UNIVERSITY OF MACEDONIA

SCHOOL OF INFORMATION SCIENCES

DEPARTMENT OF APPLIED INFORMATICS

Utilizing Deep Learning and Natural Language Processing to Recognise Chat-Based Social Engineering Attacks for Cyber Security Situational Awareness.

Ph.D. Dissertation

Nikolaos Tsinganos

Thessaloniki Greece

2023

Supervisor

Mavridis Ioannis

Professor, Department of Applied Informatics, University of Macedonia

Advisory Committee

Mavridis Ioannis

Professor, Department of Applied Informatics, University of Macedonia

Fouliras Panagiotis

Assistant Professor, Department of Applied Informatics, University of Macedonia

Gritzalis Dimitris

Professor, Department of Informatics, Athens University of Economics and Business (AUEB)

Examination Committee

Mavridis Ioannis

Professor, Department of Applied Informatics, University of Macedonia

Fouliras Panagiotis

Assistant Professor, Department of Applied Informatics, University of Macedonia

Gritzalis Dimitris

Professor, Department of Informatics, Athens University of Economics and Business (AUEB)

Rantos Konstantinos

Associate Professor, Department of Computer Science, International Hellenic University

Stergiopoulos George

Assistant Professor, Department of Information and Communication Systems Engineering, University of the Aegean

Protopapadakis Eftychios

Assistant Professor, Department of Applied Informatics, University of Macedonia

Refanidis Ioannis

Professor, Department of Applied Informatics, University of Macedonia

to Aris…

“O Captain! my Captain!…”

~ Walt Whitman

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Professor Ioannis Mavridis, for his guidance, support, and encouragement throughout my PhD journey. I have learned so much from him, both academically and personally. I am grateful for his patience, wisdom, and help in shaping me into a better scientist and a better person.

I would also like to thank the members of my advisory committee, Professor Dimitris Gritzalis, and Assistant Professor Panagiotis Fouliras, for their valuable advice and feedback. Your insights and support have been invaluable to me.

I am deeply grateful to my wife, Anastasia, and my son, Aris, for their love and support throughout my PhD journey. You have always been there for me, even when things were tough. Your sacrifices and understanding during the long hours of research and writing are deeply appreciated. I am so lucky to have you in my life.

Thank you all.

ABSTRACT

In an era dominated by digital communication, the escalating threat of chat-based social engineering (CSE) attacks looms large. These attacks, characterized by manipulation, cheating, and psychological exploitation, pose a grave danger to individuals and organizations alike. To confront this burgeoning menace, this doctoral thesis presents an all-encompassing system for recognizing CSE attacks, under the banner of the Chat-based Social Engineering Attack Recognition System (CSE-ARS).

The foundation for this research is laid in an exhaustive exploration of the theoretical landscape. This comprehensive survey delves into the core concepts and principles essential for grasping the context of CSE attack recognition. Topics encompassed here range from the broader realm of cybersecurity, particularly in the context of social engineering, to the intricacies of the attack cycle and the profound impact of social engineering attacks. We further examine the pivotal role of advanced technologies such as artificial intelligence, deep learning, and natural language processing. Notably, this investigation scrutinizes the metrics used to evaluate the performance of recognition models, including accuracy, precision, recall, and the F1 score. The aim is to establish a strong theoretical grounding, emphasizing the significance of deep learning models in identifying and addressing the multifaceted challenges of CSE attacks.

The identified enablers of successful CSE attacks are then thoroughly examined. One key enabler lies in personality traits, as social engineers strategically exploit their understanding of human behavior to manipulate their targets. Understanding the dynamics of persuasion is also crucial for defense, with machine learning algorithms leveraged to recognize persuasive strategies and enhance resilience against CSE attacks.

Persistent behavior, including paraphrasing, is another central strategy used by social engineers to manipulate their targets. Recognizing and characterizing this behavior is crucial for developing effective defenses. Deception is a vital enabler, and investigating deception cues and developing machine learning models for recognition is an essential component of defense. Additionally, recognizing speech acts and the role of chat history in providing insights into the structure and context of conversations is emphasized. Deep learning models are deployed to enhance the accuracy of CSE attack recognition and prevention by studying speech acts and incorporating chat history analysis too.

The creation of the CSE Corpus serves as a fundamental resource for studying and understanding CSE attacks. This meticulous process begins with data source selection, dialogues collection, enrichment, linguistic analysis, and finally annotation. The CSE Corpus serves as a valuable asset for researchers and practitioners alike, facilitating the

development and evaluation of robust models and methodologies for recognizing and mitigating social engineering attacks.

Next, each enabler recognizer is introduced starting with a specialized recognition model, CRINL-R, for the identification of critical information leakage in CSE attacks. By employing deep learning techniques and a carefully curated dataset, CRINL-R demonstrates promising performance in identifying instances of critical information leakage.

Personality traits remain at the forefront of the investigation in the development of the PERST-R model. This model leverages a pre-trained BERT model and a rich corpus of labeled text data to specialize in the accurate recognition of individual traits. This recognition plays a pivotal role in understanding social engineering tactics and further fortifying defenses.

The recognition of persuasion techniques in CSE attacks takes center stage with the introduction of the PERSU-R model. This model integrates persuasion principles and convolutional neural networks to identify and categorize persuasive elements within textual interactions. Its efficacy in characterizing persuasion techniques contributes significantly to bolstering defenses against social engineering attacks.

Recognition of persistence in CSE attacks is addressed through the PERSI-R model, which leverages natural language processing techniques and neural networks. This model accurately identifies and characterizes persistence cues within textual interactions, underlining the significance of recognizing persistence as a critical factor in social engineering attacks.

The culmination of this research is presented with the introduction of the Chat-based Social Engineering Attack Recognition System (CSE-ARS). CSE-ARS leverages a late fusion approach to identify and recognize CSE attacks by combining multiple sources of information. By integrating individual recognizers specialized in different facets of CSE attacks, such as critical information leakage, personality traits, dialogue acts, persuasion techniques, and persistence, CSE-ARS achieves a comprehensive understanding of chat-based interactions. The system's performance is rigorously evaluated across various chat-based scenarios, demonstrating its potential real-world applicability.

This doctoral thesis endeavors to provide a comprehensive framework for recognizing and mitigating social engineering attacks in the realm of digital communication. The integration of deep learning techniques, multimodal information fusion, and ethical considerations underscores the potential for advanced defense mechanisms against the pervasive challenges of social engineering threats. This interdisciplinary approach empowers individuals and organizations to counteract these attacks effectively,

enhancing security and preserving personal and organizational integrity in the digital age. Future research may continue to refine and expand upon these models, contributing to practical deployment and wider adoption in real-world scenarios.

Keywords: Cybersecurity, Chat-based Social Engineering, Deep Learning, Machine Learning, Natural Language Processing, Transformers, Transfer Learning, Corpus, Annotation

ΠΕΡΙΛΗΨΗ

Σε μια εποχή που κυριαρχείται από την ψηφιακή επικοινωνία, η αύξηση των Επιθέσεων Κοινωνικής Μηχανικής βασισμένων σε Συνομιλίες (ΕΚΜΣ) είναι προδιαγεγραμμένη. Αυτές οι επιθέσεις, που χαρακτηρίζονται από την ψυχολογική εκμετάλλευση, και την εξαπάτηση, αποτελούν μια σοβαρή απειλή τόσο για τα άτομα όσο και για τις επιχειρήσεις. Για την αντιμετώπιση αυτής της αυξανόμενης απειλής , η παρούσα διδακτορική διατριβή παρουσιάζει ένα σύστημα αναγνώρισης επιθέσεων ΕΚΜΣ, υπό την αιγίδα του Συστήματος Αναγνώρισης Επιθέσεων Κοινωνικής Μηχανικής βασισμένων σε Συνομιλίες (CSE-ARS).

Τα θεμέλια αυτής της έρευνας τίθενται ξεκινώντας μια εκτενή εξερεύνηση του σχετικού θεωρητικού υπόβαθρου. Η έρευνα εξετάζει τις βασικές έννοιες και αρχές που είναι απαραίτητες για την κατανόηση του πλαισίου της αναγνώρισης των επιθέσεων ΕΚΜΣ. Τα θέματα που εξετάζονται εκτείνονται από τον ευρύτερο τομέα της κυβερνοασφάλειας αλλά με έμφαση στο πλαίσιο της κοινωνικής μηχανικής, μέχρι τις λεπτομέρειες του κύκλου μιας επίθεσης ΕΚΜΣ και τις επιπτώσεις της. Εξετάζεται επίσης ο κρίσιμος ρόλος των προηγμένων τεχνολογιών όπως η τεχνητή νοημοσύνη, η βαθιά μάθηση και η επεξεργασία φυσικής γλώσσας. Σημαντική είναι επίσης η εξέταση των μετρικών που χρησιμοποιούνται για την αξιολόγηση της απόδοσης των μοντέλων αναγνώρισης επιθέσεων ΕΚΜΣ. Στόχος είναι να δημιουργηθεί μια ισχυρή θεωρητική βάση, η οποία αναδεικνύει τη σημασία των μοντέλων βαθιάς μάθησης στην αναγνώριση και αντιμετώπιση των επιθέσεων ΕΚΜΣ.

Στη συνέχεια, εξετάζονται εξονυχιστικά οι παράγοντες που διευκολύνουν την επιτυχία των επιθέσεων ΕΚΜΣ. Ένας βασικός παράγοντας συναντάται στα χαρακτηριστικά της προσωπικότητας του ατόμου, καθώς οι κοινωνικοί μηχανικοί εκμεταλλεύονται στρατηγικά την κατανόησή τους για την ανθρώπινη συμπεριφορά προκειμένου να εξαπατήσουν τους στόχους τους. Η επίμονη συμπεριφορά, είναι άλλος ένας κεντρικός παράγοντας που χρησιμοποιούν οι κοινωνικοί μηχανικοί για να εκμεταλλευτούν τις αδυναμίες των στόχων τους. Η δυνατότητα αναγνώρισης της επίμονης συμπεριφοράς είναι κρίσιμη για την ανάπτυξη αποτελεσματικών μέτρων αμυντικής φύσης. Η εξαπάτηση αποτελεί έναν ακόμη κρίσιμο παράγοντα, και η ανάπτυξη μοντέλων μηχανικής μάθησης για την αναγνώριση της είναι ένα επιπλέον στοιχείο της άμυνας. Ακόμη, η αναγνώριση των πράξεων ομιλίας και ο ρόλος του ιστορικού των συνομιλιών είναι ιδιαίτερα σημαντικές. Τα μοντέλα βαθιάς μάθησης χρησιμοποιούνται για την ενίσχυση της ακρίβειας αναγνώρισης και πρόληψης των επιθέσεων ΕΚΜΣ μελετώντας τις πράξεις ομιλίας και συμπεριλαμβάνοντας την ανάλυση του ιστορικού των συνομιλιών.

Η δημιουργία του σώματος κειμένων CSE Corpus αποτελεί ένα θεμελιώδες εργαλείο για τη μελέτη και κατανόηση των επιθέσεων ΕΚΜΣ. Η διαδικασία παραγωγής του σώματος κειμένου αρχίζει με την επιλογή πηγών δεδομένων, τη συλλογή διαλόγων, τον εμπλουτισμό τους, τη γλωσσική ανάλυση και την τελική επισήμανση τους. Το σώμα κειμένων CSE Corpus αποτελεί ένα πολύτιμο εργαλείο για τους ερευνητές, διευκολύνοντας την ανάπτυξη και την αξιολόγηση αξιόπιστων μοντέλων και μεθοδολογιών για την αναγνώριση και την αντιμετώπιση των επιθέσεων ΕΚΜΣ.

Στη συνέχεια, οι ανιχνευτές για κάθε επιμέρους παράγοντα παρουσιάζονται ξεκινώντας με το εξειδικευμένο μοντέλο αναγνώρισης, ο ανιχνευτής CRINL-R, για την αναγνώριση της διαρροής κρίσιμων πληροφοριών. Με τη χρήση τεχνικών βαθιάς μάθησης και ενός προσεκτικά επιλεγμένου συνόλου δεδομένων, το CRINL-R εκπαιδεύεται και επιδεικνύει ελπιδοφόρα απόδοση στην αναγνώριση περιπτώσεων διαρροής κρίσιμων πληροφοριών.

Τα χαρακτηριστικά της προσωπικότητας παραμένουν στο επίκεντρο της έρευνας στην ανάπτυξη του ανιχνευτή PERST-R. Αυτό το μοντέλο εκμεταλλεύεται ένα προ-εκπαιδευμένο μοντέλο BERT και ένα πλούσιο σύνολο δεδομένων με ετικέτες για την ακριβή αναγνώριση των ατομικών χαρακτηριστικών προσωπικότητας. Αυτή η αναγνώριση παίζει κρίσιμο ρόλο στην κατανόηση των τακτικών της κοινωνικής μηχανικής και στην ενίσχυση των μέτρων αμυντικής φύσης.

Η αναγνώριση τεχνικών πειθούς στις επιθέσεις ΕΚΜΣ καταλαμβάνει την κεντρική θέση στον ανιχνευτή PERSU-R. Αυτό το μοντέλο αξιοποιεί νευρωνικά δίκτυα για την αναγνώριση και κατηγοριοποίηση στοιχείων πειθούς. Η αποτελεσματικότητά του στην αναγνώριση των τεχνικών πειθούς συμβάλλει σημαντικά στην ενίσχυση των μέτρων αμυντικής φύσης έναντι των επιθέσεων κοινωνικής μηχανικής.

Η αναγνώριση της επιμονής στις επιθέσεις ΕΚΜΣ αντιμετωπίζεται μέσω του ανιχνευτή PERSI-R, το οποίο χρησιμοποιεί τεχνικές επεξεργασίας φυσικής γλώσσας και νευρωνικά δίκτυα. Αυτό το μοντέλο αναγνωρίζει και χαρακτηρίζει με ακρίβεια στοιχεία επιμονής, υπογραμμίζοντας τη σημασία της αναγνώρισης της επιμονής ως κρίσιμου παράγοντα στις επιθέσεις κοινωνικής μηχανικής.

Η κορύφωση αυτής της έρευνας πραγματοποιείται με την παρουσίαση του Συστήματος Αναγνώρισης Επιθέσεων Κοινωνικής Μηχανικής βασισμένου σε Συνομιλίες (CSE-ARS). Το CSE-ARS χρησιμοποιεί μια προσέγγιση με χρήση της τεχνικής αργής συγχώνευσης πληροφοριών για την αναγνώριση των επιθέσεων ΕΚΜΣ, συνδυάζοντας τα συμπεράσματα των επιμέρους ανιχνευτών. Ενσωματώνοντας τους επιμέρους ανιχνευτές εξειδικευμένους στην αναγνώριση παραγόντων επιτυχίας μιας επίθεσης ΕΚΜΣ, όπως η διαρροή κρίσιμων πληροφοριών, τα χαρακτηριστικά της προσωπικότητας, οι πράξεις ομιλίας, οι τεχνικές πειθούς και η επιμονή, το CSE-ARS

επιτυγχάνει μια σφαιρική κατανόηση των ΕΚΜΣ. Η απόδοση του συστήματος αξιολογείται αυστηρά σε διάφορα σενάρια βασισμένα σε συνομιλίες, επιδεικνύοντας την πραγματική δυνατότητά του για εφαρμογή στον πραγματικό κόσμο.

Αυτή η διδακτορική διατριβή παράσχει ένα σύστημα για την αναγνώριση και τη μείωση των επιθέσεων κοινωνικής μηχανικής στον τομέα της ψηφιακής επικοινωνίας. Η ενσωμάτωση τεχνικών βαθιάς μάθησης, και συνένωσης πληροφοριών υπογραμμίζει τη δυνατότητα για προηγμένα μέτρα αμυντικής φύσης έναντι των επικρατούντων προκλήσεων και απειλών κοινωνικής μηχανικής. Αυτή η διεπιστημονική προσέγγιση αντιμετωπίζει αποτελεσματικά τις επιθέσεις ΕΚΜΣ, ενισχύοντας την ασφάλεια και διατηρώντας την εμπιστευτικότητα της πληροφορίας στην ψηφιακή εποχή. Μελλοντικές έρευνες μπορούν να συνεχίσουν να βελτιώνουν και να επεκτείνουν τα επιμέρους μοντέλα, συμβάλλοντας στην πρακτική εφαρμογή και την ευρύτερη υιοθέτησή τους σε πραγματικά σενάρια ΕΚΜΣ.

TABLE OF CONTENTS

Contents

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. Motivation and Research Questions

Social engineering is a versatile and complex phenomenon that appears in real life and the digital world as well. It is related to a manipulative form of communication that exploits human personality traits either in a mass personal or interpersonal way ((Gehl and Lawson 2022). From “fake news” to psychographic advertisements and from cognitive hacking to spear-phishing, there is a plethora of different types of social engineering attacks. Verizon’s key annual report (‘DBIR Report 2023 - Master’s Guide’), consistently states social engineering attacks as the most usual and hazardous vector by which malicious users gain access to secured and restricted computer networks. Contemporary practices of attackers have been studied thoroughly (Wang et al., 2021) and numerous solutions have been suggested within the academic domain as well as within the public sector.(Syafitri et al. 2022). Recently, Chat-based Social Engineering (CSE) attacks increased due to the widespread use of electronic medium communication tools that have been boosted due to the COVID-19 pandemic (Venkatesha, Reddy, and Chandavarkar 2021)and are now considered mainstream.

CSE attacks are tightly related to pretexting, a type of social engineering attack, in which a storyline is methodically planned out in advance and the attacker builds a persona with specific characteristics to approach the human target. The most known pretexts were created by Kevin Mitnick, a legendary social engineer, whose stories met wide media coverage and can be found in (K. Mitnick 2011). In CSE attacks, pretexts often exploit a simple fact: human personality has vulnerabilities that can be exploited broader using cultural dynamics, social stereotypes, and gender roles. The complexity of the phenomenon imposes an interdisciplinary approach to deal with the many different factors that are engaged.

Although several cyber-defense mechanisms have been proposed (Vishwanath 2022) to defend from social engineering attacks, most of the solutions focus on technical countermeasures to improve users’ protection. Being technical these mechanisms do not account for known CSE attack enablers as identified and illuminated in (Tsinganos et al. 2018). Currently, the majority of CSE attacks include phishing attacks, spear-phishing, social media scams, covid-19 scams, and whaling. The attackers’ favorite method of approach is impersonation where they pretend to be someone else to deceive and gain unauthorized access to sensitive information. Furthermore, the attackers recently use artificial technology techniques to create realistic videos and audio of individuals (deepfakes), which can be used to impersonate them in CSE attacks.

Chat-based social engineering attacks are a growing threat that can lead to various negative consequences, including financial loss, damage to reputation, loss of productivity, or other legal consequences. Financial loss can occur due to fraudulent transactions or data breaches, which can be costly for organizations. A successful CSE attack can damage the reputation of an organization, leading to a decline in business and loss of customer trust. Loss of productivity can occur when important data is lost, leading to the need for system rebuilding, and decreased efficiency. Legal and regulatory penalties can be incurred if an organization is held liable for data breaches. CSE attacks can target sensitive information essential for daily operations, such as financial data, personal data of employees and clients, and trade secrets. Given the significant impact of CSE attacks, it is crucial to develop effective recognition and prevention mechanisms. These mechanisms should be cost-effective, easy to use, and enable individuals and organizations to better protect themselves against imminent risks.

Thus, research towards the design and implementation of a system that recognizes CSE attacks is important for the following reasons:

1. **Protection against CSE attacks:** CSE attacks are a significant threat to organizations and individuals alike, as they exploit human vulnerabilities to gain unauthorized access to sensitive information or systems. An effective recognition system can help prevent such attacks and protect individuals and organizations against loss of critical information or financial resources.
2. **Recognition of Sophisticated CSE attacks:** CSE attacks are often sophisticated, and traditional security measures are not sufficient to recognize them. Research in this area can help identify new attack vectors and develop countermeasures to recognize and prevent these sophisticated attacks.
3. **Automation of Recognition:** Currently, the most common approach to recognizing CSE attacks is through manual inspection, which is time-consuming and error-prone. An automated recognition system can help reduce the workload on security personnel and improve the accuracy and speed of recognition.
4. **Advancements in Deep Learning:** Deep learning models have shown promising results in various domains, including natural language processing (NLP), image and speech recognition. The application of these models to CSE attack recognition can lead to more accurate and efficient recognition systems.
5. **Improved Security Posture:** Effective recognition systems can enhance the security posture of organizations and individuals against social engineering attacks, and help prevent data breaches and other security incidents.

6. **Rising Threat of Social Engineering:** With the increasing dependence on digital technologies and the growing sophistication of cybercriminals, CSE attacks have become more prevalent and sophisticated. Therefore, there is a pressing need to develop effective methods to recognize and prevent these attacks.
7. **Limitations of Traditional Security Measures:** Traditional security measures, such as firewalls and antivirus software, are essential for protecting against known cyber threats. However, they are not always effective in recognizing social engineering attacks, as these attacks rely on manipulating human behavior rather than exploiting technical vulnerabilities. Therefore, there is a need to complement traditional security measures with more advanced recognition techniques that can identify social engineering tactics.
8. **Need for Interdisciplinary Research:** CSE attacks recognition research requires a multidisciplinary approach that combines expertise in psychology, human behavior, cybersecurity, linguistics and data analytics. By bringing together researchers from different fields, we can gain a deeper understanding of the social engineering tactics used by attackers and develop more effective recognition techniques.
9. **Importance of Data Collection and Analysis**: CSE attacks recognition research requires large-scale data collection and analysis to identify patterns and trends in social engineering attacks. By analyzing the data collected from real-world attacks, researchers can develop more accurate and effective recognition techniques and inform the development of policies and guidelines to prevent these attacks.
10. **Potential for Positive Impact:** Effective CSE attack recognition techniques can have a positive impact on individuals, organizations, and society as a whole. By recognizing and preventing CSE attacks, we can protect sensitive information, prevent financial losses, and safeguard critical infrastructure. Additionally, research in this area can contribute to a greater understanding of human behavior and inform the development of more effective cybersecurity strategies.

Provided the context mentioned above, the following research questions arise:

RQ1: *What are the most common types of CSE attacks, and what are the existing methods for recognizing them?*

RQ2: *What are the most relevant features and data sources for CSE attack recognition, and how can they be extracted and pre-processed to train machine learning models?*

RQ3: *What are the most suitable machine learning models for CSE attack recognition, and how can they be trained and optimized?*

RQ4: *How different machine learning models can be combined in an automated CSE attack recognition system?*

RQ5: *How does the performance of the proposed CSE attack recognition system compare to existing CSE attack recognition methods, in terms of accuracy, efficiency, and false positive rates?*

RQ5: *What are the limitations of the proposed CSE attack recognition system, and what are the potential future research directions in this area?*

1.2. Aim and Objectives

This PhD thesis aims to identify the critical enablers of successful CSE attacks, develop and evaluate machine learning models to recognize them and combine them in an automated system able to recognize CSE attacks. Chat-based Social Engineering Attack Recognition System (CSE-ARS) is an ensemble deep learning-based late fusion system for CSE attack recognition. Thus, the research aims to enhance the effectiveness and accuracy of recognizing CSE attacks by leveraging the combined power of multiple individual enabler recognizers in chat-based communication.

To achieve the aim stated above, the following objectives were pursued:

1. *Investigate existing approaches in current literature*: Conduct a comprehensive review of the existing literature on CSE attack recognition and information fusion techniques. Analyze the strengths and limitations of current methodologies in order to identify gaps and opportunities for improvement.
2. *Design the CSE-ARS system*: Design a deep learning-based solution that incorporates late fusion of information, to effectively recognize CSE attacks.
3. *Collection and Building of corpus*: Develop a methodology to assemble a representative corpus of chat-based conversations containing instances of social engineering attacks. Standardize and apply text preprocessing techniques to clean the data, ensuring its quality and relevance for training and evaluation purposes.
4. *Implement and optimize the individual enabler recognizers*: Implement the appropriate deep learning models to facilitate enablers' recognition and train

them using the collected corpus. Optimize the models by fine-tuning hyperparameters and exploring techniques to improve their accuracy, robustness, and efficiency in recognizing different types of enablers.

5. *Evaluate and compare performance of the individual deep learning models*: Conduct extensive experiments and evaluations to assess the performance of individual deep learning models in terms of accuracy, precision, recall, and F1-score. Compare their results with existing approaches to validate their effectiveness in recognizing CSE attacks.
6. *Implement and optimize CSE-ARS system*: Design, implement and optimize the CSE-ARS system leveraging appropriate ensemble technique to combine the output of the individual deep learning models.
7. *Evaluate CSE-ARS system.* Assess thoroughly a wide range of CSE-attacks, to offer insights into the system's abilities and its potential practical use. The evaluation of CSE-ARS must reveal comprehensive outcomes in its ability to detect CSE attacks.
8. *Analyze limitations and challenges*: Identify the limitations and potential challenges associated with the implementation and deployment of the CSE-ARS in real-world scenarios. Investigate factors such as scalability, computational requirements, and ethical considerations.
9. *Provide recommendations and future directions*: Based on the findings of the research, provide recommendations for improving the performance and practical applicability of CSE-ARS. Outline potential future directions for research in the field of CSE attack recognition.

1.3. Contributions

The contributions of this thesis can be summarized as follows:

- The CSE ontology is an asset-oriented ontology as an asset adheres to cybersecurity ontology definitions (ISO 15408:2022), and connects social engineering concepts with cybersecurity concepts. It transforms abstract ideas, concepts, entities, and relations into tags and attributes to describe the related domain. This ontology, is helpful in categorizing hierarchies, by grouping similar in-context concepts and relations.
- The enablers of a successful CSE attack
    - Critical Information Leakage: Social Engineers attempt to extract information from their victims. (e.g., sensitive data or login credentials).

- Personality traits: Social engineers exploit personality traits to manipulate their victims.
- Dialogue Acts: In a manipulated conversation, a social engineer will utilize dialogue acts to force the victim in unwanted actions.
- Persuasion: In order to convince the victim to take specific actions, a social engineer will attempt several persuasion techniques.
- Persistence: Persistent behavior is encountered by social engineers who use various language patterns to repeat questions or commands in different words.
- Deception: Deceptive language is used order to capture the victim in a fictional scenario and lead him to unreasonable decisions.
- Chat History: A CSE attack may be a multi-stage attack. Thus, the full chat dialogue history is utilized by a social engineer before he unleashes his attack.

- A chat-based social engineering recognition system, known as CSE-ARS, is being proposed. This system adopts an interdisciplinary approach and considers various factors from a diverse set of disciplines such as Psychology (personality traits), Linguistics (dialogue-acts), Behavioral Science (persuasion, persistence), and Computer Science (Deep Learning, Natural Language Processing etc.).
- The implementation and evaluation of diverse deep learning models, such as recurrent neural networks (RNNs), convolutional neural networks (CNNs), and transformer-based models, is proposed for the recognition of individual CSE attack enablers. More specifically, the following enabler recognizers are presented:
  - Critical Information Leakage Recognizer (CRINL-R): a Long -Short Term Model (LSTM) recognizing critical data information leakage utilizing named entity recognition (NER) techniques.
  - Personality Traits Recognizer (PERST-R): a Bidirectional Encoder Representation for Transformers (BERT) model that predicts personality traits.
  - Dialogue-act Recognizer (DIACT-R): a BERT model able to recognize dialogue acts that can lead to deception taking into account dialogue history.
  - Persuasion Recognizer (PERSU-R): a CNN model that predicts persuasion attempts.

    - Persistence Recognizer (PERSI-R): a BERT model that predicts persistent behavior by identifying paraphrasing.

- The collection of the CSE Corpus, a corpus comprising both realized and fictional CSE attacks is introduced.
- A methodology for collecting, building, and annotating a corpus where the data sources are identified, dialogues are collected, enriched, linguistically analyzed, and processed. Moreover, after declaring the annotation task's goal, an abstract model and ontology are created, specifications are defined, annotator guidelines are produced, the annotation task is performed, the inter-annotator agreement is validated, and the final annotated corpus is established.
- During the research CSE Corpus was enhanced, annotated, and transformed in different ways for different downstream tasks. Thus, five different corpora were developed to be used for training and evaluating the different enabler recognizers and the CSE-ARS system.
- To substantiate the notion that various types of enablers are necessary for the successful execution of a CSE attack, a late fusion approach is proposed. This approach combines the final predictions from each enabler recognizer to generate a unified prediction, leveraging the unique strengths of each recognizer.
- A late fusion optimization approach is proposed, wherein the outputs of the individual recognizers are fed into a weighted linear aggregation. The weights for aggregation are optimized using the metaheuristic algorithm of simulated annealing and k-fold cross-validation technique.
- Recommendations and Future Directions:
    - Based on the findings and analysis, recommendations for improving the performance, robustness, and practical applicability of CSE-ARS are provided.
    - Potential future directions for research in CSE attack recognition were identified, such as exploring novel deep learning architectures (Large Language Models (LLMs) or considering real-time implementation challenges.

1.4. Overall Research Approach

To achieve the objectives outlined in section 1.2, the research followed a systematic and iterative approach, encompassing data collection and preparation, model development, training and optimization, evaluation, and analysis. The overall research approach is divided in three main phases as follows:

PHASE 1: CSE-ARS Design.

PHASE 2: CSE attack enabler recognizers implementation & evaluation.

PHASE 3: CSE-ARS ensemble model design, implementation & evaluation.

In each phase several tasks were accomplished as depicted in **Figure 1-1**. More specifically, the tasks per phase can be summarized as follows:

PHASE 1:

- Acquire CSE attack domain knowledge.
- Identify CSE attack attributes and enablers.
- CSE attack oriented corpus characteristics.
- Corpus Collection and Building:
    - Assemble a diverse and representative corpus of chat-based conversations, containing instances of CSE attacks.
    - Build and annotate the dataset by standardizing the data format, cleaning noise, and ensuring data quality and relevance for deep learning model.
- Propose CSE attack recognition system design

PHASE 2:

- Development of individual enabler recognizers (deep learning models):
    - CRINL-R (CRitical INformation Leakage Recogniser): Designed to identify critical information disclosure in chat-based communication.
    - PERST-R (PERSonality Traits Recognizer): Aimed at recognizing personality traits of individuals engaged in the conversation, which may indicate susceptibility to social engineering attacks.

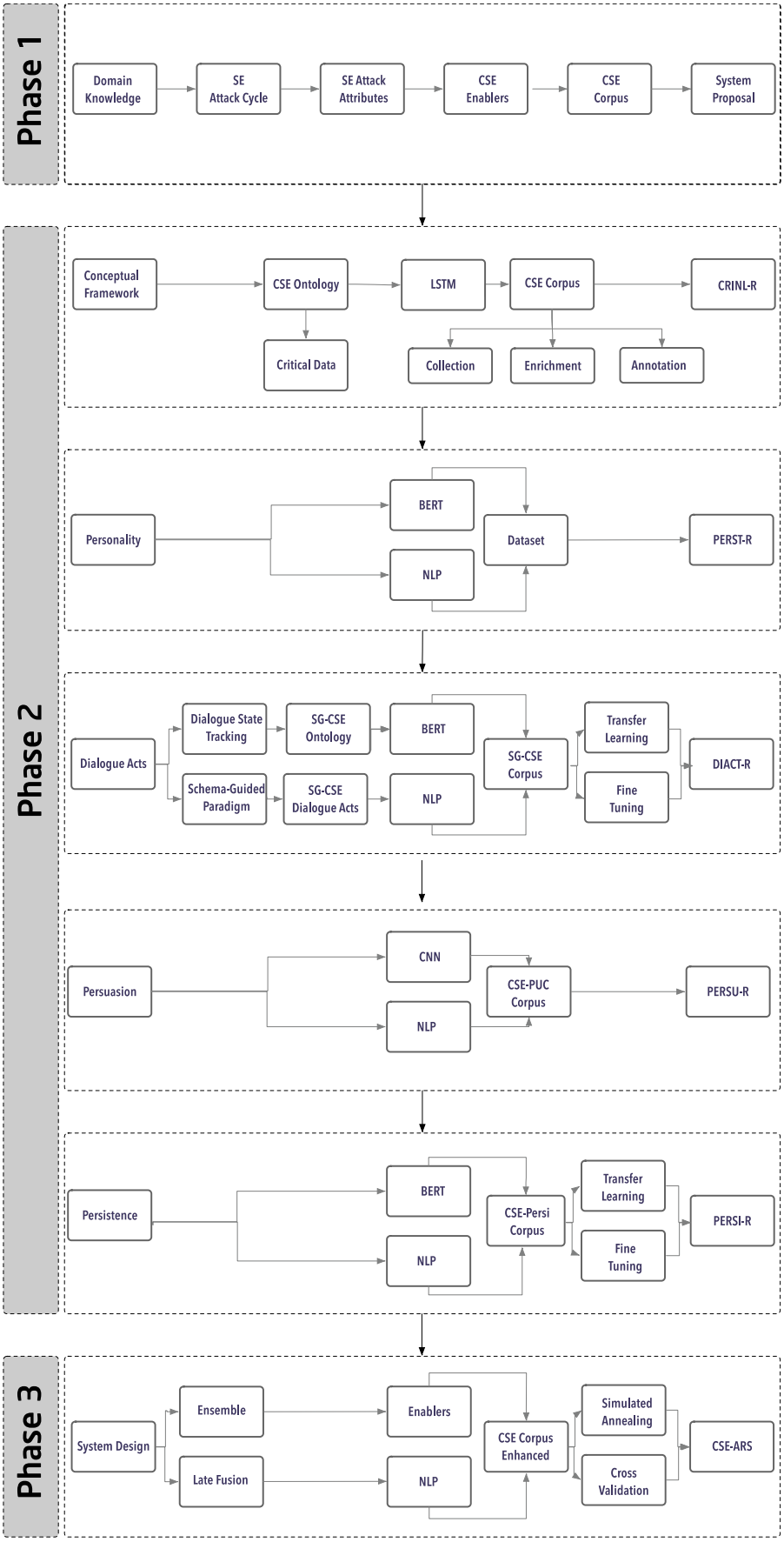

***Figure 1-1**. Research roadmap*

- DIACT-R (DIalogue ACTs Recogniser): Developed to identify different dialogue acts, such as requests for sensitive information or attempts to build rapport.
- PERSU-R (PERSUasion Recogniser): Focused on recognizing persuasive language or manipulative techniques used in social engineering attacks.
- PERSI-R (PERSIstence Recogniser): Designed to recognize persistent attempts made by the attacker to extract sensitive information.

- For each individual enabler recognizer, the following steps were taken:
  - Training and Optimization:
    - Training of deep learning model using the appropriate preprocessed corpus, employing appropriate optimization techniques and loss functions specific to each model's objective.
    - Fine-tuning of hyperparameters and exploration of techniques such as transfer learning or ensembling to enhance the performance and generalization capabilities of the models.

PHASE 3:

- Integration and Late Fusion:
  - Integrate the outputs of the developed deep learning models, combining their predictions using a late fusion approach to leverage the multimodal information effectively.
  - Explore fusion strategies, such as weighted averaging or decision-level fusion, to combine the outputs of individual models into a unified decision-making framework.
- Evaluation and Analysis:
  - Conduct comprehensive evaluations of the CSE-ARS system by assessing its performance metrics, including accuracy, precision, recall, and F1-score.
  - Compare the results of CSE-ARS with existing approaches to evaluate its effectiveness in recognizing CSE attacks.

- Analyze the limitations, challenges, and potential biases associated with the developed models and propose strategies for mitigating them.

1.5. Thesis Structure

The rest of this thesis is structured as follows:

- Chapter 2, “Theoretical Background”, delves into the relevant theoretical foundations, including social engineering, artificial intelligence, deep learning, and natural language processing.
- Chapter 3, “Conceptual Framework”, presents all the in-context information regarding the CSE attack and its enablers setting the stage for the subsequent chapters.
- Chapter 4, "Related Work” critically reviews and synthesizes existing research and literature relevant to the thesis topic. It provides a comprehensive overview of the current state of knowledge in the field, highlighting key studies, methodologies, and gaps in the literature. This section serves as the foundation for the thesis, helping to contextualize the research within the broader academic landscape and demonstrating the need for the proposed study.
- Chapter 5, “CSE Corpus” delves into the CSE Corpus, a pivotal resource for studying and combating social engineering attacks. It outlines the meticulous steps taken in its creation, including source selection, dialogue collection, enrichment, linguistic analysis, and annotation.
- Chapter 6, “Critical Information Leakage Recognizer” explores the CRINL-R model for CSE attack recognition. Evaluation results demonstrate its potential in flagging CSE attacks, and enhancing overall security.
- Chapter 7, “Personality Traits Recognizer” focuses on the development and assessment of the PERST-R model, aimed at identifying personality traits in CSE attacks. The chapter offers a comprehensive view of the model, describing its use of a pre-trained BERT model, the fine-tuning process, corpus selection, training methods, and evaluation metrics.
- Chapter 8, “Dialogue Acts Recognizer” delves into the development and evaluation of the DIACT-R model for recognizing dialogue acts in the context of CSE attacks. The chapter emphasizes the schema-guided paradigm, dialogue state tracking, and the SG-CSE BERT model architecture.
- Chapter 9, “Persuasion Recognizer’ delves into the development and assessment of the PERSU-R model for recognizing persuasion techniques in CSE attacks. The chapter highlights the model’s incorporation of persuasion principles, and the use of Convolutional Neural Networks (CNNs).

- Chapter 10, “Persistence Recognizer” details the development and assessment of the PERSI-R model for recognizing persistence in CSE attacks. The chapter explores the model’s use of Natural Language Processing (NLP) techniques, Neural Networks, and the model’s architecture.
- Chapter 11, “Chat-based Social Engineering Attack Recognition System” presents the CSE-ARS, presenting its architecture, the training procedure involving late fusion and simulated annealing, and the achieved results.
- Finally, Chapter 12, titled "Conclusions and Future Work," offers overall conclusions based on the findings, identifies potential avenues for future research, and concludes the thesis.
- The appendix section and the list of publications and references follow the main chapters, providing additional supplementary information and resources.

2. THEORETICAL BACKGROUND

2.1. Introduction

Social engineering attacks pose significant challenges in today's digital landscape. CSE attacks, in particular, exploit human vulnerabilities to manipulate individuals into divulging sensitive information or performing unintentional actions. Addressing this critical issue requires the development of advanced techniques that can recognize and mitigate such attacks effectively. This Ph.D. thesis aims to contribute to the field of cyber security and social engineering prevention by proposing CSE-ARS (Chat-based Social Engineering Attack Recognition System), a novel deep learning-based system that integrates information for accurate CSE attack recognition. By leveraging the power of deep learning models and fusing various enablers, including critical information leakage, personality traits, dialogue acts, persuasion, and persistence, the research seeks to enhance the recognition and prevention capabilities of social engineering attacks.

The thesis begins with an exploration of the theoretical background necessary to understand the complexities of cyber security in the context of social engineering, artificial intelligence (AI), deep learning (DL), and natural language processing (NLP). This foundation sets the stage for the subsequent chapters, where specific deep learning models are developed to tackle different aspects of CSE attacks.

2.2. Social Engineering

Social engineering is a tactic used by attackers to manipulate individuals into divulging sensitive information or performing actions that may compromise cyber security (Gehl and Lawson 2022; Hadnagy 2010) .It is a common tactic used in cyber-attacks, and it relies on exploiting human psychology and behavior. Social engineering attacks can take various forms, including phishing, pretexting, baiting, and quid pro quo. These attacks can be conducted through various channels or attack surfaces such as chat, email, phone, or in-person interactions. Recent years have seen an increase in CSE attacks ('DBIR Report 2023 - Master's Guide'), and they are becoming more sophisticated and harder to recognize.

Traditional technical security measures such as firewalls and antivirus software are not effective in preventing social engineering attacks, and there is a growing need for new techniques to recognize and prevent such attacks. Machine learning and deep learning techniques have been utilized as a solution for recognizing social engineering attacks.

These techniques have been shown to be effective in identifying patterns and anomalies in text, which can be used to recognize CSE attacks. Recognition methods may involve natural language processing to identify specific queries, commands, or predefined blacklisted topics.

Attackers often use social networking sites to gather information and manipulate their targets. Automated data retrieval from semi-structured web pages is a common strategy used to approach targets with useful information. To achieve high recognition accuracy and low false negative rates, it is crucial to include as many influencing factors as possible, such as psychological profiles of interlocutors, persuasion techniques etc. However, these techniques alone do not sufficiently recognize social engineering attacks, and there is a need for an ensemble approach that combines the results of multiple recognizers. The overall recognition problem can be decomposed, and averaging the weight of several factors for efficient modelling and effective inference methods.

A very common and dangerous type of social engineering attack is Spear phishing, which targets a specific victim and compromises her confidential data to get access into a sensitive system. Spear phishing attacks may be unleashed by sending malware or sending a URL link to the targets using fake identities to manipulate the victim but also through chat-based software. Social engineers know that humans are the weakest links in cyber security and they can put an organization at risk of a cyber-attack. By targeting employees with various social engineering techniques, they can manipulate their target to provide sensitive data.

A social engineering attack is mainly related to manipulation and concerns human behavior, making it difficult to precisely predefine and recognize it by only syntactic or semantic analysis of the chat messages. Furthermore, human language ambiguity makes discriminating a sentence as malicious or not, even harder. To cope with this challenge, a researcher has to employ a toolkit (e.g., machine learning tools) to process all available data and to infer in a probabilistic manner whether a conversation contains elements indicative of a social engineering attack.

2.2.1. Attack Lifecycle & Attributes

In a typical social engineering attack, the attacker acts in a predetermined manner, where she initially gathers information using every possible technique or tool, then approaches the potential victim and develops a trust relationship. Next, she exploits this trust relationship to manipulate the victim to perform an action that would enable her to violate the respective information system. At the final stage, the attacker reaches her original target violating a CIA triad member (confidentiality, integrity, availability) of

informational resources. This lifecycle was formally described in (K. D. Mitnick and Simon 2011) as having the following phases:

- *Reconnaissance*: The attacker gathers information about the target, such as their name, job title, and contact information. This can be done through publicly available information or by using social media or other online platforms.
- *Building trus*t: The attacker establishes a relationship with the target by posing as a trustworthy individual or organization. They may use tactics such as creating a sense of urgency or offering a reward to gain the target's trust.
- *Attac*k: The attacker uses the information and trust they have gained to launch the attack. This can take many forms, such as tricking the target into disclosing sensitive information, convincing them to click on a malicious link or open a malware-laden attachment, or even gaining physical access to a facility.
- *Consolidatio*n: After the attack is successful, the attacker consolidates her gains. This may involve exfiltrating stolen information, using the information to gain further access to the target's systems, or even selling the information on the dark web.
- *Cover-up*: The attacker covers their tracks to avoid recognition and make it difficult to trace the attack.

Not all social engineering attacks have all these phases and also the order of the phases may vary depending on the scenario. Additionally, some attacks can be launched in a very short time, while others may take months or even years to be completed.

Figure 2-1 depicts the different attributes of a social engineering attacks namely actor, approach, method, route, technique and distribution method. This thesis targets the attributes in box.

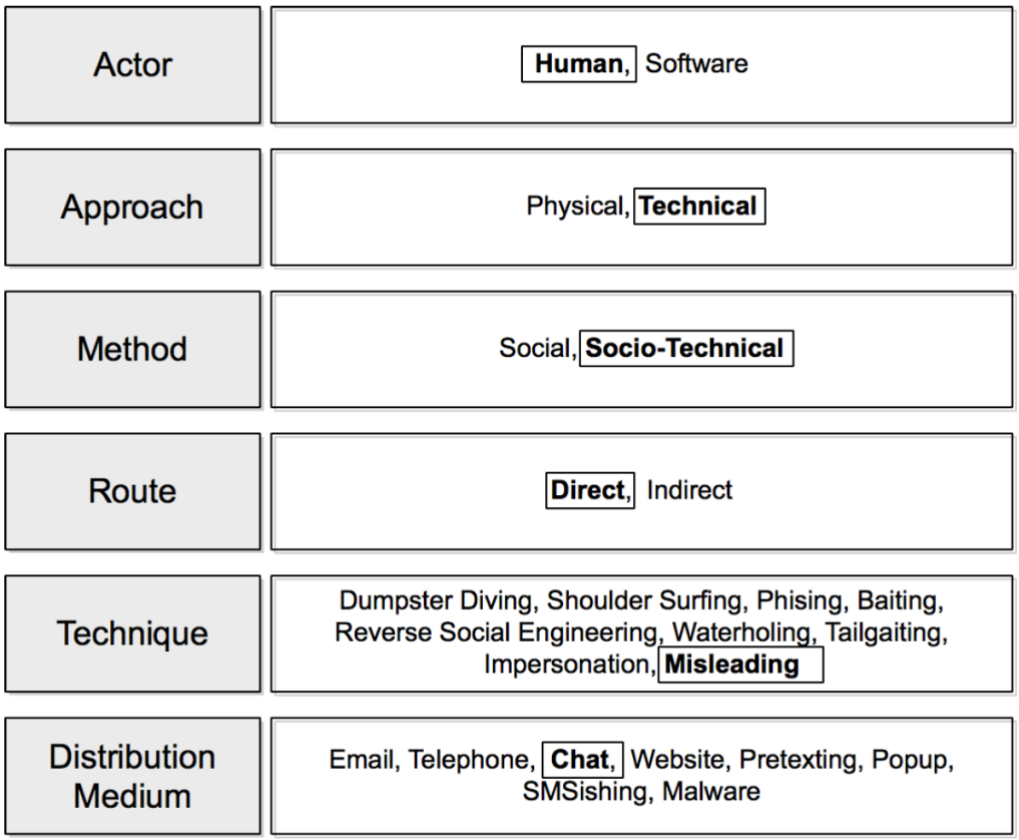

***Figure 2-1**. Social Engineering Attributes*

2.2.2. CSE Attack

Chat-based social engineering attacks are manipulative tactics employed by malicious actors to exploit human psychology, trust, and vulnerabilities in various forms of online communication channels, such as messaging apps, email, or social media messaging platforms. These attacks aim to manipulate individuals into divulging sensitive information, performing actions, or making decisions that can compromise their security, privacy, or financial well-being. An example of a typical CSE attack, that borrows phases and attributes from section 2.2.1 follows:

1. Initial Contact: The attacker creates a fake profile on a popular messaging app, posing as a colleague, friend, or a trustworthy entity. They carefully choose a profile picture and display a name to mimic the person they are impersonating.
2. Building Trust: The attacker initiates a conversation with the target, often with a friendly greeting. They may also provide some context to make their approach seem legitimate. For instance, they might say, “Hey [Target’s Name], it’s been a while. I wanted to catch up.”

3. Creating Urgency: To manipulate the target into acting quickly, the attacker fabricates a situation that demands immediate attention. They might claim that the target's account has been compromised, or that there's an urgent matter that requires confidentiality. This urgency puts pressure on the target to respond promptly.
4. Request for Information: The attacker eventually leads the conversation to the main objective—extracting sensitive information. They might say, "I need to verify your identity. Can you please confirm your date of birth and the last four digits of your Social Security Number?"
5. Leveraging Psychological Tricks: To make the target more compliant, the attacker may employ psychological tactics. They could use flattery, create a sense of fear or excitement, or appeal to the target's desire to help a friend in need.
6. Establishing Credibility: The attacker might claim to have access to sensitive information or details about the target's life, gained through social media stalking or other means. This can make the impersonation more convincing.
7. Encouraging Action: Once they have acquired the desired information, the attacker might ask for additional favors, such as clicking on a malicious link or transferring money. They could claim it's related to the urgent matter they previously mentioned.
8. Concealing Identity: Throughout the conversation, the attacker takes measures to hide their true identity, such as using proxy servers or disposable phone numbers, making it challenging to trace them.
9. Termination: After achieving their goals or sensing suspicion from the target, the attacker may abruptly end the conversation to avoid detection. They may also block the target or delete their fake profile to cover their tracks.

