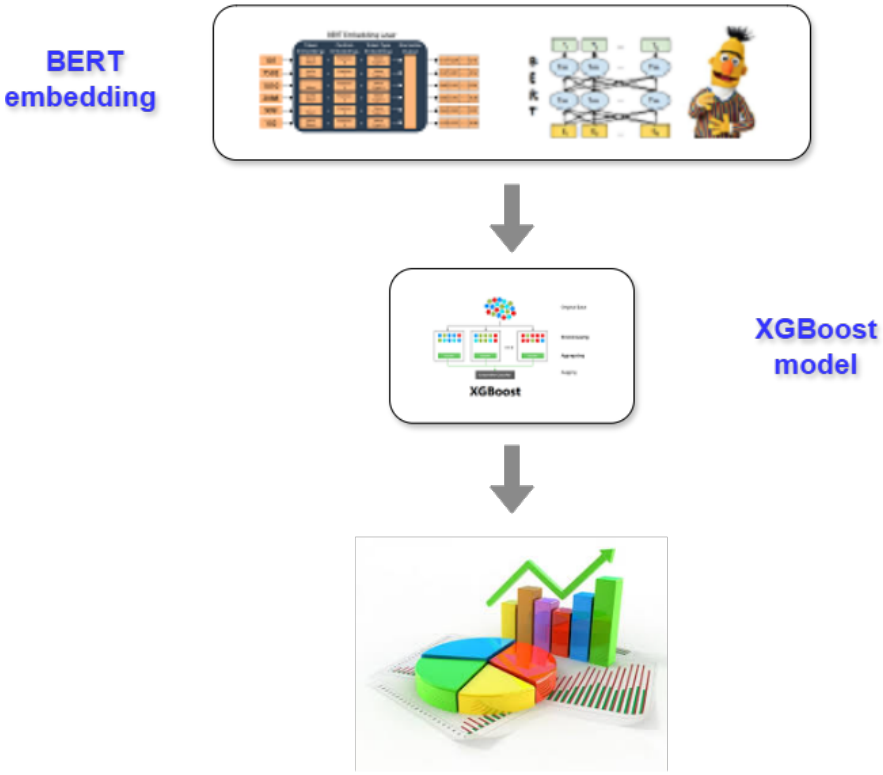

FIGURE 2.17 – Approche hybrid BERT et machine learning traditionnel

2.10 Conclusion

Ce chapitre propose une méthodologie complète pour la détection des attaques CSE, combinant approches modernes et classiques de machine learning, après un prétraitement minutieux des données, nous avons évalué et comparé trois stratégies principales : une approche machine learning traditionnel, une approche basée uniquement sur les embeddings BERT et une dernière méthode hybride exploitant ces mêmes embeddings comme entrées pour l'algorithme de classification (XGBoost).

Dans le chapitre suivant, nous présenterons la configuration expérimentale avec les résultats obtenus pour chaque modèle proposé.

Chapitre 3

Testes et résultats

3.1 Introduction

Ce chapitre présente une évaluation approfondie des performances comparées de trois approches de détection d'attaques d'ingénierie sociale CSE : les modèles de machine learning classiques, l'approche BERT et notre architecture hybride BERT_ML.

L'expérimentation s'appuie sur un corpus dédié de conversations annotées, après avoir optimisé les réglages des hyperparamètres, une analyse comparative des résultats est effectuer.

Il faut noter que pour les deux approches BERT et BERT_ML, vu qu'on a travaillé avec un dataset équilibré, il est inutile d'utiliser SMOTE.

3.2 Métriques d'évaluation

Passage à la phase d'évaluation, où nous évaluons les performances de chaque modèle sur de nouvelles données non vues auparavant, à savoir l'ensemble de test, ainsi que sa capacité à faire des prédictions sur des données distinctes de l'ensemble d'entraînement, en utilisant des métriques de performance telles que :

3.2.1 Matrice de confusion

Une matrice de confusion est un tableau qui indique le nombre des prédictions faites par le modèle par rapport aux vraies valeurs (réelles), il s'agit d'un tableau carré de taille n × n, où n représente le nombre de classes dans un dataset. [Choudhary and Gianey, 2017].

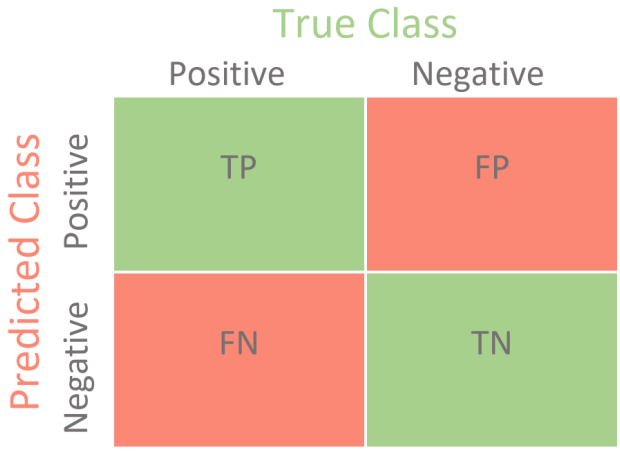

FIGURE 3.1 – Matrice de confusion typique pour un problème de classification binaire

— **VP (Vrais Positifs)** : Proportion d'attaques CSE correctement identifiées comme telles par le modèle.
— **VN (Vrais Négatifs)** : Proportion d'activités légitimes correctement classifiées comme normales.
— **FP (Faux Positifs)** : Proportion d'activités légitimes incorrectement flaggées comme attaques CSE.
— **FN (Faux Négatifs)** : Proportion d'attaques CSE incorrectement classées comme activités normales.

3.2.2 Définitions fondamentales

— **Accuracy** : Elle représente la proportion de prédictions correctes parmi l'ensemble des prédictions effectuées [Hossin and Sulaiman, 2015].

$$Accuracy = \frac{TN + TP}{TN + FN + TP + FP} \quad (4.1)$$

— **Précision** : La précision mesure la capacité d'un modèle à ne prédire "positif" que lorsqu'il a raison, autrement dit, parmi tous les exemples que le modèle a prédits comme positifs, combien sont vraiment positifs [Hossin and Sulaiman, 2015].

$$\text{Précision} = \frac{TP}{TP + FP} \quad (4.2)$$

— **Recall** : Capacité à détecter toutes les attaques, autrement dit, parmi tous les vrais positifs existants dans les données, combien ont été correctement détectés par le modèle [Hossin and Sulaiman, 2015].

Un recall élevé signifie que le modèle rate peu de cas positifs, ce qui est souvent plus important que l'accuracy dans des contextes sensibles.

$$Recall = \frac{TP}{TP + FN} \quad (4.3)$$

— **F1-score** : est la moyenne harmonique entre la précision et le recall.

Il fournit une mesure unique qui équilibre la capacité du modèle à identifier correctement les cas positifs (recall) et à éviter les fausses alertes (précision) [Hossin and Sulaiman, 2015].

$$F1 - score = 2 \times \frac{Prcision \times Rappel}{Prcision + Rappel} \quad (4.4)$$

Dans le domaine de la détection des attaques CSE, l'évaluation du modèle doit privilégier le recall plutôt que d'autres métriques, car ce critère reflète la capacité du modèle à identifier correctement les véritables attaques, un faux négatif, où le modèle ne détecte pas une attaque réelle, représente un risque de sécurité critique [Vennila et al., 2018].

3.3 Division des données

1. **Approche ML classique**

   Cette étape consiste à séparer les données en deux ensembles distincts : l'ensemble d'entraînement (80%) et l'ensemble de test (20%), comme le montre la figure 3.2.

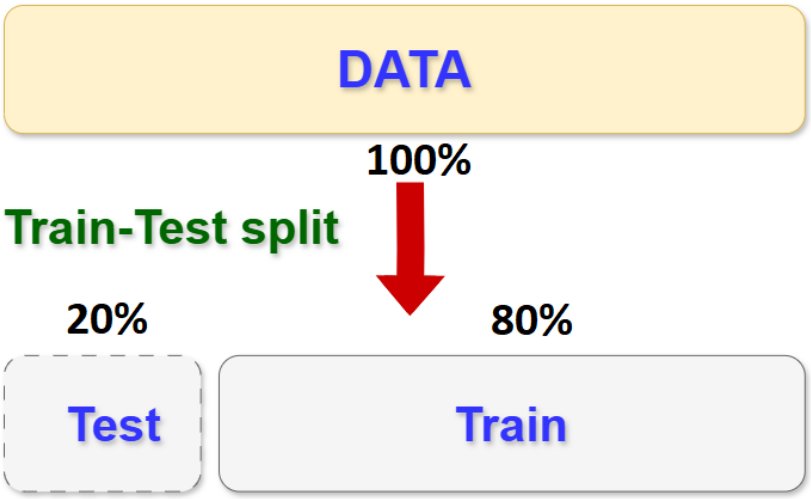

FIGURE 3.2 – Division des données : ensemble d'entraînement et ensemble de test

2. **Approche BERT et BERT_XGBoost**

Dans cette étape les données son séparer en trois ensembles distincts : l'ensemble d'entraînement (70%), l'ensemble de validation (15%) et l'ensemble de test (15%), comme le montre la figure 3.3.

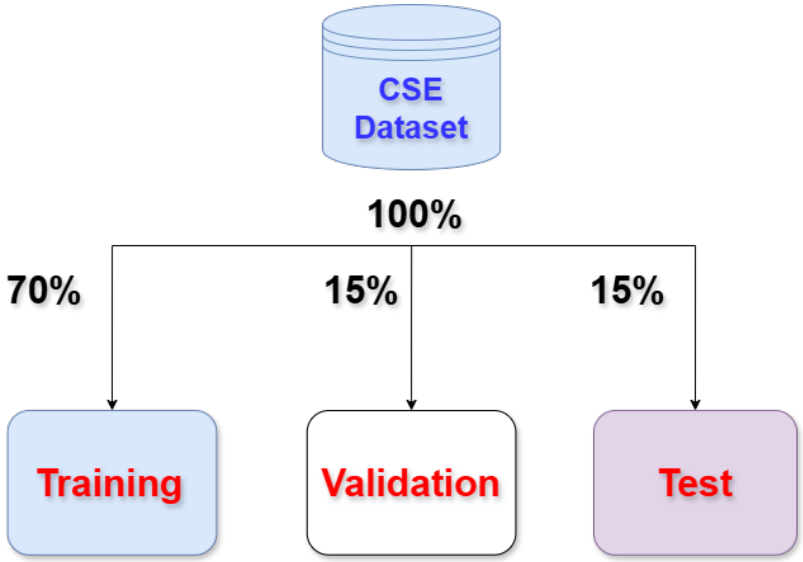

FIGURE 3.3 – Division des données : ensemble d'entraînement, ensemble de validation et ensemble de test

3.4 Configuration des hyperparamètres

Les modèles d'apprentissage automatique possèdent à la fois des paramètres et des hyperparamètres, les paramètres sont appris pendant l'entraînement tendi que les hyperparamètres sont généralement fixés par l'utilisateur avant l'entraînement [Arnold et al., 2024].