If successful, this CSE attack can have a serious impact on the victim.

2.2.3. Impact

Small-medium enterprises (SMEs) are particularly vulnerable to CSE attacks due to their limited resources and lack of cyber security expertise. Social engineering attacks can have a significant impact on SMEs, including:

- Financial Loss: Social engineering attacks can result in financial loss for SMEs. This can include the loss of company funds through fraudulent transactions or the loss of customer data, which can result in costly data breaches.

- Loss of Reputation: A successful social engineering attack can damage the reputation of an SME. This can lead to a loss of customer trust and a decline in business.
- Loss of Productivity: Social engineering attacks can disrupt the normal operation of an SME. This can include the loss of important data or the need to rebuild systems, which can result in a loss of productivity.
- Legal Consequences: SMEs can be held liable for data breaches and can face significant legal and regulatory penalties.

For SMEs to mitigate the risks of social engineering attacks several actions can be taken like educating employees about social engineering tactics, implementing cyber security controls, and regularly monitoring for suspicious activity.

2.3. CSE Attack Enablers

For the attacker to develop a trust relationship, she relies on specific human (victim) personality traits treating them as vulnerabilities and adapting her tactics accordingly. She aims is to influence the victim's way of thinking, and to persuade him to behave mistakenly. The act of deception is underlying throughout the at-tacker's effort. A communication scenario between the attacker and her victims involves message exchange through an electronic chat system. This is the point where the effort of this thesis on recognizing social engineering attacks is focusing.

For a successful CSE attack several enablers must be triggered. By definition (Bernard 2012) an enabler is something that enables or facilitates an action. In the context of CSE attacks, enablers refer to the various tactics and techniques used by attackers to trick their targets into disclosing sensitive information or performing certain actions. These enablers such as persuasion techniques, deception techniques, critical information leverage tricks, paraphrasing methods, and specific dialogue acts, which are often used in combination to increase the chances for a successful CSE attack. Recognition, in early stage, of these enablers is important to recognize and prevent CSE attacks. In (Tsinganos et al. 2018) and (Tsinganos, Fouliras, and Mavridis 2023; 2022; Tsinganos, Mavridis, and Gritzalis 2022; Tsinganos and Mavridis 2021) the following enablers were identified:

- Critical information leakage: Social engineers may attempt to extract sensitive or confidential information from their targets.

- Personality traits: Social engineers may attempt to exploit certain personality characteristics of their targets to gain their trust. For example, they may appeal to a person's sense of authority or their need for social validation.
- Dialogue acts: Social engineers may use certain dialogue acts, such as questions or commands, to elicit information or manipulate the conversation.
- Persuasion attempts: Social engineering attacks often involve attempts to persuade the target to take a certain action, such as providing personal information or clicking on a link.
- Persistence attempts: Social engineers may attempt to paraphrase information they've already obtained in order to confirm its accuracy or to obtain additional information.
- Deception attempts: Social engineering attacks often involve deception, such as pretending to be someone else or providing false information.
- Chat Dialogue History: As said, many CSE attacks are performed in several stages and it is very important for a recognizer mechanism to have access to the full dialogue history in order to make inferences.

Deception attempts and chat dialogue history can easily be utilized in accordance with dialogue acts recognition.

2.4. Artificial Intelligence & Machine Learning

Artificial Intelligence (AI) is an exciting field with many practical applications and many more active research topics. During the last years, scientists relied on AI to tackle with problems difficult for humans to solve but straight-forward to be solved by machines (Russell and Norvig 2022). Historically, and due to the interdisciplinary nature of AI, there was a debate about how scientists should approach AI, meaning to build a machine that acts or a machine that thinks intelligently. Many disciplines such as Philosophy, Mathematics, Neuroscience, Psychology and of course Computer Science have contributed to that debate and to what AI is today. The modern AI approach that prevailed is focused on building rational agents, which means intelligent entities that act so as to achieve the best outcome or, when there is uncertainty, best expected outcome.

AI is a vast and universal scientific field and it encompasses several subfields such as Machine Learning, Computer Vision, Robotics, Expert Systems, Speech Processing, Natural Language Processing and others. Machine learning is a subfield of AI that studies the ability to improve the agent's performance based on experience. A statement

credited to Tom Mitchel (Mitchell 1997) defines ML as “A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.”. This statement still holds true and it is the effort to improve this performance measure P that led the scientific community to explore new directions.

2.5. Deep Learning & Neural Networks

In recent years, deep learning (Goodfellow, Bengio, and Courville 2016) has emerged as a powerful paradigm in the field of artificial intelligence, revolutionizing various domains by enabling machines to automatically learn intricate patterns and representations from large-scale data. Deep learning models offer the advantage of capturing complex patterns and dependencies in multimodal data, including text, speech, and behavioral cues.

At their core, deep learning models consist of multiple layers of interconnected nodes, known as artificial neurons or units. These layers are organized in a hierarchical fashion, with each layer extracting increasingly complex features from the input data. The input to a deep learning model is typically a high-dimensional data representation, such as images, text, or audio. Each input feature is assigned a weight, and these weights are learned during a training phase to optimize the model’s performance. The first layer of the model, known as the input layer, receives the raw input data and passes it through to the subsequent layers.

As the data flows through the network, each neuron in a given layer computes a weighted sum of the inputs it receives, including the weighted outputs from the previous layer. This sum is then passed through an activation function, which introduces non-linearity and allows the model to learn complex relationships and patterns within the data. The output of each neuron in a layer becomes the input for the neurons in the next layer, and this process is repeated until the data reaches the final layer, known as the output layer. The output layer produces the model’s prediction or classification based on the learned weights and patterns. During training, the model’s predictions are compared to the true values of the training data, and an optimization algorithm, such as gradient descent, is used to adjust the weights iteratively, minimizing the difference between predicted and true values.

One of the key strengths of deep learning models is their ability to automatically learn hierarchical representations of data. By having multiple layers, each layer can learn increasingly abstract and complex features. Lower layers typically capture low-level features, such as edges or textures in images, while higher layers learn more abstract representations, such as shapes or objects. This hierarchical feature extraction enables

deep learning models to excel at tasks such as image recognition, natural language processing, and speech recognition.

Overall, deep learning models leverage the power of interconnected layers of artificial neurons, trainable weights, and non-linear activation functions to learn complex patterns and relationships within high-dimensional data. Through an iterative training process, these models can make accurate predictions, recognize patterns, and generalize well to unseen data, making them a valuable tool in various domains, including computer vision, natural language processing, and cyber security.

Using deep learning models for natural language processing tasks is a relatively new paradigm that recently presented breakthrough results in many NLP tasks such as part-of-speech-tagging, parsing, text classification and others. Since then, a massive interest has been generated applying deep learning models to several different domains, including the cybersecurity domain.

The deep learning models proposed in this thesis are trained using carefully constructed corpora derived from CSE Corpus, which represents real-world chat dialogues involving social engineering attacks. This corpus captures the diverse nature of attacks, incorporating different social engineering techniques, linguistic patterns, and contextual variations. The availability of such annotated data facilitates the training and evaluation of the deep learning models, enabling the exploration of nuanced attack strategies and the development of effective defense mechanisms. **Figure 2-2** presents a typical deep learning training pipeline and training dataset split where X+Y+Z=100.

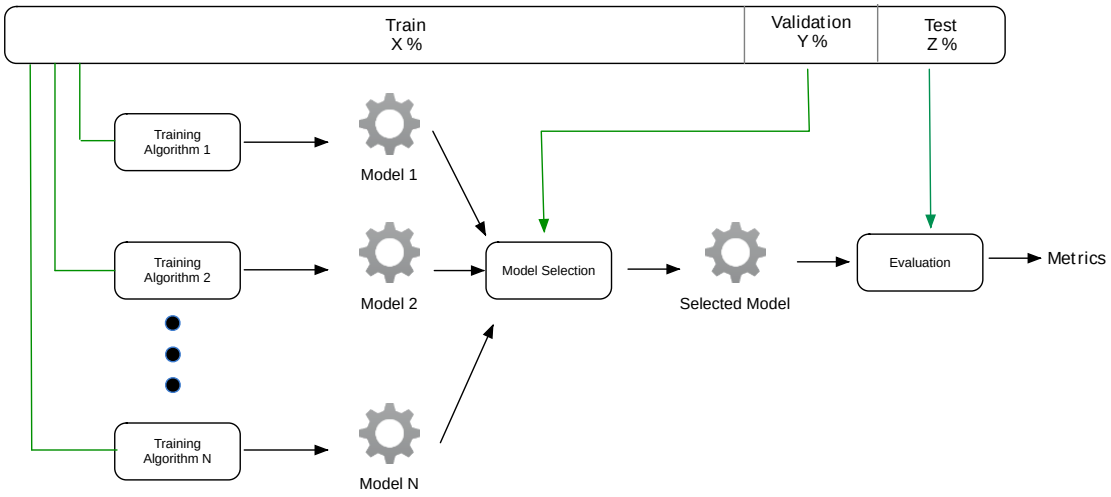

***Figure 2-2**. Train/Validation/Test split and the training pipeline*

Recognition tasks are common to cybersecurity domain where threats must be recognized in order to protect valuable assets whether the input is numeric (i.e., network

addresses) or unstructured text (i.e., chat-based dialog). For example, deep learning can discover complex structure existing in a corpus and using a learning algorithm (i.e., backpropagation) can indicate how a model should change its internal parameters that are used to compute the representation in each layer from the representation in the previous layer. There are several cybersecurity recognition problems that can be casted to text classification problems such as those mentioned in (Tsinganos et al. 2018). More specifically, deception recognition, persuasion recognition, personality traits recognition, dialogue-acts recognition, and persistence recognition can be seen as text classification problems where the input is a sentence written by one of two interlocutors in a chat-based client.

2.6. Natural Language Processing

Natural Language Processing (NLP) is a subfield of artificial intelligence and an intersection with computational linguistics that focuses on enabling computers to understand, interpret, and generate human language. It encompasses a wide range of techniques, algorithms, and models that facilitate the interaction between humans and machines using natural language, such as speech or text. The goal of NLP is to bridge the gap between human communication and machine understanding, enabling computers to process, analyze, and generate language in a way that is meaningful and useful.

NLP involves various tasks, including but not limited to, natural language understanding (NLU) (Jurafsky and Martin 2022), natural language generation (NLG) (Jurafsky and Martin 2022), information extraction (Clark, Fox, and Lappin 2013), sentiment analysis (Dos Santos and Gatti 2014), machine translation (Luong, Pham, and Manning 2015), and question answering (Chakravarty 2019). NLU involves the comprehension of human language, enabling machines to extract meaning, identify entities, recognize grammatical structures, and understand the intent behind user queries. NLG, on the other hand, focuses on generating human-like language, allowing machines to produce coherent and contextually appropriate responses or written content.

To achieve these tasks, NLP employs a range of techniques and methodologies. One approach is rule-based systems, where explicit rules and linguistic patterns are defined to handle language processing tasks. Another approach is using statistical methods, which utilize large amounts of annotated data to train probabilistic models that capture patterns and associations within the language. More recently, deep learning techniques, such as recurrent neural networks (RNNs) and transformers, have demonstrated remarkable success in NLP tasks by leveraging the power of neural networks to learn complex language representations.

NLP faces numerous challenges due to the inherent complexity and ambiguity of natural language. These challenges include dealing with variations in sentence structure, understanding context and semantics, resolving references, handling idiomatic expressions, and addressing the nuances of sentiment and tone. Researchers and practitioners in NLP continually strive to develop innovative algorithms and models that can overcome these challenges and improve the accuracy and effectiveness of language processing tasks.

Thus, natural language processing is a field dedicated to developing computational methods that enable computers to understand, analyze, and generate human language. By leveraging techniques from linguistics, artificial intelligence, and machine learning, NLP plays a crucial role in facilitating human-computer interaction, powering applications such as chatbots, virtual assistants, sentiment analysis systems, machine translation services, and more. As the demand for efficient and natural language communication grows, advancements in NLP continue to shape the way we interact with and benefit from technology.

2.7. DL Models Evaluation

The evaluation of deep learning models is essential to assess their performance and effectiveness in solving specific tasks. In the context of this thesis, where the focus lies on developing deep learning models for CSE attack recognition, the selection and application of appropriate metrics are of paramount importance. This section introduces the concept of metrics and their significance in evaluating the performance of the proposed models.

Metrics serve as quantitative measures that allow for the objective assessment of model performance. They provide insights into various aspects of model behavior, such as accuracy, precision, recall, and F1 score, among others. In the context of CSE attack recognition, metrics help determine the models' ability to correctly identify and classify different types of attacks, while minimizing false positives and false negatives.

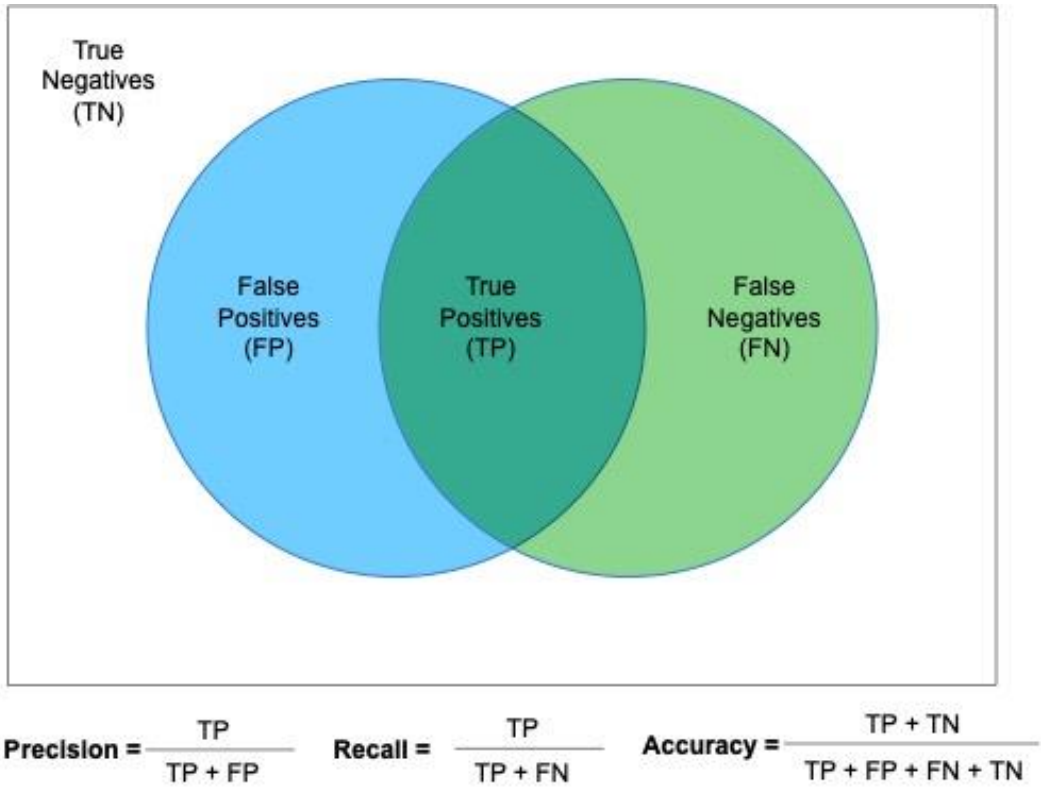

Figure 2-3. *Precision, Recall, and Accuracy*

Several performance metrics are employed to evaluate the quality of the proposed deep learning models or systems and assess their ability to make accurate predictions on new, unseen data. More specifically, throughout the experiments, the following performance metrics were used:

- *Receiver Operating Characteristic (ROC)* is a graphical representation of the performance of a binary classifier model at different classification thresholds. The ROC curve plots the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings, where the TPR is the proportion of actual positive instances that are correctly identified by the model, and the FPR is the proportion of negative instances that are incorrectly identified as positive.
- *Accuracy* is a measure of the proportion of correct predictions made by the model, expressed as a percentage. It is calculated as the ratio of the number of correct predictions to the total number of predictions made by the model.
- *Precision* is a measure of the proportion of true positive predictions among all positive predictions made by the model. It is calculated as the ratio of the number of true positive predictions to the total number of positive predictions made by the model.

- *Recall*, also known as sensitivity, is a measure of the proportion of true positive predictions among all actual positive instances. It is calculated as the ratio of the number of true positive predictions to the total number of actual positive instances
- *F1*, The F1 metric is an evaluation measure that combines precision and recall to assess the performance of a binary classification model. It provides a single numerical value that represents the harmonic mean of precision and recall, giving equal importance to both measures. The F1 metric is particularly useful when there is an imbalance between the positive and negative classes in the dataset.

Where appropriate additional metrics are introduced and utilized in place.

2.8. Chapter Conclusion

Chapter 2 provided a comprehensive exploration of the theoretical background necessary for understanding the research context of CSE attack recognition. Various key aspects were discussed, encompassing topics such as cyber security in the context of social engineering, the attack cycle and the impact of social engineering attacks, the role of artificial intelligence, deep learning and neural networks, natural language processing, and metrics for evaluation.

Throughout this chapter, the significance of comprehending the fundamental principles and concepts associated with CSE attacks and the utilization of deep learning models for their recognition and identification was emphasized. Through the examination of the attack cycle and the potential impact of these attacks, valuable insights were acquired regarding the severity and significance of the problem at hand. Furthermore, an exploration was conducted on the integration of artificial intelligence, deep learning, and natural language processing techniques to tackle the challenges involved in recognizing and mitigating CSE attacks.

In addition, a detailed discussion was held regarding the metrics employed to evaluate the performance of the proposed models. These metrics, such as accuracy, precision, recall, and F1 score, enable a quantitative assessment of the models' capabilities in recognizing and classifying different types of attacks. Moreover, metrics that take into account the multimodal nature of the data were considered, encompassing speech, text, and behavioral cues. Additionally, metrics related to sentiment analysis and dialogue acts alignment were also taken into consideration.

Chapter 2 provided the necessary theoretical foundation for understanding the research context of CSE attack recognition. The insights obtained from this chapter form the

foundation for the subsequent chapters, where the proposed models, methodologies, experimental results, and analysis are presented. By leveraging the theoretical knowledge and metrics discussed in this chapter, the objective is to develop effective deep learning models and make contributions to the advancement of CSE attack recognition techniques.

3. RELATED WORK

In this chapter a summarization of the most interesting works is presented. Although there is no research work dedicated to CSE attack recognition, there is valuable information in the following works.

Researchers performed several successful classifications of the social engineering attacks (Heartfield and Loukas 2016; Kumar, Chaudhary, and Kumar 2015; Salahdine and Kaabouch 2019) based on criteria such as the entity involved, the medium used to unleash the attack or the number of steps an attack can take. All these classifications end up in fine-grained taxonomies that include different methods of social engineering attacks. Verizon in (Verizon Enterprise) reports that in 2023 social engineering attacks were at 21%, the first step of every cyber-attack that led to a major data breach. Furthermore, the same report mentions that social engineering attackers' methods of choice are phishing and pretexting. The pretext method uses an invented scenario to facilitate the attacker to persuade the target to do what the attacker wants (Salahdine and Kaabouch 2019).

In 2005, Hoeschele and Rogers ((Hoeschele and Rogers 2005) introduced the Social Engineering Defense Architecture (SEDA) which is designed to recognize social engineering attacks during real-time phone conversations. Although the model was successful in recognizing the attacks, it did not incorporate previous activity history or personality recognition characteristics of both the attacker and the victim. In their 2010 paper, Bezuidenhout et al. (Bezuidenhout, Mouton, and Venter 2010) introduced an architecture named the Social Engineering Attack Recognition Model (SEADM), which assists users in making decisions through the use of a simple binary decision tree model. However, the authors rely on several unrealistic assumptions to justify the logic of their proposed system. SEADM was revisited in a subsequent paper by Mouton et al. in 2015 (Mouton et al. 2015) where the system was adapted to account for social engineering attacks that include unidirectional, and bidirectional communication between the attacker and the victim. According to Bhakta and Harris(Bhakta and Harris 2015), the most successful social engineering attacks involve a conversation between the attacker and the victim. Their methodology involves utilizing a predefined Topic Blacklist (TBL) to check dialogue sentences. The authors report achieving a precision rate of a recall rate of 88.9% with their approach. Sawa et al. (Sawa et al. 2016)expanded upon the aforementioned work by implementing advanced language processing techniques that balance syntactic and semantic analysis. The reported results demonstrate 100% precision and 60% recall. However, they used a small dataset with only three conversations which limited the precision and recall's success as a measure. Moreover, the algorithm did not consider any contextual information during the

classification process, making it unaware of the specific environment in which it operates. Uebelacker and Quiel (Uebelacker and Quiel 2014) have proposed a taxonomy of social engineering attacks based on Cialdini's Influence principles. They investigated the relationship between the Big-5 Theory, which pertains to personality traits, and Cialdini's influence principles. They proposed a Social Engineering Personality Framework (SEPF) and outlined a complete research roadmap for future work on SEPF.

In (Smutz and Stavrou 2012), the authors recognize malicious documents using a framework that is based on machine learning algorithms. A Random Forests ensemble classifier is used which selects in random the features for each classification tree. The features are selected and extracted from the document's metadata and structure. This Random Forests classifier appears to be resilient against mimicry attacks and highly efficient in recognition rate.

(Peng, Harris, and Sawa 2018) the authors present a method to recognize malicious statements using natural language processing techniques. They analyze the statements and discriminate the malicious ones that imply a phishing attack. The malicious intent of the statements is recognized by analyzing the statements. The proposed algorithm is evaluated using a phishing email dataset that is used as a benchmark set.

In Lansley et al. (Lansley et al. 2020; Lansley, Kapetanakis, and Polatidis 2020; Lansley, Polatidis, and Kapetanakis 2019), the authors use NLP methods and neural networks in an attempt to recognize social engineering attacks. They describe a method where offline or online text documents are processed using NLP tools and through artificial neural networks, they discriminate whether the text is a social engineering attack or not. The text in the first stage is parsed and checked for syntactical/grammatical errors using natural language techniques while later an artificial neural network classifies possible social engineering attacks. The proposed method has presented high accuracy results during the evaluation where a real and a semi-synthetic dataset were used for the model training. Furthermore, several other classification models have been tried to compare the two datasets.

(Lee, Saxe, and Harang 2020) the authors describe a method to represent the syntactic and semantic characteristics of the natural language by using a pre-trained BERT model. The proposed model seems to be resilient to adversarial attacks where the attackers intentionally replace the keywords with synonyms.

In (Catak, Sahinbas, and Dörtkardeş 2021) the authors describe a URL classifier based on Random Forest models and gradient boosting classifier. The URL classifier, in order to improve the algorithm's efficiency recognizing malicious web sites, makes use of features related to the host and the linguistic characteristics of the URL. By using

machine learning algorithms, the authors succeed in drastically reducing the recognition time of a malicious URL. Thus, they offer real-time protection for harmless web browsing while at the same time-saving computational resources.

Lan (Lan 2021) proposes a social engineering recognition model based on deep neural network. The model is able to recognize deception attempts and phishing attempts by analyzing the text content. The chat history in the first stage is processed and analyzed using natural language techniques and the context semantics are captured and mined using a bi-directional Long Short-Term Model (bi-LSTM). The integration of the user characteristics and chat content characteristics as features for the classification is done by ResNet.

Social engineering attacks are not only unleashed by humans; recent advances in Chatbot technology escalate even more the problem of social engineering. Chatbots (Adamopoulou and Moussiades 2020) are conversational agents that communicate through artificial intelligence and natural language processing . A chatbot dedicated to accomplishing a malicious task can utilize multiple personalities in an attempt to deceive a human and extract sensitive data. Chatbots are also capable of approaching all types of humans presenting different personalities based on the seven principles of persuasion defined by R. Cialdini (Cialdini 2021) (reciprocity, scarcity, authority, commitment, liking, social proof and unity). Moreover, due to the COVID-19 pandemic the use of conversational agents increased, and it is a rather common practice for an employee to socialize with a chatbot as part of daily routine operations. Users od chat software must be aware of the possible dangers and be protected from malicious questions which aim to extract sensitive data. Although chatbots are a novel tool in the hands of social engineers, the rapid pace in AI and NLP makes them an important factor due to their capabilities. Today, the majority of CSE attacks are operated by humans, but AI-driven social engineering attacks are expected to happen in the near future through chatbots too (Pogrebna and Skilton 2019).

In (Dalton et al. 2020), the authors describe a system that protects against CSE attacks by deploying a pipeline of NLP components. The NLP components include NER, Dialogue Engineering, Stylometry and Ask and Framing Question. They use an active defense approach to recognize the social engineer's intention and then waste her time and resources. In (Saleilles and Aïmeur 2021), the authors also present a different approach with a chatbot that is dedicated to educating the users and raising their awareness regarding social engineering attacks. The chatbot first performs an assessment of the cybersecurity concepts knowledge of the user using a quiz, and then based on the answers it proposes specific training paths to fill the knowledge gaps. During the education, the chatbot makes use of malicious questions, in an attempt to extract sensitive data from the users to make them more aware of the possible dangers.

4. CONCEPTUAL FRAMEWORK

Social engineering characterizes the general phenomenon of human manipulation involving the field of information systems. Its success depends on specific traits of human personality. These personality traits define the way of human behavior. The interested lies in traits that:

- Enhance the attacker's ability to manipulate.
- Make the victim vulnerable to manipulation.

A human's previous conversations can also help us draw a more complete picture of his vulnerability level and trigger an alarm with more confidence if a threshold is exceeded. In the following sub-sections, the main entities related to this study are presented, the attack surface and the CSE attack enablers that are decisive for the success or failure of a CSE attack.

4.1. Entities and attack surface

A chat may begin with the user or the potential social engineer, who initiates the conversation by entering utterances into the chat software, which serves as their means of communication. As shown in **Figure 4-1** the CSE-ARS system is able to analyze the content of the utterances utilizing deep learning algorithms and techniques and make inference regarding enablers that can lead to successful CSE attacks.

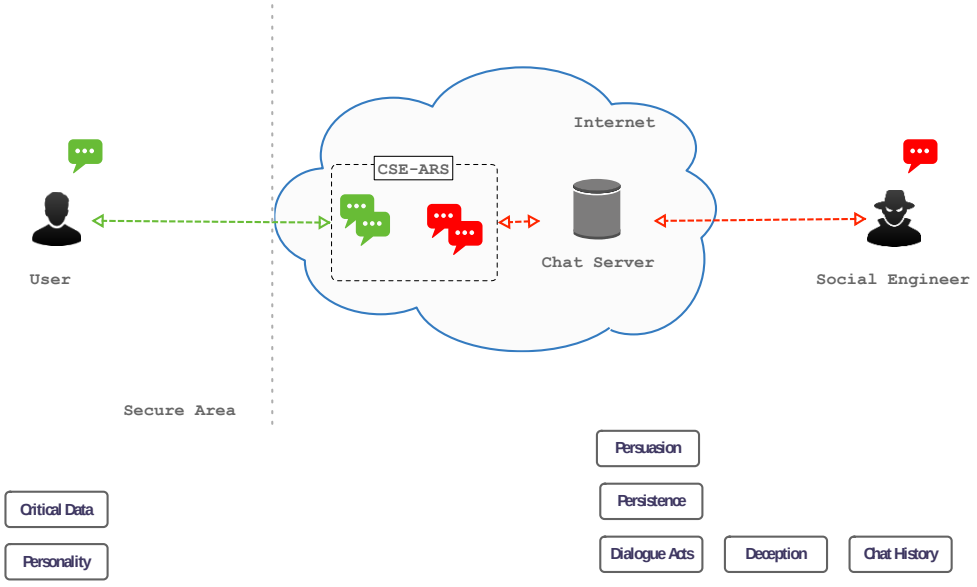

Figure 4-1. *Critical enablers that can lead to a successful CSE attack*

Throughout this process, CSE-ARS acts as a safeguard, aiming to protect the user by analyzing his personality characteristics and detecting critical information leakage. On the social engineer's side, CSE-ARS is able to recognize malicious persuasion attempts, persistent behavior and recognition of dialogue acts that can lead to unwanted actions by analyzing the full chat history and thus making inference regarding deception attempts.

For a successful CSE attack, several enablers can be triggered. By definition (Bernard 2012), an enabler is something that enables or facilitates an action. In the context of CSE attacks, enablers refer to the various tactics, techniques, and artifacts used by attackers to trick their targets into performing certain actions. These enablers include critical information leverage, personality traits that can be exploited, specific dialogue acts that can lead to information leakage, persuasion techniques, deception techniques, and paraphrasing methods. All these enablers are often used in combination to increase the chances of a successful CSE attack. Recognition of these enablers in an early stage is important to recognize CSE attacks. Following the work of (Tsinganos et al. 2018) the following critical enablers are identified:

- Critical information leverage: Social engineers attempt to extract sensitive or confidential information from their targets.
- Personality characteristics: Social engineers attempt to exploit certain personality characteristics of their victims in order to manipulate them and gain their trust.
- Dialogue acts: Social engineers use certain dialogue acts, such as questions or commands, to elicit information or manipulate the conversation.
- Persuasion attempts: Social engineering attacks often involve attempts to persuade the target to take a certain action, such as providing critical information or clicking on a link.
- Persistent behavior: Social engineers may attempt to paraphrase information to obtain additional information by repeatedly turning the discussion to the subject of interest.
- Deception attempts: Social engineering attacks often involve deception, such as pretending to be someone else or providing false information.

- Chat History: A CSE attack can be unleashed in separate and distinct phases. A recognition mechanism should be able to assess the full conversation between two interlocutors.

In the next sections the enablers that can lead to a successful CSE attack, collectively known as CSE attack enablers, are presented.

4.2. Critical Information Leakage

Critical information leakage is a significant enabler for successful CSE attacks. Social engineers employ various tactics to extract sensitive or confidential information from their targets, posing a significant threat to individuals and organizations. Recognizing the importance of addressing this challenge, deep learning algorithms can play a pivotal role in recognizing patterns in language that indicate the leakage of critical information.

Deep learning models can be trained to analyze chat-based conversations and identify linguistic cues or behavioral patterns that hint at the disclosure of e.g., personal information or login credentials. These models can learn from a vast amount of labeled data, encompassing both legitimate conversations and examples of information leakage. The training process enables the models to develop an understanding of the specific linguistic markers associated with critical information leakage, thereby enhancing their ability to recognize such instances in real-time.

The application of deep learning models in identifying critical information leakage holds great promise for bolstering the security and resilience of individuals and organizations against CSE attacks. By swiftly recognizing and flagging instances of information leakage, these algorithms can provide an added layer of defense, enabling timely intervention and mitigating potential risks. Moreover, the continuous learning capabilities of machine learning models allow them to adapt to evolving attack strategies, thereby staying one step ahead of malicious actors and safeguarding sensitive information from being compromised.

4.3. Personality Traits

In psychology, human personality *"refers to individual differences in characteristic patterns of thinking, feeling and behaving" (Kazdin, A.E. 2004)* and, although there is no universal acceptance, the Big-5 Theory analyzes a five-factor model (FFM) of the personality traits, or otherwise called factors to classify personalities. These factors are believed to capture most of the individual differences in terms of personality. The five factors, usually measured between 0 and 1, are (Spielberger 2004):

- conscientiousness: *"The degree to which individuals are hard-working, organized, dependable, reliable, and persevering versus lazy, unorganized, and unreliable."*
- extraversion: *"The extent to which individuals are gregarious, assertive, and sociable versus reserved, timid, and quiet."*
- agreeableness: *"The degree to which individuals are cooperative, warm, and agreeable versus cold, disagreeable, rude, and antagonistic."*
- openness: *"the extent to which an individual has richness in fantasy life, aesthetic sensitivity, awareness of inner feelings, need for variety in actions, intellectual curiosity, and liberal values."*
- neuroticism: *"the degree to which one has negative effect, and also disturbed thoughts and behaviors that accompany emotional distress"*

The degree of susceptibility to CSE attacks has also been connected to the five personality traits. Responsible behavior concerning security best practices has been positively correlated with low openness (Sudzina and Pavlicek 2023), meaning that high levels of openness could potentially facilitate risky security behavior. Lower levels of Conscientiousness have been found to predict deviant workplace behavior such as irresponsible conduct or rule-breaking (Salgado 2002). High levels of Extraversion are predictive of increased vulnerability to phishing attacks (Albladi and Weir 2017). Agreeableness has consistently been associated with phishing in multiple studies (Anawar et al. 2019; Uebelacker and Quiel 2014). Agreeable individuals may be more susceptible to manipulation due to their tendency to establish trust with the target, which is a characteristic of agreeableness. On the other hand, lower levels of Neuroticism are associated with higher susceptibility (Exploring the Role of Individual Employee Characteristics and Personality on Employee Compliance with Cybersecurity Policies, 2012).

The results are contradictory in many situations and they do not lead to a direct conclusion. Up till now, researchers have examined the relation between personality traits and social engineering by combining knowledge of human behavior in other fields (marketing, etc.). It would be of great benefit to analyze and measure the exact relation of personality traits with specific SE techniques.

The following table depicts when a personality trait can lead to increased susceptibility of CSE attacks.

Table 4-1 *- Personality traits & CSE Susceptibility*

Personality Trait	Value	Vulnerability
Openness	High	High
Consciousness	Low	High
Extraversion	High	High
Agreeableness	High	High
Neuroticism	Low	High

4.4. Speech-Acts & Dialogue Acts

Theoretical linguistics inquire into the nature of human language and seek to answer fundamental questions as to what a language is, or the inner workings of it. Several different levels of analysis are defined, such as syntactic (studies the structure of the visible/audible form of the language), semantic (studies the relations and dependencies between different language structures and their potential meanings), and pragmatic (studies the issues related to language use due to context and uncovers the intention of the speaker in an utterance).

This study on chat-based conversations can benefit by finding the ordering and patterns of interaction between two interlocutors. The interest is in uncovering the actions that are hidden between the words and pragmatic analysis seems to be the appropriate approach from such a language/action perspective (Winograd 1986). The starting point to study the pragmatics of language action is Speech Act Theory (SAT). According to SAT (Searle et al. 1980), the uttering of a sentence is an action, and in short, the form says that "saying is doing" or similarly "words are deeds". Austin (Austin 1975) claimed *"all utterances, in addition to meaning, perform specific acts via the specific communicative force of an utterance"* and introduced a three-fold distinction among the acts one simultaneously performs when saying something:

- Locutionary act: the production of a meaningful linguistic expression.
- Illocutionary act: the action intended to be performed by a speaker in uttering a linguistic expression, either explicitly or implicitly. Examples include: accusing, apologizing, refusing, ordering, etc.
- Perlocutionary act: the effect of the illocutionary act on the hearer such as persuading, deterring, surprising, misleading or convincing.

For example, the phrase of an IT technician: *"The operating system will reboot in five minutes."* results in saying that the OS will reboot in 5 minutes (locutionary act) and informs the users of the imminent rebooting of the OS (illocutionary act). By producing his utterance, the IT technician intents to make users believe that the OS will reboot in 5 minutes and urges them to do housekeeping activities (perlocutionary act). The IT technician performs all these speech acts, at all three levels, just by uttering the above sentence.

Searle proposed speech acts to be classified into the following five categories along four dimensions (illocutionary point, direction of fit between words and world, psychological state, and propositional content):

- *Representatives* express the speaker's beliefs. Examples include claiming, reporting, asserting, stating and concluding. Using representatives, the speaker makes words fit the world by representing the world as he believes it is.
- *Directives* express the speaker's desire to get the hearer act in a specific way. Examples include commands, advice, orders and requests. Using directives, the speaker intends to make the world match the words via the hearer. E.g., "Double-click this file."
- *Commissives* are used to express the speaker's intention and commitment to do something in the future. Examples include offers, pledges, promises, refusals, and threats. Using Commissives, the speaker adapts the world to the words; e.g., "I'll never give you access to your account."
- *Expressives* express the psychological state of the speaker such as joy and sorrow. Examples include praising, blaming, apologizing, and congratulating. There is no direction of fit for expressives; e.g. "Well done, John!"
- *Declaratives* are used to express immediate changes in the current state of some affair. Examples are firing (from employment), declaring war, etc. Both directions of fit, suit this type of speech act (words-to-world and world-to-words). E.g., "I object, Your Honor."

Speech acts and dialogue acts play a crucial role in CSE attack recognition. Speech acts involve the communicative intent behind a statement, while dialogue acts represent the function of an utterance in a conversation. Recognizing these acts is essential for identifying CSE attacks as attackers often use specific speech and dialogue acts to manipulate victims. Understanding the interplay between speech and dialogue acts allows for the detection of manipulative or coercive language patterns, helping to enhance the accuracy of CSE attack recognition systems and ultimately bolstering cybersecurity measures in chat-based interactions.

4.5. Influence & Persuasion

As Schneier points out (Samuel Galice Marine Minier 2008), human risk perception has evolved over thousands of years. Nevertheless, progress in technology has changed our lives very fast without allowing enough time for our risk perception to adjust to new threats. This vulnerability in human design is exploited by social engineers and then transferred to information systems to compromise them. Schneier discusses also heuristics (called shortcuts) in human behavior and biases. Both are causal factors for wrong appraisals and decisions. Robert Cialdini (Ganzha and Polskie Towarzystwo Informatyczne 2010) agrees with Schneier and discusses the principles of influence and how heuristics and biases are exploited by a human to manipulate another human. Cialdini also argues that there are two types of influence: *compliance* and *persuasion*. Using persuasion, the attacker sends a message and then the victim changes his behavior, attitude or knowledge as a result of the received message. Compliance forces the change of a behavior as a result of a direct request. The request can be explicit (hard) or implicit (soft). Cialdini (Cialdini and Goldstein 2002) conducted experiments and field studies on sales and marketing department employees, and defined six influence principles. In a recent work (Cialdini 2021) Cialdini added a seventh principle to his famous taxonomy. Overall, the seven persuasion principles are:

- *Reciprocation*: a social norm that makes us repay others for what we have received. It builds trust between humans and we are all trained to adhere or suffer severe social disapproval. Humans feel obliged after receiving a gift.
- *Commitment and Consistency*: humans commit by stating who they are, based on what they do or think. They also like to be consistent because that builds character. Attackers exploit that kind of belief by initially asking for a small favor, then a bigger one and finally the big bad favor. Humans that have already served an attacker feel they have to show commitment and be consistent with their prior behavior.
- *Social Proof*: humans tend to believe what others do or think as right.
- *Liking*: if someone likes us and makes it obvious, it is hard to resist not to like him back. After that it is easier for him to ask us a favor and difficult for us to deny him one. On the opposite direction we all want to be liked
- *Authority*: humans tend to trust and obey experts or someone in a high hierarchical position. It is difficult for an employee to deny a request from an IT manager, for example.
- *Scarcity*: limited information leads to wrong decisions and limited resources are more desirable. If an attacker knows that an employee wants a specific application then she can offer it (after injecting an exploit), or claim a reason to request a favor based on evidence that only the user possesses.

- *Unity*: a perception that we share a common identity, that we are all part of "us."

Apart from Cialdini, many researchers tried to capture the psychological aspects of human behavior related to persuasion. Gragg (Gragg 2003) presents a list of such principles and calls them triggers: Strong affect, Overloading, Reciprocation, Deceptive Relationships, Diffusion of Responsibility, Authority, Integrity and Consistency. Scheeres (Scheeres 2008) makes obvious the relationship between Gragg's and Cialdini's treatment by correlating all these principles and triggers. Granger (Granger 2001; 2002) and Peltier(Peltier 2006) present similar factors of persuasion based on their point of view.

Table 4-2 summarizes the mapping of the above factors along with Cialdini's principles. In this thesis' approach Cialdini's influence principles are adopted because there is a major overlap with all of the factors proposed by the other researchers.

Table 4-2. *Mapping of Persuasion Principles and Factors.*

Cialdini's Principles	**Harl's (1997) Communication Strategies**	**Gragg's (2003) Compliance-Gaining Strategies**	**Granger (2001) Persuasion Principles**	**Peltier's (2006) Persuasion Principles**
Reciprocity	Giving and receiving	Offering concessions	Exchange	Exchange
Scarcity	Limited availability	Creating a sense of urgency	Scarcity	Scarcity
Authority	Expertise and credibility	Appealing to expertise	Authority	Authority
Liking	Similarity and attraction	Making yourself likable	Liking	Liking
Social proof	Consensus and conformity	Appealing to others' behavior	Social proof	Social proof
Consistency	Commitment and consistency	Getting someone to commit to a small step	Commitment	Commitment

4.6. Deception

An (An 2015) describes Deception as *"an act or statement intended to make people believe something that the speaker does not believe to be true, or not the whole truth"*. A more precise definition for Deception is given in (Granhag and Strömwall 2004) where *"Deception is a successful or unsuccessful attempt, without forewarning, to create in another a belief which the communicator considers to be untrue"*. Over the years the research community became very interested in the recognition of deception. Due to the interdisciplinary nature of the phenomenon, researchers from various

scientific fields (psychology, computer science, linguistics, philosophy, etc.) have already presented their results by studying and analyzing several different deceptive cues (e.g., bio-metric indicators, facial indicators or gestural indicators).

There are two categories of deception (An 2015):

- face-saving: when humans lie to protect themselves, to avoid tension and conflict in a social interaction, or to minimize hurt feelings and ill will,
- malicious: when humans lie with harmful intent.

The primary interest is in recognizing a malicious deception attempt in a text-based conversation and using this finding as an extra indicator for recognizing a social engineering attempt. So far, several research attempts have been made to study verbal or nonverbal cues to recognize deceptive behavior (Ott et al. 2011), (Feng, Banerjee, and Choi 2012). Current work in deception recognition is mainly based on verbal cues and has shown that it is possible to reliably predict a deception attempt (Vrij 2014). In most of the works researchers have collected data and manually annotated them for deceptive status. After that, the labeled data were fed to a classification algorithm for supervised learning. The features extracted for text-based deception recognition are critical and directly connected to prediction accuracy (Ott et al. 2011), (Feng, Banerjee, and Choi 2012).

The common scientific approach for persuasion recognizing is to use three types of features, namely lexical, acoustic, and speech features. The most frequently used techniques for lexical analysis are: Linguistic Inquiry and Word Count (LIWC), N-gram, Part-of-speech (POS), and Dictionary of Affect in Language (DAL). LIWC is primarily used for recognizing psychological characteristics by calculating several metrics for the usage of different word categories, the usage of casual words, the existence of positive or negative emotions in text, etc. In (Hirschberg et al. 2005), and (Ott et al. 2011) researchers used LIWC to examine text-based communication and managed to extract valuable knowledge regarding people's personality, and cognitive and emotional characteristics. The above research works differ in accuracy results due to the use of different datasets that lead to accurate or less accurate machine learning algorithms. DAL is mostly used to analyze emotive content and its main difference from LIWC is that it has a narrower focus. N-gram is usually combined with other more advanced techniques, like LIWC to train binary classifiers (e.g., Naive Bayes, SVM, etc.) during a lexical analysis.

4.7. Chat History

This enabler poses a significant technical challenge in assessing the risk of a potential CSE attack by analyzing the user's chat dialogues. CSE attacks often unfold over multiple phases, where the attacker strategically prepares and builds a sense of trust with the victim over time. During this process, the attacker carefully explores the victim's vulnerabilities and seeks the opportune moment to launch the attack. Therefore, it becomes crucial to leverage the entire history of chat dialogues between the same individuals, converting them into quantifiable metrics that can serve as additional indicators for recognizing CSE attacks.

By analyzing the chat history, we can capture valuable insights into the dynamics and patterns of the communication between the attacker and the victim. This historical information provides a contextual understanding of the relationship between the interlocutors and unveils the progression of the conversation over time. It allows us to identify any anomalous or suspicious behavior that deviates from the norm, potentially indicating the presence of a social engineering attack.

Transforming the chat history into measurable values involves the extraction of relevant features and the application of suitable techniques, such as natural language processing and machine learning. These techniques enable us to quantify various aspects of the dialogues, such as linguistic patterns, sentiment analysis, topic modeling, and user behavior. By quantifying the chat history, we can create a comprehensive representation of the communication dynamics, enabling the development of effective algorithms and models for CSE attack recognition.

The analysis of chat history provides an additional layer of insight in recognizing CSE attacks. By incorporating the historical context of the interlocutors' communication and transforming it into measurable values, we can leverage this enabler to enhance the accuracy and robustness of CSE attack recognition systems. The utilization of chat history as an extra indicator strengthens the ability to identify suspicious patterns, behaviors, and manipulative techniques employed by attackers. Future research can further explore advanced techniques for feature extraction and modeling to optimize the utilization of chat history in CSE attack recognition and prevention.

4.8. Persistence

The persistent behavior of an attacker serves as a critical enabler for successful CSE attacks. Social engineers often employ tactics (Kubal and Nimkar 2019)such as paraphrasing to confirm the accuracy of information they have already obtained or to

extract additional sensitive details from their targets. Recognizing the significance of addressing this challenge, machine learning algorithms can be trained to recognize patterns in language that indicate paraphrasing attempts, thus enhancing the recognition and prevention of such persistent behavior.

By leveraging machine learning techniques (Thompson 2017), algorithms can learn to analyze chat-based conversations and identify linguistic cues that suggest paraphrasing. These algorithms can be trained on labeled data that encompasses instances of paraphrasing, allowing them to develop an understanding of the various linguistic markers associated with this persistent behavior. By examining patterns such as the repetition of information in different words or syntactical variations used to convey the same meaning, machine learning models can effectively flag instances of paraphrasing, thereby alerting users to potential social engineering attacks.

The application of machine learning algorithms in recognizing persistent behavior, such as paraphrasing, plays a crucial role in mitigating the risks associated with CSE attacks. By proactively identifying and highlighting instances of paraphrasing, these algorithms empower individuals and organizations to recognize suspicious behavior and take appropriate countermeasures. The continuous learning capabilities of machine learning models also enable them to adapt to evolving techniques employed by attackers, ensuring a robust defense against persistent social engineering tactics.

4.9. Chapter Conclusions

Chapter 4 delved into the enablers of CSE attacks, exploring various factors that contribute to the success of these malicious activities. The focus of the study was directed toward the exploration of key enablers, namely critical information leverage, personality traits, persuasion, deception, speech acts, chat history, and persistence. By thoroughly examining each enabler, valuable insights were obtained into the tactics employed by social engineers and the significance of addressing these enablers in the development of effective defense mechanisms.

Personality traits play a crucial role in social engineering attacks, as social engineers strategically leverage their understanding of human behavior to manipulate targets. The examination of personality traits and their recognition through machine learning algorithms contributes to the development of robust models for recognizing and mitigating social engineering attacks.

Furthermore, persuasion is a powerful enabler that social engineers exploit to manipulate targets' decisions and actions. Understanding the dynamics of influence and

employing machine learning algorithms to recognize influential strategies contributes to enhancing the resilience against social engineering attacks.

Persistent behavior, including the use of paraphrasing, is a key strategy for social engineers to manipulate their targets and increase the likelihood of a successful attack. It allows them to build trust, adapt to the target's responses, and maintain a consistent deception, ultimately making it more challenging for the target to recognize and resist the manipulation.

Deception is another vital enabler, as social engineers often employ techniques to deceive targets and gain their trust. By investigating deception and developing machine learning models that recognize deceptive cues, we can fortify defenses against CSE attacks.