Pour l'optimisation des modèles, nous allons adopter la recherche GridSearchCV, qui présente plusieurs avantages spécifiques par rapport à d'autres méthodes de recherche dans le cadre de l'optimisation des hyperparamètres, tant pour les modèles BERT, hybride que pour d'autres approches de machine learning (DT, RF, AdBoost, XGBoost, SVM...Ext.).

en ce qui suit, nous allons présenter pour chaque approche les meilleurs hyperparamètres trouvés par GridSearchCV :

1. **Machine learnning**

   Nous proposons, dans un premier temps, une synthèse des meilleurs hyperparamètres identifiés à l'aide de la méthode GridSearch pour chacun des modèles étudiés, cette analyse est menée à la fois avant et après l'application de la technique d'équilibrage SMOTE, il faut noter qu'à cette étape, nous avons utilisé un

dataset déséquilibré (premier dataset).

— **Sans SMOTE**

Modèle	**Hyperparamètres optimaux**
Decision Tree	`class_weight` : balanced, `criterion` : gini, `max_depth` : 5, `max_features` : `null`, `min_samples_leaf` : 4, `min_samples_split` : 2
Random Forest	`bootstrap` : `true`, `class_weight` : balanced_subsample, `max_depth` : 5, `max_features` : `null`, `min_samples_leaf` : 4, `min_samples_split` : 10, `n_estimators` : 200
AdaBoost	`algorithm` : SAMME, `estimator` : `null`, `learning_rate` : 1.0, `n_estimators` : 50, `random_state` : 42
XGBoost	`colsample_bytree` : 0.8, `gamma` : 0, `learning_rate` : 0.01, `max_depth` : 3, `n_estimators` : 300, `reg_alpha` : 1, `reg_lambda` : 0, `scale_pos_weight` : 1, `subsample` : 1.0
SVM	`C` : 0.1, `class_weight` : balanced, `degree` : 3, `gamma` : scale, `kernel` : poly
TinyML(XGBoost)	`colsample_bytree` : 0.8, `gamma` : 0, `learning_rate` : 0.01, `max_depth` : 3, `n_estimators` : 300, `reg_alpha` : 1, `reg_lambda` : 0, `scale_pos_weight` : 1, `subsample` : 1.0

Tableau 3.1 – Hyperparamètres optimaux des modèles ML sans SMOTE

— **Avec SMOTE**

Modèle	Hyperparamètres optimaux
Decision Tree	`class_weight` : `null`, `criterion` : entropy, `max_depth` : 5, `max_features` : sqrt, `min_samples_leaf` : 4, `min_samples_split` : 2, `smote__k_neighbors` : 7, `smote__sampling_strategy` : 0.5
Random Forest	`bootstrap` : `true`, `class_weight` : balanced, `max_depth` : 5, `max_features` : `null`, `min_samples_leaf` : 4, `min_samples_split` : 2, `n_estimators` : 300, `smote__k_neighbors` : 7, `smote__sampling_strategy` : 0.5
AdaBoost	`algorithm` : SAMME.R, `estimator` : `null`, `learning_rate` : 0.1, `n_estimators` : 100, `random_state` : 42
XGBoost	`colsample_bytree` : 1.0, `gamma` : 0, `learning_rate` : 0.01, `max_depth` : 3, `n_estimators` : 300, `reg_alpha` : 0, `reg_lambda` : 0, `scale_pos_weight` : 8.645, `subsample` : 1.0
SVM	`C` : 1, `class_weight` : balanced, `degree` : 3, `gamma` : auto, `kernel` : poly, `smote__k_neighbors` : 7, `smote__sampling_strategy` : 0.5
TinyML(SVM)	`C` : 1, `gamma` : scale, `kernel` : linear

Tableau 3.2 – Hyperparamètres optimaux des modèles ML avec SMOTE

2. **Modèle BERT**

   En ce qui suit, nous présentons les meilleurs hyperparamètres utilisés avec le modèle BERT, il faut toutefois noter qu'à cette étape, nous avons utilisé un dataset équilibré (le deuxième dataset), donc SMOTE n'a pas été appliqué.

Modèle	Paramètre	Valeur optimale
BERT	`batch_size`	16
	`learning_rate`	2.25×10^{-5}
	`num_epochs`	4
	`warmup_steps`	349
	`weight_decay`	0.096

Tableau 3.3 – Hyperparamètres optimaux du modèle BERT pur

3. **Modèle BERT_ML** Cette section présente la configuration optimale des hyperparamètres du modèle BERT_ML, déterminée à partir d'un dataset équilibré (deuxième dataset), ce qui rend l'utilisation de SMOTE inutile dans notre cas.

Modèle	Paramètre	Valeur optimale
BERT	batch_size	16
	learning_rate	2.25×10^{-5}
	num_epochs	4
	warmup_steps	349
	weight_decay	0.096
XGBoost	colsample_bytree	0.8
	gamma	0
	learning_rate	0.01
	max_depth	3
	n_estimators	300
	reg_alpha	1
	reg_lambda	0
	scale_pos_weight	1
	subsample	1.0

Tableau 3.4 – Hyperparamètres optimaux du modèle BERT_ML

3.5 Experimentations

L'objectif de cette section est d'évaluer les performances des modèles proposés à travers différentes expérimentations et de sélectionner l'architecture optimale parmi eux.

Dans le domaine de la détection d'attaque CSE, l'évaluation du modèle doit privilégier le recall par rapport aux autres métriques, en effet, cette mesure reflète la capacité du modèle à identifier correctement les véritables incidents d'attaque CSE, un faux négatif (où le modèle ne détecte pas une attaque réelle) représente un risque de sécurité critique.

3.5.1 Performances des modèles ML classiques

Nous exposons dans ce suit les résultats des modèles classiques avec et sont SMOTE.

3.5.1.1 Avec SMOTE

Modèle	Accuracy	Précision	Rappel	F1-score
Decision Tree	84.67%	37.04%	62.5%	46.51%
Random Forest	82%	33.33%	68.75%	44.9%
AdaBoost	82%	33.33%	68.75%	44.9%
XGBoost	64%	21.21%	87.5%	34.15%
SVM	**87.33%**	**44.83%**	**81.25%**	**57.8%**
TinyML(SVM)	83%	36.11%	81.25%	50%

Tableau 3.5 – Performances comparées des modèles de ML avec SMOTE

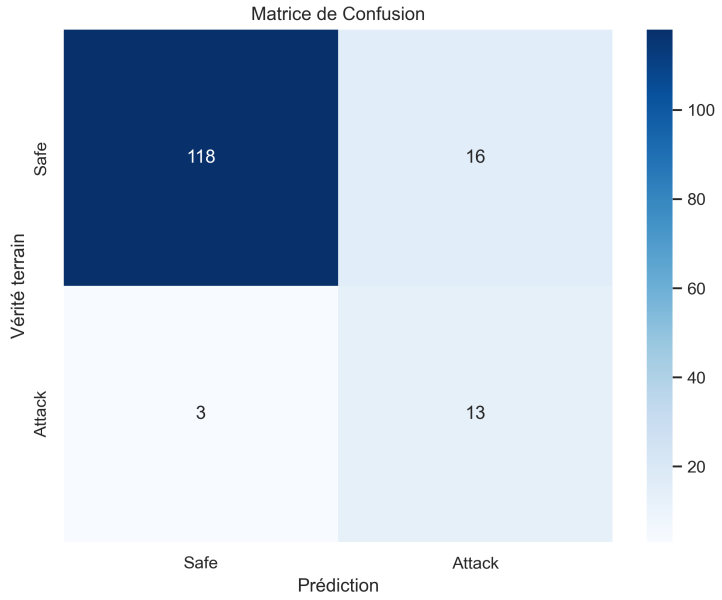

Figure 3.4 – Matrice de confusion SVM avec SMOTE

3.5.1.2 Sans SMOTE

Modèle	Accuracy	Précision	Rappel	F1-score
Decision Tree	82.67%	35.29%	75%	48%
Random Forest	82.67%	35.29%	75%	48%
AdBoost	92%	66.66%	50%	57.14%
XGBoost	**93.33%**	**87.5%**	**43.75%**	**58.34%**
SVM	86.67%	42.86%	75%	54.55%
TinyML(XGBoost)	93.33%	87.5%	43.75%	58.34%

Tableau 3.6 – Performances comparées des modèles de ML sans SMOTE

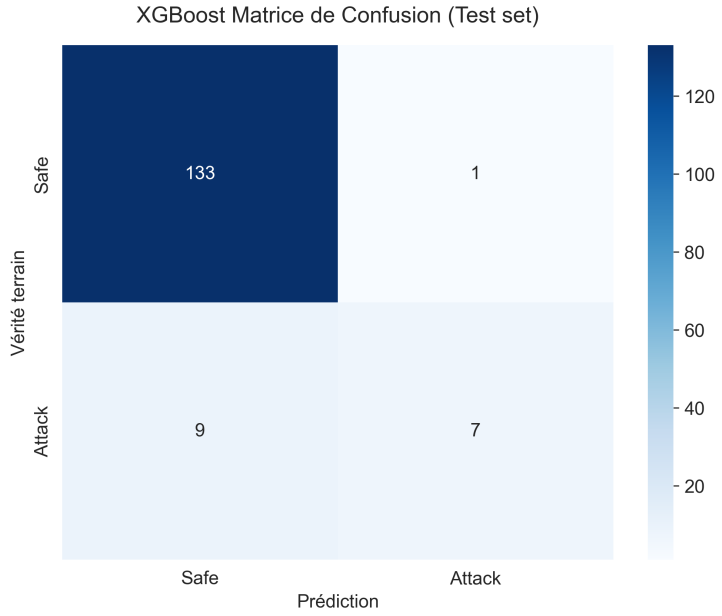

FIGURE 3.5 – Matrice de confusion XGBoost

Les résultats montrent que SVM avec SMOTE offre les meilleures performances globales, avec un F1-score de 57.8%, un rappel élevé (81.25%) et une précision acceptable (44.83%), ce qui en fait le modèle le plus équilibré pour détecter les attaques tout en minimisant les faux positifs.