Moreover, speech acts and chat history provide valuable insights into the structure and context of conversations, which can be leveraged to recognize suspicious or malicious activities. By studying speech acts and incorporating chat history analysis into machine learning models, we can improve the accuracy of attack recognition and prevention.

The insights obtained from this chapter establish a strong basis for the following chapters, where the proposed models and methodologies for recognizing and mitigating these enablers are presented. By effectively addressing these enablers, we aim to enhance the resilience of individuals and organizations against CSE attacks.

5. CSE CORPUS

5.1. Introduction

It is a fact that humans nowadays are extensively using Information and Communication Technologies (ICT) and especially Electronic Medium Communication (EMC) software for almost every aspect of their daily activities. Although EMC software is an advantageous tool, it also constitutes a large and vulnerable attack surface. The vulnerabilities lay in human personality characteristics, and the way adversaries exploit them to extort valuable information for their benefit. As stated by ENISA ('What Is Social Engineering?'), the term Social Engineering "refers to all techniques aimed at taking a target into revealing specific information or performing a specific action for illegitimate reasons".

A CSE attack cannot be described using only technical terms due to the nature of the human vulnerabilities. Therefore, the implementation of a multi-factor recognition system is imperative to enable the automatic identification of a CSE (CSE) attack. An ideal multi-factor recognizer should process the dialogue in real-time and protect the users by recognizing potential CSE attacks in early stages, e.g., by alerting the user if the probability of sensitive data leakage exceeds a predefined threshold.

Such recognition can be based either on a pure statistical approach or a machine learning approach. Considering the importance of recognizing a CSE attack in real-time, the ML approach can qualify as an efficient solution. Furthermore, ML algorithms can be combined perfectly with NLP algorithms which can be used to process the language used in a natural communication setting. Such ML/NLP algorithms need labelled datasets to be trained and to be efficient in their predictions. For example, a set of processed dialogues (usually named 'corpus') of realized CSE attacks can be tagged at the sentence-level to discriminate the malicious sentences from the benign ones. A ML/NLP algorithm trained with this corpus will be able to efficiently recognize a malicious sentence in a future dialogue. The efficiency of the algorithm is related to the level of consistency between the content of the dataset and the researcher's domain of interest, i.e., if one asks how to recognize phishing emails, then she needs a corpus of emails that gives insight into the question asked.

Currently, there is a lack of corpus composed of realized CSE attacks, in contrast with other types of attacks like network intrusions, because CSE dialogues are rarely going public. Furthermore, a CSE Corpus should be of sufficient quantity so the algorithms can be trained well and be efficient. In case the collected dialogues are not of sufficient

quantity, researchers apply resampling techniques (Lateh et al. 2017; D.-C. Li et al. 2018) to enlarge the corpus, or they use annotation (Wilcock 2009) to add useful in-context information as metadata to every dialogue.

The metadata information augments the algorithms' effectiveness by providing pointers to what is important in a corpus. The ML/NLP algorithms extrapolate rules from the metadata provided in order to apply those rules later to unannotated text dialogues. An annotation task frequently employs a custom eXtensible Markup Language (XML) representation scheme depending on the context used. While there are some well-known annotation schemes, there is no universal standard for annotating dialogues. Furthermore, there are multiple ways for this metadata information to be stored, either as *inline* annotation (metadata are stored in the same file as the dialogue), or *stand-off* annotation (metadata are stored in a separate file).

This chapter answers the aforementioned concerns by investigating:

- The feasibility of collecting and building a CSE attack corpus to address the lack of training data composed of realized CSE attacks.
- The feasibility of annotating the linguistic characteristics of social engineers' language in order to constitute an adjuvant factor for the ML algorithms' training.

5.2. Foundations & Methodology

This research's approach to CSE Corpus design was based on a conceptual framework describing the CSE attack domain and a methodology for building and annotating a CSE corpus.

5.2.1. Foundations

According to ENISA , the ISO/IEC 15408-1:2009(E) standard ('Evaluation Criteria for IT Security — Part 1: Introduction and General Model - ISO/IEC 15408-1:2009') is commonly used as a resource for evaluating the security of IT products and systems, and it is also used for procurement decisions concerning such products by providing an abstract cybersecurity concept map. This concept map (**Figure 5-1**) is focused on the protection of assets from threats. Threats are related to malicious or other human activities that seek to abuse the assets. In this high-level concept map, the cybersecurity concepts are interconnected through relationships depicting the strong interrelation between them.

After a thorough text analysis of the collected raw dialogues that is presented in Section 5.3, it was concluded that social engineers aimed to extract information that could be categorized into the following three sensitive data categories.:

- **Personal**: Details related to the victim, e.g., first or last name, telephone number etc.
- **IT**: Details related to the Information Technology (IT) ecosystem, e.g., network-share names, hardware/software characteristics, etc.
- **Enterprise**: Details related to the enterprise operation, e.g., department names, Office numbers, etc.

By integrating the aforementioned findings and narrowing the scope to enterprise environments, significant enhancements were made to the concept map of CSE attacks within the framework provided by ENISA. These improvements resulted in a more comprehensive understanding of CSE attacks specifically tailored to enterprise settings, enabling organizations to better identify and address potential vulnerabilities and risks. This enrichment of the concept map added valuable in-context details that are relevant to the thesis' specific goal.

The concept map in **Figure 5-1** introduces seven new concepts that are essential for describing the domain of CSE attacks. By utilizing this concept map, we can discern that Small-Medium Enterprises (SMEs) prioritize the protection of their assets against potential threats as their main concern. *Threat agents,* such as *Social Engineers,* give rise to these threats by exploiting *vulnerabilities* leading to greater *risk* for the assets. It is the responsibility of the SMEs to safeguard the assets where, in the case of a CSE attack the most critical asset is the *Sensitive Data* that can leak. In addition, sensitive data can include *Personal*, *IT,* or *Enterprise* details.

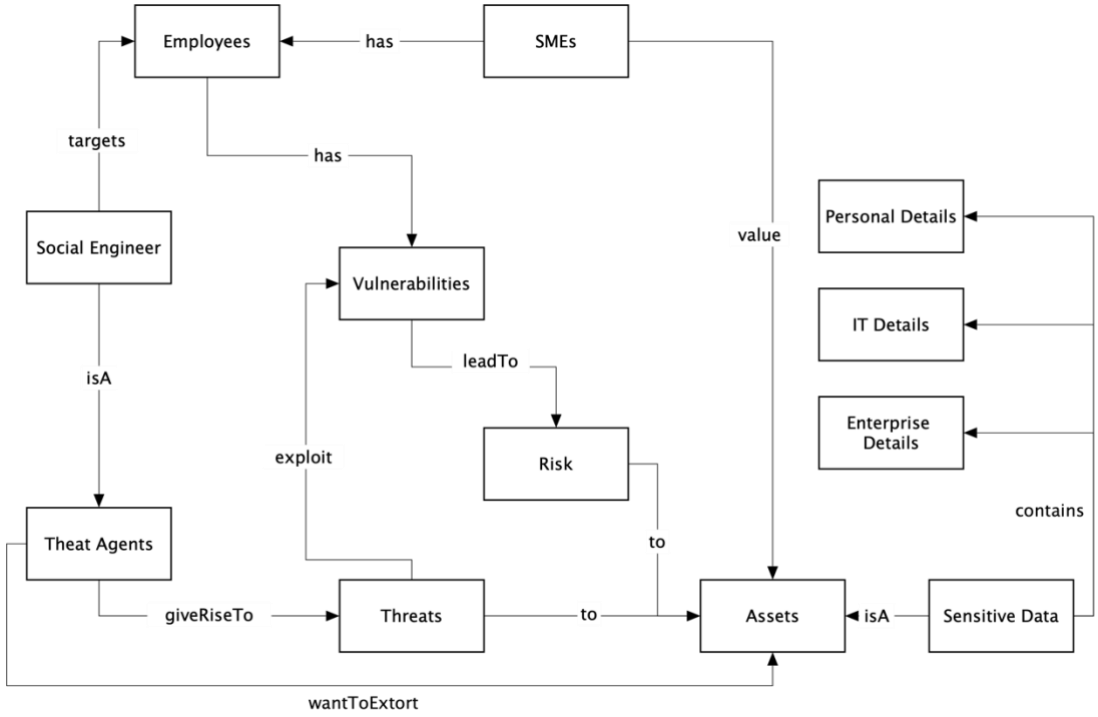

Figure 5-1. *The concept map of CSE Attack.*

5.2.2. Methodology

The methodology followed for building and annotating a CSE Corpus contains two phases, each comprising several steps, as described below:

- PHASE 1: CSE Cor*pus Building*
    - STEP1-*Sources Selection*: the CSE attack dialogue sources are identified.
    - STEP2-*Dialogues Collection:* the CSE attack dialogues are collected through manual and automated web scraping methods.
    - STEP3-*Dialogues Enrichment*: the collected dialogues are enriched.
    - STEP4-*Linguistic Analysis:* the enriched dialogues are analyzed from a linguistic perspective.
    - STEP5-*Dialogues Processing*: the analyzed dialogues are further processed using NLP techniques to form the CSE corpus.
- PHASE 2: CSE Cor*pus Annotation*, based on the steps described in (Pustejovsky and Stubbs 2013).

- STEP1-*Goal Declaration:* the goal of the annotation task is declared.
- STEP2-*Model and CSE Ontology Creation*: the phenomenon in study is modeled in abstract terms and an ontology is created to be used as the base for the annotation task.
- STEP3-*Specifications Definition:* a concrete representation of the model is created based on the CSE ontology.
- STEP4-*Annotator Guidelines:* a blueprint is produced to help annotators in element identification and element association with the appropriate tag.
- STEP5-*Annotation Task:* the annotation process is performed. In case of changes in the CSE ontology, the process continues with STEP2.
- STEP6-*Inter-Annotator Agreement:* the inter-annotator agreement is validated.
- STEP7-*Adjudication Task:* the final version (gold standard) of the annotated CSE Corpus is formed.

The following sections 5.3 and 5.4 present the details of building and annotating the CSE Corpus.

5.3. PHASE 1: CSE Corpus Building

In the first phase of the methodology the data sources were identified and the dialogues were collected. Moreover, the dialogues were enriched, analyzed linguistically and preprocessed to be ready to be fed as input for the model's training.

5.3.1. STEP 1 – Sources Selection

To establish the CSE corpus, the first step entailed identifying sources for obtaining dialogues pertaining to CSE attacks. This task commenced by gathering synonyms and comparable search terms for the term "dialogue." To achieve this objective, the collection of synonyms and analogous search terms for the term "dialogue" was initiated. Using the *Word Embeddings* (Goldberg and Levy 2014) technique, similar terms were identified and selected based on a ranking of the most used words in context. The words that were selected were: *dialog*, *dialogue*, *chat*, *conversation,* and *discourse.* Additionally, the following terms in the CSE context were used: *discourse analysis*,

data leakage, *sensitive data exposure*, *corpus*, *dataset*, *online chat*, *instant messenger*, *excerpts*, and *conversation transcripts*. By utilizing combinations of the previously mentioned search terms, numerous websites, books, logs, and forums containing pertinent content were identified. The discovered sources belong to one of the following categories:

- Social engineering dark websites (tutorials, guides, and others)
- Social engineering dark forums (text dialogues)
- Social engineering books
- Instant Messaging logs
- Telephone conversations

The following table (**Table 5-1**) presents the top twenty web sites, forums and books used to acquire dialogues useful in producing the CSE corpus.

Table 5-1. *Top twenty sources for collecting CSE attack dialogues.*

No#	Web Sites/Forums
1	'What is the bloody point ?' https://www.whatsthebloodypoint.com/ (accessed on 15 November 2021)
2	'Social Engineering', Nulled. https://www.nulled.to/forum/69-social-engineering/ (accessed on 15 November 2021)
3	'Sinisterly-Social Engineering' https://sinister.ly/Forum-Social-Engineering (accessed on 15 November 2021)
4	'Ripoff Scams Report | Consumer Complaints & Reviews | Ripandscam.com' https://www.ripandscam.com/ (accessed on 15 November 2021)
5	MPGH-MultiPlayer Game Hacking & Cheats https://www.mpgh.net/forum/forumdisplay.php?f=670 (accessed on 15 November 2021)
6	'Home', BlackHatWorld. https://www.blackhatworld.com/ (accessed on 15 November 2021)

7	'Hack Forums' https://hackforums.net/index.php (accessed on 15 November 2021)
8	'Demonforums.net', demonforums.net. https://demonforums.net/ (accessed on 15 November 2021)
9	419 Eater-The largest scambaiting community on the planet! https://www.419eater.com/ (accessed on 15 November 2021)
10	Socialengineered Forum https://web.archive.org/web/20200119025819/https://socialengineered.net/ (accessed on 15 November 2021)
11	SE Forums https://web.archive.org/web/20180401044855/https://seforums.se/ (accessed on 15 November 2021)
12	Leak Forums Net https://web.archive.org/web/20190123070424/https://leakforums.net/ (accessed on 15 November 2021)
	Books
13	Advanced Persistent Threat Hacking: The Art and Science of Hacking-Tyler Wrightson
14	G. Watson, A. Mason, and R. Ackroyd, Social Engineering Penetration Testing: Executing Social Engineering Pen Tests, Assessments and Defense. Syngress, 2014
15	C. Hadnagy, Unmasking the Social Engineer: The Human Element of Security. John Wiley & Sons, 2014
16	M. I. Mann, Hacking the Human: Social Engineering Techniques and Security Countermeasures. Gower Publishing, Ltd., 2012.
17	K. D. Mitnick and W. L. Simon, The Art of Deception: Controlling the Human Element of Security. John Wiley & Sons, 2011.
18	K. Mitnick, Ghost in the Wires: My Adventures as the World's Most Wanted Hacker. Hachette UK, 2011.
19	J. Long, No Tech Hacking: A Guide to Social Engineering, Dumpster Diving, and Shoulder Surfing. Syngress, 2011.

20 C. Hadnagy, Social Engineering: The Art of Human Hacking. John Wiley & Sons, 2010.

5.3.2. STEP 2 – Dialogues Collection

To perform the collection of dialogues (Tsinganos 2021) a cloud-based infrastructure was established to host all the required software services and custom scripts. More specifically, the infrastructure was deployed in a Cloudstack environment by provisioning virtual machines with discrete roles. A dissection of the infrastructure based on functionality is illustrated in **Figure 5-2**.

The *Corpus Building* section includes the *n Web Scrapers* used, which host custom scripts that scan and collect text from different web sources. The Web Scrapers are the only members of the infrastructure that communicate with the Internet to locate dialogues. This section also hosts every server that stores information at all stages of the project. The *Raw Content* server contains the scraped content before any preprocessing, while the *Database* server and the *Corpus* server contain selected information based on specific criteria and the final corpus, respectively. The *Processor* server applies the custom scripts in different processing stages and stores the results in the appropriate target. Finally, the *Corpus Annotation* section includes the workstations of the cybersecurity experts acting as annotators.

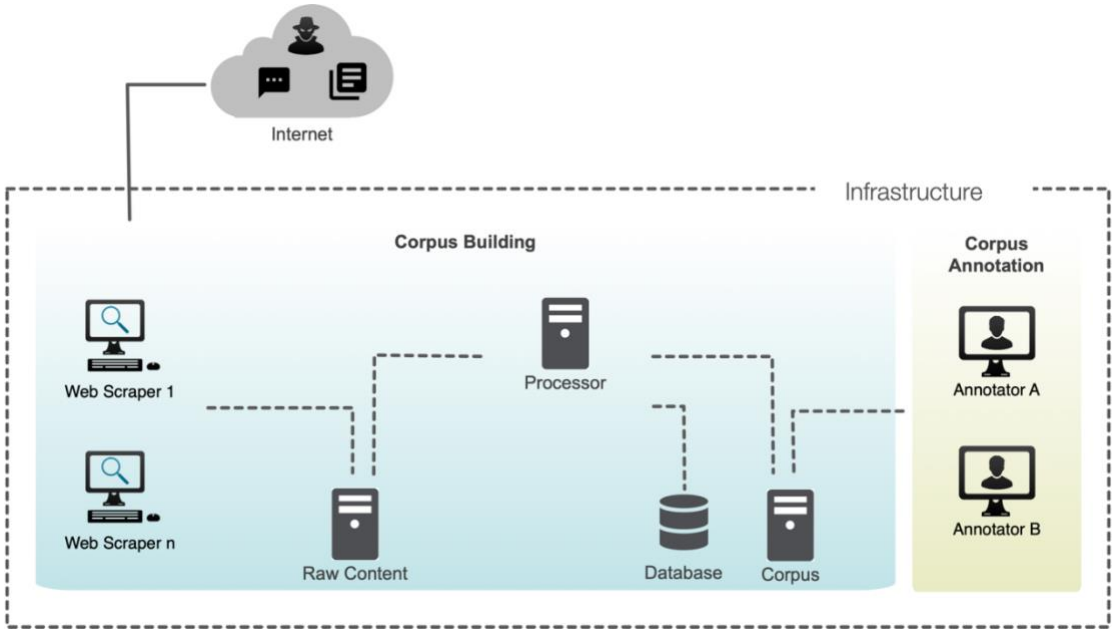

***Figure 5-2**. Infrastructure Setup.*

5.3.3. STEP 3: Enrichment

As mentioned earlier, a corpus can be small in size, and this can have a negative influence on ML/NLP algorithms' efficiency. To enhance the collected dialogues, a parsing process was conducted to generate a list of 200 crucial nouns associated with one of the three categories of sensitive data. Afterward, for each noun, ten new sentences were added where the noun identified as critical was substituted by a similar word. Thus, the dialogues were enriched by 2000 new sentences. The discovery of similar words was based on the Word Embeddings technique (Goldberg and Levy 2014), which is used to capture the meaning of the words using a dense vector representation. Each point in the embedding space represents a word and, based on the surrounding words of the target word, these points are learned and moved around.

A pre-trained word2vec model (Mikolov, Chen, et al. 2013) was employed to identify synonyms using the distributional hypothesis, which posits that linguistic elements such as words exhibiting similar distributions in a specific context (dialogue/document) tend to have similar meanings. This way, a segregation of the different domain words is created using their vector values. Words with similar meanings were grouped due to their similar distribution in the dialogues. A vector space was constructed in which each distinct word in a dialogue was assigned a corresponding vector. Hence, the vector space is a vector representation of the words in the collected dialogues.

The Individual dimensions of these vectors have no inherent meaning. However, it is the overall patterns of location and distance between vectors that ML/NLP algorithms take advantage of. For example, the application of the word embedding technique in order to locate the ten most similar words to the word 'password' gave us the list illustrated in **Figure 5-3**:

```
['password'] [
    ('passwords', 0.769),
    ('passphrase', 0.669),
    ('login', 0.644),
    ('Password', 0.637),
    ('passcodes', 0.637),
    ('logon', 0.632),
    ('username_password', 0.619),
    ('Passwords', 0.615),
    ('logon_credentials', 0.609),
    ('passphrases', 0.602)
]
```

Figure 5-3. *Ten most similar words of 'password' term.*

One can notice that the word 'password' and the word 'logon' have an Euclidean distance of 0.632 while the word 'password' and the word 'passphrases' have a smaller Euclidean distance of 0.602.

5.3.4. STEP 4 - Linguistic Analysis

The linguistic analysis was performed according to the following levels of observation (see **Figure 5-4**). A sample of the CSE dialogues was analyzed from a linguistic perspective and served as a baseline to ensure that the software tools and libraries were able to achieve the desired level of quality. Initially, as seen in **Figure 5-4**, a *lexical analysis* was performed by breaking down a sentence into words, phrases, or other meaningful part, a task known as *chunking*. Afterward, a *morphological analysis* was performed where the structure and the form of each word were the main concern. Part-of-speech tagging (POS), stems and lemmas were identified, and a *syntactic analysis* followed that focused on grammar and syntax patterns. Subsequently, the meaning of the words and phrases was examined, and the semantics of words and phrases were investigated. The most crucial step, though, was the *pragmatic analysis* that took place to identify the *actual meaning* of the utterances. This is reasonable because an automation tool cannot understand the hidden intent of a speaker or a writer.

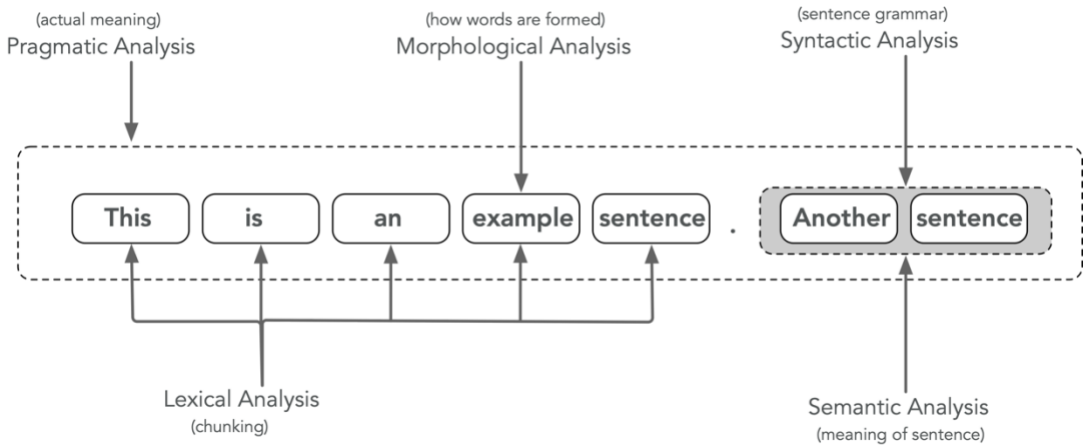

Figure 5-4. *Example of linguistic analysis.*

5.3.5. Preprocessing

The collected CSE dialogues were prepared using the text processing workflow depicted in **Figure 5-5**. At first, all dialogues were converted to use the UTF-8 encoding scheme. Slang and abbreviations were removed or substituted with corresponding phrases using open-source slang databases. Spelling correction and removal of emojis were executed using dictionaries and the TextBlob library (‘TextBlob: Simplified Text Processing — TextBlob 0.16.0 Documentation’). Empty lines, specific stopwords and specific punctuation marks were removed using traditional NLP libraries like NLTK (‘NLTK :: Natural Language Toolkit’) and spaCy(‘spaCy · Industrial-Strength Natural Language Processing in Python’). Moreover, all HTML or other programming code and URLs or paths were stored in a separate database and substituted with _code_, _url_,

and _path_ placeholders in the dialogue. Any illegal characters were also stripped and all text was transformed to lowercase.

The standard Penn Treebank (Taylor, Marcus, and Santorini 2003) tokenization rules were utilized for sentence tokenization, and finally, standardization processes using regular expressions and lookup tables were applied to tune the CSE dialogues. **Figure 5-5** depicts the dialogues processing workflow where each stage, along with the individual tasks below, is shown.

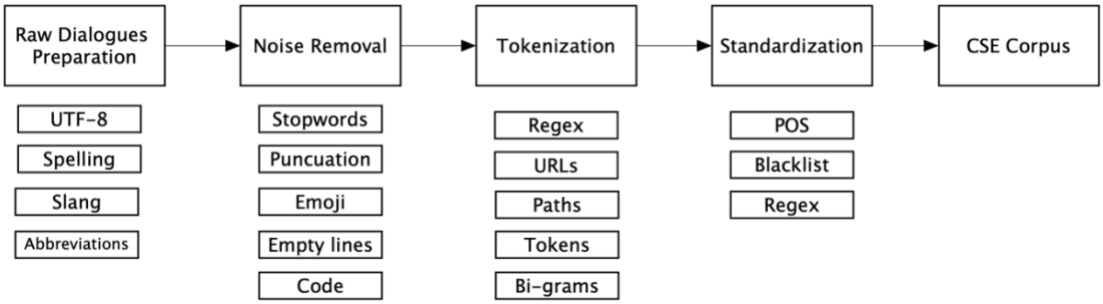

Figure 5-5. *The CSE dialogues processing workflow.*

At the end of this step, a corpus composed of CSE dialogues was formed. **Table 5-2** presents the summary of the produced corpus, named *CSE corpus*.

Table 5-2. *CSE Corpus summary.*

Corpus Name	CSE Corpus
Collection methods	Web scraping, pattern-matching text extraction
Corpus size	(N) 56 text dialogues/3380 sentences
Vocabulary size	(V) 4500 terms
Content	chat-based dialogues
Collection date	June 2018–December 2020
Creation date	June 2021

The CSE Corpus is composed of realized and fictional CSE attack dialogues. The existence of fictional attack dialogues does not affect its quality and capability because, similarly, a social engineer does not always act spontaneously but frequently prepares a fictional scenario (pretext attack) to guide the conversation and unleash his attack. The same applies to the CSE corpus, which incorporates a combination of confirmed and fictional dialogues of CSE attacks.

5.4. PHASE 2 - CSE Corpus Annotation

Following the above methodology outlined in Section 5.2.2, a plan was devised, and a specific goal was established for annotating the CSE corpus. Afterward, a simple model was developed to represent the annotation task in abstract terms, and a CSE ontology

was created to represent the in-context entities and their interconnection. Next, a specifications file was created where the entities and interconnections were described in a formal way and became tags. This file, along with the corpus guidelines, was given to the annotators to perform the annotation task. After the annotators finished their job, an inter-annotator agreement assessment took place using Cohen's Kappa metric, and the gold standard version of the CSE Corpus was finally produced.

5.4.1. STEP 1 - Goal Declaration

The annotation goal was to create the appropriate semantic target to facilitate the CSE attack recognition by assigning the correct tag to in-context words in a sentence. The labeling of all pertinent words, word sequences, or text spans within the context of CSE attacks was imperative to facilitate efficient Named Entity Recognition (NER) (Shelar et al. 2020) or text classification processes. Each word or text span was labelled with a type identifier (tag) drawn from a vocabulary created based on the CSE ontology and indicated *what* various terms denote in the context of a CSE attack and *how* they interconnect between them.

As mentioned before, SME employees are vulnerable to sensitive data leakage of type *Personal*, *IT*, and *Enterprise* details. Moreover, ML/NLP algorithms are more efficient for recognizing terms that belong to abstract classes than terms that belong to more specific subclasses. Thus, the aim during annotation was to identify and assign the correct ontology tag to words or text based on the CSE ontology.

Further refinement of the goal resulted in defining the following objectives:

- Identify keywords, syntax, and semantic characteristics to recognize *Personal* data leakage. If found, label with appropriate tag.
- Identify keywords, syntax, and semantic characteristics to recognize *IT* data leakage. If found, label with appropriate tag.
- Identify keywords, syntax, and semantic characteristics to recognize *Enterprise* data leakage. If found, label with appropriate tag.
- Identify noun-verb combinations and verb repetitions based on blacklists.

5.4.2. STEP 2- Model and CSE Ontology Creation

An abstract model that practically represented the aforementioned goal was defined. A three-category classification (Personal, IT, Enterprise) was introduced to be the basis of this abstract model for identifying CSE-related terms in a dialogue. The model M consists of a vocabulary of terms T, the relations between these terms R, and their interpretation I. The triple in Formula (1) represents the model

$$M = < T, R, I > \quad (1)$$

where:

T = {CSE_Ontology_term, Personal, Enterprise, IT}

R = {CSE_Ontology_term:: = Personal | Enterprise | IT}

I = {Personal = "list of personal terms in vocabulary ", IT = "list of Information Technology terms in vocabulary ", Enterprise = "list of enterprise/corporate terms in vocabulary "}

The CSE ontology was based on the concept map presented in 5.2 and is asset-oriented as an *asset* is defined in cybersecurity ontologies (Souag, Salinesi, and Comyn-Wattiau 2012). The development of this ontology provided valuable support to this work as it enables the grouping of similar concepts and relationships within the context. This CSE ontology, in order to be useful, must meet the following requirements:

- focus on assets (sensitive data) that could leak from an SME employee.
- include only the necessary concepts in-context.
- not exceed three levels of depth because that would lead to difficulties for algorithms and annotators to recognize the concepts.

The ontology was created in a semi-automated manner using a custom Information Extraction System (IES) to acquire structured knowledge about the related concepts. Several documents (Corporate IT Policies, IT professionals' CVs, ICT Manuals, etc.) were used as input to the IES. Subsequently, regular expressions rules were used to extract related concepts and relations.

Figure 5-6 illustrates the process of the IES workflow.

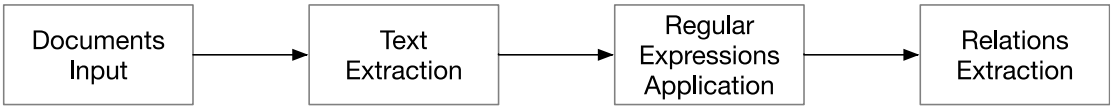

Figure 5-6. *The IES workflow.*

The proposed CSE ontology extends the ontology presented in (T. Li and Ni 2019), which connects social engineering concepts with cybersecurity concepts. The interest is centered around protecting sensitive data leakage in personal, IT, and enterprise contexts. The implemented ontology was finalized using Protégé ('Protégé') software.

An excerpt of the CSE ontology created in Protégé, is illustrated in **Figure 5-7** along with the arc types.

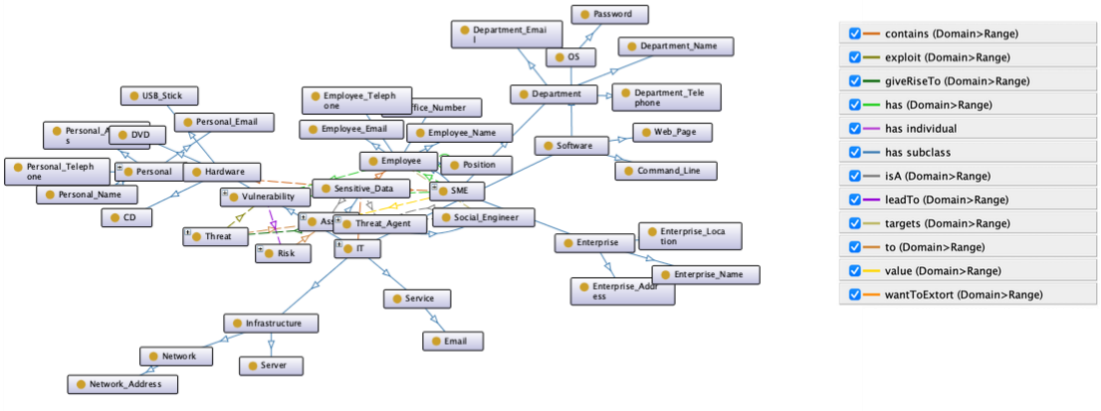

Figure 5-7. *Excerpt of the proposed CSE ontology.*

In the following **Figure 5-8**, the core concepts of the proposed CSE ontology are presented.

```
Sensitive Data ⊑ Asset                    Personal Address ⊑ Personal
Hardware ⊑ IT                             Personal Name ⊑ Personal
Infrastructure ⊑ IT                       Personal Telephone ⊑ Personal
Service ⊑ IT                              Risk
Software ⊑ IT                             Department ⊑ SME
Personal Email ⊑ Personal                 Employee ⊑ SME
Threat                                    Enterprise ⊑ SME
Vulnerability                             Social Engineer ⊑ Threat Agent
```

Figure 5-8. *The CSE ontology core concepts.*

Figure 5-9 presents an excerpt of the CSE ontology's object properties.

```
contains
∃ contains Thing ⊑ Information
⊤ ⊑ ∀ contains SME
⊤ ⊑ ∀ contains Employee
⊤ ⊑ ∀ contains Personal
⊤ ⊑ ∀ contains IT

exploit
∃ exploit Thing ⊑ Threat
⊤ ⊑ ∀ exploit Vulnerability

giveRiseTo
∃ giveRiseTo Thing ⊑ Threat Agent
⊤ ⊑ ∀ giveRiseTo Threat

has
∃ has Thing ⊑ Employee
∃ has Thing ⊑ SME
⊤ ⊑ ∀ has Employee
⊤ ⊑ ∀ has Vulnerability

wantToExtort
∃ wantToExtort Thing ⊑ Threat Agent
⊤ ⊑ ∀ wantToExtort Asset

isA
∃ isA Thing ⊑ Social Engineer
∃ isA Thing ⊑ Information
⊤ ⊑ ∀ isA Threat Agent

⊤ ⊑ ∀ isA Asset

leadTo
∃ leadTo Thing ⊑ Vulnerability
⊤ ⊑ ∀ leadTo Risk

targets
∃ targets Thing ⊑ Social Engineer
⊤ ⊑ ∀ targets Employee

to
∃ to Thing ⊑ Threat
∃ to Thing ⊑ Risk
⊤ ⊑ ∀ to Asset

value
∃ value Thing ⊑ SME
⊤ ⊑ ∀ value Asset
```

Figure 5-9. *Excerpt of Object properties.*

5.4.3. STEP 3 - Specifications Definition

The produced specification file holds the annotation schema and uses XML DTD (Murata, Laurent, and Kohn 2021) representation. It describes the model and the CSE ontology by turning the abstract ideas and entities/relations into tags and attributes. Following a clear and simple approach for the specifications file helps to build better and more accurate prediction models at a later stage. An excerpt of the produced specification file is depicted in **Figure 5-10**:

```
<!ENTITY name "CSE-Corpus_v1.0">
<!ELEMENT actor (#PCDATA)>
<!ATTLIST actor id ID prefix="ACTOR_" #REQUIRED>
<!ATTLIST actor role (attacker|victim)>
<!ELEMENT personal (#PCDATA)>
<!ATTLIST personal id ID prefix="PERSONAL_" #IMPLIED>
<!ATTLIST personal type (Personal_Address|Personal_Email|Personal_Name|Personal_Telephone)>
<!ELEMENT enterprise (#PCDATA)>
<!ATTLIST enterprise id ID prefix="ENTERPRISE_" #IMPLIED >
<!ATTLIST corporate type (Enterprise_Address Enterprise_Location|Enterprise_Name )>
<!ELEMENT it (#PCDATA)>
<!ATTLIST it id ID prefix="IT_" #IMPLIED>
<!ATTLIST it type (Hardware| Infrastructure |Service |Software)>
```

Figure 5-10*. Excerpt of the specifications file.*

5.4.4. STEP 4 - Annotator Guidelines

Two cybersecurity experts with heavy knowledge of the SMEs ecosystem were selected as the annotators and assigned the task of performing annotation at the sub-sentence

level. The annotators were also members of the annotation schema development team. They performed a stand-off annotation by character location because, in this case, the metadata type tags are stored separately from the original chat text. By employing this approach, the original text format was preserved, enhancing the efficiency of reading the original text. Moreover, this way of work enables effective handling of overlapping tags in the event of merging different annotation schemes in the future.

Over twenty different annotation tools (open source and commercial) were tested during the annotation project. The decision was made to utilize GATE ((Wilcox and Bhattacharya 2016)) and Prodigy ('Prodigy · An Annotation Tool for AI, Machine Learning & NLP') for their comprehensive functionality and ability to construct comprehensive natural language pipelines encompassing various NLP tools.

Annotators were given guidelines to direct them about tagging the important terms in a compliant way based on the schema. Several examples were also given to help resolve ambiguous term cases. First, the different tags were explained, and prioritization guidelines were given for the case of an ambiguous term that could belong to more than one category. For example, if an outsider asks for the (First or Last) Name of a department's supervisor, the answer can be tagged by Personal and Enterprise tags simultaneously. Annotators were advised to follow this order:

1. prefer the IT category over (Enterprise category or Personal category)
2. prefer the Enterprise category over the Personal category

This priority order ensures that IT details have greater importance over all other categories, and Enterprise details have greater importance over Personal details.

More specifically, an excerpt from the annotators guidelines follows:

1. Each text span may be tagged with *Personal*, *IT* or *Enterprise* main tags
2. If a sentence has entities that can fit in two or more tags, then follow the following priority IT > Enterprise > Personal
3. Prefer individual words to word combinations
4. The tag *Personal* is assigned to whatever information is related to a person. e.g., first name, Last Name. Consult CSE ontology
5. The tag *IT* is assigned to whatever information is related to the Information Technology, e.g., USB stick, computer, software and others. Consult Ontology
6. The tag *Enterprise* is assigned to whatever information is related to enterprises and enterprise environment, e.g., Department, office, positions, resumes and others. Consult CSE ontology
7. A sentence can have no important words. This sentence is called Neutral
8. Noun-verb combinations and verb repetitions will be identified automatically based on blacklists

Table 5-3 lists a sample of the examples that were given to the annotators where the important terms are in red colors, and the tag assigned to each one of them is on the right column.

Table 5-3*.Examples of CSE Corpus*

No#	Sentence	Tag
1	You will find contact numbers on the *Intranet*	IT
2	Let me get that name again and give me your *employee number*	Enterprise
3	Our *project leader* is Jerry Mendel	Enterprise
4	I can cut some corners and save some time, but I'll need your *username* and *password*	IT
5	Okay, can you tell me again your *employee ID number*.	Enterprise
6	I'll just *hit reset* and the old one will be wiped out	IT
7	For *account identification* may you please provide your *account number*	IT
8	Tom, Its Eddie… go ahead and try your *network connection*	IT
9	May I have your *full name* and *email address* please	Personal
10	Rest assured that they will be able to read the chat transcript as well as the documentation in your case	-

5.4.5. STEP 5 - Annotation Task

The annotation task performed had the target to label the words of CSE Corpus based on their semantic and syntactic characteristics. The two cybersecurity experts were responsible for labelling the words based on their semantic characteristics, and thus performed semantic annotation. By annotating the semantic characteristics of the words, the background information in each dialogue was linked with the CSE ontology. The syntactic characteristics of the words were labeled using a custom annotation software designed specifically for this purpose. This software enabled syntactic annotation by extracting statistical information related to the words employed by social engineers. **Table 5-4** showcases the two distinct types of annotation and the specific responses that were sought after.

Table 5-4. *Questions per annotation type.*

Syntactic	Semantic
What Part-of Speech (POS) type is it?	Is it a CSE ontology entity?
Is it a bi-gram?	What role does it play in the CSE ontology?

The syntactic characteristics that were annotated are the POS type, and the bi-grams which denote the syntactic relationship between pairs of words. Through the process of tagging the POS type and bigrams, subsequent phrase structure analysis became feasible, enabling the extraction of dependency trees associated with the respective phrases. This extra information can be used by parsers for Named Entity Recognition, n-grams identification, and Bag-of-Words representation.

The labelled words or text spans were further processed in order to extract statistical information valuable for the task of CSE attack recognition. Furthermore, for each sentence, a plethora of counters was also attached as metadata. This metadata information included vocabulary words frequency, words sequence pattern, words context relativity, urgency indicators, number of URLs or paths appeared, blacklisted verb and noun combinations, and backlisted bi-grams (e.g., tech support, USB stick and others). Moreover, the sentences similarity was measured to be used as attacker persistence metric; i.e., if the attacker uses different sentences but the sentences are highly similar, this means that the attacker persists on his initial intent; this information counts also as metadata information for the annotation task.

GATE and Prodigy handled all of the annotation steps along with WordNet (Miller 1995) and LIWC (Pennebaker et al., 2021) software. **Figure 5-11** illustrates a tree-structured taxonomy of the syntactic and semantic annotation targets.

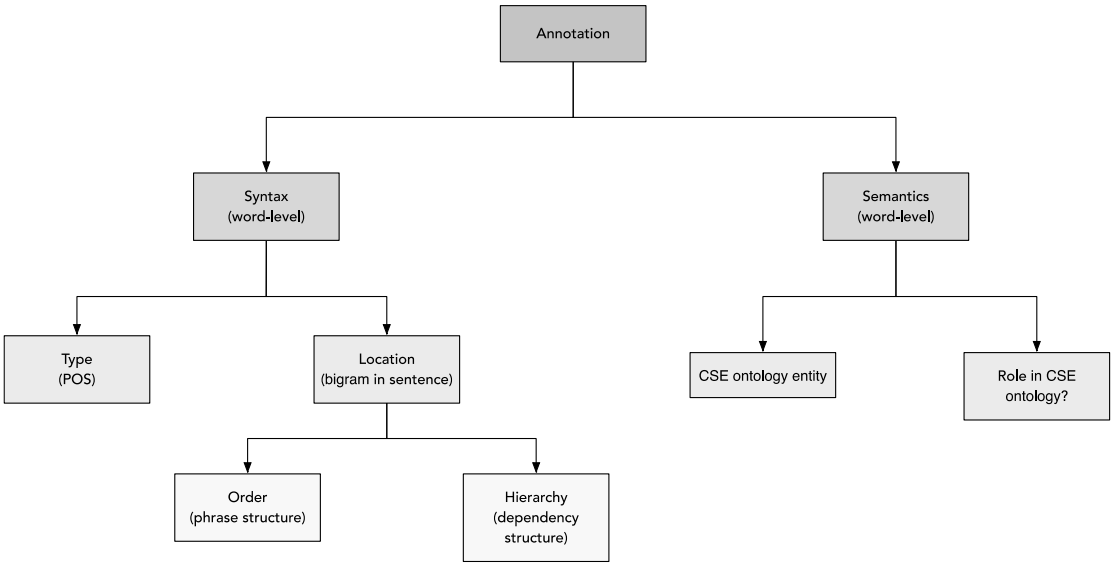

***Figure 5-11**. Annotation targets.*

The preferred encoding scheme to tag the existing chunks of text (word, or text spans) was based on the IOB format (Ramshaw and Marcus 1995). In the IOB encoding scheme, the "*I-* "prefix of a tag indicates that the tag is inside a chunk, "*O*" indicates that the tag does not belong to a chunk and the "*B-* "prefix of a tag indicates that a token is the beginning of a chunk.

For example, the sentence 'Mr. Robinson is the Head of Financial Department, and his office is on the second floor' using the IOB format is presented in **Table 5-5**.

***Table 5-5.** IOB encoding example.*

Chunk	**Encoding**
Mr	B-EMP-NAME
Robinson	I-EMP-NAME
is	O
the	O
head	I-POS
of	O
Financial	B-DEPT-NAME
Department	I-DEPT-NAME
and	O
his	O
office	I-DEPT
is	O
on	O
the	O
second	I-OFF-NUM
floor	I-OFF-NUM

During the annotation task, the words were labelled based on their semantic and syntactic characteristics. This way, relationships were extracted between words or text spans belonging to different branches of the ontology. Moreover, hidden patterns that attackers use in a conversation were discovered and valuable information about how different CSE concepts and ontology entities interact was extracted. For example, in **Figure 5-12**, where the semantic categories are labelled with tags in brackets, we can discover that an *Employee_Name* has an "IS" relation with *Position* and an "OF" relation with *Department_Name.*

```
Mr Robinson [Employee_Name] is the Head [Position] of Financial Department [Department_Name]
 and his office [Department] is on the second floor [Office_Number]
```

Figure 5-12. *Discovery example.*

5.4.6. STEP 6 - Inter-Annotator Agreement

The inter-annotator agreement (IAA) was validated using Cohen's Kappa (McHugh 2012), which is one of the most popular statistics to measure the agreement between two annotators of N terms on m categories (Craig 1981). Formula (2) was used to calculate Cohen's Kappa for two annotators:

$$k = \frac{p_0 - p_e}{1 - p_e} = 1 - \frac{1 - p_0}{1 - p_e} \qquad (2)$$

where p_0 is the relative observed agreement among annotators and p_e is the hypothetical probability of chance agreement.

The produced CSE Corpus has N = 4500 terms and m = 3 categories, and both annotators (A and B) agreed for the *personal* category 1665 times, for the *enterprise* category 1442 times and for the *IT* category 1194 times. **Table 5-6** contains a contingency matrix where each x_{ij} represents the multitude of terms that annotator A classified in category i, but Annotator B is classified in category j, with i, j = 1, 2, 3. The proportions on the diagonal, x_{ii}, represent the proportion of terms in each category for which the two annotators agreed on the assignment.

Table 5-6. *Contingency matrix.*

Annotator		**B**	**B**	**B**	**Total**
	Category	**personal**	**enterprise**	**IT**	
A	**personal**	1665	63	13	1741
A	**enterprise**	59	1442	31	1532
A	**IT**	14	19	1194	1227
Total		1738	1524	1238	4500

The observed agreement is calculated as: $P(o) = \frac{1665+1442+1194}{4500} = 0.956\ (95.6\%)$

The expected change agreement, thus the proportion of terms that would be expected to agree by chance is given by formula (3):

$$P(e) = \frac{\frac{(1741x1738)}{4500} + \frac{(1532x1524)}{4500} + \frac{(1227x1238)}{4500}}{4500} = 0.339. \quad (3)$$

thus, the Cohen's Kappa metric is, $k = \frac{p_0 - p_e}{1 - p_e} = \frac{0.617}{0.661} = 0.933$

Interpreting the Cohen's kappa value of 0.933, we can safely conclude that the level of agreement for the CSE Corpus annotation task was 'Almost Perfect'.

5.4.7. STEP 7 - Adjudication Task

After the two annotators finished the annotation task, having succeeded in an almost perfect level of inter-annotator agreement, the gold standard corpus was created by performing adjudication over the data. The gold standard corpus was the final annotated version of the CSE corpus. No annotator was part of the adjudication process. As anticipated, no difficulties were encountered owing to the significant level of agreement among the annotators. Furthermore, it is noteworthy to mention that there were no instances where the annotators agreed while the adjudicators, held a different opinion regarding the annotation tag. Thus, after the adjudication, the final *CSE Corpus* was produced with the annotation information stored in a different file accompanying each dialogue.

5.5. CSE Corpus presentation

The annotated CSE Corpus is composed of 56 dialogues, and 3380 sentences. The words that are in context and can be tagged by the CSE ontology are 4500. **Table 5-7** presents ten random utterances from the CSE corpus.

Table 5-7. *Ten random utterances from the CSE corpus.*

No #	Utterance
1	which computer servers does your group use
2	do you sign in under the username Rosemary
3	ok i am trying to logon now

4	trace it back to where its plugged
5	it could save us both big headaches the next time this network problem happens
6	did you turn your computer off
7	thanks a lot please wait for my email
8	take care and bye for now
9	my flex 2 is charged but wont turn on is it normal
10	are you experiencing any computer issues presently

As depicted in **Figure 5-13**a, the distribution of the sentence categories based on the aforementioned tags shows that the produced CSE Corpus is well balanced. Valuable information can also be extracted by observing the average word count per category, as seen in **Figure 5-13**b.

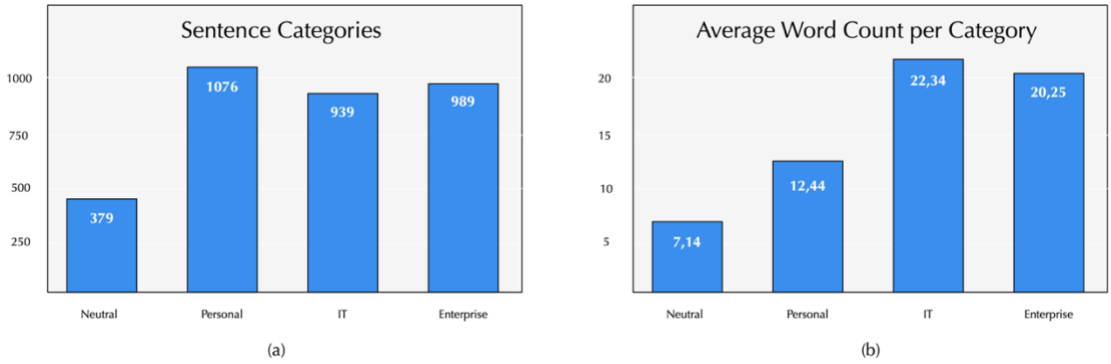

Figure 5-13. *(a) Distribution of sentence categories, (b) Average word count per category.*

Several interesting statistics were extracted that gave us information regarding the content of the sentences. Indicatively, **Figure 5-14** illustrates the five most frequent words per category and **Figure 5-15** illustrates the five most frequent bi-grams per category.

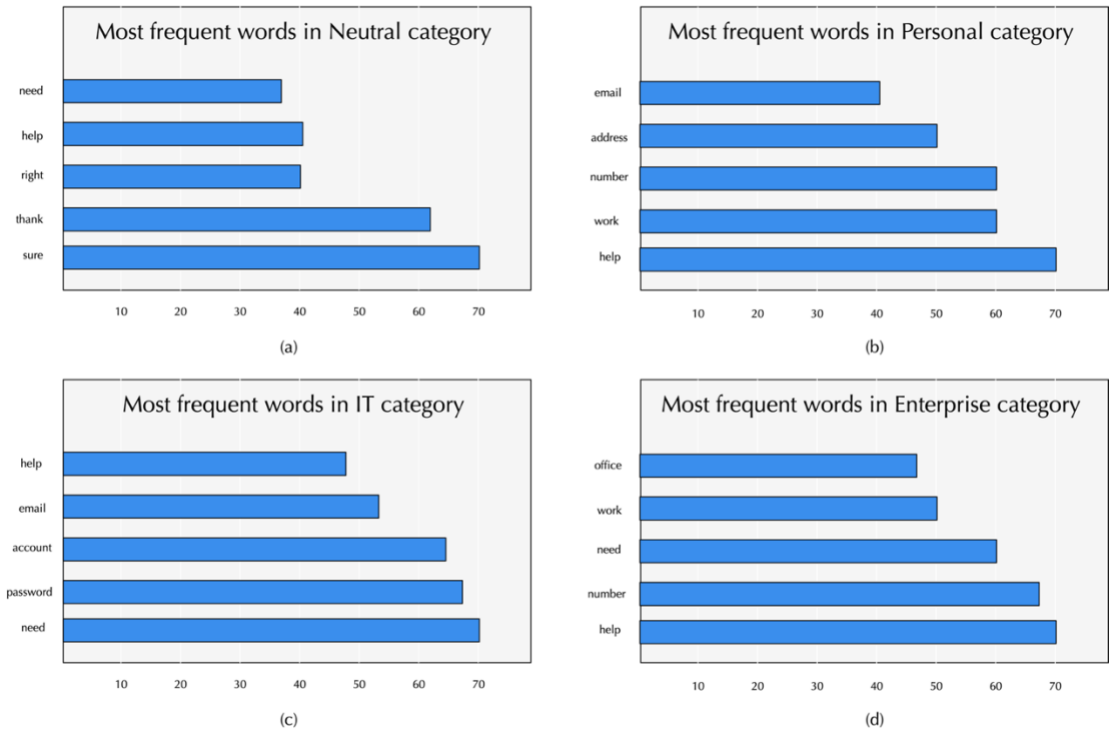

Figure 5-14. *Five most frequent words per category. (a) Neutral, (b) Personal, (c) IT, (d) Enterprise*

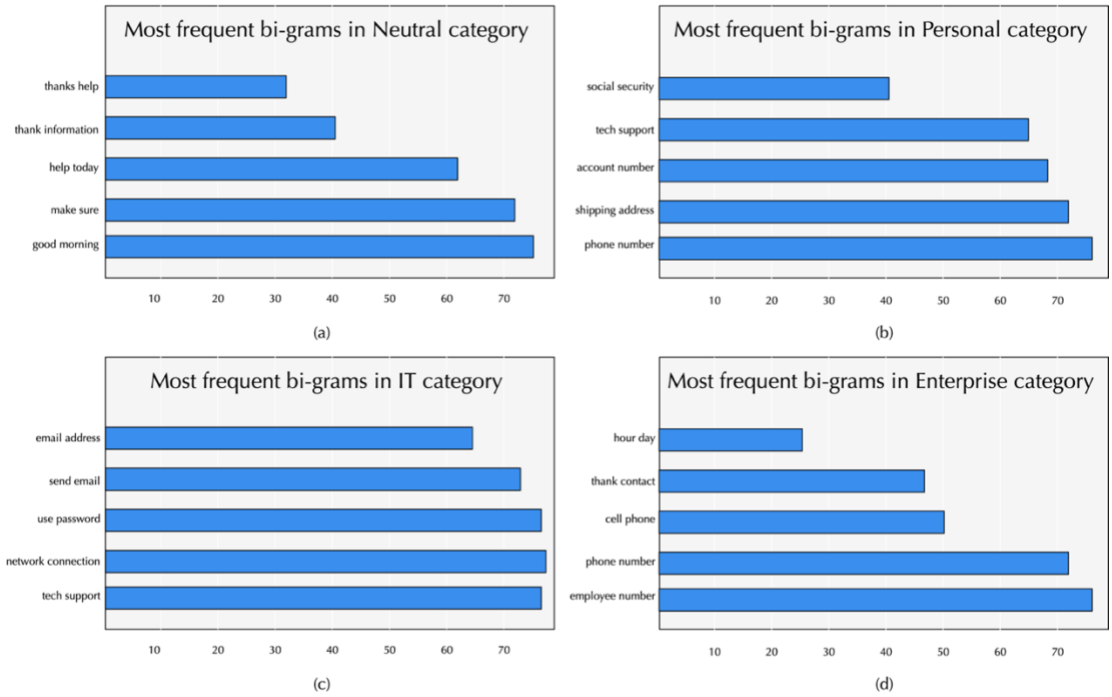

Figure 5-15. *Five most frequent bi-grams per category. (a) Neutral, (b) Personal, (c) IT, (d) Enterprise*

5.1. Chapter Conclusion

Chapter 5 focused on the comprehensive exploration of the CSE Corpus, which serves as a fundamental resource for studying and understanding CSE attacks. The chapter provided valuable insights into the building process of the corpus, including source selection, dialogue collection, enrichment, linguistic analysis, and annotation. The thorough examination of the corpus-building process contributes to the development of robust models and methodologies for recognizing and mitigating social engineering attacks.

The creation of the CSE Corpus involved meticulous steps, beginning with the selection of reliable and diverse sources to capture the real-world dynamics of CSE attacks. The dialogues were then collected, enriched, and subjected to rigorous linguistic analysis and preprocessing techniques to ensure the quality and integrity of the corpus. The annotation process involved the declaration of goals, the creation of models and ontology, specification definition, and the development of annotator guidelines. The inter-annotator agreement and adjudication tasks were performed to enhance the reliability and consistency of the corpus annotations.

The availability of the CSE Corpus provides researchers and practitioners with a valuable resource for studying CSE attacks, enabling the development and evaluation of robust models and methodologies. By leveraging the annotated corpus, researchers can train machine learning algorithms and perform comprehensive analyses to gain deeper insights into the patterns, strategies, and vulnerabilities associated with social engineering attacks.

Chapter 5 presented a detailed exploration of the CSE Corpus, encompassing its creation process, linguistic analysis, and annotation. The corpus constitutes a crucial resource for advancing research in the domain of CSE attacks, laying the groundwork for forthcoming chapters where innovative models and techniques are proposed and evaluated. The availability of such a corpus greatly contributes to the development of effective defense mechanisms, enabling the recognition and prevention of social engineering attacks in real-world scenarios.

6. CRITICAL INFORMATION LEAKAGE RECOGNIZER (CRINL-R)

6.1. Introduction

Critical information leakage refers to the unintentional or unauthorized disclosure of sensitive information during a conversation. CRINL-R utilizes the named-entity recognition (NER) technique (Collobert et al.,), a technique that extracts contextual information from unstructured text data, such as chat text. Pre-trained NER models can be fine-tuned to recognize additional entities by training them with an annotated corpus containing the relevant tags. CRINL-R utilizes NER to process dialogue sentences of the CSE corpus where personal, IT, and enterprise information have been identified and labeled. CRINL-R identifies the words or text spans that are labeled with a tag associated with an entity from the CSE ontology (Tsinganos and Mavridis 2021).

CRINL-R utlizes a bi-directional Long Short-Term Memory neural network (bi-LSTM) (Lample et al. 2016) for CSE attack named entities recognition. LSTMs were designed to overcome Recurrent Neural Networks (RNNs) inefficiency, since RNNs fail to learn long dependencies due to bias toward their most recent inputs. RNNs are a family of neural networks that operate on sequential data; they take as input a sequence of vectors (x_1, x_2…, x_N) and produce as output another sequence that represents information about every step in the input sequence. LSTMs incorporate a memory cell to capture long-range dependencies. Using several gates, LSTMs control the proportion of input that is given to the memory cell, as well as the proportion from the previous state that should be forgotten.

6.2. Model

Named Entity Recognition (NER) (Shelar et al. 2020) is a natural language processing method to extract in-context information from unstructured text data such as e-mails, web articles or chat-based dialogues. Using NER we can identify entities, like people, organizations, places etc., which exist in text. There are several well-known libraries like SpaCy NLP library (‘spaCy · Industrial-Strength Natural Language Processing in Python’), Apache OpenNLP (‘Apache OpenNLP’) and TensorFlow (‘TensorFlow’) with pretrained NER models that can be used to build a NER system that identifies the aforementioned entities. Furthermore, these pretrained models can be modified to recognize additional entities by training them with an annotated corpus that contains

the additional entities (tags). Such a corpus is the already presented CSE Corpus, where personal, IT and enterprise information has been identified and tagged.

CRINL-R takes as input preprocessed CSE dialogue sentences and identifies, in each sentence, the words or text spans that can be labelled by a tag that is associated with an entity of the CSE Ontology. This kind of task is a sequence-labelling task where a token is classified as belonging to one or none of the annotation classes. Keras ('Keras: The Python Deep Learning API') and TensorFlow were used to build, train and evaluate a standard bi-directional Long Short-Term Memory neural network (bi-LSTM) (Lample et al. 2016) for CSE attack named entities recognition.

To build the CRINL-R NER system the following steps were taken:

1. Preprocessing*:* The CSE Corpus, which has already been annotated in IOB format, contains lines that can easily be read and stored as a list of token-tag pairs. Then, each token was represented by a word embedding using a pre-trained English language model of the SpaCy NLP library.
2. Building*:* Using Keras, a bi-LSTM model was constructed, which is comprised of two compound layers:
    - *Bi-directional LSTM layer*, where the forward and backward pass were encapsulated and the input and output sequences returned were stacked.
    - *Classification layer*, where classification was performed to every position of the sequences in the stack. The SoftMax (Bridle 1989) activation function was used to scale the output and obtain sequences of probability distributions.
3. Training: To train the model in Keras, a loss function for the model was specified to measure the distance between prediction and truth, and a batch-wise gradient descent algorithm was specified for optimization.
4. Assessing*:* The performance assessment of the model was conducted by applying the model to the preprocessed validation data. For each sample dialogue sentence and each token in a sentence, a tensor was obtained that contained a predicted probability distribution over the tags. The tag with the highest probability was chosen, and for each sentence and each token the true and predicted tags were returned.

More specifically, inside the bi-LSTM neural network the following actions occurred in sequence:

1. Each dialogue sentence was split into a sequence of token-tag pairs.

2. Each token-tag pair was represented by a word embedding vector.
3. The word embeddings were pretrained to encode semantic information. This approach is known as Transfer Learning (Lu et al. 2015).
4. Going forward the bi-LSTM model at each step:
    a. the input vector was read and combined with the internal memory
    b. the output vector was produced and the internal memory was updated
5. This sequence of actions was performed by the bi-LSTM model to the input sequence and produced an output sequence of the same length.
6. Going backwards, the model read the input sequence again and produced another output sequence.
7. At each position, the outputs of steps 4 and 5 were combined and fed into a classifier which outputted the probability for the input word, at this position, which should be annotated with the Personal, IT or Enterprise tag.

6.3. Corpus

CRINL-R was trained using the CSE Corpus and its characteristics are presented in **Table 6-1**

Table 6-1. *CSE Corpus details*

Characteristic	Value
Corpus Name	CSE Corpus
Collection Method	Web-scraping, pattern-matching text extraction
Corpus size	(N) 56 text dialogues/3380 sentences
Vocabulary size	(V) 4500 terms
Total no. of turns	3380
Avg, tokens per turn	7.97
Content	Chat-based dialogues
Collection date	Jun 2018 – Dec 2020
Language	English
Release year	Jun 2021
License	Private

6.4. Training

The training of the bi-LSTM neural network involved a sequential execution of several steps. Each sentence within the dialogue was split into a sequence of token-tag pairs

where each pair represented a token and its corresponding tag. Next, the token-tag pairs were transformed into word embedding vectors. These word embeddings were generated using the spaCy NLP library, which had pre-trained models capable of encoding semantic information into the embeddings. Each token-tag pair was mapped to its respective word embedding vector. At each step of the bi-LSTM model, an input vector, representing a token-tag pair, was read and combined with the internal memory of the model. This combination allowed the model to take into account the information from previous steps. Subsequently, the model produced an output vector while updating its internal memory. This sequence of actions was performed by the bi-LSTM model for each input token-tag pair, resulting in an output sequence of the same length as the input sequence. To further refine the training process, the model underwent a backward pass. It re-read the input sequence in reverse and generated another output sequence. At each position in both the forward and backward passes, the outputs were combined. These combined outputs were then fed into a classifier, which produced a probability for each input word at that particular position. This probability indicated the likelihood of the word being annotated with the Personal, IT, or Enterprise tag. By following this series of steps, the bi-LSTM model learned to understand the relationships between the input tokens and their corresponding tags, enabling it to predict the appropriate tags for new, unseen data.

6.5. Results

The training and validation performance results of CRINL-R are presented in **Figure 6-1**

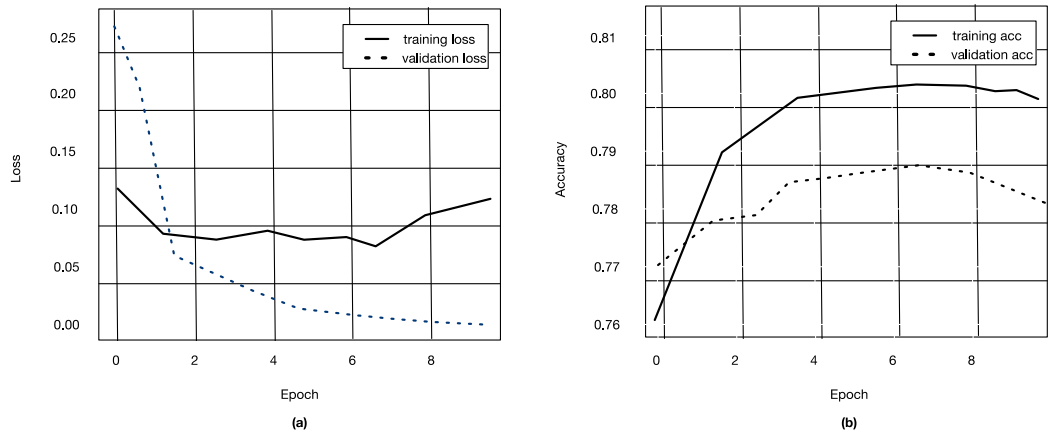

Figure 6-1. *CRINL-R (a) Loss metrics, (b) Accuracy metrics*

After training the model for 10 epochs, the achieved scores are presented in **Table 6-2**:

Table 6-2. *CRINL-R performance results on identifying terms related to Personal, Enterprise, or IT*

tag	F1	Precision	Recall	Support
PERSONAL	0.67	0.63	0.51	65
IT	0.86	0.93	0.64	122
ENTERPRISE	0.71	0.84	0.56	89

The results are satisfactory, taking into account the relatively small size of the CSE corpus. Further performance improvement can be achieved by tuning the model in one of the following ways:

- Enriching the CSE Corpus, i.e., adding sentences where more critical nouns belonging to one of the three sensitive data categories are substituted by similar words.
- Replacing the last feed-forward layer with a conditional random field (CRF) model,
- Reducing the imbalance of the tag distribution, e.g., by using a different loss function,
- Providing more input by tagging new coming CSE dialogues and thus training more data.

6.6. Chapter Conclusion

Chapter 6 delved into the development and evaluation of the CRINL-R model for CSE attack recognition. The chapter presented an in-depth exploration of the model, including its architecture, training process, and evaluation results. Through this analysis, valuable insights were gained regarding the effectiveness of the CRINL-R model in recognizing critical information leakage and its potential impact on mitigating social engineering attacks.

The CRINL-R model showcased a sophisticated architecture designed to analyze and recognize patterns in language that indicate the leakage of critical information during chat-based interactions. By leveraging deep learning techniques and training the model on a carefully curated dataset, the CRINL-R model exhibited promising performance in identifying instances of critical information leakage. The evaluation results demonstrated its ability to effectively recognize and flag potential social engineering attacks, thereby enhancing the overall security posture in chat-based communication channels.

The development and evaluation of the CRINL-R model significantly contribute to the field of CSE attack recognition. By successfully integrating deep learning algorithms

into the recognition process, the model showcases the potential of leveraging advanced machine learning techniques for combating social engineering threats. The findings of this chapter pave the way for further research and refinement of the CRINL-R model, as well as the exploration of other deep learning approaches for improved attack recognition and prevention.

Chapter 6 provided a comprehensive examination of the CRINL-R model, highlighting its architecture, training process, and evaluation results. The successful development and evaluation of the model underscore its potential in recognizing critical information leakage and mitigating CSE attacks. The insights gained from this chapter contribute to the ongoing efforts to enhance the security of communication channels and foster the development of robust defense mechanisms against social engineering threats.

7. PERST-R

7.1. Introduction

In psychology, the term "human personality" denotes the unique variations in patterns of cognition, affect, and behavior that distinguish one individual from another. The Five-Factor Model (FFM) (Salgado 2002), also known as the Big-5 Theory, represents a widely recognized framework for the classification of personality traits. The Big-5 comprises five factors that are thought to account for the majority of individual differences in personality.

The five personality traits have also been linked with high or low suppressibility to CSE attacks as presented in section 4.3. Having a system that recognizes personality traits can also provide valuable insights into the behavior and decision-making of humans. The main objective of the PERST-R model is to identify the personality traits, as defined in the Big-5 theory utilizing the chat dialogue.

Psychology plays a significant role in the context of CSE attacks. Social engineers employ various psychological tactics to manipulate and exploit the vulnerabilities of their targets. By understanding the principles of psychology, including cognitive biases, emotional triggers, and social influence, researchers can gain valuable insights into the techniques employed by social engineers. Psychological factors such as trust, authority, and reciprocity are exploited by social engineers to establish rapport and gain the trust of their targets. They may use persuasive language, mimic the communication style of the target, or employ social norms to create a sense of familiarity and credibility. Additionally, social engineers often leverage cognitive biases, such as the confirmation bias or the halo effect, to manipulate the perceptions and decision-making processes of their targets.

Furthermore, an understanding of human emotions and their influence on behavior is crucial in combating CSE attacks. Social engineers may exploit emotions such as fear, excitement, or curiosity to manipulate their targets into divulging sensitive information or performing certain actions. By studying psychological theories and models, researchers can develop robust models and algorithms that recognize emotional manipulation attempts in chat-based interactions.

7.2. Model

The PERST-R model incorporates a pre-trained Bidirectional Encoder Representations from the Transformers (BERT) model to recognize personality traits. Initially, the BERT model is fine-tuned using a large corpus of text data that has been meticulously labeled for each of the five personality traits: Openness, Conscientiousness, Extraversion, Agreeableness, and Neuroticism. This fine-tuning process enables the model to specialize in accurately identifying and characterizing these personality traits based on textual inputs.

To develop the PERST-R model, a carefully selected corpus of textual data is utilized. This corpus comprises instances of individual behavior that can be leveraged to deduce various personality traits. The collected data undergoes a preprocessing stage where it is transformed and formatted to be compatible with the training requirements of the BERT model. The BERT model is then trained on this preprocessed corpus, to predict the likelihood of each of the five personality traits for a given input text.

The performance evaluation of the trained PERST-R model involves rigorous assessment measures, including accuracy, precision, recall, and F1 score. These metrics are computed on a held-out test dataset that serves as an independent benchmark for evaluating the model's proficiency in accurately recognizing and categorizing personality traits. Through this evaluation process, the effectiveness and reliability of the PERST-R model can be determined, providing insights into its ability to successfully identify and classify personality traits based on textual inputs.

In **Figure 7-1** the training workflow of the PERST-R model is depicted where the corpus is preprocessed and fed into the BERT model. The final output is a probability distribution over the five personality traits.

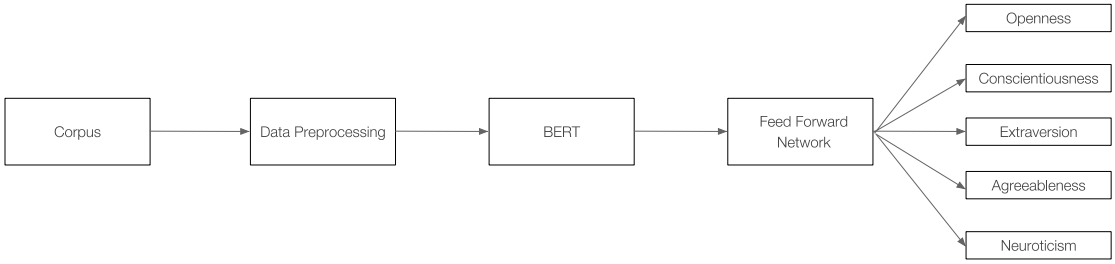

***Figure 7-1** - PERST-R model workflow*

The values of the hyperparameters for the training and optimization are presented in the following Table 7-1

Table 7-1 - PERST-R training and Optimization details

Parameter	**Value**
Hidden size	768
Hidden layers	12
Attention Heads	12
Activation function	gelu
Dropout probability	0.1
Positional Embedding	512
Optimizer	Adam
Learning rate	0.00001
Batch size	32
Loss Function	Cross-Entropy

To conclude regarding parameter values we followed a k-cross validation approach utilized according to the procedure depicted in Figure 7-2

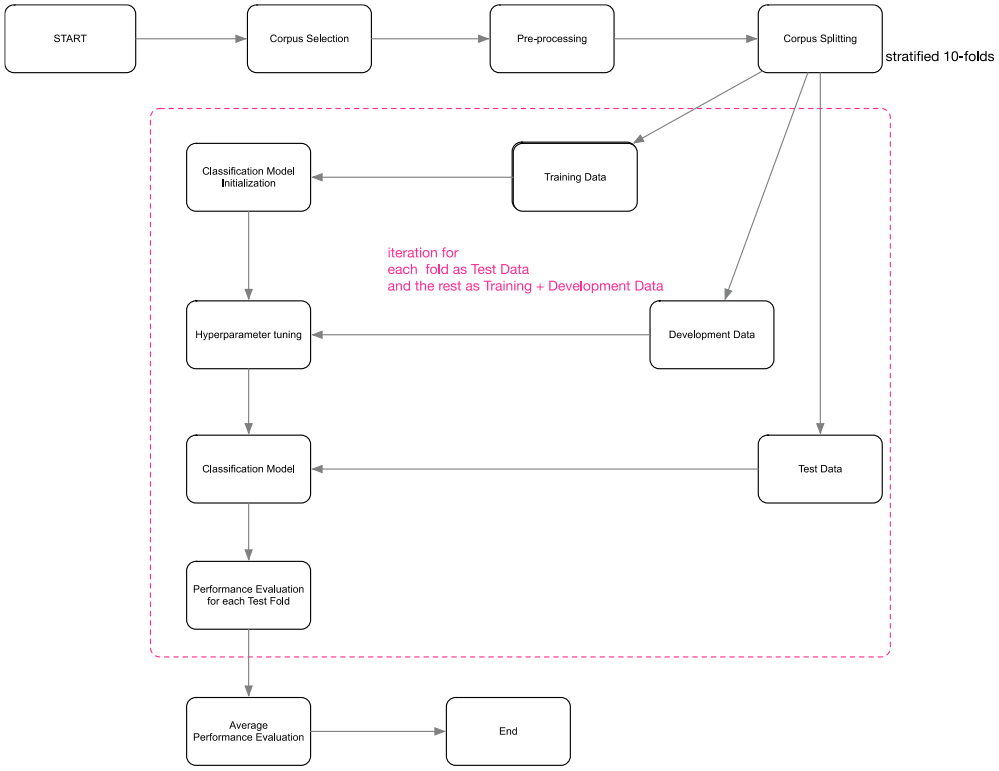

Figure 7-2 – 10-k Cross Validation

7.3. Corpus

The training corpus used is the FriendsPersona (Jiang, Zhang, and Choi, 2022) which is a large-scale conversational dataset that was constructed using scripts from the popular American TV show "Friends". It contains 1,175 dialogues between pairs of characters, totaling 105,784 utterances. The dataset is annotated with the Big-5 personality traits, with each dialogue annotated by three human raters. The FriendsPersona dataset is unique in that it provides both conversational data and personality trait annotations, which enables researchers to explore the relationship between personality traits and conversational behavior. In terms of inter-annotator agreement, the creators achieved an average pair-wise kappa of 54.92% among 2 annotators and Fleiss' kappa of 20.54% among 3 annotators across five personality traits. **Table 7-2** presents the corpus details

Table 7-2. *FriendsPersona corpus details*

Characteristic	**Value**
Total dialogues	1,175
Total utterances	105,784
Annotated personality traits	Openness, Conscientiousness, Extraversion, Agreeableness, Neuroticism
Annotation method	Three human raters per dialogue
Demographic information	Age, gender, occupation
Relationship labels	Friends, family, romantic partners
Source	Scripts from the TV show "Friends"
Language	English
Release year	2021
License	Creative Commons Attribution 4.0 International (CC BY 4.0)

7.4. Results

The PERST-R model achieves satisfactory accuracy in recognizing Big-5 personality traits, with an overall accuracy of 71,12%. **Table 7-3** and **Figure 7-3** present the performance results and the ROC graph. The area under the ROC curve (AUC) for each of the Big-5 personality traits was also impressive, with values of 0.83 for Openness, 0.62 for Conscientiousness, 0.79 for Extraversion, 0.57 for Agreeableness, and 0,80 for Neuroticism. These results demonstrate the effectiveness of the PERST-R model in accurately recognizing Big-5 personality traits and suggest that this approach could be useful in CSE attack recognition.

Table 7-3. *PERST-R Accuracy*

Model	Openness	Conscientiousness	Extraversion	Agreeableness	Neuroticism	Average
BERT	80,12	66,79	71,09	65,37	72,27	71,12

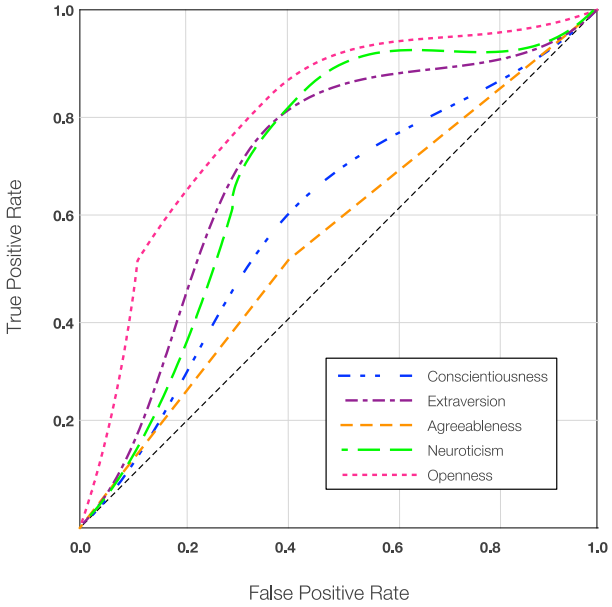

Figure 7-3. *PERST-R ROC*

7.5. Chapter Conclusion

Chapter 7 focused on the development and evaluation of the PERST-R model for recognizing personality traits in CSE attacks. The chapter provided a detailed description of the model, including its utilization of a pre-trained BERT model, fine-tuning process, corpus selection, training methodology, and evaluation metrics. Through this analysis, significant insights were gained into the efficacy of the PERST-R model in accurately identifying and characterizing personality traits based on textual inputs.

The PERST-R model leverages a pre-trained BERT model as its foundation, which is subsequently fine-tuned using a large corpus of labeled text data encompassing the five major personality traits. This fine-tuning process allows the model to specialize in the accurate recognition of individual traits such as Openness, Conscientiousness, Extraversion, Agreeableness, and Neuroticism. The selected corpus plays a crucial role in providing relevant behavioral instances that facilitate the deduction of personality traits.

During the development of the PERST-R model, the collected textual data undergoes preprocessing to transform it into a suitable format for training the BERT model. The model is then trained to predict the likelihood of each personality trait for a given input text. Evaluation of the PERST-R model's performance is conducted using

comprehensive metrics such as accuracy, precision, recall, and F1 score on a held-out test dataset, providing a robust assessment of its efficacy in personality trait recognition.

8. DIACT-R

8.1. Introduction

Language stands as one of humanity's most remarkable accomplishments and has been a driving force in the evolution of human society. Today, it is an indispensable component of both our professional and social lives. Dialogue, as a term, refers to interactive communication between two or more individuals or groups in the context of human language. It is a two-way intentional communication that can take on spoken or written forms. Language assumes a prominent role, especially in the context of cybersecurity and social engineering, as malicious users employ it to deceive and manipulate unsuspecting individuals. Among various social engineering attacks, CSE is recognized as a critical factor in the success of cyber-attacks, particularly in small and medium-sized enterprise (SME) environments. This type of attack is attracting increasing attention due to its potential impact and ease of exploitation. According to Verizon's 2023 report (‘DBIR Report 2023 - Master’s Guide’), the human element is a significant factor in driving 82% of cybersecurity breaches.

During a CSE attack, malicious users utilize linguistic manipulation to deceive their targets by exploiting human personality traits and technical misconfigurations. From a strategic standpoint, it is more effective (Tsinganos and Mavridis 2021) to isolate individual CSE attack enablers and investigate recognition methods for each enabler separately. In a chat-based dialogue, such as one between an SME employee and a potential customer, interlocutors communicate through written sentences. The ability to identify one or more characteristics that can reveal the malicious intent of an interlocutor can sufficiently safeguard the SME employee against potential CSE attacks. Dialogue act (DA) is a term used to describe the function or intention of a specific utterance in a conversation, and it has already been identified (Tsinganos and Mavridis 2021) as one of the critical enablers of successful CSE attacks. As such, recognizing dangerous DAs that may lead to a successful CSE attack is of paramount importance.

The widespread use of dialogue systems (such as Alexa and Siri) has led to a surge in research in this field. Thus, it is advantageous to transfer knowledge and terminology from this field to CSE recognition tasks. The formalism used to describe dialogue systems can facilitate the modeling of human-to-human conversations, especially CSE attacks, as both types of dialogues share common structural characteristics. Dialogue systems rely on dialogue state tracking (DST) to monitor the state of the conversation. Similarly, in human-to-human dialogues and CSE attacks, the state of the dialogue can be tracked through the DST process, which in this case is the state of the CSE attack. CSE attacks involve deception and manipulation in order to acquire sensitive

information, such as an attacker posing as a trusted individual (e.g., a bank representative) to trick the target into disclosing personal information. By using DST to track the dialogue state and identify when the attacker is attempting to extract sensitive information or deceive the victim, we can recognize and prevent CSE attacks.

By leveraging the advancements made in dialogue systems, a recognizer is developed to carry out CSE attack state tracking. The main component in the system's architecture is a BERT-based model that is trained and fine-tuned to recognize the intent of an interlocutor utilizing a multi-schema ontology and dialogue acts related to deception

8.2. Background

8.2.1. Dialogue Systems

According to (Jurafsky and Martin 2022), human-to-human dialogues possess several properties. A crucial structural attribute is the 'turn', which is a singular contribution made by one speaker in the dialogue. A full dialogue comprises a sequence of turns. For instance, in **Table 8-1**, two interlocutors identified as T(arget) and A(ttacker) exchange utterances in a small six-turn dialogue excerpted from the CSE Corpus (Tsinganos and Mavridis 2021). One or more utterances from the same speaker can be grouped into a single turn. Typically, an utterance from speaker T is followed by a response from speaker A, constituting an exchange. A dialogue consists of multiple exchanges.

Table 8-1. *A sample dialogue in turns*

Turn	Utterance
T1	We don't allow Telnet, especially from the Internet, it's not secure. If you can use SSH, that'd be okay
A2	Yeah, we have SSH.
A3	So, what's the IP address?
T4	IP is [ANON]
A5	Username and password?
T6	Username is [ANON] and password is [ANON]

Nowadays, a dialogue can be conducted between humans or between a machine and a human and the latter is called a dialogue system. Dialogue systems communicate with humans in natural language, both spoken and written and can be classified as:

- task-based dialogue systems where the system helps humans to complete a task.
- conversational dialogue systems where the systems answer questions.

Task-based systems can be further classified, based on their architecture, into systems that use:

- Genial Understander System (GUS) architecture (Bobrow et al., 2022) which is an old simple approach to describing a dialogue structure.
- Dialogue-State (DS) architecture, in which the dialogue is modeled as a series of different states.

The primary objective of dialogue systems when interacting with a human is to elicit three key pieces of information from the user's utterance: *domain classification*, *user intention*, and *requested information*. Domain classification pertains to the conversation's context and can be determined through the use of a domain ontology.

To ensure proper interpretation and response to user input, a formal representation of the knowledge that a conversational system has to comprehend must exist. This knowledge comprises concepts and entities mentioned in the conversation, the relationships between them, and the potential dialogue states, which are encapsulated in the domain ontology. Dialogue systems rely on ontologies to link the user's input to the corresponding concepts and entities, as well as to monitor the current dialogue state. By tracking the state of the conversation and the user's objectives, the system can generate more precise and pertinent responses. The domain ontology can be defined by means of a static schema, where the dialogue system operates within a single domain, or a dynamic schema, where the dialogue system is capable of operating within multiple domains. In this context, the schema represents a framework or structure that defines the arrangement of data.

Typically, a dialogue system consists of the following units (Jurafsky and Martin 2022):

- Natural Language Understanding (NLU) unit, which uses machine learning to interpret the user's utterance.
- Dialogue-State Tracking (DST) unit, which maintains the entire dialogue history.
- Dialogue Policy unit, which defines a set of rules to drive the interactions between the user and the system.
- Natural Language Generation (NLG), which generates responses in natural language.
- Text-to-Speech unit, which transforms text to speech.
- Automated Speech Recognition unit, which transforms speech into text

8.2.2. Dialogue Acts

The Speech Act theory, was introduced by Austin (Austin 1975) and Searle (Searle 1969), (Searle 2010) in the 1960s, and has become a widely used concept in linguistics

and literature studies. Today, the modern notion of speech act (Bach and Harnish 1979) has found applications in diverse fields such as ethics, epistemology, and clinical psychology (Kissine, 2022). Dialogue Acts are a type of Speech Act that represents the communicative intention behind a speaker's utterance in a dialogue. Hence, identifying the dialogue acts of each speaker in a conversation is an essential initial step in automatically determining intention.

The Switchboard-1 corpus ('Computational Pragmatics | The Switchboard Dialog Act Corpus'), which consists of telephone speech conversations, was one of the first corpora related to dialogue and dialogue acts. It contains approximately 2,400 two-way telephone conversations involving 543 speakers. The Switchboard Dialogue Act Corpus (SwDA) (Godfrey, John J. and Holliman, Edward 1993) extended the Switchboard-1 corpus with tags from the SWBD-DAMSL tagset. The SWBD-DAMSL tagset was created by augmenting the Discourse Annotation and Markup System of Labelling (DAMSL) tagset. Through clustering, 220 tags were reduced to 42 tags to improve the language model on the Switchboard corpus. The resulting tags include dialogue acts such as statement-non-opinion, acknowledge, statement-opinion, and agree/accept, among others. The size of the reduced set has been a matter of debate, with some studies using a 42-label set, while others have used the most commonly used four-class classification.

- Constatives: committing the interlocutor to something's being the case.
- Directives: attempts by the interlocutor to get the addressee to do something.
- Commissives: committing the interlocutor to some future course of action.
- Acknowledgments: express the interlocutor's attitude regarding the addressee with respect to some social action.

Thus, we can think of DAs as a tagset that can be used to classify utterances based on a combination of pragmatic, semantic, and syntactic criteria.

8.2.3. Schema-Guided Paradigm

Rastogi et al. (Rastogi et al. 2020a) proposed a novel approach named schema-guided dialogue to facilitate dialogue state tracking by using natural language descriptions to define a dynamic set of service schemas. A schema-guided ontology is an ontology that characterizes the domain and is confined by a specific schema or set of schemas. In this context, a schema specifies the organization of the data and the associations between the concepts and entities that are expected to be present in the dialogue. A frame is a data structure that represents the current state of the dialogue at a specific time point. It is comprised of *key-value* pairs, where the keys represent the entities and concepts that are relevant to the conversation, and the values represent the corresponding values of

these concepts. Typically, a frame corresponds to a specific schema or set of schemas, and it can be updated as the dialogue progresses. For instance, in a restaurant booking system, a frame may contain information such as the date and time of the reservation, the number of diners, and the desired cuisine type. In the schema-guided paradigm, frames are predetermined and the system anticipates the user's input to conform to one of the predefined frames. Although this approach can improve the robustness and accuracy of the dialogue system by providing a structured and domain-specific representation of knowledge, it also limits the flexibility of the system to handle novel and unforeseen input. A frame is typically comprised of the following components:

- *Intents*: represent the goal of the interlocutor, it defines the task or action that the dialogue systems trying to accomplish.
- *Slots*: represent the information that is being requested or provided, they describe the properties of the entities or concepts that are relevant to the task.
- *Slot values*: represent the value of the slots, they are the actual information that has been extracted or provided during the conversation in order to deceive the target.
- *Constraints*: represent the additional information that is useful for the task, they are used to guide the dialogue towards a successful outcome.

8.2.4. Dialogue State Tracking

Dialogue state tracking (Williams and Young 2007) is the process of maintaining an accurate representation of the current state of a conversation. This involves identifying the user's intentions and goals, as well as the entities and concepts that are mentioned in the conversation, in order to provide relevant and precise responses. To tackle the challenge of representing the dialogue state, Young et al. (Young et al. 2010) proposed a method that leveraged dialogue acts. They employed a partially observable Markov decision process (POMDP) to build systems that can handle uncertainty, such as dialogue systems. To achieve a practical and feasible implementation of a POMDP-based dialogue system, the state can be factorized into discrete components that can be effectively represented by probability distributions over each factor (Williams, Raux, and Henderson 2016). These factorized distributions make it more feasible to represent the most common slot-filling applications of POMDP-based systems, where the complete dialogue state is reduced to the state of a small number of slots that require filling.

In 2020, Rastogi et al. (Rastogi et al. 2020b) proposed the schema-guided paradigm to tackle dialogue-state tracking, which involves predicting a dynamic set of intents and slots by using their natural language descriptions as input. They also introduced a

schema-guided dataset to evaluate the dialogue-state tracking tasks of dialogue systems, including domain prediction, intent prediction, and slot filling. Additionally, they developed a DST model capable of zero-shot generalization. In zero-shot generalization, a model is trained on a diverse set of tasks or domains, and it learns to understand the underlying patterns or relationships between them. By leveraging this learned knowledge, the model can make predictions or generate outputs for new tasks or data points it has never encountered before. The schema-guided design utilizes modern language models, such as BERT, to create a unified dialogue model for all APIs and services by inputting a service's schema, which enables the model to predict the dynamic set of intents and slots within the schema. This approach has gained significant attention, and annual competitions such as the Dialogue Systems Technology Challenge (DSTC) track the progress in this field (Rastogi et al. 2020b).

To achieve slot filling, a sequence model can be trained using different types of dialogue acts as classification labels for each domain. Additionally, pretrained language models, such as GPT, ELMo, BERT, and XLNet, have shown significant promise in recent years with regards to natural language processing. These models have outperformed prior algorithms in terms of generalization and zero-shot learning. Consequently, they offer an effective approach to performing zero-shot learning for language understanding. Furthermore, by leveraging pretrained language models in the schema-guided paradigm, dialogue systems can generalize to unseen service schema elements and improve their accuracy and robustness.

The primary objectives of dialogue state tracking encompass predicting the active user intention, requested slots, and values of slots in a given conversation turn. Within the context of DST, the user intention closely relates to the service supplied by the dialogue system. The intention refers to the user input's purpose or goal, while the service represents the functionality offered by the dialogue system to achieve the user's intent. The user's intention is typically deduced from the input and ongoing conversation state and can be expressed as a label or a set of labels that indicate the user's objective. A service denotes the task or action the dialogue system intends to perform and can be defined by a group of intentions and slots. The slots indicate the requested or supplied information. For example, an intention might involve "booking a flight," while the slots could consist of "destination," "departure date," "return date," and other related information. Services may vary over time, making it critical to have a flexible and adaptable ontology that can be modified as required.

DST's capacity to handle either a closed set of slots (static ontology) or an open set of slots (dynamic ontology) is a crucial attribute. In the former, the model is capable of predicting only those pre-defined slot values, and cannot assimilate new slot values from example data. The model generally comprises three modules: an input layer that

translates each input token into an embedding vector, an encoder layer that encodes the input to a hidden state, and an output layer that predicts the slot value based on the hidden state. In the former, where the set of possible slot values is predefined, the output layer may be approached in two ways: i) a feed-forward layer that generates all possible values associated with a specific slot; or ii) an output layer that contrasts both the input representation and slot values, and provides scores for each slot value. The scores can be normalized by applying non-linear activation functions, such as SoftMax or sigmoid, to convert them into probability distributions or individual probabilities.

In DST, the zero-shot setting enables a model to handle new intents or slots that it has not seen before, without requiring additional training data. This allows the model to generalize to new contexts or situations based on the knowledge gained during training. Unlike traditional supervised learning approaches, where a DST model is trained on a specific dataset with a specific set of intents and slots, a zero-shot DST model can handle new intents and slots without prior exposure. Techniques such as transfer learning, pre-training, or meta-learning can be used to achieve this goal and learn a more general and robust model. For instance, a zero-shot DST model can utilize pre-trained language models or embeddings, which have already learned significant knowledge about language from a large corpus of text data. The model can then be fine-tuned on a smaller dataset specific to the DST task at hand.

8.3. Model

SG-CSE BERT is part of an SG-CSE Attack State Tracker (SG-CSEAST) a system that estimates the dialogue state during a CSE attack state by predicting the intention and slot-value pairs at turn t of the dialogue. The SG-CSEAST consists of the following four units depicted in **Figure 8-1**.

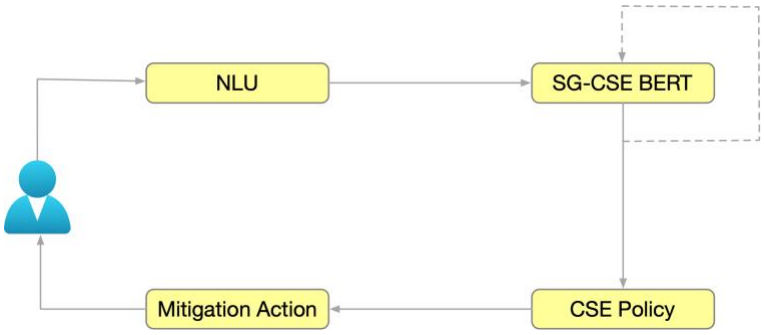

Figure 8-1. *The SG-CSE Attack State Tracker main units*

- NLU: converts the utterances into a meaningful representation.

- SG-CSE BERT: takes into account the dialogue history, and outputs the estimated state of the CSE attack.
- CSE Policy: decides which mitigation action to take.
- Mitigation Action: applies the selected mitigation action

This chapter, presents the SG-CSE BERT unit which recognizes information leakage and deception attempts by examining the full chat history and it is the deep learning model underneath DIACT-R. The information pertains to Small-Medium Enterprises (SMEs), Information Technology (IT), and personal information, whereas deception, within this thesis' context, is defined as the endeavor of a malicious user to convince a user to utilize or disclose an IT resource. The schema-guided paradigm is employed, incorporating dialogue acts (DAs) and dialogue history. The steps towards SG-CSE BERT implementation are the following (see **Figure 8-2**):

- Related dialogue acts (SG-CSE DAs) are extracted from the CSE Corpus (Tsinganos and Mavridis 2021) by mapping the intention of the utterance to a proposed dialogue act.
- A new corpus (SG-CSE Corpus) is created specifically for Dialogue State Tracking (DST), annotated with SG-CSE DAs. This corpus is utilized for fine-tuning and evaluating the recognition model.
- The CSE conceptual model and CSE ontology (Tsinganos and Mavridis 2021) are utilized to construct the SG-CSE Ontology. Thus, different attack types are extracted.
    - Four different CSE attack types (*CSE SME Info Extraction, CSE IT Info Extraction, CSE Personal Extraction,* and *CSE Department Extraction*) are extracted and they are going to be represented as services.
    - Each service is mapped to a specific schema.
    - The schema has several intents which are mapped to the corresponding DA.

The schema-guided paradigm makes use of a domain's ontology to create the required schema that describes each service existing in the domain. In this research, the domain is the CSE attacks and we predict the service, user intention, and requested slot-value pairs. For example, for an utterance like: "What is the CEO's first name?" the SG-CSE BERT model should be able to build a representation like the one presented in **Table 8-2**

Table 8-2. *Sample representation of frame slots*

Acquire	Value
Service	CSE_SME_Info_Extraction
Intent	WH-SME_Question
Employee_name	Null
Employee_position	CEO

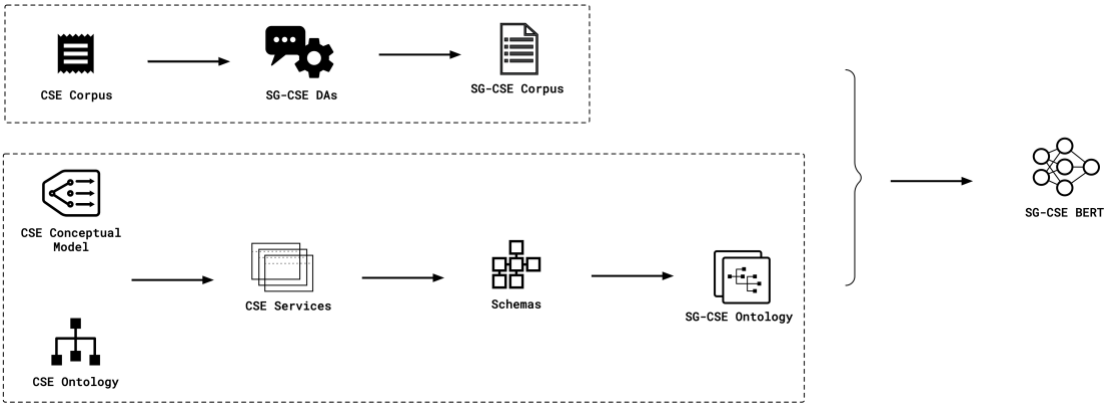

Figure 8-2. *Implementing SG-CSE BERT*

A CSE attack is unleashed through the exchange of utterances in turns of a dialogue. Time is counted in turns, and thus at any turn t, the CSE attack is at the state s_t and this state comprises the summary of all CSE attack history until time t. State s_t encodes the attacker's goal in the form of (slot, value) pairs. The different slot and value pairs are produced by the CSE ontology (Tsinganos and Mavridis 2021) and the proposed DAs that represent the in-context entities, intents, and their interconnection. The values for each slot are provided by the attacker during the CSE attack and represent her goal. E.g., during a CSE attack at turn t the state s_t could be

$$s_t = \{(employee\_name, NULL), (employee\_position, CEO)\}\}$$

In such a state the attacker's goal has been encoded for slots $employee\_name$, $employee\_position$ during the dialogue. **Figure 8-2** depicts the dialogue system concepts and evolution that are related to the research and the concepts transferred from the dialogue systems field to the CSE attack domain for CSE attack recognition via CSE attack state tracking. As **Figure 8-2** depicts, a dialogue can be realized between humans, or between humans and machines. The latter is called dialogue systems and can be task-oriented or conversational. The architecture of dialogue systems can be described using frames, and among their properties, dialogue state tracking is a property that represents the state of the dialogue at any given moment. The schema is created based on the SG-CSE Ontology and we can perform prediction tasks using state-of-the-art language models such as BERT.