En revanche, XGBoost sans SMOTE présente une accuracy plus élevée (93.33%) et une meilleure précision (87.5%), mais son rappel nettement plus faible (43.75%) signifie qu'il manque plus de la moitié des attaques, ce qui est critique en cybersécurité.

TinyML offre une solution flexible adaptée aux dispositifs embarqués, avec deux profils complémentaires, en version SVM+SMOTE, il maintient un excellent rappel (81.25%) comparable aux modèles standards, idéal pour la détection d'anomalies critiques, malgré une légère baisse de précision (36.11%), en version XGBoost, il reproduit à l'identique les performances du modèle standard (93.33% accuracy, 87.5% précision), optimal pour les applications sensibles aux faux positifs, bien que son rappel limité (43.75%) reste un point faible.

3.5.2 Performances du modèle BERT

Les résultats obtenus par le modèle BERT sont présentés dans le tableau 3.7 représenté ci-dessous.

Modèle	Accuracy	Précision	Rappel	F1-score
BERT	**70%**	**55%**	**75%**	**63%**

TABLEAU 3.7 – Résultats de l'approche BERT

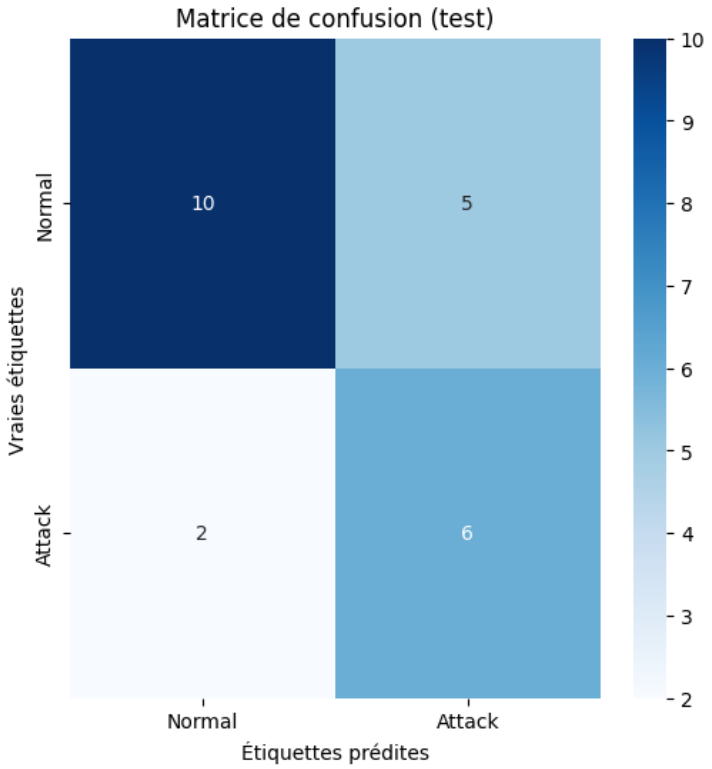

FIGURE 3.6 – Matrice de confusion BERT

Le modèle BERT présente un profil adapté aux besoins de sécurité avec 75% de rappel, garantissant la détection de la majorité des menaces, bien que sa précision modérée (55%) génère des faux positifs, a l'inverse, XGBoost affiche des performances déséquilibrées : malgré une accuracy impressionnante (93.33%) et une bonne précision (87.5%), son rappel extrêmement faible (43.75%) le rend inutilisable en pratique, car il manque plus de la moitié des attaques, tandis que SVM avec SMOTE offre un meilleur équilibre avec un rappel accru (81.25%) et un F1-score compétitif (57.8%).

3.5.3 Performances de l'approche hybride BERT_ML

Les résultats des différentes mesures d'évaluation (Accuracy, précision, Recall, et F1-score) pour l'approche hybride BERT_ML sont présentés dans le tableau 3.8 ci-dessous.

Configuration	Accuracy	Précision	Rappel	F1-score
BERT + XGBoost	**83%**	**82%**	**88%**	**85%**

TABLEAU 3.8 – Résultats de l'approche hybride BERT_ML

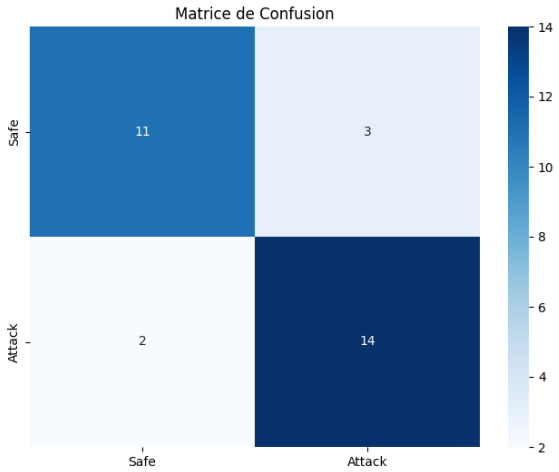

FIGURE 3.7 – Matrice de confusion BERT_ML

Les résultats expérimentaux démontrent que l'approche hybride BERT_ML combinant BERT et XGBoost, surpasse significativement les méthodes unimodales, atteignant un équilibre optimal entre précision (82%) et rappel (88%), avec un F1-score de 85%, l'analyse comparative révèle que ce modèle résout les limitations des approches existantes (BERT et ML classique (XGBoot)) : il évite à la fois la faible précision de BERT seul (55%) et le rappel insuffisant de XGBoost isolé (43.75%).

3.6 Conclusion

Cette étude compare trois approches pour la détection d'attaques CSE : les méthodes de ML classiques (avec analyse de l'impact de SMOTE), BERT seul, et une architecture hybride BERT-XGBoost, l'évaluation des algorithmes classiques a permis de sélectionner XGBoost comme meilleur candidat pour l'intégration hybride, tout en démontrant l'amélioration apportée par SMOTE (rappel de 81.25% pour SVM + SMOTE), le modèle BERT seul montre déjà des performances intéressantes (rappel 75%), mais c'est l'approche hybride BERT_ML qui établit un nouveau standard avec des résultats exceptionnels (précision 82%, rappel 88%, F1-score 85%).

Les versions TinyML (SVM + SMOTE et XGBoost) restent pertinentes pour des déploiements embarqués, mais l'architecture hybride se positionne comme la référence pour les systèmes critiques nécessitant une détection exhaustive et fiable.

Conclusion générale

Cette thèse a mis en lumière l'impact croissant des technologies émergentes sur la sophistication des attaques d'ingénierie sociale dans les chats CSE, en proposant une approche hybride, BERT_ML, combinant l'analyse sémantique profonde BERT et l'apprentissage automatique XGBoost, cette méthode répond à un besoin critique identifié face à l'explosion des menaces automatisées (+300%) et l'émergence de vecteurs sophistiqués comme les deepfakes conversationnels, le cadre théorique et expérimental développé ici fournit une base solide pour détecter ces attaques avec une compréhension fine du contexte linguistique et des schémas comportementaux.

Les performances de BERT_ML (précision : 82%, rappel : 88%) démontrent sa supériorité face aux approches traditionnelles (DT, RF, AdaBoost, XGBoost, SVM, TinyML), validant l'hypothèse qu'une synergie entre NLP (comme BERT) avancé et modèles discriminants est essentielle pour contrer les CSE modernes, ces résultats, bien que prometteurs, révèlent aussi des limites, notamment la généralisation restreinte due au dataset limité (147 échantillons) et l'adaptabilité aux futures évolutions tactiques des attaquants, néanmoins, cette recherche prouve qu'une détection robuste passe par une analyse multidimensionnelle intégrant sémantique, métadonnées et anomalies comportementales.

Cette étude ouvre des voies concrètes pour renforcer la lutte contre les attaques d'ingénierie sociale dans les chats, son intégration opérationnelle dans des environnements réels - notamment les plateformes de messagerie et les infrastructures critiques - permettrait une validation à grande échelle et une amélioration continue du système, l'évolution technologique devrait s'orienter vers l'extension des capacités de détection pour couvrir les deepfakes vocaux et visuels, ainsi que d'autres menaces émergentes, grâce à des architectures hybrides enrichies, la sensibilisation par l'adoption d'une approche combinant détection automatisée et campagnes éducatives ciblées s'avère essentielle pour protéger les utilisateurs vulnérables comme les personnes âgées ou les employés d'entreprises sensibles.

Finalement cette étude jette les bases d'une défense proactive contre les CSE, mais appelle une approche dynamique alliant technologie et interdisciplinarité(cybersécurité, psychologie, IA), les prochaines étapes clés concerneront l'élargissement des données (Da-

taset) et l'optimisation temps réel des contre-mesures.

Bibliographie

Bibliographie

[Abobor and Josyula, 2023] Abobor, M. and Josyula, D. P. (2023). Socialbert a transformer based model used for detection of social engineering characteristics. In *2023 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 174–178. IEEE.

[Ai et al., 2024] Ai, L., Kumarage, T., Bhattacharjee, A., Liu, Z., Hui, Z., Davinroy, M., Cook, J., Cassani, L., Trapeznikov, K., Kirchner, M., et al. (2024). Defending against social engineering attacks in the age of llms. *arXiv preprint arXiv :2406.12263*.

[Alemayehu, 2023] Alemayehu, Y. (2023). *SOCIAL ENGINEERING ATTACK DETECTION MODEL*. PhD thesis, St. Mary's University.

[Alshingiti et al., 2023] Alshingiti, Z., Alaqel, R., Al-Muhtadi, J., Haq, Q. E. U., Saleem, K., and Faheem, M. H. (2023). A deep learning-based phishing detection system using cnn, lstm, and lstm-cnn. *Electronics*, 12(1) :232.

[Alsufyani and Alzahrani, 2021] Alsufyani, A. A. and Alzahrani, S. (2021). Social engineering attack detection using machine learning : Text phishing attack. *Indian J. Comput. Sci. Eng*, 12(3) :743–751.

[Arnold et al., 2024] Arnold, C., Biedebach, L., Küpfer, A., and Neunhoeffer, M. (2024). The role of hyperparameters in machine learning models and how to tune them. *Political Science Research and Methods*, 12(4) :841–848.

[Basiri et al., 2021] Basiri, M. E., Nemati, S., Abdar, M., Cambria, E., and Acharya, U. R. (2021). Abcdm : An attention-based bidirectional cnn-rnn deep model for sentiment analysis. *Future Generation Computer Systems*, 115 :279–294.

[Benavides-Astudillo et al., 2023] Benavides-Astudillo, E., Fuertes, W., Sanchez-Gordon, S., Nuñez-Agurto, D., and Rodríguez-Galán, G. (2023). A phishing-attack-detection model using natural language processing and deep learning. *Applied Sciences*, 13(9) :5275.