The SG-CSE BERT unit, as proposed, adheres to the schema-guided approach for zero-shot CSE attack state tracking by employing a fine-tuned BERT model. By taking various CSE attack schemas as input, the model can make predictions on a dynamic set of intents and slots. These schemas contain the description of supported intents and slots in natural language, which are then used to obtain a semantic representation of these schema elements. The use of a large pre-trained model such as BERT enables the SG-CSE BERT unit to generalize to previously unseen intents and slot-value pairs, resulting in satisfactory performance outcomes on the SG-CSE Corpus. The SG-CSE BERT unit, is made up of two different components.

- A pre-trained BERT base cased model from Hugging Face Transformers (Wolf et al. 2020) to obtain embedded representations for schema elements, as well as to encode user utterances. The model is capable of returning both the encoding of the entire user utterance using the <CLS> token, as well as the embedded representation of each individual token in the utterance including intents, slots, and categorical slot values. This is achieved by utilizing the natural language descriptions provided in the schema files in the dataset. These embedded representations are pre-computed and are not fine-tuned during the optimization of model parameters in the state update module.
- The second component, is a decoder that serves to return logits for predicted elements by conditioning on the encoded utterance. In essence, this component utilizes the encoded user utterance to make predictions regarding the user's intent and the relevant slot values. Together, these various components comprise the SG-CSE BERT recognizer.

8.3.1. The SG-CSE Dialogue Acts

Given the inherent differences between CSE attack dialogues and standard dialogues, a carefully designed set of dialogue-act labels is necessary to meet the requirements of CSE attack recognition. These dialogue acts should be able to capture the intentions of the attacker while remaining easily understandable. To create a set of appropriate dialogue acts, the utterances of both interlocutors in the CSE Corpus were analyzed, classified, and combined into pairs. Based on Young's paradigm outlined in (Young et al. 2010), a set of fourteen dialogue acts is proposed. Each SG-CSE DA is represented by an intent slot and has a data type and a set of values that it can take. The full list of the dialogue acts is presented in the following **Table 8-3**. The percentage of each dialogue act frequency is also given in the last column for the total number of utterances in the SG-CSE Corpus.

Table 8-3. *Frequency of Dialogue Acts in SG-CSE Corpus*

Dialogue Act	Example	%
Greeting	Hello, my name is John	6
Statement	I've lost the connection	21
Uninterpreted	:-)	2
Agreement	Sure, I can	7
Question	Is this the Sales department?	11
Yes-No Question	Can you give me a copy?	9
WH-SME-Question	What this the HQ Address?	5
WH-IT-Question	What is your IP address?	8
WH-Personal Question	Which one is your personal email?	4
Rephrase	Do you have a network address?	1
Directive	Click this link, please	16
Yes Answer	No, not possible	5
Reject	I can't do this	3
Bye	Cheers!	2

The following **Table 8-4** contains examples of dialogue acts mapped to real word utterances from the SG-CSE Corpus.

Table 8-4. *Sample utterances and corresponding dialogue acts*

Utterance	Dialogue Act
Hello Angel, I'm experiencing a problem with my Charge 2	Greeting
I saw that the battery was leaking	Statement
Can you confirm my email associated with my [ANON] account?	Directive
Cheers!	Bye

8.3.2. The SG-CSE Ontology

The SG-CSE ontology comprises a collection of schemas that describe the CSE attack domain, based on the CSE ontology and the proposed set of fourteen SG-CSE DAs. The use of a schema-based ontology enables a structured and domain-specific representation of knowledge, enhancing the robustness and accuracy of the recognition task. This is achieved through the provision of predefined frames, encompassing predetermined entities and concepts, which the system can expect to encounter as input. **Table 8-5** presents the 18 slots existing in SG-CSE Corpus in tabular form with accompanying example values

Table 8-5. *Slots in SG-CSE Corpus*

Slot	Type	Example Values
Hardware	Dictionary	CD, DVD, USB stick
Network	Numbers	192.168.13.21
Server	String	subdomain.domain.tld
Service	String	Email, ftp, ssh
Software	String	RDP, Firefox
Personal_Email	String	user@domain.tld
Personal_Name	String	George, Sandra
Personal_Telephone	Numbers	0123456789
Dept_Email	String	dept@domain.tld
Dept_Name	String	Sales, Marketing
Employee_Name	String	George, Maria
Employee_Email	String	name@domain.tld
Employee_Telephone	Numbers	0123456789
Employee_Position	String	Executive, Manager
Office_Number	Numbers	12, 23
Enterprise_Address	String	26th Av. Somewhere
Enterprise_Location	String	Athens, Thessaloniki
Enterprise_Name	String	ACME, Other

The services present in training set are the following four:

- CSE_SME_Info_Extraction
- CSE_IT_Info_Extraction
- CSE_Personal_Extraction
- CSE_Department_Extracrtion

Where the slots related to each service are combinations of the aforementioned slots in **Table 8-5**.

Figure 8-3 depicts an excerpt of the example schema for the CSE_SME_Info_Extraction service. The dots between the boxes denote that more slots and intents exist.

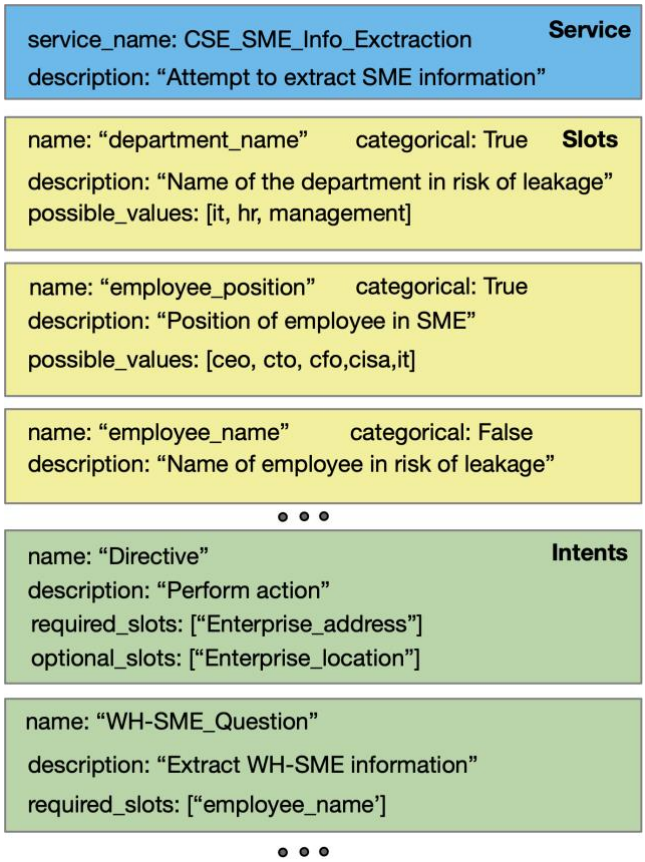

Figure 8-3. *Example schema for CSE_SME_Info_Extraction*

8.3.3. The SG-CSE BERT Model

This section introduces the SG-CSE BERT model, which operates in a schema-guided setting and can condition the CSE attack service schema using the descriptions of intents and slots. To enable the model to represent unseen intents and slots, a BERT-based model pre-trained on large corpora is employed. As a result, the proposed SG-CSE is a model for zero-shot schema-guided dialogue state tracking in CSE attack dialogues. The SG-CSE BERT model encodes all schema intents, slots, and categorical slot values into embedded representations. Since social engineering attacks can take many forms, schemas may differ in their number of intents and slots. Therefore, predictions are made over a dynamic set of intents and slots by conditioning them on the corresponding schema embedding.

The sequence pairs used for the embeddings of intent, slot, and slot value are presented in **Table 8-6**, and they are fed as input to the pre-trained BERT model (see **Figure 8-4**) where $a_1, \ldots, a_n$ and $v, \ldots, v_n$ are the two sequence tokens that are fed as a pair to SG-CSE BERT encoder. The U_{CLS} is the embedded representation of the schema and $t, \ldots, t_{n+m}$ are the token-level representations.

Table 8-6. *Sequence pair used for embeddings*

	Sequence 1	Sequence 2
Intent	CSE attack description	Intent description
Slot	CSE attack description	Slot description
Value	Slot description	value

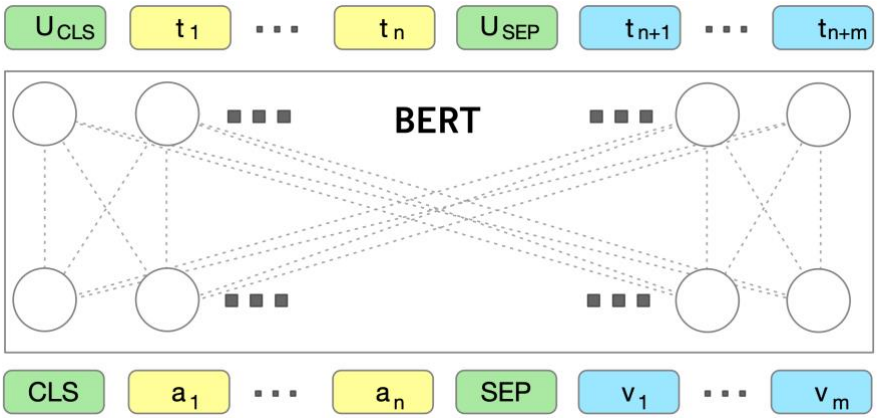

Figure 8-4. *Pre-trained BERT model*

Schema Embeddings: Let I, S be the intents and slots of CSE attack service and $\{i_j\}$ where $1 \leq j \leq I$ and $\{s_j\}$ where $1 \leq j \leq S$ their embeddings. The embeddings of the N non-categorical slots are denoted by $\{s_j^n\}$ where $1 \leq j \leq N \leq S$ and the embeddings for all possible values that the k^{th} categorical slot can take are denoted by $\{v_j^k\}$ where $1 \leq j \leq V^k$ and $1 \leq k \leq C$. C is the number of categorical slots and $N + C = S$.

Utterance Embedding: A pair of two consecutive utterances between the two interlocutors is encoded and represented to embeddings as u_{CLS} and the token level representations $\{t_i\}$ where $1 \leq i \leq M$ and M the total number of tokens in the pair of utterances.

The model utilizes the schema and utterance embeddings and a set of projections (Rastogi et al. 2020a) to proceed to predictions for active intent, requested slot, and user goal. Regarding the **active intent**, which pertains to the intent requested by the attacker and the one being recognized, it assumes the value 'NONE' when no intent is currently being processed by the model. Otherwise, if i_0 is the trainable parameter in $\mathbb{R}^d$ then the intent is given by

$$l_{int}^j = \mathcal{F}_{int}(u, i_j, 1), 0 \leq j \leq I$$

SoftMax function is used to normalize the logits l_{int}^{j} are normalized and produce a distribution over the set of intents plus the "NONE" intent. The intent with the highest probability is predicted as active

The **requested slots,** that means the slots whose values are requested by the user in the current utterance are given by:

$$l_{req\_slot}^{j} = \mathcal{F}_{req\_slot}(u, s_j, 1), 0 \leq j \leq S$$

The sigmoid function I is used to normalize the logits $l_{req\_slot}^{j}$ and get a score in the range of [0, 1]. During inference, all slots with a score > 0.6 are predicted as requested.

The **user goal** is defined as the user constraints specified over the dialogue context till the current user utterance and it is predicted in two stages. In the first stage, a distribution of size 3 denoting the slot status taking values *none*, *harmless* and *current* is obtained using

$$l_{status}^{j} = \mathcal{F}_{status}(u, s_j, 3), 1 \leq j \leq S$$

If the status is predicted as *none*, its value is assumed unchanged. If the predicted status is *harmless* then the slot gets the value *harmless*. Otherwise, the slot value is predicted using the following

$$l_{value}^{j,k} = \mathcal{F}_{status}(u, v_j^k, 1), 1 \leq j \leq V^k, 1 \leq k \leq C$$

The SoftMax algorithm is used to map the categorical values of a variable into a distribution over the entire range of possible values. For each non-categorical slot, logits are obtained using

$$l_{start}^{j,k} = \mathcal{F}_{start}(t_k, s_j^n, 1), 1 \leq j \leq N, 1 \leq k \leq M$$
$$l_{end}^{j,k} = \mathcal{F}_{end}(t_k, s_j^n, 1), 1 \leq j \leq N, 1 \leq k \leq M$$

Each of these two distributions above represents the starting and end points for the actual span of text that references the specific slot. The indices a $\leq$ v maximizing start[a] + end[v] will be the boundary between spans, and the value associated with that span is assigned to that slot.

Figure 8-5. *Two example pairs of utterances with predicted dialogue states*

In **Figure 8-5** above we see the predicted CSE attack dialogue state using two turns from two different utterance pairs. In green boxes, the active intent and slot assignments are shown and in the orange box, we can see the related schema of the CSE attack service. The CSE attack state representation is conditioned on the CSE attack schema which is provided as input along with the victim and attacker utterances.

8.4. Corpus

The SG-CSE Corpus is a task-oriented dialogue corpus specifically designed for CSE attacks and is derived from the CSE corpus. Its main purpose is to serve as a training set for intent prediction, slot-value prediction, and dialogue state tracking in the context of CSE attacks. The evaluation set of the SG-Corpus contains previously unseen services from the CSE domain, which allows us to evaluate the SG-CSE BERT model's ability to generalize in zero-shot settings. The corpus comprises various types of CSE attacks, including real-life cases and fictional scenarios. The hybrid approach used to create the SG-CSE Corpus combines characteristics of both balanced and opportunistic corpora, and it is based on the schema-guided paradigm. The SG-CSE Corpus contains 90 dialogues and is presented in ***Table 8-7***.

Table 8-7. *SG-CSE Corpus identity*

Characteristic	**Value**
Corpus name	SG-CSE Corpus
Collection Methods	Web scraping, pattern-matching text extraction
Corpus size	(N) 90 text dialogues/5798 sentences
Vocabulary size	(V) 4500 terms
Total no. of turns	5798
Avg. tokens per turn	8.42
No. of slots	18
No. of slot values	234
Content	chat-based dialogues
Collection date	June 2018–December 2020
Creation date	Aug 2022

The original CSE Corpus was preprocessed and all dialogues were converted into pairs of utterances and annotated to enhance the corpus, paraphrasing techniques were employed, replacing important verbs and nouns to generate additional dialogues. To mitigate the potential limitations of a small corpus, a list of 100 critical nouns and 100 critical verbs was compiled, and categorized under three sensitive data categories. For each noun and verb, ten new sentences were generated by substituting the noun with a similar word. To address this, the Word Embeddings technique was employed, utilizing dense vector representations to capture the semantic meaning of words. The embeddings are learned by moving points in the vector space based on the surrounding words of the target word. To identify synonyms, a pre-trained word2vec model was utilized, leveraging the distributional hypothesis that suggests linguistic items sharing similar contextual distributions exhibit similar meanings. Segregation of domain words was achieved by grouping words that exhibited similar contextual distributions based on their corresponding vector values. This resulted in a vector space where each unique word in the dialogues was assigned a corresponding vector, representing a vector representation of the words in the collected dialogues. The critical verbs and nouns were replaced and combined to create new phrases. The following ***Figure 8-6*** depicts the distribution of the number of turns in the dialogues involved.

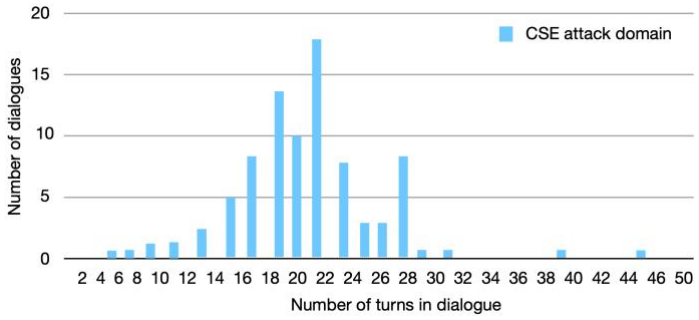

Figure 8-6. *Distribution of turns in the SG-CSE Corpus*

The following ***Figure 8-7*** presents the distribution of types of dialogue acts existing in the SG-CSE Corpus

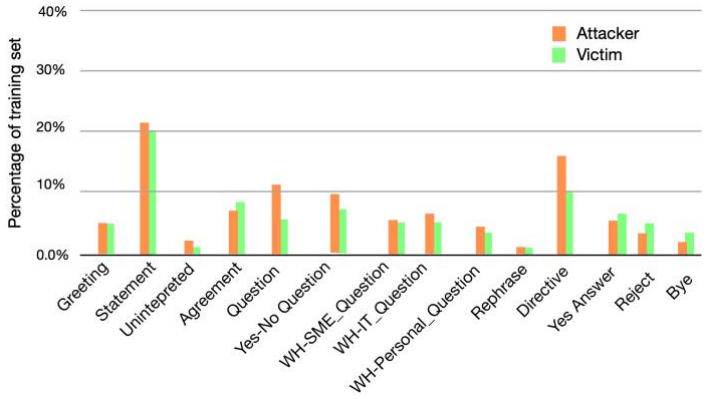

Figure 8-7. *Distribution of types of dialogue acts in the SG-CSE Corpus*

The annotation task has significantly enhanced the value and quality of the SG-CSE Corpus, allowing it to be utilized for task-oriented dialogue tasks, such as DST. Additionally, natural language descriptions of the various schema elements (i.e., services, slots, and intents) are included in the corpus. To test the zero-shot generalization ability, the evaluation set includes at least five services that are not present in the training set. The SG-CSE Corpus is composed of CSE attack dialogues between two interlocutors, where each dialogue pertains to a specific CSE attack service in the form of a sequence of turns. Each turn is annotated with the active intent, dialogue state, and slot spans for the different slot values mentioned in the turn. The schema includes information such as the service name, the supported tasks (intents), and the attributes of the entities used by the service (slots).

8.5. Results

The following metrics are considered for evaluating the CSE attack state tracking:

- Active Intent Accuracy: The fraction of user turns for which the active intent has been correctly predicted.
- Requested Slot F1: The macro-averaged F1 score for requested slots overall eligible turns. Turns with no requested slots in ground truth and predictions are skipped.
- Average Goal Accuracy: The average accuracy is calculated based on the correct prediction of slot values in each turn of the dialogue state.
- Joint Goal Accuracy: This is the average accuracy of predicting all slot assignments for a given service in a turn correctly. Also, Harmonic mean between seen and unseen classes.

The SG-CSE BERT model was implemented using the Hugging Face library and the BERT uncased model, which consists of 12 layers, 768 hidden dimensions, and 12 self-attention heads. The model was trained using a batch size of 32 and a dropout rate of 0.2 for all classification heads. A linear warmup strategy was applied with a duration of 10% of the training steps. The AdamW optimizer (Kingma and Ba 2014) with a learning rate of 2e-5 was used during training.

In **Figure 8-8** the performance of the SG-CSE BERT model is depicted. SG-CSE BERT shows efficiency in Active Intent Accuracy and Requested Slots F1 and less efficiency in Average Goal Accuracy and Joint Goal Accuracy.

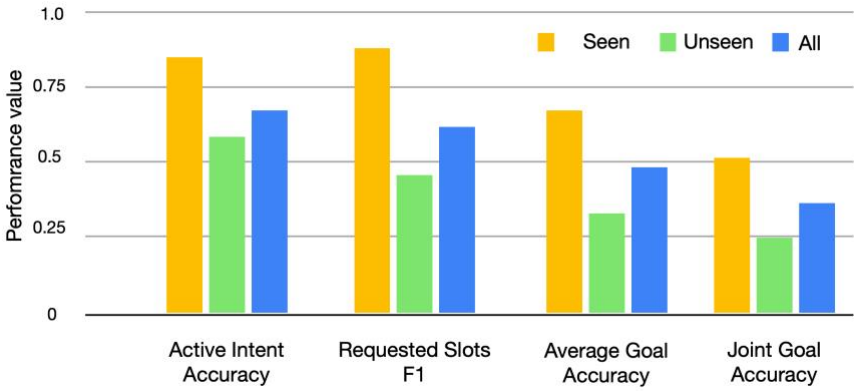

Figure 8-8. *The SG-CSE BERT model's performance*

The following **Table 8-8** presents the same performance results in tabular form.

Table 8-8. *SG-CSE Attack state tracker performance*

System	Model	Parameters	Active Intent Accuracy	Req Slot F1	Acg Goal Accuracy	Joint Goal Accuracy
Seen	$BERT_{BASE}$	110M	85.2	89.6	74.1	56.7
Unseen	$BERT_{BASE}$	110M	53.8	48.3	31.4	24.9
All	$BERT_{BASE}$	110M	69.5	68.9	52.7	40.8

8.6. Discussion

Early dialogue state tracking datasets were developed to be specific to a particular domain due to the difficulty in building models that can effectively track dialogue states for multiple domains. However, with the recent release of multidomain datasets and the incorporation of machine learning-based methods, it has become possible to build models that can track states for multiple domains using a single set of training data. To evaluate the models' generalization capability in zero-shot settings, the SG-CSE Corpus was generated. This corpus includes evaluation sets that consist of previously unseen services. By testing the models on these unseen services, their ability to generalize beyond the training data can be assessed, providing insights into their robustness and adaptability. This allowed the evaluation of the models' performance on tasks they had not been explicitly trained on. To address the issue of limited dialogue resources, data augmentation can be explored as an option. Augmenting the training dataset by adding more diverse examples can improve performance. Source-based augmentation generates sentences by changing a single variable value in a sample utterance, while target-based augmentation takes portions of sentences from different places in the training data and recombines them.

DIACT-R is built around BERT, a pre-trained transformer-based model that has been trained on a large corpus of text data. BERT has demonstrated strong generalization ability across a wide range of natural language processing tasks and domains. When fine-tuned on a specific task and domain, BERT is able to learn specific patterns and features of the task and domain, which allows it to achieve good performance. However, if BERT is not fine-tuned, it may be able to recognize new unseen intents but it would not have enough information to generate the corresponding slot values. Moreover, it may not be able to recognize new unseen services or new unseen domains. Large-scale neural language models trained on massive corpora of text data have achieved state-of-the-art results on a variety of traditional NLP tasks. However, although the standard pre-trained BERT is capable of generalizing, task-specific fine-tuning is essential for achieving good performance. This is confirmed by recent research which has shown

that pre-training alone may not be sufficient to achieve high accuracy in NLP tasks. For example, the performance of a pre-trained model on a downstream task may be significantly improved by fine-tuning it on a smaller in-domain dataset.

The SG-CSE Corpus is designed to test and evaluate the ability of dialogue systems to generalize in zero-shot settings. The evaluation set of the corpus contains unseen services, which is important to test the generalization ability of the model. This evaluation set does not expose the set of all possible values for certain slots. It is impractical to have such a list for slots like IP addresses or time because they have infinitely many possible values, or for slots like names or telephone numbers, for which new values are periodically added. Such slots are specifically identified as non-categorical slots. In the evaluation sets, a significant number of values that were not present in the training set were meticulously selected. This was done to specifically evaluate how well the models performed on unseen values, providing a robust assessment of their generalization capabilities. Some slots like hardware, software, etc. are classified as categorical, and a list of all possible values for them is provided in the schema.

DIACT-R is designed to handle dialogue state tracking for a larger number of related services within the same domain. Its schema-guided paradigm provides a structured and domain-specific representation of knowledge, which increases the model's robustness and accuracy. This, in turn, allows the system to better understand and interpret user input, and track the dialogue state more accurately. The schema and frames work together to create this representation, with the schema constraining the possible states of the dialogue, and the frames tracking the current state. The proposed system has shown satisfactory performance, and the experimental results demonstrate its effectiveness. Its simplistic approach also leads to more computational efficiency, which makes it a good candidate for use as a separate component in a holistic CSE attack recognition system, as proposed in previous works (Tsinganos et al. 2018), (Tsinganos, Mavridis, and Gritzalis 2022), and (Tsinganos, Fouliras, and Mavridis 2022). Additionally, SG-CSE BERT is computationally efficient and scalable to handle large schemata and dialogues. It should be noted that the performance of SG-CSE BERT is dependent on the quality and completeness of the training data. Moreover, incorporating additional sources of information such as user context and sentiment can enhance the system's performance in real-world scenarios. Overall, SG-CSE BERT provides a promising solution for zero-shot schema-guided dialogue state tracking in the domain of CSE attack recognition.

A static ontology, which is a predefined and unchanging set of intents and slots, can be used for zero-shot recognition in DST. However, there are some limitations to its effectiveness. While a comprehensive ontology can cover a wide range of possible

situations and contexts, it may not be able to encompass all possible scenarios. As a result, a model trained on such an ontology may struggle to generalize to new and unseen intents and slots. Moreover, since the ontology remains static and is not updated during the training process, it may not be able to adapt to changes in the domain over time or new types of attacks. In contrast, a dynamic ontology can be updated during the training process to adapt to new situations and contexts. This can be achieved by using unsupervised methods to extract the ontology from the data or by incorporating active learning methods that allow the system to query human experts when encountering unseen intents and slots. By using a dynamic ontology, the system can learn to recognize new intents and slots over time, improving its ability to generalize to new and unseen scenarios. Zero-shot recognition in DST remains an active area of research, with new methods and approaches being developed to improve the performance of DST models in recognizing unseen intents and slots. By incorporating dynamic ontologies and other techniques, future DST models may be better equipped to recognize and respond to previously unseen user input.

8.7. Chapter Conclusion

This chapter aimed to investigate schema-guided dialogue state tracking in the context of CSE attacks. A set of fourteen dialogue acts was formulated and the SG-CSE Corpus was constructed using the CSE Corpus. By adopting the schema-guided paradigm, DIACT-R's model was developed as a straightforward method for zero-shot CSE attack state tracking. The obtained performance results exhibited promise and confirmed the efficacy of the proposed approach. To establish the necessary background for the study, an analysis of dialogue systems and their attributes was conducted, which facilitated the formulation of this approach. Furthermore, an exploration was conducted on how task-based dialogue systems and dialogue state tracking concepts and terminology can be adapted to the CSE attack domain.

The primary focus of this work revolved around the creation of the SG-CSE DAs (dialogue acts) and the SG-CSE Corpus. These efforts included mapping slots and intents, as well as proposing DIACT-R. The model achieved satisfactory performance results using a small model and input encoding. Although various model enhancements were attempted, no significant improvement was observed.

The study suggests that data augmentation and the addition of hand-crafted features could improve the performance of the CSE attack state tracking, but further experimentation is necessary to explore these methods. The DIACT-R model offers an advantage in the few-shot experiments, where only limited labeled data is available. Such an approach can help to create a more comprehensive and accurate understanding

of CSE attacks and their underlying mechanisms, as well as to develop more effective recognition and prevention strategies. For example, experts in natural language processing can help to improve the performance of dialogue state tracking models, while experts in psychology and sociology can provide insights into the social engineering tactics used in CSE attacks and the psychological factors that make users susceptible to these attacks. Cyber-security experts can provide a deeper understanding of the technical aspects of CSE attacks and help to develop more robust and effective defense mechanisms.

9. PERSU-R

9.1. Introduction

A diverse group of vulnerabilities are targeted in a CSE attack, ranging from technical misconfigurations to human psychological characteristics. However, it is more efficient to isolate the enablers of a successful CSE attack and to investigate the methods of recognition and defense separately. As mentioned in (Tsinganos et al. 2018) persuasion is only one of the enablers, and it is critical to be able to recognize it in the early stage and obtain information about it. Collectively, our knowledge about persuasion coupled with our knowledge about other enablers will guide our decisions regarding the response to the CSE attack. Persuasion is a crucial factor, but alone is not an adequate criterion to conclude whether a CSE attack is occurring. However, if persuasion methods are recognized in a cyber-security context using an appropriate dataset related to digital work environments, then the expected outcome will be rather useful.

Persuasion is a well-known method used by social engineers and the latest research regarding persuasion as an influence tactic (Cacioppo, Cacioppo, and Petty 2018; Cawood 2021), confirms and emphasizes persuasion's role as a CSE attack enabler. ISACA ('Information Technology - Information Security - Information Assurance | ISACA' 2018) defines enablers as the "factors that, individually and collectively, influence whether something will work". In (Tsinganos et al. 2018), persuasion was identified as one of the most critical factors that can lead to a successful CSE attack. Thus, in any given chat, it is of utmost importance to detect and recognize persuasion attempts at an early stage to deter a consequent attack and prevent sensitive data from being compromised.

Cialdini's latest work (Cialdini 2021) added a seventh persuasion principle called Unity to his famous taxonomy. Overall, the seven persuasion principles are reciprocity, commitment, social proof, liking, authority, scarcity, and unity. These principles distract people from thinking deliberately and analytically because of the amount of disinformation they inject into a normal communication flow (Ross, Rand, and Pennycook 2021). Therefore, from a cybersecurity perspective, it is critical to identify if a sentence in a natural communication setting contains a degree of disinformation. The term "persuasive payload" (pp) is defined as any information deliberately designed to deceive a human by modifying their opinion or leading them to make erroneous actions. Additionally, the concept of a "pp-container" is established to refer to every sentence that conveys a persuasive payload. In other words, any sentence that contains information aligned with one of the seven persuasion principles defined by Cialdini will be classified as a pp-container.

The main objective of this chapter is to guide cybersecurity defense mechanisms in recognizing early-stage CSE attacks by determining if sentences in the chat-based conversation contain persuasive payload. The timely recognition of pp-containers during a chat will raise awareness of the social engineering cues which permeate chat-based conversations. To achieve the intended goal, deep learning and natural language processing (NLP) techniques, specifically convolutional neural networks (CNN) and word embeddings, were utilized.

This chapter describes the design and implementation of a persuasion classifier that utilizes deep learning and natural language processing techniques. For this purpose, a convolutional neural network was trained on CSE-PUC Corpus , specifically annotated for recognizing Cialdini's persuasion principles. The proposed persuasion classifier network, named CSE-PUC, is the main algorithm of PERSU-R, and can determine whether a sentence carries a persuasive payload by producing a probability distribution over the sentence classes as a persuasion container. The focus is on classifying sentences as pp-containers or not, treating all persuasion principles as equally important. Therefore, persuasion recognition can be understood as a sentence classification task that attempts to recognize the existence of any type of persuasion payload, that is, whether the sentence is a pp-container or not.

9.2. Background

9.2.1. Persuasion principles

Persuasion as a process has been an interdisciplinary field of study for many years owing to the universality of its applications. There are exceptional works that have set the theoretical background, such as Cialdini’s (Cialdini 2021; Cialdini and Goldstein 2002; Resnik and Cialdini 1986). Cialdini discussed the principles of influence and concluded that there are two types of influence: compliance and persuasion. The difference between them lies in the fact that in compliance, a direct request is used to force a change in someone's behavior while in ‘persuasion,’ we are sending a message to force someone to change his/her behavior because of the message reception. Additionally, in (Siddiqi, Pak, and Siddiqi 2022) the authors further elaborate on influence methodologies, where they present the following categories: social influence, persuasion, attitude and behavior, trust and deception, language, and reasoning, countering social engineering-based cyberattacks, and machine learning-based countermeasures. Moreover, they classified the persuasion method into distinct types of persuasion: similarity, distraction, curiosity, and persuasion using authority. A similar classification is performed for the proposed attitude and behavior category, where

commitment influences the attitudes and behaviors of someone. The aforementioned classification is adapted in this work, and thus a way of identifying the 'Persuasion using authority' and 'Commitment' methods is proposed.

Overall, the seven Cialdini's persuasion principles are as follows:

- Reciprocation: A strong urge based on the rule of social interaction to reciprocate by giving something back to someone who gave us something first.
- Commitment and Consistency: We feel obliged to carry out the promise we have made so as not to feel untrustworthy.
- Social Proof: humans tend to believe what others do or think is right.
- Liking: we tend to like people who like us.
- Authority: Under certain circumstances, people are likely to be highly responsive to assertions of authority.
- Scarcity: We are highly responsive to indications that something we may want is in short supply.
- Unity: a perception that we share a common identity, that we are all part of "us."

After conducting a quantitative analysis to examine the existing persuasion principles in the CSE-PUC Corpus (Tsinganos and Mavridis 2021), authority and commitment arise as the most common persuasion principles used by social engineers.

9.2.2. Convolutional Neural Networks

Convolutional neural networks (Lecun et al. 1998) are feed-forward networks with an architecture inspired by the human visual cortex, and they were initially used for image recognition. They were named after a mathematical operation called convolution, which was being applied. These specialized network architectures can take arbitrarily sized data inputs and identify local patterns in a dataset that are sensitive to the word order; however, they do not consider where these patterns are located in the data. When CNNs are applied to images, the neural network architecture uses a 2-D convolution. In **Figure 9-1**, a CNN is illustrated, where we can identify its two main functional parts: the feature extractor and the classifier. The feature extractor part contains convolution and pooling layers, where the most relevant features for the task are automatically selected, saving us from the manual labor of extracting features, as in traditional machine learning algorithms. The classifier part contains fully connected and prediction layers, where the actual classification is performed using an activation function.

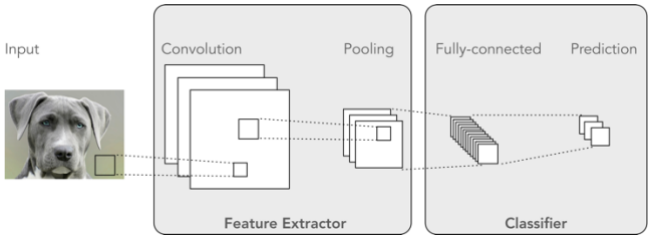

Figure 9-1. *A CNN comprised of a feature extractor and a classifier part.*

Four main operations are taking place in the layers of the aforementioned functional parts, and they are the following:

- *Convolution*: A linear operation of element-wise matrix multiplication and addition between a predefined part of the input data and a predefined matrix, called a *filter*, that captures the local dependencies underlying the original input data. This operation is executed on the convolution layer.
- *Non-Linearity*: A nonlinear operation performed by a function that enables the network architecture to represent real-world phenomena by capturing complex patterns that linear functions cannot capture. Almost all real-world phenomena are nonlinear. This operation is also executed on the convolution layer, and the final output is a matrix called a feature *map* that encodes the most notable features.
- *Pooling*: A subsampling operation that reduces the dimensionality of each resulting feature map while retaining the most valuable information. Usually, one can use max-pooling or average-pooling, that is, in 1-max-pooling, the most significant data element is selected from the feature map within a predefined window.
- *Classification*: A classification operation is performed using a fully connected layer that uses an activation function in the prediction layer. The outputs of the convolutional and pooling layers represent high-level features of the input data. In the last output layer, called prediction, these high-level features are used to classify the input data into various classes based on the training dataset. If SoftMax is used as the activation function, the output is a probability distribution over the classes.

In **Figure 9-1**, we see only one set of convolution and pooling layers; however, a CNN architecture can contain multiple sets in a row, where the second convolution layer performs convolution on the output of the previous pooling layer. Utilizing multiple convolution layers means that we use multiple filters on different input data, which results in the production of a richer feature map. In general, as we add more convolution steps, the network will be able to learn and recognize more intricate features.

While CNNs for image recognition typically use 2-D convolutions, in the context of NLP, the operation of convolution is 1-D, which means that a 1-dimensional array represents the text. Utilizing the ability of CNNs to identify local patterns in data, one can locate indicative patterns (phrases or n-grams) in larger text blocks such as sentences or documents. In the NLP context, a sentence is represented as a matrix, and each row of the matrix is associated with a language token, that is, a word (Y. Kim 2014). Using a similar representation, a CNN can learn to identify the local patterns during the training phase. Several CNN architectures (Torfi et al. 2020) have been used successfully for a variety of natural language processing tasks (text classification (Conneau et al. 2016), sentence classification (Zhang and Wallace 2015), and sentiment analysis (Dos Santos and Gatti 2014).

9.3. Model

PERSU-R is a task-specific neural network architecture composed of a CNN and a Multilayer Perceptron (MLP). The task of the classifier is to decide whether a sentence carries a persuasive payload by producing a probability distribution over the sentence classes, thus deciding whether the sentence is a pp-container. The CNN was used as a feature extractor in the first layer of PERSU-R, and it was integrated at a later stage with the rest of the architecture. PERSU-R was trained end-to-end, producing the result of the prediction task.

As mentioned in section 9.1, predicting that a written sentence carries a persuasive payload can be cast into a sentence classification task that identifies specific patterns in the sentence. These patterns are composed of specific sequences of ordered sets of tokens (i.e., words in the sentence). Thus, identifying a persuasive payload in a sentence means identifying informative local features that may be repeated, regardless of where they are placed in the sentence. Let us consider the following sentence that is part of the CSE Corpus (Tsinganos and Mavridis 2021): “I need that information to report back to my boss.”. We can easily conclude that some of the words are highly informative of a persuasive payload existence (i.e., the word *boss* denotes a possible use of the persuasion principle of *authority*), which holds true regardless of the position of this word in the sentence.

PERSU-R was developed to identify informative cues within sentences. This architecture is designed to process input sentences and extract the relevant information. CNNs with convolutional and pooling layers are used to identify such local cues (Johnson and Zhang 2016), where they have been used to identify indicative phrases for specific tasks. Furthermore, it is not important where this pattern may appear in a sentence. The main point of interest is only the existence of a specific sequence of

tokens of varying lengths that indicate a particular cue. A convolutional neural network identifies indicative local patterns (linguistic structures) and combines them to produce a fixed-size vector representation of these structures. This fixed-size vector represents the local patterns that are most informative for the prediction task at hand, which in this case is persuasion payload recognition.

When using a CNN architecture for a natural language task such as sentence classification, we initially apply a nonlinear function at the convolution layer of the CNN, which is learned over each k-sized window of tokens sliding in the sentence. This nonlinear function is called *a filter* and transforms the k-sized window of tokens into a scalar value. We can even apply multiple filters, that is, ℓ number of filters, and instead of a scalar, we can obtain a vector equal in dimensions to the number of filters applied. Next, in the pooling layer of the CNN, the pooling operation combines the resulting vectors from the different k-sized windows into a single ℓ-dimensional vector by taking the maximum (max pooling) value observed in each of the ℓ dimensions over the different k-sized windows. Each filter identifies a different feature from the input data window and the pooling operation selects the most important ones. The output of the ℓ-dimensional vector is then fed into the following parts of the overall network architecture. The parameters, which are the values of the filters applied, are tuned by back-propagating the gradients from the network loss during the training phase. In **Figure 9-2** we can see an example of a convolution with a sliding window of size $k = 3$, and 6-dimensional output $\ell = 6$ applied to the following sentence "*I need that info to report back to my boss*."

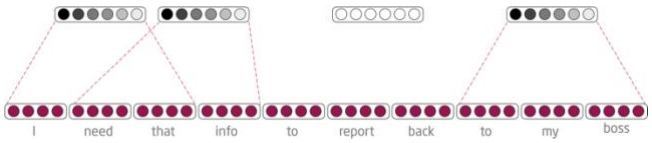

Figure 9-2. *Convolution was applied over a sentence with a window of size k=3 and an output of dimension l =6.*

9.3.1. Operation

PERSU-R takes as input the sentences exchanged between the interlocutors of the chat-based conversation. The words are represented using word vector embeddings, which are either trained by us (own trained) or by others (pre-trained). When there is a lack of a sizeable supervised training set, a common method to improve the performance of the deep learning algorithm is the word embeddings initialization using pre-trained vectors obtained from an unsupervised neural language model. During the tests, publicly available fastText word vectors were utilized (Mikolov et al. 2017) trained on 630 billion tokens on the Common Crawl. The vectors have a dimensionality of 300 and

were trained using a continuous bag-of-words architecture (Mikolov, Sutskever, et al. 2013). All words in the CSE-PUC Corpus vocabulary that were not present in the set of pre-trained words were initialized randomly with the same dimension and variance.

The operation of PERSU-R can be explained easily using a toy example. Let us assume that we have a corpus composed of just two sentences (taken from the CSE-PUC Corpus): "I need your username" and "Please read me your password." This corpus has a vocabulary (set of different words) of size $v = 8$, the length of the largest sentence was $n = 5$, and there are two sentences in the corpus; thus, $b = 2$. All sentences should have the same length to be fed as inputs to the CNN. Therefore, the first sentence is padded until it reaches a length of five. Each word in a sentence is represented by word embedding (e.g., one-hot and fastText). For ease of visualization in this example, a one-hot representation is assumed. The word embeddings are shown in **Figure 9-3**.

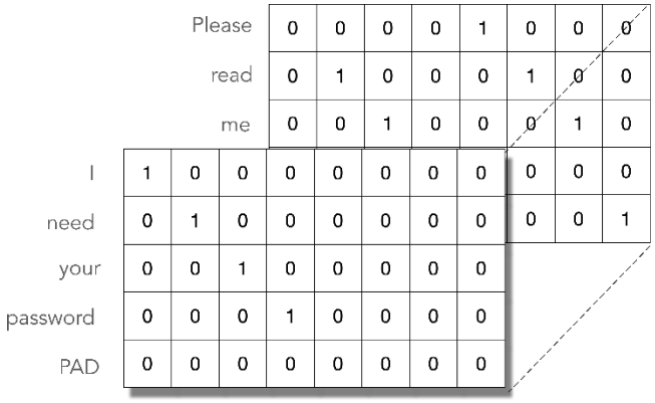

Figure 9-3. *One-hot word vector representation of a corpus composed of only two sentences; thus, b=2. The vocabulary was of size v=-8, and the largest sentence was of size n=5.*

If we assume that we are processing only one sentence (the first), then it is evident that we are dealing with an $n * v$ matrix, which in our example is 5×8. In contrast, if we process a batch, or, as in our case, the whole corpus of size b=2, then we can represent the sentences in the corpus with a $2 \times 5 \times 8$ tensor.

The convolution filter will have a size of $m * v$, where in our example m=2, to process two words simultaneously. Convolving the $n * v$ input with the $m * k$ filter provides a feature map of $1 * n$ dimensions. If multiple filters q are used, we obtain a $q * n$ matrix, as shown in **Figure 9-4**.

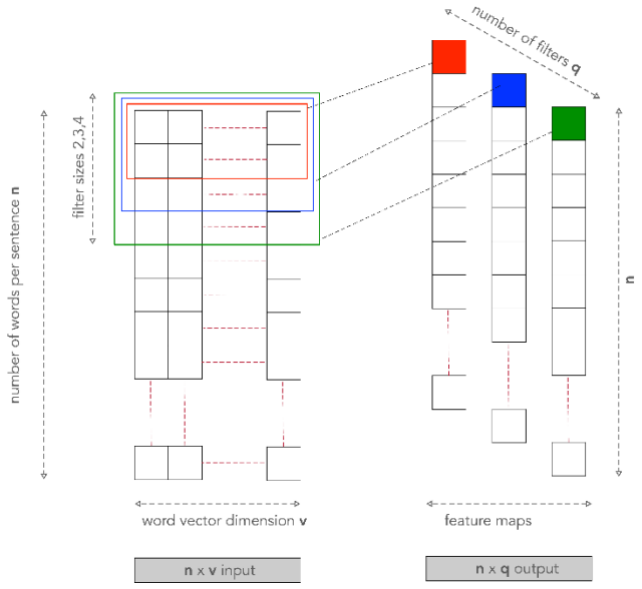

Figure 9-4. *1-D Convolution*

The convolution operation plays a vital role in preserving the spatial information of the sentences. With q different layers with different filter sizes, the network learns to extract ratings with different size phrases, leading to improved performance. Subsequently, the max-pooling operation subsamples the outputs produced by the previously discussed parallel convolution layers. Therefore, from the $q * n$ feature map, a $q * 1$ vector was produced by concatenating the maximum elements of each convolution layer (**Figure 9-5**).

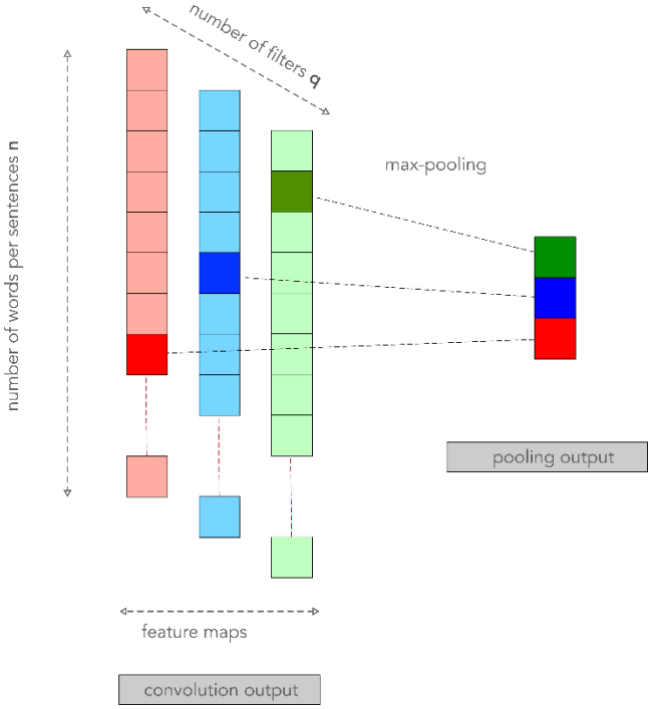

Figure 9-5. *The max-pooling operation*

The end-to-end PERSU-R architecture is shown in **Figure 9-6**.

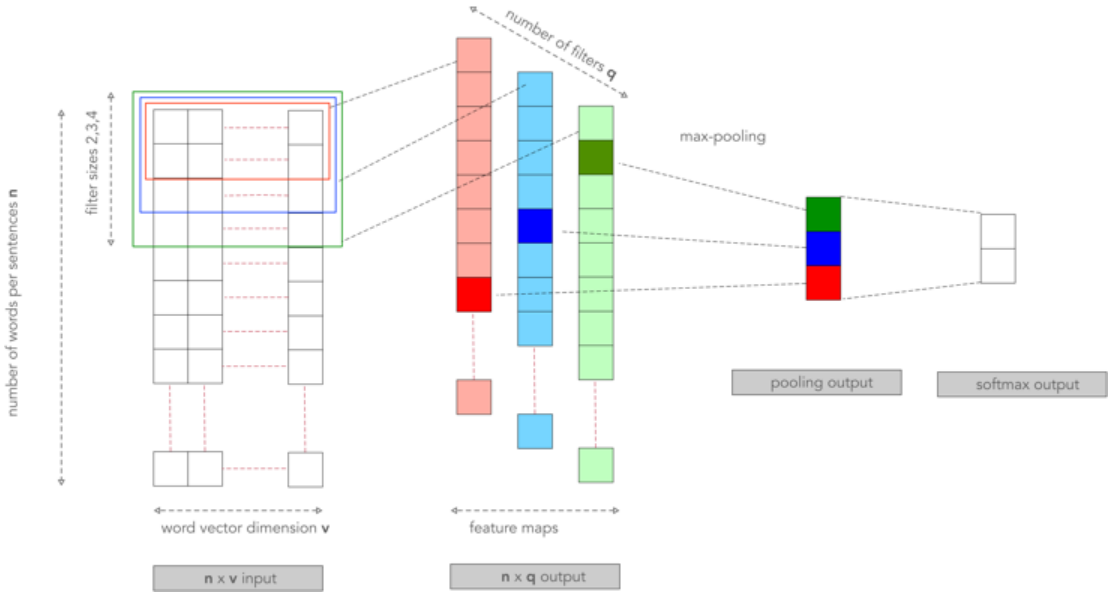

Figure 9-6. *PERSU-R architecture.*

9.3.2. Architecture

PERSU-R is trained end-to-end. As previously mentioned, a CNN is employed as a feature extractor to generate a sequence of vectors. These vectors are then passed to subsequent components of the network to make predictions. Every interesting pattern in each sentence that is informative for the pp-container classification task is captured by PERSU-R.

A single-layer convolutional neural network (CNN) is utilized in this approach. This CNN is trained on word vector embeddings obtained from fastText, which includes both the own trained embeddings as well as pre-trained embeddings. The CSE Corpus is appropriately annotated with labels corresponding to the two classes of sentences: pp-container and neutral. In the case of pre-trained word embeddings, the CSE Corpus was also used for transfer learning on top of the pre-trained word vectors. The word vocabulary of size v contains words projected from a $1-of-v$ encoding to a reduced dimensionality vector space via a hidden layer. As already mentioned, these reduced dimensionality word vectors encode the semantic features of vocabulary words.

Figure 9-7 illustrates the functional end-to-end architecture of PERSU-R, where only two filters per window are shown for ease of visualization. The labels above the layers represent the operations performed in each layer.

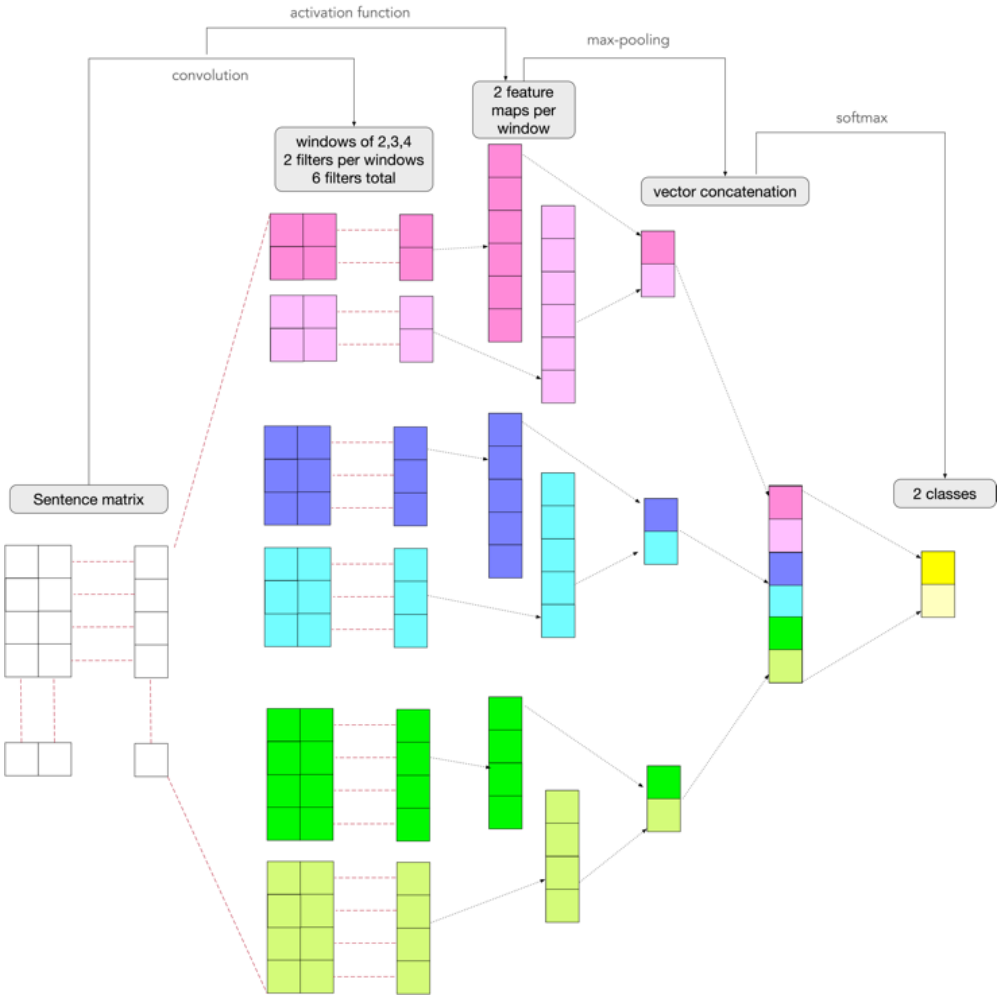

Figure 9-7. *Functional end-to-end architecture of PERSU-R*

The operations of the convolutional and pooling layers are explained in the following subsections.

9.3.2.1. Convolution

Suppose that we have a sequence of n words $w_{1:n} = w_1, w_2, \dots, w_n$, which constitute a sentence, and each word w_i is projected in a d-dimensional space, and thus is associated with a d-dimensional word vector (word embedding) $E_{[w_i]} = w_i$. To apply a 1-D convolution of width k, we slide a k-width window over the sentence and apply the convolution filter (also called a kernel) to each window over the sentence. The convolution filter is a dot product with a weight vector u followed by a nonlinear linear activation function, which in this case is a rectified linear unit (ReLU) (Agarap 2018).

Therefore, the i-th window of the k words $w_i, w_{i+1}, \dots, w_{i+k}$ results in the concatenated vector $x_i = [w_i, w_{i+1}, \dots, w_{i+k}] \in \mathbb{R}^{k*d}$. On each concatenated vector, the convolution

filter is applied, and the vector is transformed to a scalar value $r_i = f(x_i \circ u)$, where f is the nonlinear-linear activation function, $r_i \in \mathbb{R}, x_i \in \mathbb{R}^{k*d}$ and $u \in \mathbb{R}^{k*d}$. When multiple ℓ filters are used, arranged in matrix **U**, then $\boldsymbol{r}_i = f(x_i \circ \boldsymbol{U} + \boldsymbol{b})$, where $r_i \in \mathbb{R}^{\ell}, x_i \in \mathbb{R}^{k*d}$, $\boldsymbol{U} \in \mathbb{R}^{k*d*\ell}$, and $b \in \mathbb{R}$ is the bias. The vector summarizes the i-th window and captures various kinds of informative information in each dimension.

9.3.2.2. Pooling

When the convolution operation is completed, the output is q vectors $\boldsymbol{p}_{i:q}$where $\boldsymbol{p}_i \in \mathbb{R}^{\ell}$. Next, these vectors are pooled (combined) to form a vector $\boldsymbol{c} \in \mathbb{R}^{\ell}$, representing the entire sequence and capturing and encoding all informative cues for the persuasive payload recognition task. We use 1-*max-pooling,* which takes the maximum value across each dimension, which equally means that it selects the most important feature. Vector $\boldsymbol{c}$ is fed to a fully connected layer that uses the SoftMax function to output a probability distribution over the pp-container classes.

The loss for this network architecture is calculated for the persuasive payload recognition task by back-propagating the error all the way back through the pooling and convolution layers, as well as the word embedding layer. During the training, the convolution matrix U, the bias vector b, the fully connected layer, and potentially the embedding matrix E such that the vector c resulting from the convolution and pooling process indeed encodes information relevant to the task at hand.

Returning to the example sentence "I need that info to report back to boss" and adapting to the inspiring work of Goldberg (Goldberg 2016), **Figure 9-8** illustrates the convolution, pooling, and max-pooling operations. In the illustration, there is a window of size three, and each word has been transformed into a 2-dim embedding vector (not shown). The word-embedding vectors are concatenated, resulting in a 6-dimensional window representation. Each of the eight windows was transferred through a 6 × 3 filter (linear transformation MUL followed by ReLU), resulting in eight 3-dimensional filtered representations. MUL is a multiplication operation between the input feature map and a weight matrix and ReLU (Rectified Linear Unit), is a non-linear activation function. Finally, the max-pooling operation is applied, which takes the maximum over each dimension (feature), resulting in the final 3-dimensional pooled vector.

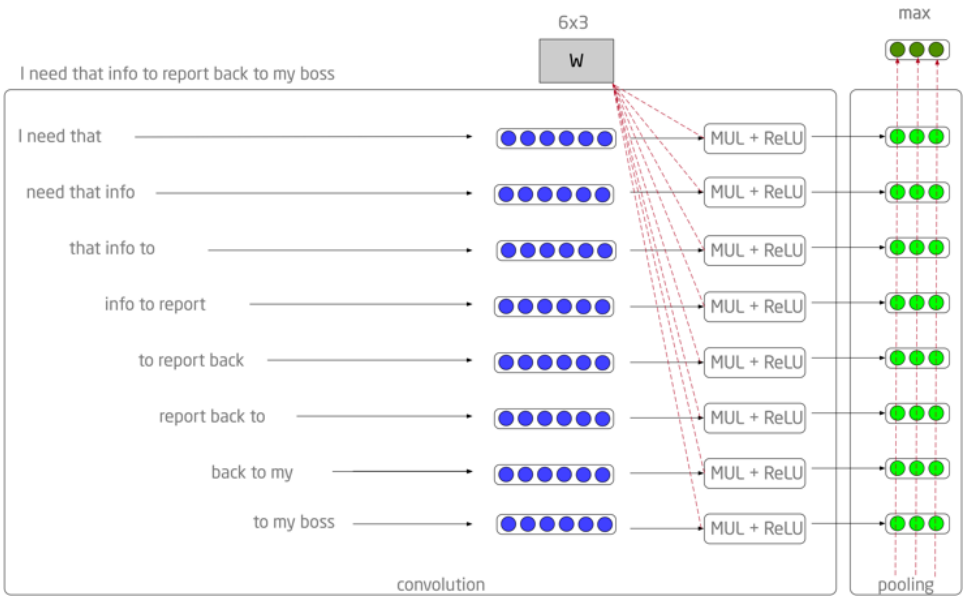

***Figure 9-8**. Illustration of convolution, pooling, and max-pooling*

9.4. Corpus

PERSU-R was trained using the CSE-PUC Corpus which was created by collecting realized and fictional social engineering attack dialogues from social engineering dark websites (forums, tutorials, etc.), social engineering books, and several logs. The corpus was enriched using the word-embedding technique by adding sentences with synonymous or similar words based on a pre-defined ranking.

After a pre-processing pipeline composed of noise removal (stopwords, empty lines, etc.), tokenization, and standardization, the CSE-PUC Corpus was created having the characteristics presented in **Table 9-1**.

***Table 9-1.** CSE-PUC Corpus*

Characteristic	**Value**
Corpus name	CSE-PUC Corpus
Collection Methods	Web scraping, pattern-matching text extraction
Corpus size	(N) 56 text dialogues/3380 sentences
Vocabulary size	(V) 4500 terms
Content	chat-based dialogues
Collection date	June 2018–December 2020
Creation date	June 2021

The CSE-PUC Corpus was explicitly labeled for the pp-container prediction task. Each of the 3880 sentences was labeled as a pp-container based on the criterion of whether any of Cialdini's persuasion characteristics exist in the sentence or not. After the annotation task, the final dataset was well-balanced, as depicted in **Figure 9-9** where 51,1% of the sentences were pp-containers and 48,9% were not (denoted in the figure as neutral).

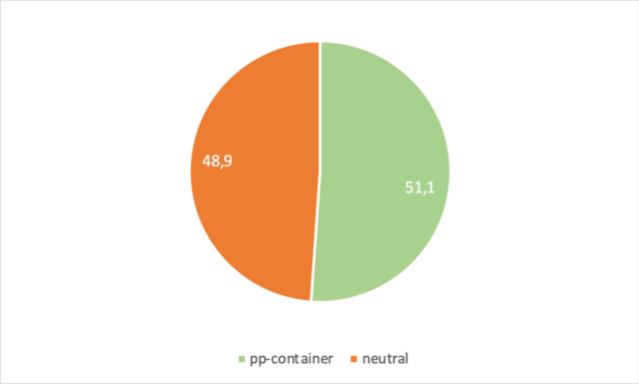

Figure 9-9. *The CSE-PUC Corpus is composed of 51,1% pp-container sentences and 48,9% neutral sentences.*

The distribution of the number of words in the sentences in each class is shown in **Figure 9-10**. We can observe that most social engineers utilize short sentences to unleash their attack, and they use more words to become friendly before their actual attack.

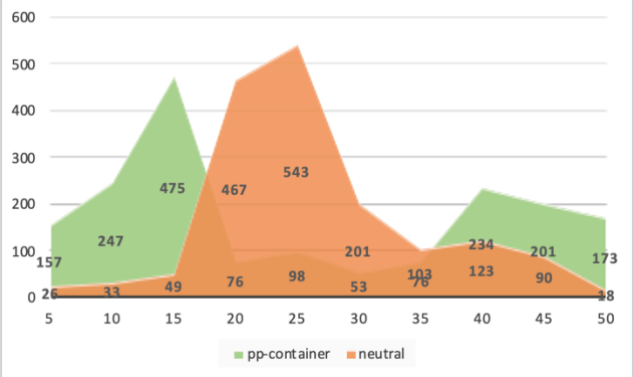

Figure 9-10. *Distribution of words per sentence class in the CSE-PUC Corpus*

9.5. Training

Parameters like the number of filters, filter sizes, the architecture of the network, etc., have all been fixed before Step 1. They did not change during the training. A summary of the training process follows, while **Table 9-2** presents the same process in pseudocode:

- STEP1**:** All filter values and weights were initialized using fastText word vector representations (either own-trained on the CSE-PUC Corpus or pre-trained).
- STEP2: The network takes a training sentence as input, then the forward propagation step is executed where the convolution operation, ReLU, and pooling operations take place in the fully connected layer and outputs the probabilities over the classes. Let us suppose that the output probabilities for the sentence "I need that info to report back to my boss" are at the end of the first epoch [0.9, 0.1]. For the first training example, the weights are randomly initialized; thus, the output probabilities are also random.
- STEP3: Calculate the total error at the prediction layer with the following formula

$$Total\ Error\ = \sum \frac{1}{2}(target\ probability - output\ probability)^2 \qquad 9.1$$

- STEP4: A back-propagation algorithm was used to calculate the gradients of the error regarding all weights in the network. Then, gradient descent is used to update all the weights and filter values to minimize the output error.
    - The weights are adjusted in proportion to their contribution to the total error.
    - The values of the filter matrix get updated.
- STEP5: Repeat steps 2-4 with all sentences in the training set.

When a new (unseen) sentence reaches PERSU-R as input, the network goes through the forward propagation step and outputs a probability for each class (for a new sentence, the probabilities over the classes are calculated using the weights that have been optimized to classify all previous training examples correctly).

Table 9-2. *Algorithmic steps of PERSU-R training*

Algorithm PERSU-R
Require labeled SOURCE corpus, unlabeled TARGET corpus, hyperparameters
Input dataset $\mathcal{D}$ ← CSE Corpus, n words $w_{1:n} = w_1, w_2, \dots, w_n$, sentence $E_{[w_i]} = w_i$, concatenated vector of k words x_i , weight vector u, nonlinear-linear activation function f, $r_i \in \mathbb{R}, x_i \in \mathbb{R}^{k*d}$ and $u \in \mathbb{R}^{k*d}$.
Initialization *CSE-PUC training hyperparameters initialization: batch size, training epochs, learning rate, dropout rate, optimizer*

Begin Training:

Step 1*: CSE-PUC network hyperparameters initialization: filter size, number of filters, weight values, padding, activation function, stride length, pooling method,*

Loop: ***for each*** *sentence E in dataset* $\mathcal{D}$

Step 2*:* ***do*** *forward propagation:*

1-D Convolution operation, transform vector $x_i = [w_i, w_{i+1}, \dots, w_{i+k}] \in \mathbb{R}^{k*d}$ *to* $r_i = f(x_i \circ u)$

Pooling operation: output q vectors $\boldsymbol{p}_{i:q}$ *where* $\boldsymbol{p}_i \in \mathbb{R}^{\ell}$

Step 3*: Calculate Total Error*

$$Total\ Error = \sum \frac{1}{2}(target\ probability - output\ probability)^2$$

Step 4*: Backpropagate the error and adjust the model parameters.*

Calculate gradients of the error regarding all weights in the network

Execute gradient descent:

Update all filter and weight values

End Loop

End Training

Output: *Probabilities over the classes*

The above steps train PERSU-R, which means that by the end of the training process, all weights and filter values of the network will be optimized and ready to classify sentences from the training set.

The training process of the example input sentence "I need that information to report back to my boss" from a higher-level perspective is illustrated in **Figure 9-11**.

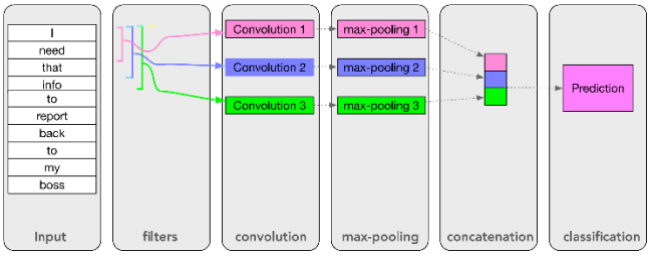

Figure 9-11. *End-to-end training of the persuasion classifier for recognizing whether a sentence is a pp-container.*

9.6. Results

PyTorch (Paszke et al. 2019) was used to implement the PERSU-R architecture. This Python library is dedicated to facilitating rapid research on deep learning models by easing the implementation of innovative neural network architectures. *AllenNLP* (Gardner et al. 2017), a Pytorch-based NLP library designed to support researchers who want to build novel natural language models, is also used. *Pytorch Lightning* ('PyTorch Lightning') and *wandb* (Biewald 2020) completed the toolset to manipulate the training pipelines and Bayesian hyper-parameter sweep.

The critical difference between the different variations of PERSU-R architectures that were tested in the word vector representation layer. Initially, word vectors were created using the cutting-edge fastText algorithm trained on the CSE-PUC corpus. These word vectors were specifically trained for this purpose. fastText (Joulin et al. 2016) is an algorithm and a word-embedding library developed by Facebook that uses information from linguistic units smaller than words to train high-quality word embeddings. In addition, fastText pre-trained word vectors were utilized, which were treated either as fixed (they are not updated during training) or as updated parameters of PERSU-R. Lastly, randomly initialized word vectors were employed in one test turn. Thus, the four variations of the PERSU-R's model were:

- Own-trained CSE-PUC: word embeddings are created using the CSE Corpus and fastText
- Fixed pre-trained CSE-PUC: fastText pre-trained vectors are used and remain fixed during the pp-container CNN training.
- Updated pre-trained CSE-PUC: fastText pre-trained vectors are used and updated during network training.
- Random CSE-PUC: where the word embeddings are initialized using random numbers

A standard 80/10/10 split of the CSE-PUC Corpus was made for the training/development/test sets respectively, and a five-fold validation was conducted using the average scores across folds to compare the performance of the different model variations. The learning rate was 0.001, the training batch size was 16, and all models were trained for ten epochs. Additionally, a dropout 0.1 probability was applied to reduce overfitting.

The four different learner implementations were also compared against an SVM baseline model, which is a traditional machine learning technique that is simple and flexible in addressing a wide range of classification tasks.

Table 9-3. *Training results*

Model	Accuracy	Macro F1
SVM	70.4 %	57,2 %
Own-trained CSE-PUC	62.2 %	54.8 %
Fixed pre-trained CSE-PUC	66.4 %	57.3 %
Updated pre-trained CSE-PUC	71.6 %	58.2 %
Random CSE-PUC	60.5 %	51.7 %

The experimental results are presented in **Table 9-3** and illustrated in **Figure 9-12** and **Figure 9-13**. In the former figure, the x-axis represents the number of epochs in training the variations in the CSE-PUC network. The y-axis represents the accuracy ratio of the CSE-PUC model on the validation set. The *Updated pre-trained CSE-PUC* yielded the best results among the four different CSE-PUC variations. It outperformed the *Fixed pre-trained CSE-PUC*, Own-*trained CSE-PUC*, and *Random CSE-PUC* by a clear margin, achieving an accuracy of 71.6%. The *Fixed pre-trained CSE-PUC* gave the second maximum accuracy of 66.4%, while the *Own-trained CSE-PUC* attained an accuracy of 62.2%, and the *Random CSE-PUC* had the lowest accuracy. These experimental results confirm the significance of word vectors learned from extensive unlabeled text data in capturing syntactic and semantic information.

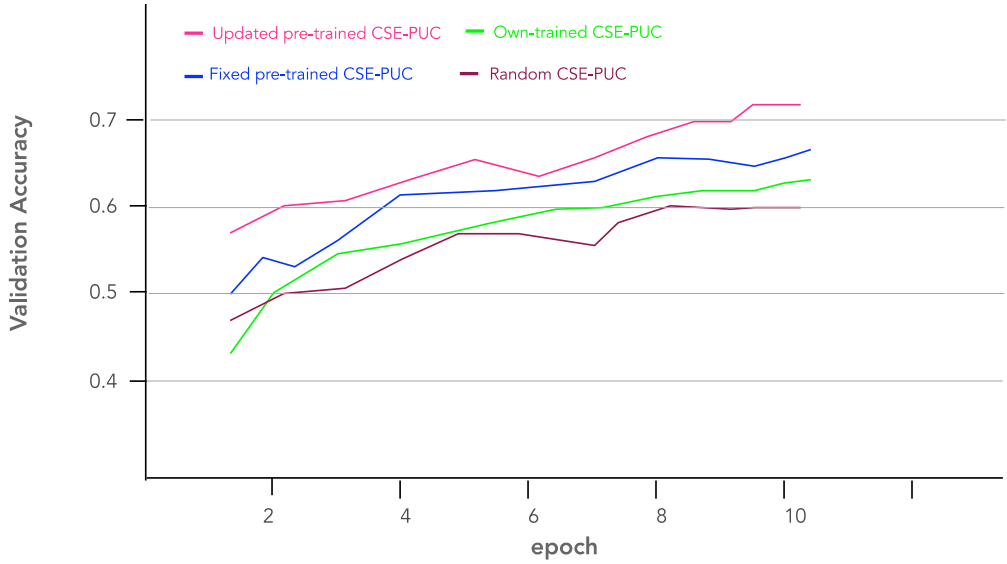

Figure 9-12. *Accuracy ratio of the validation set for four different CSE-PUC variations*

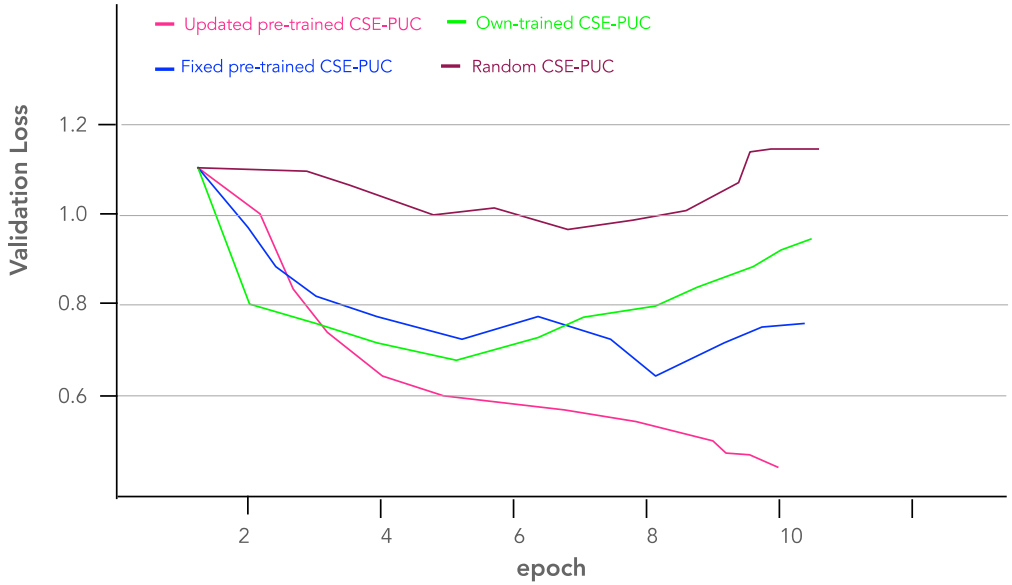

Figure 9-13. *Loss on the validation set for CSE-PUC model variations.*

The minor improvement of the pp-container CSE model against the pp-container random model can be attributed to the insignificant influence of the context information, which calls for further research regarding the extraction of context-related features. Nevertheless, the competitive results of the updated and fixed models are promising and can be used as part of a multifactor CSE recognition system, such as that proposed in a previous work (Tsinganos et al. 2018).

These results also suggest that fastText pre-trained vectors are suitable, 'universal' feature extractors, and can be utilized across datasets and corpora. Finetuning the pre-trained vectors for the pp-container classification task yielded satisfactory improvements.

9.7. Discussion

PERSU-R's architecture details and training choices are presented in **Table 9-4**.

Table 9-4. *PERSU-R architecture & training choices*

Description	Values
Batch size	16
Word Embeddings	fastText
Word Embeddings size	300
Filter sizes	2,3,4,5
Number of filters	100,100,100,100

Stride length	1
Zero padding	Yes
Activation function	ReLU
Pooling method	1-max
Dropout rate	0.1
Training epochs	10

Several different architectures of PERSU-R were tested (e.g., multiple layers of convolution), but no significant increase in performance was observed; thus, a simpler architecture was selected. This experiment acted as a proof-of-concept for the eligibility of simple neural network architectures as aid tools for cybersecurity defense mechanisms. The fast pace of research regarding deep learning algorithms and NLP techniques offers the opportunity for cybersecurity researchers to quickly adapt and apply new innovative tools. What is usually missing is the context awareness of the specific task at hand. In this case, recognizing whether persuasion methods existed in a sentence during a potential CSE attack was cast as a sentence classification problem. Furthermore, the NLP and deep learning toolset had to be tuned to align with the specific requirements of this approach. This customization was essential to ensure the successful functioning of the method. The toolset was enhanced to incorporate knowledge from the cybersecurity domain, particularly in the realm of CSE attack recognition. This customization enabled the toolset to be more attuned and effective in addressing the specific challenges of this domain. The CSE Corpus and persuasion-oriented annotation played a critical role in this awareness. The annotation transforms a corpus composed of CSE attacks into a corpus capable of recognizing persuasion attempts while remaining in the realm of CSE attacks.

Furthermore, the word-embedding layer of the overall PERSU-R architecture, which encodes the written sentences of the interlocutors was trained over the CSE-PUC Corpus. During this process contextual knowledge from the digital battlefield is injected into PERSU-R. In addition, the use of pre-trained word embeddings (fastText) yielded more efficient results than CSE pre-trained or randomly initialized word embeddings. This was expected because the vast corpus (600+ billions of words) used for training was not easy to beat. Nevertheless, the CSE-PUC variation that used word embeddings trained on CSE-PUC Corpus gave satisfactory results, and future enrichment of CSE-PUC Corpus would be beneficial for transfer learning.

The convolutional neural network as an encoder and a feature extractor has already been tested, measured (Zhang and Wallace 2015), and trusted for text classification. The convolution operation plays an essential role in preserving the spatial information of a sentence. In this study, a convolutional neural network with one convolution layer

was trained, employing multiple filters and filter sizes to capture multiple features. Different filters were able to capture distinctive features from each sentence. The CSE-PUC variations independently learned the values of these filters on their own during the training process. Prior to the training process, several hyperparameters need to be specified, including the number of filters and filter sizes.

All hyperparameters were chosen via a grid search of the development set. As the number of filters increases, more features are extracted, and the network's recognition capability improves for unseen sentences. However, to avoid overfitting, constraints were applied to the l2 norms of the weight vectors as a regularization technique by employing a dropout rate of 0.1. Various dropout rates were tested, that is, when the number of feature maps was increased, the dropout rate was also increased to avoid overfitting effects. A rectified linear unit (ReLU) was selected and used as the activation function for the convolution layer, but tanh was another excellent candidate with slightly worse performance results. Finally, mini-batches of size 16 were fed to PERSU-R, and filter sizes of 2, 3, 4, and 5 were used with 100 feature maps each. During training, the Adadelta optimizer was selected.

Due to the small size of the training corpus, addressing the out-of-vocabulary (OOV) problem was crucial. To mitigate this challenge, the fastText algorithm was employed, utilizing sub word information to alleviate the OOV problem. This approach effectively handled unseen words, enhancing the overall robustness of the system. It is common for a neural network to be trained for days or even weeks but such a training duration would be a limitation for a network like PERSU-R. Thus, training efficiency and speed are critical for an application like CSE attack recognition and choices must be made with care and after exhausting grid search. A neural network's complexity analysis is necessary because the dimensionality of a neural network is a key factor in its learning and performance ability. It is a design decision that should be taken with care as it plays a vital role in the computational cost. Although the computational cost involves several computer resources such as central processing unit speed, random access memory, etc. it usually refers to the time required to complete a certain operation. Concerning neural networks, the computational cost is a measure of the number of computing resources used for training or inference which leads us to know how much power or time we need to train or use a neural network. There are several ways to measure computational cost such as floating-point operations (FLOPS) or multiply and accumulate operations (MACs). Considering the operation of a neural network, measuring the multiplication of inputs and weights and adding them is critical to conclude about their computational cost. The convolutional layers that PERSU-R uses are very efficient for one-dimensional sequences analysis that we apply for the recognition of CSE attacks. Furthermore, even for larger text sequences, convolutional layers can be used as a pre-

processing step that extracts higher-level features that later will consume further processing cycles. PERSU-R has one convolutional layer and according to [57] the complexity is $O(n * d * k * f)$ where n is the sequence length, d is the representation dimension, k is the filter size and f is the number of filters. The fact that PERSU-R has only one convolutional layer keeps complexity low and makes the model efficient for implementation.

9.8. Chapter Conclusion

Chapter 9 focused on the development and evaluation of PERSU-R for recognizing persuasion techniques in CSE attacks. The chapter provided a comprehensive overview of the CSE-PUC model which is the model that PERSU-R utilizes, including its incorporation of persuasion principles, the utilization of Convolutional Neural Networks (CNNs), and the architecture of PERSU-R. Through this analysis, significant insights were gained into the effectiveness of PERSU-R in accurately identifying and characterizing persuasion techniques based on textual inputs.

PERSU-R integrates persuasion principles as a foundation for recognizing persuasive techniques employed by social engineers in chat-based interactions. By leveraging CNNs, the CSE-PUC model captures meaningful patterns and features within the text, enabling accurate identification of persuasive elements. The architecture of the CSE-PUC model forms the core of PERSU-R, incorporating the operation and architecture components to effectively recognize and categorize persuasion techniques.

During the development process, CSE-PUC comprising instances of persuasive interactions was carefully constructed for training and fine-tuning. The model underwent training to learn the nuances of different persuasion techniques and their associated patterns in the text. The evaluation of PERSU-R's performance was conducted using comprehensive metrics, including accuracy, precision, recall, and F1 score, on a held-out test dataset. These metrics provided valuable insights into the model's effectiveness in identifying and categorizing persuasion techniques accurately.

The results suggest that satisfactory performance levels can be achieved while keeping a simple model architecture and low computational costs. This approach broadens the understanding of embracing machine learning models to respond to real-life cyber threats and might help fellow researchers adapt similar machine learning algorithms to solve cybersecurity problems. This research indicates that the same approach can be used as a generic solution framework within which we can cast cybersecurity problems related to natural language into text classification problems. Following a similar pattern, we can also try to recognize deception acts, recognize personality traits, or recognize speech acts by using modern text-classification machine-learning techniques.

10. PERSI-R

10.1. Introduction

In a chat-based dialogue e.g., between an SME employee and a potential customer, the interlocutors exchange written sentences during their communication. The ability to identify one or more characteristics (Tsinganos et al. 2018) that can discriminate a normal interlocutor from a malicious one can lead to sufficiently protect the SME employee from a potential CSE attack. To achieve her goal, a malicious interlocutor repeats her arguments many times in a conversation to convince and manipulate her partner. This was also confirmed, after processing the CSE Corpus (Tsinganos and Mavridis 2021), where it was observed that 83% of social engineering attackers tend to insist on their arguments to exfiltrate the targeted type of critical information. The persistence of an interlocutor to regurgitate the same topic that could lead to sensitive data exfiltration is an enabler of a successful CSE attack. Thus, there is a need to develop a mechanism for recognizing when an interlocutor is continuously trying to lead the conversation to a specific and previously mentioned topic.

To cope with such a problem, we can adopt different approaches depending on the pros and cons of each one. Traditional machine learning models require substantial amounts of labeled data to be trained for a recognition task, but labeling data is a time-consuming and expensive activity. To achieve persistence recognition, the latest advancements in deep learning, particularly the Transformers (Lin et al. 2021), were leveraged. These cutting-edge techniques were employed to enhance the effectiveness of the persistence recognition process. Transformers use substantial amounts of unlabeled raw text for training and take advantage of the linguistic information contained to overcome the difficulty of creating a large amount of labeled data. The Transformer models are very efficient in learning the context and relationships between sequential data, such as words in a sentence. Even if supervised learning using labeled data is an option, learning robust word vector representations in an unsupervised manner of learning can lead to a significant performance boost.

In this chapter, an approach is described for the paraphrase recognition task, in the context of CSE attacks, based on a combination of unsupervised pre-training and supervised fine-tuning based on the Transformer-based BERT model (Devlin et al. 2019). This approach results in a model, called CSE-PersistenceBERT, which learns a universal representation that transfers knowledge with only minimal adaptation to the paraphrase recognition task and it is the main model of PERSI-R recognizer. Initially, the CSE-PersistenceBERT model has access to a large corpus of unlabeled text on which BERT has been pre-trained, and later it uses the CSE-Persistence Corpus, which

emerged from the CSE Corpus (Tsinganos and Mavridis 2021). CSE-Persistence Corpus was generated after manually annotating the CSE Corpus and utilizing the CSE ontology (Tsinganos and Mavridis 2021) . This corpus which is relevant to the downstream task was used to fine-tune PERSI-R. The complete training procedure was performed in two stages: firstly, the BERT model parameters were learned by training the model using unlabeled data on a language modeling objective, and secondly, a part of the pre-trained parameters was fine-tuned on the paraphrase recognition task using labeled data from the CSE-Persistence Corpus. The evaluation results confirmed that PERSI-R can recognize the persistence of an interlocutor in an early stage of a chat-based conversation. This can result in an indicator of a potential CSE attack, given that the topic of the conversation is an entity in the CSE ontology.

This chapter contributes to improving knowledge about utilizing existing state-of-the-art deep learning models as cyber defense mechanisms. This study provides a better understanding of how we can tailor a BERT-based language model to expose the malicious intent of an interlocutor. Furthermore, detailed implementation details are given about fine-tuning such a model using an in-context and appropriately annotated corpus for the paraphrase recognition task.

10.2. Background

10.2.1. Text Similarity

In Natural Language Processing (NLP) terms, persistence recognition can be cast as a Natural Language Understanding (NLU) task. Furthermore, there are a plethora of relative NLU tasks that can be utilized to describe persistence recognition, such as Semantic Textual Similarity, Natural Language Inference, and Paraphrase Recognition. All these NLU tasks are closely related, but they also differ in several aspects:

- Semantic Textual Similarity (STS) (Chandrasekaran and Mago 2022; Agirre et al. 2012) is the semantic task of inferring the relation between different text data units. It is usually measured as a numerical score in the range of [0,1] and quantifies the semantic similarity between different text data units. In earlier days, techniques like Bag of Words (BoW) and Term Frequency - Inverse Document Frequency (TF-IDF) were used to represent text as vectors to aid the calculation of semantic similarity. These techniques were designed to identify if two text units contained the same words or a similar group of characters. However, sentences can have different meanings while containing the exact, same words or can contain different words while representing semantically similar concepts.

- Natural Language Inference, sometimes also called Textual Entailment (Manning 2006; Marelli et al. 2014; Dagan, Glickman, and Magnini 2006), is the classification task of determining, given a text premise and a text hypothesis, whether the hypothesis is entailed by, contradictory to, or independent of the premise. The most popular categories used for textual entailment relationships are entailment, contradiction, and neutral.
- Paraphrase Recognition (Vrbanec and Meštrović 2020) can be described as another text classification task that determines whether two text data units have a similar meaning (or not) in a specific context.

Semantic textual similarity identifies the semantic equivalence between text data units as a continuous value, while textual entailment and paraphrase recognition produce a categorical output. However, it is possible to quickly transfer, e.g., from the STS measure area to the Paraphrase Recognition task, by introducing a paraphrase recognition threshold. For example, one can set a certain value of the STS measurement above which two different text data units can be considered as related. The aforementioned NLU tasks can also broadly be classified based on the approach used to achieve the objective, as follows:

- knowledge-based, where ontologies, databases, or dictionaries are used (e.g., WordNet ('WordNet | A Lexical Database for English'))
- corpus-based, where information retrieved from large corpora is used (e.g., word2vec (Mikolov, Chen, et al. 2013))
- deep neural network-based, where recent developments of neural networks are utilized (e.g., CNNs (Y. Kim 2014), Transformers (Lin et al. 2021), etc.), and
- hybrid-based, where characteristics from all the above-mentioned methods are combined (Mohamed, Gomaa, and Abdalhakim 2019).

For each of these NLU tasks, there are several popular datasets used for the performance evaluation of the corresponding algorithms. These datasets are composed of sentences or pairs of sentences with associated classification labels.

10.2.2. Transformers

A neural network model that transforms one sequence of tokens into another performs a sequence-to-sequence (Seq2Seq) task, as it accepts a sequence of tokens as input and produces another sequence of tokens as output. Known applications of such tasks are machine translation (McCann et al. 2017), spell checking (Singh and Singh 2020), etc. This paradigm, until recently, was implemented by neural networks based on an encoder-decoder architecture like recurrent neural networks (RNNs) (Lipton, Berkowitz, and Elkan 2015). The encoder e.g., a Long Short-Term Memory (LSTM)

(Yu et al. 2019) neural network, takes a sequence of tokens and outputs a fixed-size vector representation of the input. Then, the decoder, which is also an RNN, uses this vector to produce the output sequence of tokens. The limitation of this workflow lies in the information compression that occurs during the transformation of the input sequence into a fixed-length vector. No matter how lengthy the input sequence is, its representation will always have to fit into a fixed vector size, and consequently a significant part of the information is lost.

This bottleneck was overcome using a mechanism called *Attention* (Vaswani et al. 2017; Bahdanau, Cho, and Bengio 2014). Attention can produce a summary for each input token, which is context-dependent. This summary is a list of vectors that act as a memory that the decoder can look up during output production. These vectors represent the hidden states of the encoder and, in NLP, we can think of them as key-value pairs representing input tokens. An attention function *f* is applied on every key producing a weight, and then the input values are weighted by the corresponding weight and summed up to create the summary vector. This weighted sum is then appended to the decoder's hidden states to produce the final output sequence.

A *Transformer* [19] is a relatively new type of encoder-decoder neural network architecture that utilizes a specific type of attention mechanism based on the concept of *self-attention*. Due to their outstanding performance over RNNs, Transformers are now the de facto standard architecture to use in related NLP tasks (machine translation, etc.). The self-attention mechanism produces, for each input token, a summary of the entire input using as context the specific token. Furthermore, a Transformer uses *multi-head self-attention* by using multiple sets of key-value pairs and queries per token, thus producing multiple sets of weights focused on different input sequence characteristics. By repeatedly applying several layers, the input sequence is transformed from a raw word embedding to a more abstract form representing the input's semantics. While a Transformer is a powerful model by itself, it also acts as the underlying architecture of well-known pre-trained models such as GPT-2 (Budzianowski and Vulić 2022) and BERT (Devlin et al. 2019).

Pre-trained models are used in the *transfer learning* (Ruder 2019), which is a collection of techniques that improve the performance of a neural network model in a task using data and/or a model trained for a different task. Transfer learning consists of at least two steps (Peters, Ruder, and Smith 2019);

- pre-training, where the model learns a general-purpose representation of inputs, and
- adaptation, where the input representation is transferred to a downstream task. The adaptation has two main paradigms:

- Feature extraction, where the model's weights remain unchanged and are used as features in a downstream task.
- Fine-tuning, where (some of) the model's weights are unfrozen and fine-tuned for the new downstream task.

In the first step, the model is trained for one task (pre-training), and in the second step, it is adjusted for another task (fine-tuning). Transfer learning uses word embeddings, which are learned vector representations. Thus, semantically similar words share similar representations. Although these word embeddings can be used for downstream NLP tasks, they are limited in the sense that they are trained per token and thus ignore context.

BERT is a transformer-based pre-trained language model that employs the transformer encoder part to transform the input into contextualized embeddings through a series of layers that gradually summarize the input sequence. Due to its transformer-based architecture, BERT can capture long-term dependencies between input tokens, considering the context of the token in both directions. BERT is trained using self-supervised learning, which means there is no need for human intervention, e.g., data annotation. BERT's training comes from the data itself, as the humungous datasets used for training contain deep linguistic knowledge such as collocation, syntactic, grammatical, and semantic information.

In the literature, the big pre-trained language models are sometimes referred to as base or Foundation models (Bommasani et al. 2021), and their availability gave rise to the terms of fine-tuning, transfer learning, and classification heads. Fine-tuning implies updates to some of the model's layers, while classification heads treat the last layer of the model as input features: they take input X and predict the outcome Y performing classification if the labels are categorical or regression if the labels are continuous (Church, Chen, and Ma 2021).

10.3. Model

The approach proposed in this thesis, incorporates the technologies of Transformers, self-supervised learning, pre-trained language models, and transfer learning to boost the performance of the paraphrase recognition task in the context of a CSE attack. For this purpose, the PERSI-R recognizer was developed, a fine-tuned version of the BERT model. PERSI-R defines a sentence vector-based model that performs text classification on pairs of fixed-size sentence representations that are computed independently of one another. PERSI-R takes as input an instance composed of two sentences and outputs a classification result for them.

BERT is available in two variants: BERT-base and BERT-large. For this work, the decision was made to utilize BERT-base, which consists of twelve layers of transformer encoder blocks and a total of 110 million parameters. This choice of model architecture provides a powerful and expressive framework for capturing and understanding the contextual information within the text data, enabling effective analysis and processing of the CSE attack domain. The transfer learning technique is used to improve the performance of the downstream NLP task, which in this study is the paraphrase recognition task, using a model that is already trained (pre-trained) on a different task (e.g., language modeling). Transfer learning, as adopted in this work, is depicted in **Figure 10-1**.

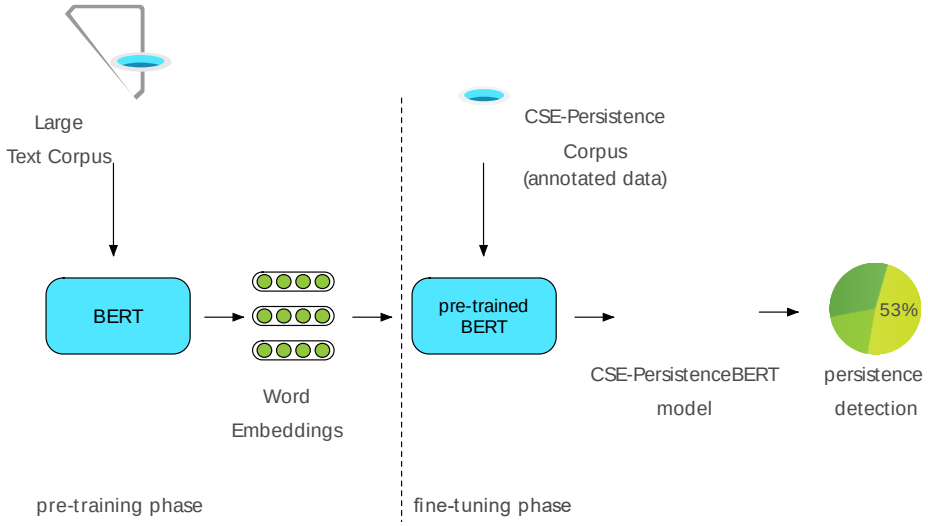

Figure 10-1. *The proposed Transfer Learning approach*

The end-to-end learning process of the complete neural network is divided into a pre-training phase where the BERT-base model is trained by masking word tokens in each sentence of a large text corpus and a fine-tuning phase where the pre-trained BERT model is learning the persistence recognition labels, once again, from the CSE-Persistence corpus. This last training is performed only on the last BERT layer to extract features that will allow the model to use the representations of the pre-trained model.

BERT uses the [CLS] token that is placed at the beginning of a token indicating the start of the input sequence. In addition, the [SEP] token separates the two sentences, and the [PAD] token is used for padding. Thus, an example instance would be the following:

```
[CLS] <sentence 1> [SEP] <sentence 2> [SEP] [PAD]
```

The [CLS] vector acts as an input to the dense layer, and the similarity is calculated using a cosine similarity measure. The cosine similarity of two vectors a and b is computed according to formula (10.1):

$$cosine\_similarity(a,b) = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}}, \quad 10.1$$

where a, and b are vectors of dimension n.

During the fine-tuning phase, no modifications are made to the neural network architecture. The focus is primarily on adjusting the model's parameters and training it on domain-specific data to improve its performance on the target task. More specifically, the classification pipeline (see **Figure 10-2**) is as follows: The pre-trained BERT tokenizer sends to the transformer encoder a sequence of tokens, which is composed of the two input sentences concatenated and separated by a special [SEP] token. A [CLS] token is prepended to the sequence denoting the start of the sequence. This token is used to extract the final embedding of the input instance. [PAD] token is a way to keep the input size constant. The loss function is minimized by continuously training the entire neural network on the CSE-Persistence Corpus data. The loss function used is cross-entropy, which is computed according to formula (2):

$$l_{cross-entropy}(\hat{y}, y) = -\sum_{i=1}^{C} t_i \log(\hat{y}_i), \quad (2)$$

where C is the number of different classes in the data, $\hat{y}$ is the predicted probabilities vector over the classes for the instance, y is the correct label for the instance, t_i indicates binary if i is the correct label for the specific instance, and $\hat{y}_i$ is the predicted probability that the instance belongs to class i.

The training objective is to find the parameters that minimize the function, as mentioned above, which equally translates to:

$$arg_\theta min \frac{1}{|D|} \sum_{(x,y)\in D} l(\hat{y}, y),$$

where D is the dataset consisting of n data points (x_i, y_i) usually referred as input vectors. x_i is the vector inputted into the neural network and y_i is the accompanying label.