[Boser et al., 1992] Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152.

[Chen and Guestrin, 2016] Chen, T. and Guestrin, C. (2016). Xgboost : A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

[Chouaib, 2011] Chouaib, H. (2011). Sélection de caractéristiques : méthodes et applications. *Paris Descartes University : Paris, France*.

[Choudhary and Gianey, 2017] Choudhary, R. and Gianey, H. K. (2017). Comprehensive review on supervised machine learning algorithms. In *2017 International Conference on Machine Learning and Data Science (MLDS)*, pages 37–43. IEEE.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20 :273–297.

[Côté et al., 2024] Côté, P.-O., Nikanjam, A., Ahmed, N., Humeniuk, D., and Khomh, F. (2024). Data cleaning and machine learning : A systematic literature review.

[Dang et al., 2022] Dang, T., Sakai, Y., Tabaru, T., and Kasagi, A. (2022). Regularizing data for improving execution time of nlp model. *The International FLAIRS Conference Proceedings*, 35.

[David et al., 2021] David, R., Duke, J., Jain, A., Janapa Reddi, V., Jeffries, N., Li, J., Kreeger, N., Nappier, I., Natraj, M., Wang, T., et al. (2021). Tensorflow lite micro : Embedded machine learning for tinyml systems. *Proceedings of Machine Learning and Systems*, 3 :800–811.

[Devlin et al., 2019] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert : Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics : human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

[Do et al., 2022] Do, N. Q., Selamat, A., Krejcar, O., Herrera-Viedma, E., and Fujita, H. (2022). Deep learning for phishing detection : Taxonomy, current challenges and future directions. *Ieee Access*, 10 :36429–36463.

[Držík and Šteflovič, 2023] Držík, D. and Šteflovič, K. (2023). Text vectorization techniques based on wordnet. *Jazykovedny Casopis*, 74(1) :310–322.

[Egigogo et al., 2024] Egigogo, A. R., Idris, I., Olalere, M., and Aderiike, A. O. (2024). Evaluating deep learning models for website phishing attack detection : A comparative analysis. *Ceddi Journal of Information System and Technology (JST)*, 3(2) :19–29.

[Falade, 2023] Falade, P. V. (2023). Decoding the threat landscape : Chatgpt, fraudgpt, and wormgpt in social engineering attacks. *arXiv preprint arXiv :2310.05595*.

[First et al., 2024] First, B. M., Shanmugavadive, K., Sathishkumar, V. E., Maruthappa, M., Subramanian, M., and Thinakaran, R. (2024). Attention mechanism-based cnn-lstm for abusive comments detection and classification in social media text. *International Journal of Advanced Computer Science and Applications*, 15(10).

[Freund and Schapire, 1997] Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1) :119–139.

[Friedman, 2001] Friedman, J. H. (2001). Greedy function approximation : a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

[Gehl and Lawson, 2022] Gehl, R. W. and Lawson, S. T. (2022). *Social Engineering : How Crowdmasters, Phreaks, Hackers, and Trolls Created a New Form of Manipulativ e Communication*. MIT Press.

[Ghafir et al., 2018] Ghafir, I., Saleem, J., Hammoudeh, M., Faour, H., Prenosil, V., Jaf, S., Jabbar, S., and Baker, T. (2018). Security threats to critical infrastructure : the human factor. *The Journal of Supercomputing*, 74 :4986–5002.

[Ghannay et al., 2016] Ghannay, S., Favre, B., Estève, Y., and Camelin, N. (2016). Word embedding evaluation and combination. In Calzolari, N., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., and Piperidis, S., editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 300–305, Portorož, Slovenia. European Language Resources Association (ELRA).

[Grefenstette, 1999] Grefenstette, G. (1999). *Tokenization*, pages 117–133. Springer Netherlands, Dordrecht.

[Gururaj et al., 2024] Gururaj, H., Janhavi, V., and Ambika, V. (2024). *Social Engineering in Cybersecurity : Threats and Defenses*. CRC Press.

[Guyon and Elisseeff, 2003] Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar) :1157–1182.

[Hastie, 2009] Hastie, T. (2009). The elements of statistical learning : data mining, inference, and prediction.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Tibshirani, R., and Friedman, J. (2009). Random forests. *The elements of statistical learning : Data mining, inference, and prediction*, pages 587–604.

[Hossin and Sulaiman, 2015] Hossin, M. and Sulaiman, M. N. (2015). A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, 5(2) :1.

[Jamal and Wimmer, 2023] Jamal, S. and Wimmer, H. (2023). An improved transformer-based model for detecting phishing, spam, and ham : A large language model approach. *arXiv preprint arXiv :2311.04913*.

[Kamruzzaman et al., 2023] Kamruzzaman, A., Thakur, K., Ismat, S., Ali, M. L., Huang, K., and Thakur, H. N. (2023). Social engineering incidents and preventions. In *2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0494–0498. IEEE.

[Koide et al., 2023] Koide, T., Fukushi, N., Nakano, H., and Chiba, D. (2023). Detecting phishing sites using chatgpt. *arXiv preprint arXiv :2306.05816*.

[Koudri, 2021] Koudri, S. (2021). Algorithme pour un ensemble de données multi-étiquettes déséquilibré. Master's thesis, Université de Badji Mokhtar - Annaba, Algérie. Master II Informatique, Analyse et Gestion des données massives.