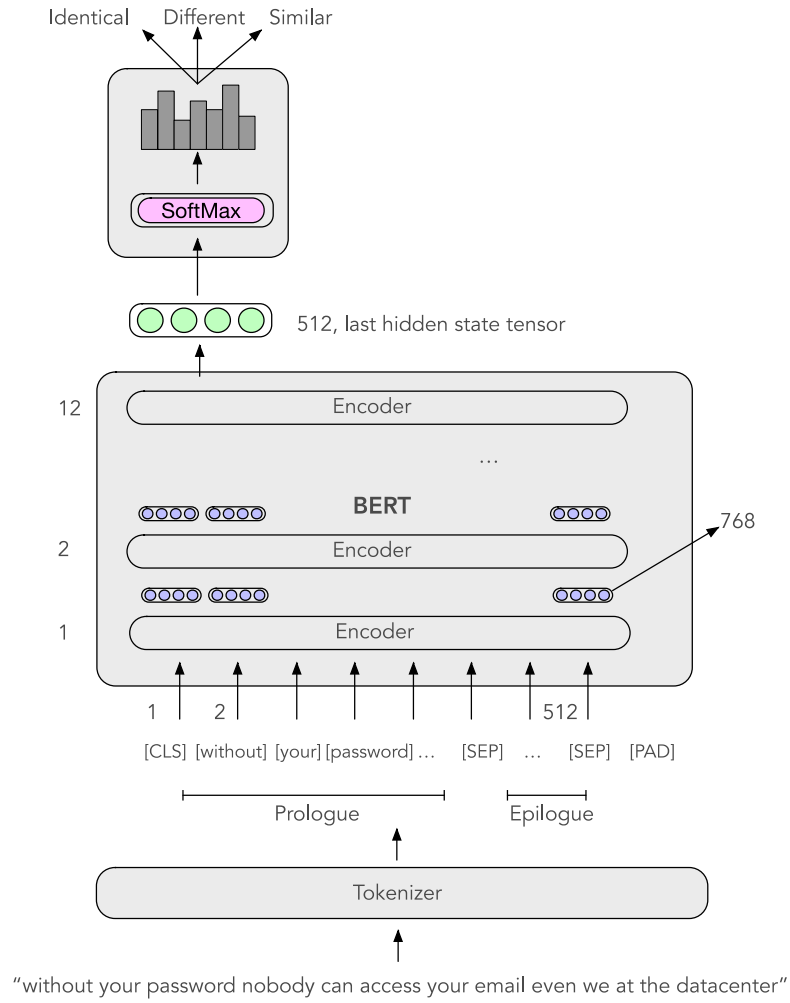

Figure 10-2. *The proposed classification pipeline.*

Instead of being initialized randomly, PERSI-R's weights are inherited by the pre-trained BERT model, and the neural network is trained from scratch. The final sentence representations, which are a set of values called *logits,* are then passed to the *SoftMax (Goodfellow, Bengio, and Courville 2016)* function to derive a probability distribution regarding the sentences' paraphrase recognition task. Some of the BERT's weights, which were initialized with the pre-trained values, are also fine-tuned through backpropagation. The combination of the linear layer with SoftMax is called a *head*. Therefore, it is said that we are attaching a classification head to BERT to solve the prediction task. The SoftMax function that outputs the probabilities of the instance belonging to each class is the following:

$$softmax(z_i) = \frac{\exp(z_i)}{\sum_j \exp(z_j)},$$

where z_i is each element in the last layer of the neural network.

For the implementation, the Hugging Face (Transformers 2022) library of transformers was used along with AllenNLP (Gardner et al. 2017) and Pytorch (Paszke et al. 2019). The Hugging Face library has become the standard library for NLP researchers, and several state-of-the-art model implementations exist, such as GPT-2, BERT, RoBERTa, etc., with or without pre-trained model parameters. Using AllenNLP and BERT-as-service (Raval 2022), a text classification model was implemented that embeds the input sequence, encodes it with a seq2vec (H. J. Kim, Hong, and Cha 2020) encoder, and finally classifies it with the help of a SoftMax layer coupled with a classification head. The embedding is done by BERT, BertPooler (Gardner et al. 2017) did the encoding, and Adam (Gardner et al. 2017) was employed as the optimizer.

For the fine-tuning of PERSI-R, the hyperparameter values were set as recommended in (Devlin et al. 2019): a maximum length of the input sentence to the model of 128 (max_length = 128), a batch size of 32 (batch_size = 32), a learning rate of 3e5; 4 training epochs (epochs = 4), and a dropout probability of 0.1.

During the persistence recognition task, an additional aspect involves maintaining a top-five ranking of the most similar sentences. This ranking enables the identification of sentences that exhibit high similarity to aid in the recognition process. E.g., for the sentence "The chief executive character wants to change her password", the top-5 ranking returns the following (**Table 10-1**):

Table 10-1. *Top 5 of similar sentences*

Index	Sentence	Similarity
1	The chief executive character is in a meeting with important clients and would like the password reset as his current email account password no longer works	0.7804
2	And listen we just installed an update that allows people to change their passwords	0.6937
3	Now go ahead and type your password but don't tell me what it is	0.5255
4	You should never tell anybody your password not even tech support	0.3490
5	In this case, would you like to reset your password	0.3488

10.4. Corpus

The Stanford Natural Language Inference (SNLI) corpus (Bowman et al. 2015) is a well-known extensive collection of English written sentence pairs manually annotated with entailment, contradiction, and neutral labels. It is usually used to determine the

semantic relationship between two different text data units, a *premise,* and a *hypothesis.* Following the SNLI paradigm, the CSE-Persistence corpus was produced by modifying and annotating the CSE Corpus (Tsinganos and Mavridis 2021) to be suitable for the task of paraphrase recognition. For this purpose, the CSE ontology (Tsinganos and Mavridis 2021) was utilized, which is asset-oriented and connects social engineering concepts with cybersecurity ones. The CSE ontology focuses on sensitive data that could leak from an SME employee during a chat-based conversation. It groups similar in-context concepts to facilitate the hierarchical categorization of an SME's assets and does not exceed three levels of depth to be efficient for text classification algorithms. The CSE ontology was created using a custom information extraction system and several text documents as input (corporate IT Policies, IT professionals' CVs, ICT Manuals, and others). An excerpt of the CSE ontology is depicted in **Figure 10-3**

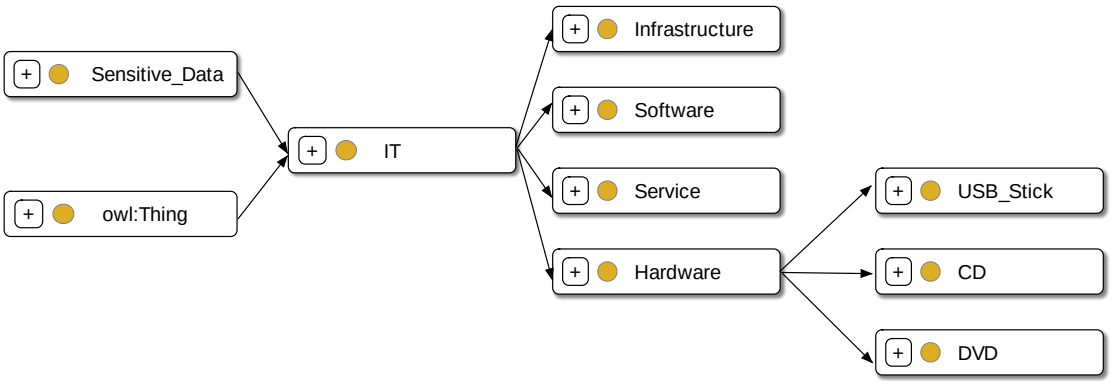

Figure 10-3. *Excerpt of the CSE ontology*

Each instance of the CSE-Persistence corpus is composed of two sentences and is manually labeled as being a member of one of the following three categories:

- *Identical (I)*: the two sentences are semantically close *and* share a common term targeting the same leaf entity in the CSE Ontology (e.g., USB_Stick in Figure 1).
- *Similar (S)*: the two sentences are semantically related *and* share a common intent, which translates into a higher-level entity in the CSE ontology, targeting a different leaf entity (e.g., Hardware as the higher-level entity and CD as the leaf entity in Figure 1).
- *Different (D)*: the two sentences are not semantically related, *and* they do not share a common higher-level or leaf entity.

To capture the essence of each instance in the CSE-Persistence corpus, an analogy is drawn to a miniature drama play composed of only two sentences. Following this concept, specific names have been assigned to the two sentences, drawing from commonly used terminology associated with dramatic structure. These assigned names

provide a descriptive framework for understanding and analyzing the dialogues within the corpus, facilitating further analysis and interpretation

- The first sentence is referred to as the Prologue in the sense that the social engineering attacker uses it to introduce her intention to the play.
- the second sentence is referred to as the *Epilogue* in the sense that it concludes the play and informs us how the story ends.

Thus, the CSE-Persistence Corpus contains instances of training examples where each instance is composed of a string for the Prologue, a string for the Epilogue, and a Paraphrase Recognition label (Identical, Similar, Different).

Table 10-2 presents the identity of the CSE-Persistence corpus, providing several key statistics:

Table 10-2. *Key Statistics of the CSE-Persistence Corpus*

Data set size	16900 sentences
Type of text units	Pairs of sentences
Source of judgment	Three judges
Training pairs	13520
Development pairs	1690
Test pairs	1690
Identical sentences	6484
Similar sentences	5023
Different sentences	5393

Several text pre-processing steps occurred on the CSE-Persistence Corpus, such as identification and removal of one-word sentences, ensuring appropriate text file encoding, and noise removal, e.g., stopwords, emoji, etc. to prepare it as input to PERSI-R.

In a separate validation phase, apart from the custom annotation, two more judgments were collected for each label of the 16900 examples, where a 96% annotator consensus emerged. In **Table 10-3**, three random sentence pairs with the selected gold label (Identical, Similar, Different) in bold, and each annotator's complete set of labels are depicted. E.g., the first sentence pair was classified as Identical (I) by us and the two annotators. Thus, the three labels are III, and the final Paraphrase Recognition Label (gold label) is Identical.

Table 10-3. *Examples of sentence pairs from the annotated CSE-Persistence corpus*

Prologue	Epilogue	Annotators Labels	Paraphrase Recognition Label
without your password,	I can cut some corners to save	III	**Identical**

nobody can access your mail, even we at the data center	some time, but I'll need your username and password		
Just double click on the icon when it downloads	You must open the file when it's done	SSS	**Similar**
Are you using a USB extension cord	Thanks for the quick replies	DDD	**Different**

The underlined word is the entity found in the CSE Ontology. The first pair of sentences in **Table 10-3** uses the leaf-entity "*password"* of the CSE ontology. The second pair of sentences does not share a leaf entity but a higher-level entity "*IT" (Fig.1).* The higher-level entity "IT" contains the leaf entities "icon" and "file". Finally, the third pair of sentences has neither a leaf entity nor a higher-level entity in common.

To produce a well-balanced dataset, each *Prologue* appears five times in the CSE-Persistence corpus with a different *Epilogue* and the corresponding label. An example of a corpus instance in JSON format is presented in **Figure 10-4**:

```
{
  "prologue": "without your password, nobody can access your mail, even we at the data center",
  "epilogue": "I can cut some corners to save some time, but I 'll need your username and password",
  "label": "Identical"
}
```

Figure 10-4. *Example of a CSE-Persistence instance in JSON format*

10.5. Training

During the training process, three columns were considered from the CSE-Persistence corpus — 'Prologue', 'Epilogue', and 'Paraphrase Recognition label'. The Paraphrase Recognition label is the 'gold_label' given to the instance after the manual annotation of the dataset. An excerpt of the CSE-Persistence corpus that is used for persistence recognition follows in **Table 10-4**:

Table 10-4. *Excerpt from the CSE-Persistence corpus*

#	Prologue	Epilogue	Paraphrase Recognition label

1	I have my resume on this USB key	Are you using a USB extension cord	identical
2	See, without your password, nobody can access your mail	As smart as they are, they didn't the password	identical
3	When did you last change your password	I can cut some corners and save some time but I ‘ll need your username and password	identical
4	My network connection just went down like you said	I am just working on an audit	different
5	Hello how may I help you	Is there anything else I can help you for today	similar
6	It's called Doctors Database and I believe that they are located in Denver Colorado	Hello John. This is Bill Jenkins from Doctors Database in Denver	identical
7	We're trying to troubleshoot a computer networking problem	In the back of the computer can you recognize the network cable	identical
8	I am sorry for interrupting you but I am experiencing a problem with my Charge 2	When I tried to turn on the Charge 2 I saw that the battery was leaking	similar
9	I need that info to report back to my boss	My boss will probably fire me if I don't have it for the morning	similar
10	The chief executive character is in a meeting with important clients and would like the password reset as his current email account	But we need the details or we can't give you any information	different

The CSE-Persistence Corpus was divided into train, validation, and test set, where all the source data were elicited from successful or unsuccessful CSE attacks gathered from websites, books, and logs as described in a previous work (Tsinganos and Mavridis 2021).

10.6. Results

The evaluation of the CSE-PersistenceBERT model was conducted using standard performance measures commonly employed to assess classification tasks.:

True Positives (TP): sentence pairs where the true tag is positive and whose category is correctly predicted to be positive.

False Positives (FP): sentence pairs where the true tag is negative and whose category is incorrectly predicted to be positive.

True Negatives (TN): sentence pairs where the true tag is negative and whose category is correctly predicted to be negative.

False Negatives (FN): sentence pairs where the true tag is positive and whose class is incorrectly predicted to be negative.

Using the above measures, *Accuracy* was calculated, defined as the number of sentence pairs correctly identified as either true positive or truly negative out of the total number of entities

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (5)$$

For the baseline model, word2vec (Mikolov, Chen, et al. 2013) is used to produce word vector representations in $\mathbb{R}^{300}$. Then, the sentence vector embeddings were generated by averaging the vector embeddings of all tokens in the sentence. Furthermore, BERT-base was employed without fine-tuning as a comparison model to support the proposed model's superiority.

During PERSI-R's fine-tuning, the only required architecture changes that are appropriate for the paraphrase recognition task concern the extra fully-connected layers. During the supervised learning of the downstream task, the parameters of these extra layers were learned from scratch, while some of the parameters of the pre-trained BERT model were fine-tuned. After fine-tuning, PERSI-R was compared to the baseline model and achieved the accuracy values presented in **Table 10-5**, which are also aligned with the work in (Phang, Févry, and Bowman 2019).

Table 10-5. *Model benchmarks on the CSE-Persistence Corpus*

Model	Training Accuracy (%)	Training Loss	Validation Accuracy (%)	Validation Loss
Baseline	82.57	0.36	73.83	0.39
BERT-base	84.01	0.24	76.79	0.37
PERSI-R	84.96	0.21	78.03	0.36

Figure 10-5 (a), depicts the accuracy of the training and validation data sets of PERSI-R, and **Figure 10-5** (b) depicts the loss of training and validation datasets.

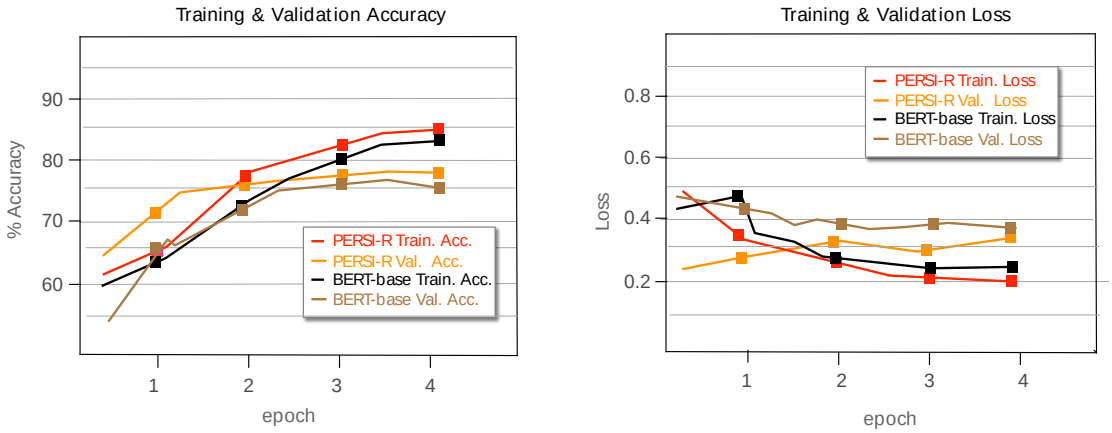

Figure 10-5. *(a) Training & Validation Accuracy (b) Training & Validation Loss*

(a) **(b)**

A comparison of the accuracy of the baseline model vs. PERSI-R is depicted in **Figure 10-6** as the validation set of the CSE-Persistence Corpus varied from 0% to 100%. In every percentage of data used, PERSI-R outperforms the baseline model. One key observation of the experimental results was that the difference between PERSI-R and baseline model's accuracy was bigger when the corpus was smaller (between 20% and 40%). This shows that even a complex BERT-based model can be efficient with a smaller corpus.

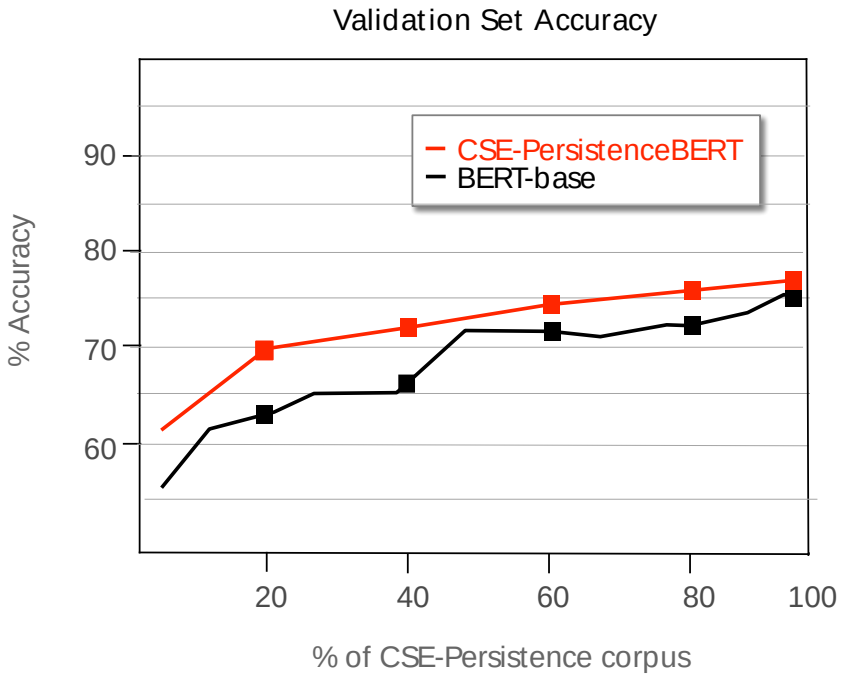

Figure 10-6. *Comparison of accuracy on the validation set as a function of corpus percentage.*

In addition, an investigation was carried out to assess the influence of the number of layers transferred during fine-tuning, specifically from the unsupervised pre-training phase to the persistence recognition task. **Figure 10-7**, depicts the performance of PERIS-R as a function of the transferred layers. As an observation, it was confirmed that each layer of the pre-trained BERT model contains valuable information for the persistence recognition task. Thus, PERSI-R improved its performance in the persistence recognition task by transferring its "knowledge".

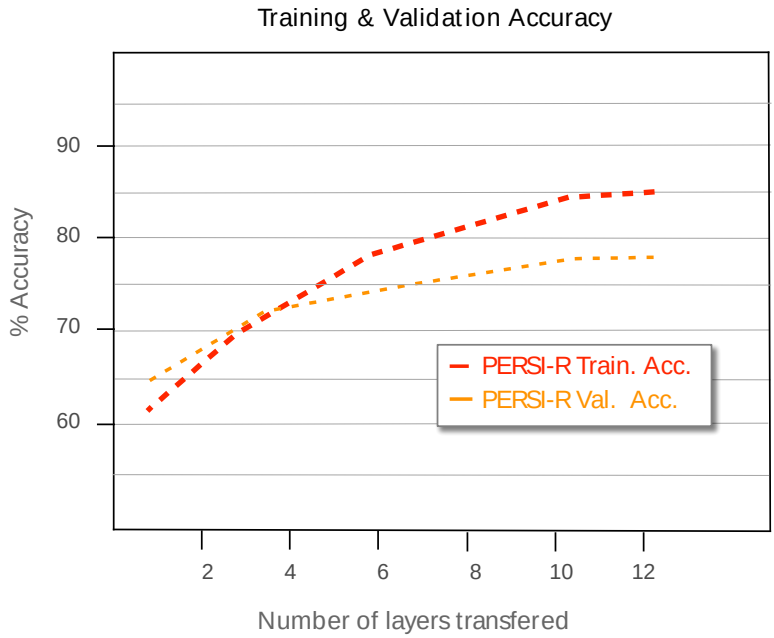

Figure 10-7. *Performance improvement during transfer-learning*

10.7. Discussion

The high accuracy of the PERSI-R model indicates that it is capable of being used as an additional module in the proposed CSE attack recognition system (Tsinganos et al. 2018). PERSI-R recognizes the persistent behavior of a malicious user by measuring the semantic similarity of the sentences uttered. Therefore, if an interlocutor tries to extract critical information by trying different approaches, her intention will be revealed by the model. Such a component is a useful add-on for a holistic system that combines several different models to recognize individual enablers of successful CSE attacks e.g., personality characteristics, persuasion and deception attempts, dialogue acts, etc.

This research concludes that pre-trained models can be used for cyber-security-oriented tasks such as the recognition of CSE attacks. The benefits are considerable:

- less development and training time compared to an RNN model approach that is trained from scratch. The model weights are pre-trained, encoding valuable information trained on extensive corpora.

- Reduced training data requirements, as the focus lies in fine-tuning the pre-trained model specifically for the targeted downstream task.
- increased accuracy on the downstream task after fine-tuning for a few epochs (e.g., 2-4).

Initially, there were instances of overestimating semantic similarity, and upon conducting further research, it was concluded that punctuation played a crucial role in many of the missed cases. Also, the lexical overlap was another reason that led the model to mistakenly classify pairs of sentences as similar when they were not. The reasons were almost analogous for the cases where PERSI-R underpredicted similarity. Thus, the similarity was difficult to recognize when there was a significant lack of lexical overlap or if completely different punctuation was used.

By conducting experiments, the use of Euclidean distance as an alternative to cosine similarity was explored, revealing slightly inferior performance metrics. This can be explained because Euclidean distance is highly effective at clustering tasks, but a smaller distance is measured if the two different vectors have no common attribute values. .

Word embedding dimensionality reduction was explored using Principal Components Analysis (PCA) and found a slight improvement in accuracy as in Huang et. al 2019). This approach may be helpful in the case of a smaller dataset; however, research is ongoing regarding making pre-trained models better few-shot learners (Gao, Fisch, and Chen 2021),(Hu et. al 2022) focusing on natural language inference tasks such as text entailment (Wang et al. 2021).

10.8. Chapter Conclusion

Chapter 10 presented the implementation and evaluation of PERSI-R for recognizing persistence in CSE attacks. The PERSI-R model leverages NLP techniques to capture the linguistic features and patterns indicative of persistence in CSE attacks. By employing neural networks, the model effectively learns and recognizes these persistence cues, enabling accurate identification and characterization of persistent behavior. The architecture of the CSE-PersistenceBERT model forms the core of the PERSI-R model, incorporating the principles and mechanisms necessary to identify and classify persistence in textual data.

During the development process, a carefully curated corpus (CSE-Persistence Corpus) of textual data containing instances of persistence in chat-based interactions was utilized to train and fine-tune the PERSI-R model. The model underwent rigorous training to understand the subtleties and nuances of persistent behavior in text. The

performance of the PERSI-R model was evaluated using various metrics, including accuracy, precision, recall, and F1 score, on a held-out test dataset. These metrics provided valuable insights into the model's effectiveness in accurately recognizing and characterizing persistence.

Through pre-training on a corpus with long sections of contiguous in-context sentences, PERSI-R acquired the necessary language knowledge related to CSE attacks. This way the model acquires the capability to process long-range linguistic and semantic dependencies which are then successfully transferred to solve NLU tasks such as persistence recognition. The presented work confirms that the generic word vector representations produced by the pre-trained models can be employed in a range of natural language processing tasks related to the cyber security domain. Moreover, the fine-tuning using the appropriately annotated CSE-Persistence Corpus generated in-context representations suitable for the paraphrase recognition task during a CSE attack. The approach taken achieved satisfactory performance results while keeping the same model size. The findings from this chapter lay the groundwork for further advancements in recognizing and mitigating persistence, ultimately bolstering the security and resilience of individuals and organizations in the face of CSE threats.

11. CSE-ARS

11.1. Introduction

In the context of this thesis a typical chat setup is described in **Figure 11-1** involving various entities and roles. This setup revolves around the interaction between the user, the social engineer, the Internet, the chat server, and CSE-ARS.

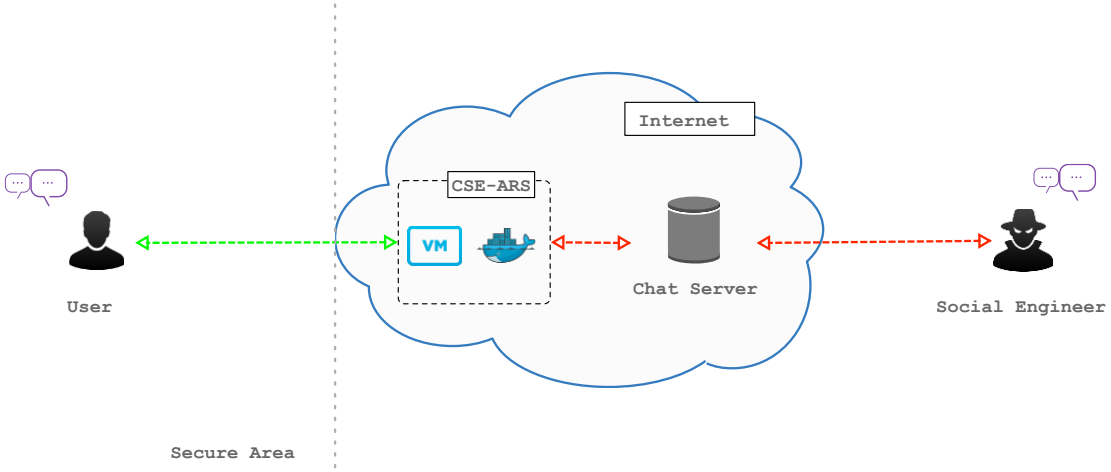

Figure 11-1. *A typical setup for chat between a user and a potential social engineer*

The entities that appear in the above figure are the following:

- *The user*: The user is an individual who actively participates in the chat setup. He uses chat software as a means of communication, engaging in conversations with other users. The user's role is crucial as he is the intended target of potential CSE attacks. He inputs his utterances or messages into the chat software, initiating (or responding to) the communication process. The user's characteristics, behavior and responses play a significant role in determining the effectiveness of the CSE attack recognition system.
- *CSE-ARS*: CSE-ARS, serving as a virtual machine (VM) or a docker container, acts as an intermediary between the user and the potential social engineer. This system is specifically designed for detecting and recognizing CSE attacks, leveraging its deep learning-based techniques and expertise. When the user enters an utterance or message into the chat software, the communication is intercepted by CSE-ARS. It performs preliminary checks and analysis on the utterance, applying various algorithms and models to identify potential CSE attacks.

- *The chat server*: The chat server serves as a central entity responsible for transmitting messages between the user and the social engineer. It also acts as an intermediary, receiving the victim's utterance from CSE-ARS and forwarding it to the social engineer. Once the social engineer formulates a response, the chat server receives the response and forwards it back to CSE-ARS for analysis. The chat server plays also a critical role in facilitating the communication process and ensuring the exchange of messages between the user and the social engineer.
- *The social engineer*: The social engineer is an individual who interacts with the user through the chat setup. Her objective is to manipulate the user into revealing sensitive information or performing malicious actions. The social engineer formulates responses based on the user's utterances, aiming to exploit vulnerabilities and achieve her malicious goals.
- *The internet* acts as the underlying infrastructure that enables the communication between the victim, the chat server, and the social engineer. It provides the connectivity and network capabilities required for the chat setup to function effectively.

The chat may begin with the victim or the potential social engineer, who initiate the conversation by entering utterances into the chat software, which serves as their means of communication. Simultaneously, the CSE-ARS system, acts as an intermediary between the two interlocutors which can detect and recognize CSE attacks. As shown in **Figure 4-1** the CSE-ARS system is able to analyze the content of the utterances utilizing deep learning algorithms and techniques and make inference regarding enablers that can lead to successful social engineering attacks.

In **Figure 11-2** CSE-ARS is depicted in action where we can see the individual enabler recognizers that are utilized to make inference regarding the state of the chat.

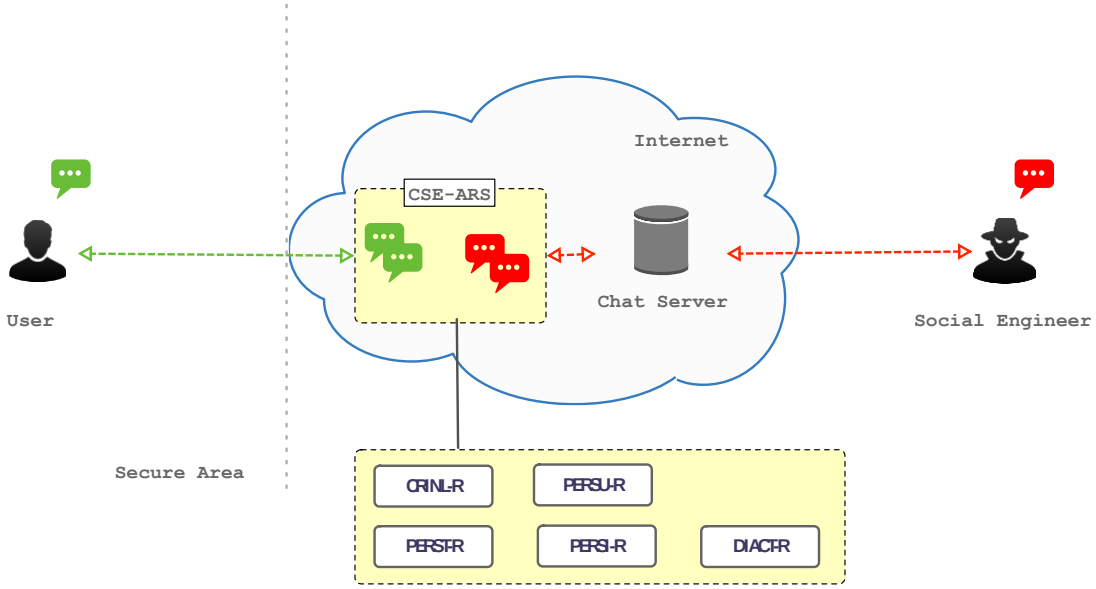

Figure 11-2. *CSE-ARS in action*

Throughout this process, CSE-ARS acts as a safeguard, aiming to protect the user by analyzing his personality characteristics and detecting critical information leakage. Furthermore, on the social engineer's side, CSE-ARS is able to recognize malicious persuasion attempts, persistent behavior and recognition of dialogue acts that can lead to unwanted actions by analyzing the full chat history and thus make inference regarding deception attempts.

11.2. Architecture

A resilient and efficient cyber-defense system was designed and implemented with the aim to predict whether a chat-based dialogue is transitioning into a social engineering attack. It decides whether an interlocutor is unleashing a CSE attack to extract critical personal, enterprise, or IT data from the other. CSE-ARS brings a multi-modal approach to solving the problem of recognizing CSE attacks. In the context of this study, multi-modal refers to the use of multiple modalities or sources of information to make predictions. The proposed system utilizes a late fusion technique, which takes as input each recognizer's output and produces a final prediction through a weighted linear aggregation. The weights were determined using a validation set and are optimized to maximize the performance of the system.

CSE-ARS, utilizes the recent trends in deep learning and natural language processing to examine and classify utterances taking into account the CSE attack enablers presented in Sections 4.2 though 4.8. The enabler recognizers were evaluated based on their performance using the same context to draw safe conclusions. This means special

attention was paid to the training data, the hyperparameters, the experimental setup, and the number of experiments conducted for the comparison to be trustworthy. In the evaluations, the emphasis was placed on prioritizing the reduction of false negatives rather than false positives. The objective was to minimize instances where potential CSE attacks go undetected, even if they resulted in a slightly higher number of false positives. This trade-off was a basic design decision for all enabler recognizers that were tested and evaluated.

This approach allowed to exploit the strengths of the different enabler recognizers and to account for the fact that different enablers may be utilized to successfully conduct a CSE attack. More analytically, CSE-ARS incorporates the following enabler recognizers:

- Critical Information Leverage Recognizer (CRINL-R): a named-entity recognizer (NER), based on a bi-directional long short-term memory (bi-LSTM) model(Schuster and Paliwal 1997), recognizing critical data information.
- Personality Recognizer (PERST-R): a BERT (Devlin et al., 2019) model that predicts personality traits of the interlocutors based on the Big-5 theory.
- Dialogue-act Recognizer (DIACT-R): a BERT model that recognizes dialogue acts that can lead to deception taking into account the whole dialogue history.
- Persuasion Recognizer (PERSU-R): a convolutional neural network (CNN) (Lecun et al. 1998) that predicts persuasion attempts.
- Persistence Recognizer (PERSI-R): a BERT model that predicts persistent behavior by identifying paraphrasing in chat dialogue.

After the individual recognizers generate their outputs, the late fusion model combines them to determine whether the chat text constitutes a CSE attack. Each recognizer has its feature extraction network customized for its specific objective and state. The outputs of the recognizers are fused using a weighted linear aggregation method. All the recognizers, including the CSE-ARS but excluding PERST-R, are trained on different variants of the custom CSE Corpus (Tsinganos and Mavridis 2021)

The multimodal fusion architecture is depicted in **Figure 11-3**. The system's design is flexible, allowing for the addition or removal of individual recognizers as needed. This allows the system to be adapted to new types of CSE attack enablers as they emerge. By using a combination of different recognizers, the system can capture a wide range of enablers and provide a comprehensive assessment of the likelihood of a particular input being a CSE attack.

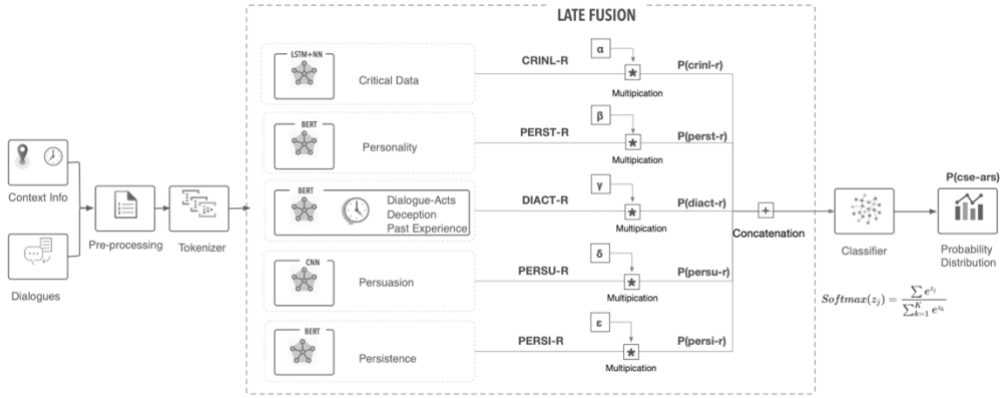

***Figure 11-3**. The CSE-ARS system architecture*

In this work, concatenation was used for the fusion layer to combine the outputs of multiple binary classifiers denoting the independence of the different enablers and keeping the system simple. The concatenation layer takes the outputs of each classifier and concatenates them into a single vector, which can then be fed into a fully connected layer for the final prediction. The outputs of the classifiers are treated as separate features and the information from each classifier is retained in the final combined vector. This is useful when the classifiers are designed to capture different aspects of the input and the outputs are independent of each other. In this way, the final combined vector provides a multi-modal representation of the input and can be used to make a prediction.

11.3. Corpus

CSE-ARS was trained on the CSE-ARS Corpus, which is an augmented version of the CSE Corpus (Tsinganos and Mavridis, 2021) composed of realized and fictional CSE attack dialogues. **Table 11-1** presents the details of the CSE-ARS Corpus.

***Table 11-1**. CSE-ARS Corpus*

Characteristic	Value
Corpus Name	CSE-ARS corpus
Collection Method	Web-scraping, pattern-matching text extraction
Corpus size	(N) 56 text dialogues/3380 sentences
Vocabulary size	(V) 5400 terms
Total no. of turns	9685
Avg. tokens per turn	9,34
Content	Chat-based dialogues
Collection date	Jun 2018 – Dec 2020

Language	English
Release year	Feb 2023
License	Private

11.4. Training

To comprehensively assess the effectiveness of the proposed CSE-ARS, a ten-fold cross-validation experiment was conducted. CSE-ARS Corpus underwent a randomization process, resulting in its division into ten distinct subsets. Throughout each iteration of the experiment, nine subsets were further subdivided into training (80%) and validation (20%) sets, while the remaining subset was designated as the testing set. By employing this methodology, prediction scores for each testing subset were obtained after ten rounds, which were subsequently amalgamated to derive an overall prediction score. To enhance the performance of this approach, several strategies were implemented. At the outset, diverse configurations were explored, and each recognizer was trained using a single domain training set. The validation set was employed to mitigate the risk of overfitting during the training process. Subsequently, the optimal configuration parameters were selected based on the evaluation criterion of the area under the receiver operating characteristic curve (AUC) value. Finally, after the training of the specific recognizers, an exhaustive examination of various coefficient combinations was conducted until the classification performance, as assessed by the AUC value, reached its maximum on the validation set.

11.4.1. Late Fusion

Ensemble techniques offer a valuable means of enhancing system performance by combining multiple machine learning models. Such techniques prove particularly beneficial when individual models exhibit high accuracy but differ in the types of errors they make, as often encountered in multimodal approaches. Within the context of this study, as "modality"(Lahat, Adali, and Jutten 2015) is considered each distinct acquisition framework that captures information regarding the same phenomenon from various types of recognizers, under different conditions, across multiple experiments or subjects, among other factors. The late fusion approach (Lahat, Adali, and Jutten 2015), akin to decision-level fusion, constitutes an ensemble technique that combines multiple models to generate a final prediction. In late fusion, the unimodal decision values are acquired and merged to derive the ultimate decision. This approach facilitates flexible training and straightforward predictions, even when one or more modalities are absent, albeit at the expense of disregarding certain low-level interactions between modalities.

The output of enabler recognizers is concatenated and subsequently fed into a final classifier to formulate the conclusive prediction. Leveraging late fusion methodology leads to enhanced performance when compared to utilizing a solitary model, rendering it a commonly adopted technique across various machine learning domains, including image classification, natural language processing, and speech recognition. The key advantage of late fusion lies in its capacity to allow individual recognizers to specialize in their respective areas of expertise, thereby contributing to a more accurate final prediction. If a recognizer m_i is used on modality i using input k_i where $\mathrm{i} = 1,2,\ldots,\mathrm{M}$ then the final prediction of a late fusion system is given by $\mathrm{p} = \mathrm{f}(m_1(k_1), m_2(k_2), \ldots, m_M(k_M))$. The strengths of late fusion systems are relatively simple to implement compared to other models, as they simply combine the outputs of different models into a single prediction.

11.4.2. Simulated Annealing

The novelty of this approach resides in the architectural design and the fusion of multi-dimensional data. The fundamental idea behind the multimodal fusion approach is to integrate the distinct output to enhance CSE attack recognition. The outputs of the individual enabler recognizers are represented by probability distributions, obtained through the application of the SoftMax classifier. The SoftMax classifier formula is as follows:

$$softmax(y)_i = \frac{e^{y_i}}{\sum_{j=1}^{n} e^{y_i}} \qquad (11.1)$$

where y_i represent the data in j_{th} column of the output vector and n represents the output vector dimension. The SoftMax layer's output constitutes a probability distribution across the two possible classes, whether an utterance is considered a CSE attack or not. Multimodal fusion was performed using weighted linear aggregation as the fusion technique; the specific construction steps are as follows:

- STEP1: The trained enabler recognizers (CRINL-R, DIACT-R, PERSI-R, PERST-R, and PERSU-R) are applied to the particular validation sets. The output (prediction results) of each recognizer ($i.e., output_{CRINL-R}$, $output_{DIACT-R}, output_{PERSI-R}$, $output_{PERST-R}$, and $output_{PERSU-R}$) has the form of a matrix with dimensions equal to the number of samples in the corresponding validation set and the number of classes.
- STEP2: To fuse the features of the individual recognizers, the following fusion methods are defined:

$$\begin{cases} output_{CSE-ARS} = \alpha * output_{CRINL-R} \\ +\beta * output_{DIACT-R} + \gamma * output_{PERSI-R} \\ +\delta * output_{PERST-R} + \varepsilon * output_{PERSU-R} \\ \\ \alpha + \beta + \gamma + \delta + \varepsilon = 1 \\ \\ 0 \le \alpha \le 1, 0 \le \beta \le 1, 0 \le \beta \le 1, 0 \le \beta \le 1, 0 \le \varepsilon \le 1 \end{cases} \tag{11.2}$$

where $output_{CSE-ARS}$ represent the result of feature fusion, α represents the weight of the CRINL-R output, β represents the weight of the DIACT-R output, γ represents the weight of the PERSI-R output, δ represents the weight of the PERST-R output and ε represents the weight of the PERSU-R output.

- STEP3: the best combination of $(\alpha, \beta, \gamma, \delta, \varepsilon)$ on the validation set is found so that the cross entropy between $output_{CSE-ARS}$ and label (one-hot encoding) is close to the theoretical minimum $(\alpha, \beta, \gamma, \delta, \varepsilon)$. The step is equivalent to a new round of feature learning.
- STEP4: In order to find the optimal solution of $(\alpha, \beta, \gamma, \delta, \varepsilon)$ on the validation set, the following optimization problem is defined:

$$\min\ (Loss(output_{CSE-ARS}, Label) \tag{11.3}$$

$$\begin{cases} \alpha + \beta + \gamma + \delta + \varepsilon = 1 \\ 0 \le \alpha \le 1, 0 \le \beta \le 1, 0 \le \beta \le 1, 0 \le \beta \le 1, 0 \le \varepsilon \le 1 \end{cases} \tag{11.4}$$

To obtain the global optimal solution, the Simulated Annealing (SA) algorithm (Rere, Fanany, and Arymurthy 2015) is utilized . SA is a metaheuristic optimization algorithm used for discovering the global optimum of a complex objective function. The probability of a particular state of x is determined by the following equation:

$$p(x) = e^{\frac{-\Delta f(x)}{kT}} \tag{11.5}$$

where f(x) is the configuration of energy, k is Boltzmann's constant, and T is temperature. The algorithm (Rere, Fanany, and Arymurthy 2015) that describes the optimization procedure is shown in **Table 11-2**

Table 11-2. *Simulated Annealing algorithm*

Algorithm: Simulated Annealing
Generate a random initial solution x_0
Calculate objective function
Parameter initialization (T, k, c)
while control condition not true **do**
for number of new states
Pick new solution $x_0 + \Delta x$ in neighborhood
# Evaluate new state
If $f(x_0 + \Delta x) > f(x_0)$ **then**
$f_{new} = f(x_0 +); x_0 = x_0 + \Delta x$
else
$\Delta f = f(x_0 + \Delta x) - f(x_0)$
$random\, r(0,1)$
$\boldsymbol{if}\ r > \exp\left(\frac{-\Delta f(x)}{kT}\right) \boldsymbol{then}$
$f_{new} = f(x_0 + \Delta x), x_0 = x_0 + \Delta x$
else
$f_{new} = f(x_0),$
end if
end if
$f = f_{new}$
Decrease the temperature periodically: T = c x T
end for
end while

To summarize, the Simulated Annealing (SA) algorithm starts with an initial solution and evaluates it using the objective function. Then, it perturbs the solution and evaluates the new solution. If the new solution is better, it becomes the new current solution. If it's worse, it may still be accepted with a probability based on the temperature parameter.

The objective function in this work is defined as shown in **formula 11.4**, and the SA algorithm is used to determine the optimal values of (α, β, γ, δ, ε) on the validation set. Finally, the tuple (α, β, γ, δ, ε) is transferred to the test set to predict CSE attacks using the fused features, and the results are compared to those obtained through single modality feature learning. The workflow of CSE-ARS' multimodal fusion is shown in **Figure 11-4**.

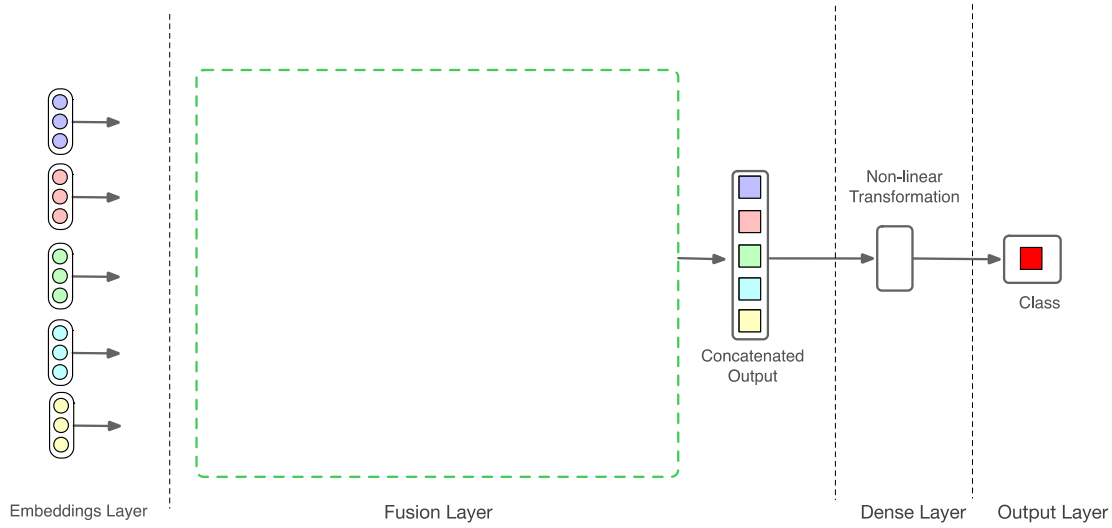

Figure 11-4. *Workflow of the CSE-ARS multimodal fusion*

The layers depicted in the figure above are as follows:

- Embeddings Layer: This layer receives input from the individual recognizers.
- Fusion Layer: This layer combines the outputs from the individual recognizers into a single representation. The fusion layer is implemented as a weighted linear aggregation.
- Dense Layer: This layer is used to apply non-linear transformations to the fused representation to produce the final prediction. The dense layer is implemented as a fully connected layer.
- Output Layer: This layer produces the final prediction, which is the probability of a given text being a CSE attack or not.

These steps were repeated during training until the system reached satisfactory performance. For this specific problem of CSE attack recognition, a binary cross-entropy loss function (Kullback and Leibler 1951) is used. For the optimizer, the method of choice was the Adam optimizer (Kingma and Ba 2014), which adaptively adjusts the learning rate for each parameter based on the historical gradient information. Furthermore, it tends to converge faster and with a better convergence minimum compared to SGD (Ruder 2016). In summary, the choice of a binary cross-entropy loss function and the Adam optimizer is a reasonable one for the late fusion system for CSE attack recognition, as it provides a robust and efficient way to measure the error and update the system parameters.

11.5. Evaluation Results

CSE-ARS has been extensively evaluated on the CSE-ARS corpus. Due to the absence of comparable CSE recognition systems, a baseline system was utilized as a point of reference employing majority voting (Breiman 1996; Dietterich 2000) a method, in which the predictions of several classifiers are combined and the class label with the highest frequency is selected as the final prediction. The objective of comparing the results of the two models is to highlight the advantages of employing a more sophisticated late fusion model over a simplistic majority voting ensemble model for CSE attack recognition. The results show that CSE-ARS is outperforming the majority voting ensemble model which is used as a baseline system.