[Lamari et al., 2021] Lamari, M., Azizi, N., Hammami, N. E., Boukhamla, A., Cheriguene, S., Dendani, N., and Benzebouchi, N. E. (2021). Smote–enn-based data sampling and improved dynamic ensemble selection for imbalanced medical data classification. In *Advances on Smart and Soft Computing : Proceedings of ICACIn 2020*, pages 37–49. Springer.

[Lansley et al., 2020] Lansley, M., Lansley, M., Lansley, M., Mouton, F., Mouton, F., Kapetanakis, S., Kapetanakis, S., Pliatsikas, N., Polatidis, N., and Polatidis, N. (2020). Seader++ : Social engineering attack detection in online environments using machine learning. *null*.

[Lansley et al., 2019] Lansley, M., Polatidis, N., and Kapetanakis, S. (2019). Seader : A social engineering attack detection method based on natural language processing and artificial neural networks. In *Computational Collective Intelligence : 11th International Conference, ICCCI 2019, Hendaye, France, September 4–6, 2019, Proceedings, Part I 11*, pages 686–696. Springer.

[Leonov et al., 2021] Leonov, P. Y., Vorobyev, A. V., Ezhova, A. A., Kotelyanets, O. S., Zavalishina, A. K., and Morozov, N. V. (2021). The main social engineering techniques aimed at hacking information systems. In *2021 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT)*, pages 0471–0473.

[Mahanta and Maringanti, 2023] Mahanta, K. and Maringanti, H. B. (2023). Social engineering attacks and countermeasures. In *Perspectives on Ethical Hacking and Penetration Testing*, pages 307–337. IGI Global.

[Mezhoud, 2024] Mezhoud, B. (2024). *Analyse comparative des modèles de vectorisation (BERT, Word2Vec, FastText, GloVe) associés au SVM pour la détection de spam*. PhD thesis, Université du Québec à Trois-Rivières.

[Mitnick and Simon, 2003] Mitnick, K. D. and Simon, W. L. (2003). *The art of deception : Controlling the human element of security*. John Wiley & Sons.

[Örberg Kätterer, 2023] Örberg Kätterer, H. (2023). Sjösäkerhet i en digitaliserad sjöfart : En studie om effekter på sjösäkerheten i en framtida digitaliserad sjöfart.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn : Machine learning in python. *the Journal of machine Learning research*, 12 :2825–2830.

[Rathod et al., 2025] Rathod, T., Jadav, N. K., Tanwar, S., Alabdulatif, A., Garg, D., and Singh, A. (2025). A comprehensive survey on social engineering attacks, countermeasures, case study, and research challenges. *Information Processing & Management*, 62(1) :103928.

[Salahdine and Kaabouch, 2019] Salahdine, F. and Kaabouch, N. (2019). Social engineering attacks : A survey. *Future internet*, 11(4) :89.

[Tsinganos, 2023] Tsinganos, N. (2023). *Utilizing deep learning and natural language processing to recognise chat-based social engineering attacks for cyber security situational awareness*. PhD thesis, Πανεπιστήμιο Μακεδονίας. Σχολή Επιστημών Πληροφορίας. Τμήμα Εφαρμοσμένης ....

[Tsinganos et al., 2023] Tsinganos, N., Fouliras, P., and Mavridis, I. (2023). Leveraging dialogue state tracking for zero-shot chat-based social engineering attack recognition. *Applied Sciences*, 13(8) :5110.

[Tsinganos et al., 2024] Tsinganos, N., Fouliras, P., Mavridis, I., and Gritzalis, D. (2024). Cse-ars : Deep learning-based late fusion of multimodal information for chat-based social engineering attack recognition. *IEEE Access*, 12 :16072–16088.

[Tsinganos et al., 2022] Tsinganos, N., Mavridis, I., and Gritzalis, D. (2022). Utilizing convolutional neural networks and word embeddings for early-stage recognition of persuasion in chat-based social engineering attacks. *IEEE Access*, 10 :108517–108529.

[Tsinganos et al., 2018] Tsinganos, N., Sakellariou, G., Fouliras, P., and Mavridis, I. (2018). Towards an automated recognition system for chat-based social engineering

attacks in enterprise environments. In *Proceedings of the 13th International Conference on Availability, Reliability and Security*, pages 1–10.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

[Vennila et al., 2018] Vennila, G., Manikandan, M., and Suresh, M. (2018). Dynamic voice spammers detection using hidden markov model for voice over internet protocol network. *Computers & Security*, 73 :1–16.

[Wang and Ghaleb, 2023] Wang, Z. and Ghaleb, F. A. (2023). An attention-based convolutional neural network for intrusion detection model. *IEEE Access*, 11 :43116–43127.

[Wang et al., 2022] Wang, Z., Ren, Y., Zhu, H., and Sun, L. (2022). Threat detection for general social engineering attack using machine learning techniques. *arXiv preprint arXiv :2203.07933*.

[Wang et al., 2020] Wang, Z., Sun, L., and Zhu, H. (2020). Defining social engineering in cybersecurity. *IEEE Access*, 8 :85094–85115.

[Zaoui et al., 2024] Zaoui, M., Yousra, B., Yassine, S., Yassine, M., and Karim, O. (2024). A comprehensive taxonomy of social engineering attacks and defense mechanisms : Towards effective mitigation strategies. *IEEE Access*.