For each distinct recognizer, the assumption is made that if the recognition is true (e.g., persuasion is correctly recognized), it indicates the presence of a CSE attack. Consequently, there are two distinct classes that need to be predicted: CSE-attack and Neutral. The predicted class label $\hat{y}$, if there are m different classifiers, is given by $\hat{y} = mode\{C_1(x), C_2(x), \dots, C_m(x)\}$. The performance of each recognizer was evaluated through 10-fold cross-validation on the training dataset before combining them in to CSE-ARS.

The evaluation process provided insights into the individual performance of each recognizer before their integration into CSE-ARS e.g., the global optimization outcome for (α, β, γ, δ, ε) on the validation set yielded (0.134, 0.294, 0.201, 0.169, 0.202). Upon acquiring the key parameters (α, β, γ, δ, ε) for the CSE-ARS, both the individual recognizers and the fusion model were tested on the test set. Their respective prediction accuracies were 56,70%, 71,20%, 68,70%, 64,90%, 58,90%, and 79,96%. CSE-ARS achieved a prediction accuracy of 8.76% greater than the optimal single modality model. The loss value of each model was calculated as shown in the following formula

$$Loss(L_t, L_t^*) = -\frac{1}{n}\sum [L_t(i) * logL_t^*(i)] + \lambda R(w) \tag{11.6}$$

where L_t is the correct label of the sample, L_t^* is the network output, and λ is the weight of the regularization term.

The loss value for each recognizer was 0.304, 0.250, 0.378, 0.401, 0.290 and the multimodal fusion model was only 0.17954. This rigorous evaluation method provided a robust assessment of the performance of each recognizer and CSE-ARS. The results of the 10-fold cross-validation are shown in **Figure 11-5**.

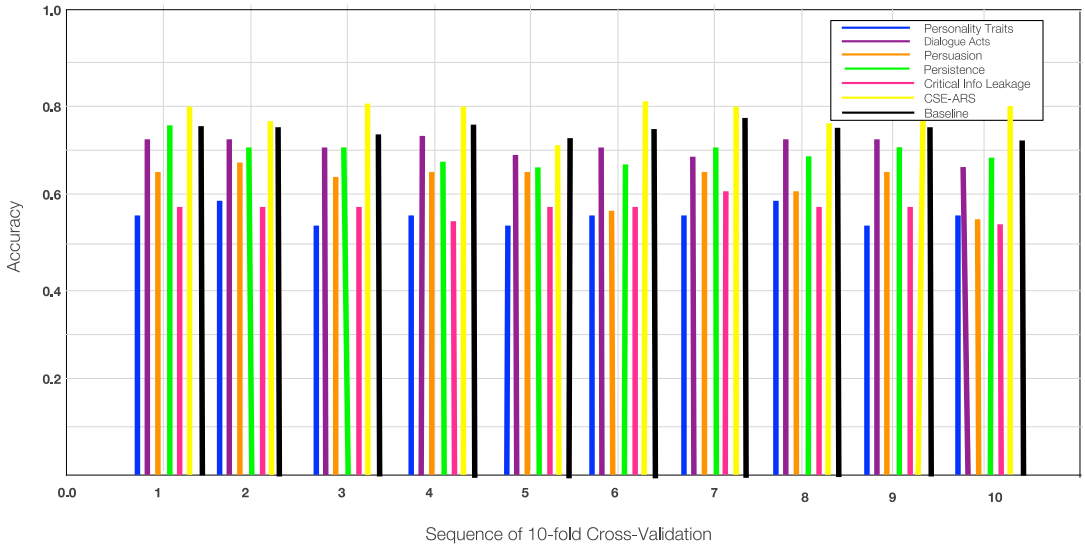

Figure 11-5. *– Sequence of 10-fold Cross-Validation*

Each set represents a 10-fold cross-validation. In the 10 times of validations, the average prediction accuracy, prediction, recall and F1 score of the enablers, majority voting model and CSE-ARS are shown in **Table 11-3**.

Table 11-3. *Average Accuracy of individual recognizers and CSE-ARS multimodal fusion in 10-fold cross-validation*

Model	Avg. Accuracy	Avg. Precision	Avg. Recall	Avg. F1-score
PERST-R	0.567	0.512	0.501	0.560
DIACT-R	0.712	0.648	0.633	0.604
PERSU-R	0.687	0.601	0.600	0.653
PERSI-R	0.749	0.667	0.640	0.630
CRINL-R	0.589	0.554	0.550	0.601
Majority Voting	0.753	0.700	0.689	0.701
CSE-ARS	**0,799**	0.702	0.702	0.701

To assess the performance of the CSE-ARS system in predicting CSE attacks, ROC curves were computed for each class, and the corresponding AUC values were determined for each model. This analysis provided insights into the discriminative power and overall performance of the system in detecting CSE attacks. **Figure 11-6** shows the ROC curves of the models used to evaluate the prediction performance of CSE attacks, and **Table 11-4** displays the calculated AUC values for each model.

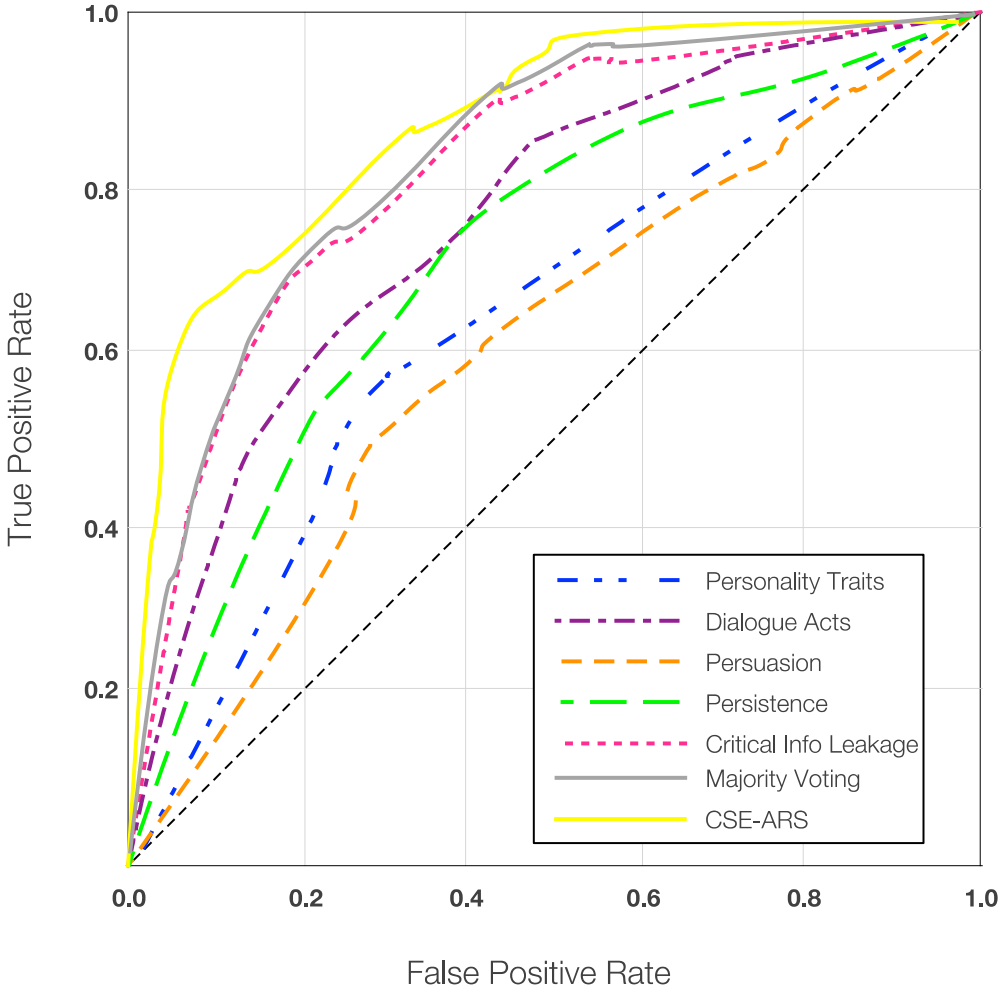

Figure 11-6. *ROC curves of CSE-ARS, Majority Voting model and recognizers*

Table 11-4. *AUC values CSE-ARS, Majority Voting model and recognizers*

Model	AUC
PERST-R	0.5479 (+/- 0.14)
DIACT-R	0.6783 (+/- 0.12)
PERSU-R	0.7010 (+/- 0.11)
PERSI-R	0.6971 (+/- 0.14)
CRINL-R	0.5783 (+/- 0.9)
Majority Voting	0.649 (+/- 0.12)
CSE-ARS	0.7432 (+/- 0.10)

11.6. Chapter Conclusion

In today's digital world, CSE attacks have emerged as a significant threat, capable of extracting sensitive information or manipulating individuals into performing certain actions. By exploiting human personality traits and employing persuasive dialogue acts and deception techniques, attackers can effectively deceive their victims.

In this chapter, CSE-ARS was presented, showcasing its efficacy in the recognition and mitigation of CSE attacks. CSE-ARS integrates multiple recognizers, including CRINL-R, PERST-R, DIACT-R, PERSU-R, and PERSI-R, to capture various aspects of CSE attacks. Through the late fusion technique it combines the outputs of these individual recognizers, allowing for a holistic analysis of multimodal information, such as critical information leakage, personality traits, dialogue acts, persuasion techniques, and persistence. The Simulated Annealing algorithm was employed to optimize the fusion process, ensuring an optimal combination of the individual model outputs.

By leveraging transfer learning, CSE-ARS effectively utilized pre-existing pre-trained models, reducing the need for extensive training data while enhancing system efficiency.

To train the CSE-ARS model, a carefully constructed corpus comprising labeled instances of CSE attacks was utilized. The model underwent an extensive training process, fine-tuning each individual deep learning model and optimizing the late fusion mechanism.

In order to assess the effectiveness of CSE-ARS, experiments were conducted on custom corpora (CSE-ARS Corpus) that consists of both real-world and fictional chat texts. This comprehensive evaluation allowed for the examination of the system's performance across different types of chat-based scenarios, providing insights into its capabilities and potential real-world applicability. The comprehensive evaluation conducted in this study substantiates the effectiveness and reliability of CSE-ARS in accurately recognizing CSE attacks. Thus, CSE-ARS renders a valuable tool for safeguarding individuals and organizations against such threats. The results indicate that the proposed approach exhibits satisfactory performance in recognizing CSE attacks.

12. CONCLUSIONS AND FUTURE WORK

12.1. Conclusions

Deep learning is a powerful approach for building recognizers as it allows for the learning of high-level abstractions from data. This is particularly useful in the case of CSE attacks, where the patterns of behavior and language used by attackers can be complex and difficult to identify. Deep learning models such as CNNs, LSTMs, and transformer-based ones such as BERT can learn these complex patterns and make accurate predictions about the likelihood of a CSE attack. Furthermore, the use of pre-trained deep learning models is beneficial as these models are already trained on large amounts of data, which enables them to generalize well to new data and new types of CSE attacks. This is important as social engineering attacks are constantly evolving, and being able to adapt to new types of CSE attacks is crucial for the effectiveness of a CSE attack recognition system.

For the implementation of the individual deep learning recognizers, the popular deep learning frameworks PyTorch (Paszke et al. 2019) and HuggingFace (Wolf et al. 2020) were mainly used . Every experiment was tracked as it relates to the progress and the results. The level of detail captured during each experiment was meticulously maintained to ensure that the experiments could be recreated accurately or compared with other experiments in the future. This comprehensive logging approach allows for reproducibility and facilitates effective comparisons between different experiments for experiment tracking (e.g., loss curve, model performance metrics, hyperparameters, etc.) and experiment versioning, and for this reason the Weight & Biases platform (Biewald 2020) was utilized. Different combinations of hyper-parameter values gave different performance results. A thorough and exhaustive search was conducted, considering all possible combinations, to identify the optimal values for hyperparameters. This approach ensured that no potential combination was overlooked and allowed for the selection of the most suitable hyperparameter values for the given task. The performance of each hyper-parameters set was evaluated against a dedicated validation set. The test split was never used for hyper-parameter tuning, to avoid overfitting and the best model in each case was selected based on its performance on the validation set. Afterward, each model's performance was measured against the test split.

Many trade-offs had to be considered during model selection such as computing requirements, and performance. As already mentioned, the design decision was to prefer models that had fewer false positives neglecting the false negatives metric. Furthermore, it was also a design decision to choose the models based solely on their performance and not on computing requirements. Neural networks demand more

powerful machines (e.g., GPU over CPU) to deliver high accuracy with an acceptable inference latency. It was crucial to maintain a comprehensive record of definitions required to replicate an experiment alongside its pertinent artifacts, which refer to the files created during an experiment such as those displaying loss curves, evaluation loss graphs, logs, or intermediate results of a model throughout a training process. This practice facilitates the comparison of distinct experiments and aids in the selection of the optimal experiment tailored to one's specific requirements.

CSE-ARS system has been extensively evaluated on the CSE-ARS corpus and outperformed the individual recognizers in terms of accuracy, precision, recall, F1 score, and AUC value. PERSI-R had the highest performance among the individual models, but CSE-ARS achieved a higher accuracy in 10-fold cross-validation. However, there was a large gap between the AUC value of CSE-ARS and PERSU-R. It is suggested that the superior performance of CSE-ARS is due to the strength of the late fusion approach to utilize multiple modalities, which may capture more comprehensive information about CSE attacks. Overall, the results suggest that the multimodal fusion approach has the potential to improve the accuracy of CSE attack prediction. The proposed system CSE-ARS has shown promising results in recognizing the various enablers such as personality traits, dialogue acts, persuasion attempts, persistent behavior, and critical information leverage.

One of the main contributions of this study is the use of the late fusion method to combine the separate outputs of the individual recognizer. This approach allows for the strengths of each technique to be leveraged and results in a more robust and accurate recognition system. The individual recognizers used in the proposed system also deserve further discussion. The use of a convolutional network for recognizing persuasion attempts is effective in identifying patterns in the language used in persuasion attempts. The use of BERT for recognizing personality traits, dialogue acts, and persistent behavior is also well-suited for these tasks as BERT is a pre-trained language model that has been shown to have strong performance in natural language understanding tasks. It is worth noting that the BERT models used in this study were fine-tuned on specific and appropriate corpora, and their performance may be improved by further fine-tuning on larger and more diverse corpora. Moreover, the use of such models allows for the proposed system to be easily updated and improved as new data and techniques become available, making it more adaptive to the constantly evolving social engineering attacks. Overall, the individual recognizers used in CSE-ARS are effective in recognizing the various enablers of social engineering attacks. The results obtained from the experiments and evaluations showcase the potential of deep learning models to recognize critical information leakage, personality traits, dialogue acts, persuasion techniques, and persistence in chat-based interactions. These models offer

valuable insights into the identification and analysis of social engineering behavior, facilitating proactive defense and enhancing cybersecurity measures. The developed CSE-ARS system, incorporating the individual models, provides a holistic approach to CSE attack recognition, equipping organizations and individuals with an effective defense mechanism.

This study emphasizes the efficiency and efficacy of the proposed system as a valuable tool in safeguarding individuals and organizations from the insidious threats of social engineering attacks. By raising awareness of the vulnerabilities inherent in human interactions and developing advanced recognition systems, we can empower individuals and organizations to counteract these attacks effectively. The implications of this research extend beyond the field of computer science, reaching into the realm of computer security and the preservation of personal and organizational integrity in the digital age.

Finally, it is important to emphasize the need for ethical considerations in the development of CSE attack recognition systems. As these systems involve the processing of sensitive and personal information, it is crucial to ensure that they are designed and used in a way that respects privacy and protects against potential biases and discrimination. This requires a careful and transparent design process, as well as ongoing monitoring and evaluation of the system's performance and impact. Overall, a multi-disciplinary and ethical approach is essential for the development of effective and responsible CSE attack recognition systems.

In conclusion, this thesis contributes to the body of knowledge in the field of CSE attack recognition. The developed deep learning models and CSE-ARS lay the foundation for effective defense strategies against social engineering threats. By combining the power of deep learning, natural language processing, and cybersecurity, we can enhance the security posture of individuals and organizations, safeguarding critical information and mitigating the risks posed by CSE attacks. With continued research and collaboration, we can foster a safer digital environment and protect users from the detrimental impacts of social engineering.

12.2. Future Work

While the current research has made significant contributions to the field of CSE attack recognition, several avenues for future work can further enhance the effectiveness and applicability of the developed models and system. These areas of future research include:

1. **Enhancing Model Generalization:** The developed deep learning models, including CRINL-R, PERST-R, DIACT-R, PERSU-R, PERSI-R, and CSE-ARS, have shown promising performance on the existing corpora. However, it is crucial to evaluate their generalization capabilities in diverse and real-world scenarios. Future work can involve collecting larger and more diverse datasets, encompassing a wider range of social engineering techniques and attack vectors. This will allow for a more comprehensive evaluation of the models' performance and their ability to generalize to real-world situations.
2. **Incremental Model Updates:** Social engineering techniques and attack vectors evolve over time, requiring continuous updates and improvements to the recognition models. Future research can explore techniques to enable the models to adapt and learn from new types of attacks and emerging trends in social engineering. This can be achieved through incremental learning approaches that allow the models to update their knowledge and adapt to evolving attack patterns without requiring a complete retraining process.
3. **User Awareness and Education:** While the developed system focuses on CSE attack recognition, raising user awareness and providing education on social engineering risks remain essential. Future work can involve the development of educational materials, training programs, or interactive tools that help users understand CSE attack techniques and learn how to identify and respond to suspicious activities. Integrating user awareness and education initiatives with the recognition models and system can create a comprehensive defense strategy against CSE attacks.
4. **Beyond Text:** The development of large language models (LLMs) has opened up new possibilities for the detection of deepfakes. One challenge in using LLMs for deepfake detection is that deepfakes are becoming increasingly sophisticated. However, the capabilities of generative models are still under research and the first results are highly promising.
5. **Interpretability**: It may be beneficial to investigate the interpretability of individual models and CSE-ARS to identify the specific features that contribute to the final prediction.

The future work outlined above presents several directions for extending the current research and enhancing the capabilities of the individual models and CSE-ARS for CSE attack recognition. By addressing these areas, researchers can contribute to the advancement of the field and provide more effective solutions to combat social engineering threats. The combination of improved model generalization, incremental updates, user awareness and beyond text initiatives will further strengthen the defense

against CSE attacks and contribute to the overall cybersecurity landscape. Additionally, collaboration between academia, industry, and cybersecurity professionals is essential to foster the exchange of knowledge and expertise, enabling the implementation of robust defense mechanisms against social engineering attacks.

PUBLICATIONS

International Journals

I. E. K. Markakis et al., 'The FORTIKA Accelerated Edge Solution for Automating SMEs Security', in Challenges in Cybersecurity and Privacy-the European Research Landscape, River Publishers, 2022, pp. 77–101.

II. Tsinganos Nikolaos, Panagiotis Fouliras, and Ioannis Mavridis. 2022. 'Applying BERT for Early-Stage Recognition of Persistence in Chat-Based Social Engineering Attacks'. Applied Sciences 12(23):12353. doi: 10.3390/app122312353.

III. Tsinganos Nikolaos, Panagiotis Fouliras, and Ioannis Mavridis. 2023. 'Leveraging Dialogue State Tracking for Zero-Shot Chat-Based Social Engineering Attack Recognition'. Applied Sciences 13(8):5110. doi: 10.3390/app13085110.

IV. Tsinganos Nikolaos, and Ioannis Mavridis. 2021. 'Building and Evaluating an Annotated Corpus for Automated Recognition of Chat-Based Social Engineering Attacks'. Applied Sciences 11(22). doi: 10/gnpb3z.

V. Tsinganos Nikolaos, Ioannis Mavridis, and Dimitris Gritzalis. 2022. 'Utilizing Convolutional Neural Networks and Word Embeddings for Early-Stage Recognition of Persuasion in Chat-Based Social Engineering Attacks'. IEEE Access 10:108517–29. doi: 10.1109/ACCESS.2022.3213681.

VI. Tsinganos Nikolaos, Panagiotis Fouliras, Ioannis Mavridis, and Dimitris Gritzalis. 2023. 'CSE-ARS: Deep learning-based late fusion of multimodal information for chat-based social engineering attack recognition'. (to be submitted)

International Conferences

VII. Tsinganos, Nikolaos, Georgios Sakellariou, Panagiotis Fouliras, and Ioannis Mavridis. 2018. 'Towards an Automated Recognition System for Chat-Based Social Engineering Attacks in Enterprise Environments'. P. 10 in.International Conferences

REFERENCES

'Transformers'. Accessed 8 April 2022. https://huggingface.co/docs/transformers/index.

Adamopoulou, Eleni, and Lefteris Moussiades. 2020. 'An Overview of Chatbot Technology'. In *Artificial Intelligence Applications and Innovations*, edited by Ilias Maglogiannis, Lazaros Iliadis, and Elias Pimenidis, 584:373–83. IFIP Advances in Information and Communication Technology. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-49186-4_31.

Agarap, Abien Fred. 2018. 'Deep Learning Using Rectified Linear Units (Relu)'. *arXiv Preprint arXiv:1803.08375*.

Agirre, Eneko, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. 'SemEval-2012 Task 6: A Pilot on Semantic Textual Similarity'. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, 385–93. Montréal, Canada: Association for Computational Linguistics. https://aclanthology.org/S12-1051.

Albladi, Samar Muslah, and George R. S. Weir. 2017. 'Personality Traits and Cyber-Attack Victimisation: Multiple Mediation Analysis'. *2017 Internet of Things Business Models, Users, and Networks*, November, 1–6. https://doi.org/10.1109/CTTE.2017.8260932.

An, Guozhen. 2015. 'Literature Review for Deception Detection'. *Dr. Diss. City Univ. New York*.

Anawar, S., Durga L. Kunasegaran, M. Z. Mas'ud, and N. A. Zakaria. 2019. 'ANALYSIS OF PHISHING SUSCEPTIBILITY IN A WORKPLACE: A BIG-FIVE PERSONALITY PERSPECTIVES'. In . https://www.semanticscholar.org/paper/ANALYSIS-OF-PHISHING-SUSCEPTIBILITY-IN-A-WORKPLACE%3A-Anawar-Kunasegaran/009115d2744cd5b195496d3704d6178388f8fbfd.

'Apache OpenNLP'. Accessed 23 June 2023. https://opennlp.apache.org/.

Austin, John Langshaw. 1975. *How to Do Things with Words*. Vol. 88. Oxford university press.

Bach, Kent, and Robert M. Harnish. 1979. *Linguistic Communication and Speech Acts*. Cambridge, MA: MIT Press.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. 2014. 'Neural Machine Translation by Jointly Learning to Align and Translate'. *arXiv Preprint arXiv:1409.0473*.

Bernard, Pierre. 2012. *COBIT® 5 - A Management Guide*. Van Haren.

Bezuidenhout, Monique, Francois Mouton, and H. S. Venter. 2010. 'Social Engineering Attack Detection Model: SEADM'. In *2010 Information Security for South Africa*, 1–8. Johannesburg, South Africa: IEEE. https://doi.org/10/czz4kx.

Bhakta, R., and I. G. Harris. 2015. 'Semantic Analysis of Dialogs to Detect Social Engineering Attacks'. In *Proceedings of the 2015 IEEE 9th International Conference on Semantic Computing (IEEE ICSC 2015)*, 424–27. https://doi.org/10/gfvb87.

Biewald, Lukas. 2020. 'Experiment Tracking with Weights and Biases'. https://www.wandb.com/.

Bobrow, Daniel G, Ronald M Kaplan, Martin Kay, Donald A Norman, Henry Thompson, and Terry Winograd. 'GUS, A Frame-Driven Dia|og System', 19.

Bommasani, Rishi, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, et al. 2021. 'On the Opportunities and Risks of Foundation Models'. *arXiv:2108.07258 [Cs]*, August. http://arxiv.org/abs/2108.07258.

Bowman, Samuel R., Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. 'A Large Annotated Corpus for Learning Natural Language Inference'. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Breiman, Leo. 1996. 'Bagging Predictors'. *Machine Learning* 24 (2): 123–40. https://doi.org/10.1007/BF00058655.

Bridle, John. 1989. 'Training Stochastic Model Recognition Algorithms as Networks Can Lead to Maximum Mutual Information Estimation of Parameters'. In *Advances in Neural Information Processing Systems*. Vol. 2. Morgan-Kaufmann. https://proceedings.neurips.cc/paper/1989/hash/0336dcbab05b9d5ad24f4333c7658a0e-Abstract.html.

Budzianowski, Paweł, and Ivan Vulić. 2019. 'Hello, It's GPT-2--How Can I Help You? Towards the Use of Pretrained Language Models for Task-Oriented Dialogue Systems'. In *arXiv Preprint arXiv:1907.05774*.

Cacioppo, J., S. Cacioppo, and R. Petty. 2018. 'The Neuroscience of Persuasion: A Review with an Emphasis on Issues and Opportunities'. *Social Neuroscience*. https://doi.org/10/ggqg25.

Catak, Ferhat Ozgur, Kevser Sahinbas, and Volkan Dörtkardeş. 2021. 'Malicious URL Detection Using Machine Learning'. Chapter. Artificial Intelligence Paradigms for Smart Cyber-Physical Systems. IGI Global. 2021. https://doi.org/10.4018/978-1-7998-5101-1.ch008.

Cawood, J. 2021. 'A Limited Literature Review of Influence and Persuasion'. In . https://doi.org/10/gnrtt9.

Chakravarty, Saurabh. 2019. 'Dialog Acts Classification for Question-Answer Corpora', 10.

Chandrasekaran, Dhivya, and Vijay Mago. 2022. 'Evolution of Semantic Similarity -- A Survey'. *ACM Computing Surveys* 54 (2): 1–37. https://doi.org/10.1145/3440755.

Church, Kenneth Ward, Zeyu Chen, and Yanjun Ma. 2021. 'Emerging Trends: A Gentle Introduction to Fine-Tuning'. *Natural Language Engineering* 27 (6): 763–78. https://doi.org/10.1017/S1351324921000322.

Cialdini, Robert B. 2021. *Influence, New and Expanded: The Psychology of Persuasion*. HarperCollins.

Cialdini, Robert B., and Noah J. Goldstein. 2002. 'The Science and Practice of Persuasion'. *Cornell Hotel and Restaurant Administration Quarterly* 43 (2): 40–50.

Clark, Alexander, Chris Fox, and Shalom Lappin. 2013. *The Handbook of Computational Linguistics and Natural Language Processing*. John Wiley & Sons.

Collobert, Ronan, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 'Natural Language Processing (Almost) from Scratch'. *NATURAL LANGUAGE PROCESSING*, 45.

‘Computational Pragmatics | The Switchboard Dialog Act Corpus’. Accessed 30 October 2022. https://compprag.christopherpotts.net/swda.html.

Conneau, Alexis, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2016. ‘Very Deep Convolutional Networks for Text Classification’. *arXiv Preprint arXiv:1606.01781*.

Craig, Robert T. 1981. ‘Generalization of Scott’s Index of Intercoder Agreement’. *Public Opinion Quarterly* 45 (2): 260–64. https://doi.org/10/fhstfb.

Dagan, Ido, Oren Glickman, and Bernardo Magnini. 2006. ‘The PASCAL Recognising Textual Entailment Challenge’. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, edited by Joaquin Quiñonero-Candela, Ido Dagan, Bernardo Magnini, and Florence d’Alché-Buc, 3944:177–90. Berlin, Heidelberg: Springer Berlin Heidelberg. http://link.springer.com/10.1007/11736790_9.

Dalton, Adam, Ehsan Aghaei, Ehab Al-Shaer, Archna Bhatia, Esteban Castillo, Zhuo Cheng, Sreekar Dhaduvai, et al. ‘Active Defense Against Social Engineering: The Case for Human Language Technology’. In , 8.

‘DBIR Report 2023 - Master’s Guide’. Verizon Business. Accessed 11 September 2023. https://www.verizon.com/business/resources/reports/dbir/2023/master-guide/.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. ‘BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding’. In *arXiv:1810.04805 [Cs]*. http://arxiv.org/abs/1810.04805.

Dietterich, Thomas G. 2000. ‘Ensemble Methods in Machine Learning’. In *Multiple Classifier Systems*, 1–15. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer. https://doi.org/10.1007/3-540-45014-9_1.

Dos Santos, Cicero, and Maira Gatti. 2014. ‘Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts’. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 69–78.

‘Evaluation Criteria for IT Security — Part 1: Introduction and General Model - ISO/IEC 15408-1:2009’. ISO. Accessed 13 July 2021. https://standards.iso.org/ittf/PubliclyAvailableStandards/c050341_ISO_IEC_15408-1_2009.zip.

Feng, Song, Ritwik Banerjee, and Yejin Choi. 2012. ‘Syntactic Stylometry for Deception Detection’. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, 171–75.

Ganzha, Maria and Polskie Towarzystwo Informatyczne, eds. 2010. *Proceedings of the 2010 International Multiconference on Computer Science and Information Technology (IMCSIT 2010): Wisla, Poland, 18 - 20 October 2010 ; [Consisting of Various Symposia/Workshops/Conferences]*. Piscataway, NJ: IEEE.

Gao, Tianyu, Adam Fisch, and Danqi Chen. 2021. ‘Making Pre-Trained Language Models Better Few-Shot Learners’. In *arXiv:2012.15723 [Cs]*. http://arxiv.org/abs/2012.15723.

Gardner, Matt, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. ‘AllenNLP: A Deep Semantic Natural Language Processing Platform’. In .

Gehl, Robert W., and Sean T. Lawson. 2022. *Social Engineering: How Crowdmasters, Phreaks, Hackers, and Trolls Created a New Form of Manipulativ e Communication*. MIT Press.

Godfrey, John J., and Holliman, Edward. 1993. 'Switchboard-1 Release 2'. Linguistic Data Consortium. https://doi.org/10.35111/SW3H-RW02.

Goldberg, Yoav. 2016. 'A Primer on Neural Network Models for Natural Language Processing'. *Journal of Artificial Intelligence Research* 57 (November): 345–420. https://doi.org/10/gfshxb.

Goldberg, Yoav, and Omer Levy. 2014. 'Word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method'. *arXiv Preprint arXiv:1402.3722*.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

Gragg, David. 2003. 'A Multi-Level Defense Against Social Engineering Social'. *SANS Inst.*, 21. https://doi.org/10.9780/22307850.

Granger, Sarah. 2001. 'Social Engineering Fundamentals, Part I: Hacker Tactics | Symantec Connect'. *Secur. Focus. December* 1527. http://www.symantec.com/connect/articles/social-engineering-fundamentals-part-i-hacker-tactics.

———. 2002. 'Social Engineering Fundamentals, Part II: Combat Strategies'. *Security Focus. Retrieved October* 12: 2006.

Granhag, Pär Anders, and Leif A. Strömwall. 2004. *The Detection of Deception in Forensic Contexts*. Edited by Pär Anders Granhag and Leif A. Strömwall. Vol. 9780521833. Cambridge: Cambridge University Press. https://www.cambridge.org/core/product/identifier/9780511490071/type/book.

Hadnagy, Christopher. 2010. *Social Engineering: The Art of Human Hacking*. John Wiley & Sons.

Heartfield, Ryan, and George Loukas. 2016. 'A Taxonomy of Attacks and a Survey of Defence Mechanisms for Semantic Social Engineering Attacks'. *ACM Computing Surveys* 48 (3): 1–39. https://doi.org/10/gcz9f7.

Hirschberg, Julia, Stefan Benus, Jason M Brenier, Frank Enos, Sarah Friedman, Sarah Gilman, Cynthia Girand, et al. 2005. 'Distinguishing Deceptive from Non-Deceptive Speech'. *Proc. Interspeech 2005*, 1833–36. https://doi.org/10.1.1.59.8634.

Hoeschele, Michael, and Marcus Rogers. 2005. 'Detecting Social Engineering (Chapter)'. In , 67–77. Springer.

'Information Technology - Information Security - Information Assurance | ISACA'. 2018. https://www.isaca.org/pages/default.aspx.

ISO/IEC JTC 1/SC 27 I. 2022. 'ISO/IEC 15408-1:2022'. ISO. 29 November 2022. https://www.iso.org/standard/72891.html.

Jiang, Hang, Xianzhe Zhang, and Jinho D Choi. 'Automatic Text-Based Personality Recognition on Monologues and Multiparty Dialogues Using Attentive Networks and Contextual Embeddings (Student Abstract)', no. 10: 2. https://doi.org/10.1609/aaai.v34i10.7182.

Johnson, Rie, and Tong Zhang. 2016. 'Semi-Supervised Convolutional Neural Networks for Text Categorization via Region Embedding', 26.

Joulin, Armand, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016. 'Fasttext. Zip: Compressing Text Classification Models'. *arXiv Preprint arXiv:1612.03651*.

Jurafsky, Dan, and James Martin. 2022. *Speech and Language Processing*. https://web.stanford.edu/~jurafsky/slp3/.

Kazdin, A.E. 2004. *Encyclopedia of Psychology*. Vol. 1. Wiley New York.

‘Keras: The Python Deep Learning API’. Accessed 9 October 2021. https://keras.io/.
Kim, Hwa Jong, Seong Eun Hong, and Kyung Jin Cha. 2020. ‘Seq2vec: Analyzing Sequential Data Using Multi-Rank Embedding Vectors’. *Electronic Commerce Research and Applications* 43: 101003. https://doi.org/10.1016/j.elerap.2020.101003.
Kim, Yoon. 2014. ‘Convolutional Neural Networks for Sentence Classification’. In *arXiv:1408.5882 [Cs]*. http://arxiv.org/abs/1408.5882.
Kingma, Diederik P., and Jimmy Ba. 2014. ‘Adam: A Method for Stochastic Optimization’. *arXiv Preprint arXiv:1412.6980*.
Kissine, Mikhail. *From Utterances to Speech Acts*.
Kubal, Divesh R., and Anant V. Nimkar. 2019. ‘A Survey on Word Embedding Techniques and Semantic Similarity for Paraphrase Identification’. *International Journal of Computational Systems Engineering* 5 (1): 36–52. https://doi.org/10.1504/IJCSYSE.2019.098417.
Kullback, S., and R. A. Leibler. 1951. ‘On Information and Sufficiency’. *The Annals of Mathematical Statistics* 22 (1): 79–86. https://doi.org/10.1214/aoms/1177729694.
Kumar, Anshul, Mansi Chaudhary, and Nagresh Kumar. 2015. ‘Social Engineering Threats and Awareness: A Survey’, 5.
Lahat, Dana, Tulay Adali, and Christian Jutten. 2015. ‘Multimodal Data Fusion: An Overview of Methods, Challenges, and Prospects’. *Proceedings of the IEEE* 103 (9): 1449–77. https://doi.org/10.1109/JPROC.2015.2460697.
Lample, Guillaume, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. ‘Neural Architectures for Named Entity Recognition’. *arXiv Preprint arXiv:1603.01360*.
Lan, Yuanyuan. 2021. ‘Chat-Oriented Social Engineering Attack Detection Using Attention-Based Bi-LSTM and CNN’. In *2021 2nd International Conference on Computing and Data Science (CDS)*, 483–87. https://doi.org/10/gmx6hr.
Lansley, Merton, Stelios Kapetanakis, and Nikolaos Polatidis. 2020. ‘SEADer++ v2: Detecting Social Engineering Attacks Using Natural Language Processing and Machine Learning’. In *2020 International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*, 1–6. Novi Sad, Serbia: IEEE. https://ieeexplore.ieee.org/document/9194623/.
Lansley, Merton, Francois Mouton, Stelios Kapetanakis, and Nikolaos Polatidis. 2020. ‘SEADer++: Social Engineering Attack Detection in Online Environments Using Machine Learning’. In *Journal of Information and Telecommunication*, 1–17. https://www.tandfonline.com/doi/full/10.1080/24751839.2020.1747001.
Lansley, Merton, Nikolaos Polatidis, and Stelios Kapetanakis. 2019. ‘SEADer: A Social Engineering Attack Detection Method Based on Natural Language Processing and Artificial Neural Networks’. In *Computational Collective Intelligence*, edited by Ngoc Thanh Nguyen, Richard Chbeir, Ernesto Exposito, Philippe Aniorté, and Bogdan Trawiński, 11683:686–96. Cham: Springer International Publishing. http://link.springer.com/10.1007/978-3-030-28377-3_57.
Lateh, Masitah Abdul, Azah Kamilah Muda, Zeratul Izzah Mohd Yusof, Noor Azilah Muda, and Mohd Sanusi Azmi. 2017. ‘Handling a Small Dataset Problem in Prediction Model by Employ Artificial Data Generation Approach: A Review’. In *Journal of Physics: Conference Series*, 892:012016. IOP Publishing.

Lecun, Y., L. Bottou, Y. Bengio, and P. Haffner. 1998. 'Gradient-Based Learning Applied to Document Recognition'. *Proceedings of the IEEE* 86 (11): 2278–2324. https://doi.org/10/d89c25.

Lee, Younghoo, Joshua Saxe, and Richard Harang. 2020. 'CATBERT: Context-Aware Tiny BERT for Detecting Social Engineering Emails'. In *arXiv:2010.03484 [Cs]*. http://arxiv.org/abs/2010.03484.

Li, Der-Chiang, Wu-Kuo Lin, Chien-Chih Chen, Hung-Yu Chen, and Liang-Sian Lin. 2018. 'Rebuilding Sample Distributions for Small Dataset Learning'. *Decision Support Systems* 105: 66–76. https://doi.org/10/gcx3vm.

Li, Tong, and Yeming Ni. 2019. 'Paving Ontological Foundation for Social Engineering Analysis'. In *Advanced Information Systems Engineering*, edited by Paolo Giorgini and Barbara Weber, 246–60. Lecture Notes in Computer Science. Cham: Springer International Publishing. https://doi.org/10/gjmgg3.

Lin, Tianyang, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. 2021. 'A Survey of Transformers'. *arXiv Preprint arXiv:2106.04554*.

Lipton, Zachary C., John Berkowitz, and Charles Elkan. 2015. 'A Critical Review of Recurrent Neural Networks for Sequence Learning'. *arXiv Preprint arXiv:1506.00019*.

Lu, Jie, Vahid Behbood, Peng Hao, Hua Zuo, Shan Xue, and Guangquan Zhang. 2015. 'Transfer Learning Using Computational Intelligence: A Survey'. *Knowledge-Based Systems* 80: 14–23. https://doi.org/10/f696h2.

Luong, Thang, Hieu Pham, and Christopher D. Manning. 2015. 'Effective Approaches to Attention-Based Neural Machine Translation'. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 1412–21. Lisbon, Portugal: Association for Computational Linguistics. https://doi.org/10.18653/v1/D15-1166.

Manning, Christopher D. 2006. 'Local Textual Inference: It's Hard to Circumscribe, but You Know It When You See It – and Nlp Needs It'.

Marelli, Marco, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014. 'SemEval-2014 Task 1: Evaluation of Compositional Distributional Semantic Models on Full Sentences through Semantic Relatedness and Textual Entailment'. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, 1–8. Dublin, Ireland: Association for Computational Linguistics. https://doi.org/10.3115/v1/S14-2001.

McCann, Bryan, James Bradbury, Caiming Xiong, and Richard Socher. 2017. 'Learned in Translation: Contextualized Word Vectors'. In *Advances in Neural Information Processing Systems*. Vol. 30.

McHugh, Mary L. 2012. 'Interrater Reliability: The Kappa Statistic'. *Biochemia Medica* 22 (3): 276–82.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. 'Efficient Estimation of Word Representations in Vector Space'. In *arXiv:1301.3781 [Cs]*. http://arxiv.org/abs/1301.3781.

Mikolov, Tomas, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2017. 'Advances in Pre-Training Distributed Word Representations'. *arXiv:1712.09405 [Cs]*, December. http://arxiv.org/abs/1712.09405.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. 'Distributed Representations of Words and Phrases and Their Compositionality'. In *Advances in Neural Information Processing Systems 26*, edited by C. J. C.

Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, 3111–19. Curran Associates, Inc. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

Miller, George A. 1995. 'WordNet: A Lexical Database for English'. *Commun. ACM* 38 (11): 39–41. https://doi.org/10.1145/219717.219748.

Mitchell, Tom M. 1997. *Machine Learning*. McGraw-Hill Series in Computer Science. New York: McGraw-Hill.

Mitnick, Kevin. 2011. *Ghost in the Wires: My Adventures as the World's Most Wanted Hacker*. Hachette UK.

Mitnick, Kevin D, and William L Simon. 2011. *The Art of Deception: Controlling the Human Element of Security*. John Wiley & Sons.

Mohamed, Idrisa, Wael Gomaa, and Hawaf Abdalhakim. 2019. 'A Hybrid Model for Paraphrase Detection Combines Pros of Text Similarity with Deep Learning'. *International Journal of Computer Applications* 178 (June): 18–23. https://doi.org/10.5120/ijca2019919011.

Mouton, Francois, Mercia M. Malan, Kai K. Kimppa, and Hein S. Venter. 2015. 'Necessity for Ethics in Social Engineering Research'. *Computers & Security* 55: 114–27.

Murata, M., S. St Laurent, and D. Kohn. 'XML Media Types'. Accessed 16 April 2021. https://www.rfc-editor.org/rfc/rfc3023.html.

'NLTK :: Natural Language Toolkit'. Accessed 13 October 2021. https://www.nltk.org/.

Ott, Myle, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. 2011. 'Finding Deceptive Opinion Spam by Any Stretch of the Imagination'. In *Proc. 49th Annu. Meet. Assoc. Comput. Linguist. Hum. Lang. Technol. 1*, 309–19. https://doi.org/10/gf3ghj.

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, et al. 2019. 'PyTorch: An Imperative Style, High-Performance Deep Learning Library'. In *Advances in Neural Information Processing Systems 32*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, 8024–35. Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

'[PDF] Short-Text Classification Detector: A Bert-Based Mental Approach | Semantic Scholar'. Accessed 23 November 2022. https://www.semanticscholar.org/paper/Short-Text-Classification-Detector%3A-A-Bert-Based-Hu-Ding/29a90ee48e2dce15809fc2b330100329a858b5a0.

'[PDF] Textual Analysis for Online Reviews: A Polymerization Topic Sentiment Model | Semantic Scholar'. Accessed 23 November 2022. https://www.semanticscholar.org/paper/Textual-Analysis-for-Online-Reviews%3A-A-Topic-Model-Huang-Dou/60ae927cfa2df3bd7d4992ff4b529d17ec5c005e.

Peltier, Thomas R. 2006. 'Social Engineering: Concepts and Solutions'. *Information Systems Security* 15 (5): 13–21. https://doi.org/10.1201/1086.1065898X/46353.15.4.20060901/95427.3.

Peng, T., I. Harris, and Y. Sawa. 2018. 'Detecting Phishing Attacks Using Natural Language Processing and Machine Learning'. In *2018 IEEE 12th International Conference on Semantic Computing (ICSC)*, 300–301. https://doi.org/10/gfvb9h.

Pennebaker, James W, Ryan L Boyd, Kayla Jordan, and Kate Blackburn. 'The Development and Psychometric Properties of LIWC2015', 26.

Peters, Matthew E., Sebastian Ruder, and Noah A. Smith. 2019. 'To Tune or Not to Tune? Adapting Pretrained Representations to Diverse Tasks'. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, 7–14. Florence, Italy: Association for Computational Linguistics. https://doi.org/10.18653/v1/W19-4302.

Phang, Jason, Thibault Févry, and Samuel R. Bowman. 2019. 'Sentence Encoders on STILTs: Supplementary Training on Intermediate Labeled-Data Tasks'. *arXiv:1811.01088 [Cs]*, February. http://arxiv.org/abs/1811.01088.

Pogrebna, Ganna, and Mark Skilton. 2019. *Navigating New Cyber Risks: How Businesses Can Plan, Build and Manage Safe Spaces in the Digital Age*. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-13527-0.

'Prodigy · An Annotation Tool for AI, Machine Learning & NLP'. Prodigy. Accessed 23 September 2021. https://prodi.gy.

'Propensity to Click on Suspicious Links: Impact of Gender, of Age, and of Personality Traits | Semantic Scholar'. Accessed 7 March 2023. https://www.semanticscholar.org/paper/Propensity-to-Click-on-Suspicious-Links%3A-Impact-of-Sudzina-Pavl%C3%AD%C4%8Dek/6663c539233a1b8e83abe4ea08df58c5a28ae696.

'Protégé'. Accessed 11 May 2021. https://protege.stanford.edu/.

Pustejovsky, J., and Amber Stubbs. 2013. *Natural Language Annotation for Machine Learning*. Sebastopol, CA: O'Reilly Media.

'PyTorch Lightning'. Accessed 13 January 2022. https://www.pytorchlightning.ai/.

Ramshaw, Lance A., and Mitchell P. Marcus. 1995. 'Text Chunking Using Transformation-Based Learning'. *arXiv:Cmp-Lg/9505040*, May. http://arxiv.org/abs/cmp-lg/9505040.

Rastogi, Abhinav, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020a. 'Towards Scalable Multi-Domain Conversational Agents: The Schema-Guided Dialogue Dataset'. arXiv. http://arxiv.org/abs/1909.05855.

———. 2020b. 'Schema-Guided Dialogue State Tracking Task at DSTC8'. arXiv. http://arxiv.org/abs/2002.01359.

Raval, Siraj. (2019) 2022. 'Bert-as-Service'. Python. https://github.com/llSourcell/bert-as-service.

Rere, L.M. Rasdi, Mohamad Ivan Fanany, and Aniati Murni Arymurthy. 2015. 'Simulated Annealing Algorithm for Deep Learning'. *Procedia Computer Science* 72: 137–44. https://doi.org/10.1016/j.procs.2015.12.114.

Resnik, Alan J., and Robert B. Cialdini. 1986. 'Influence: Science & Practice'. *J. Mark. Res.* 23 (3): 305. https://doi.org/10/crkvr5.

Ross, Robert M., David G. Rand, and Gordon Pennycook. 2021. 'Beyond" Fake News": Analytic Thinking and the Detection of False and Hyperpartisan News Headlines.' *Judgment & Decision Making* 16 (2).

Ruder, Sebastian. 2016. 'An Overview of Gradient Descent Optimization Algorithms'. *arXiv Preprint arXiv:1609.04747*.

———. 2019. 'Neural Transfer Learning for Natural Language Processing'. http://ruder.io/thesis/neural_transfer_learning_for_nlp.pdf.

Russell, Stuart J., and Peter Norvig. 2022. *Artificial Intelligence: A Modern Approach*. Fourth edition, Global edition. Pearson Series in Artificial Intelligence. Harlow: Pearson.

Salahdine, Fatima, and Naima Kaabouch. 2019. 'Social Engineering Attacks: A Survey'. *Future Internet*. https://doi.org/10/gf2772.

Saleilles, Jordi, and Esma Aïmeur. 'SecuBot, a Teacher in Appearance: How Social Chatbots Can Influence People', 19.

Salgado, Jesús F. 2002. 'The Big Five Personality Dimensions and Counterproductive Behaviors'. *International Journal of Selection and Assessment* 10 (1–2): 117–25. https://doi.org/10/b8c428.

Samuel Galice Marine Minier (auth.), Serge Vaudenay (eds.). 2008. *Progress in Cryptology - AFRICACRYPT 2008, First International Conference on Cryptology in Africa, Casablanca, Morocco, June 11-14, 2008. Proceedings*. 1st ed. Vol. 5023. Lecture Notes in Computer Science 5023 Security and Cryptology. Springer-Verlag Berlin Heidelberg.

Sawa, Y., R. Bhakta, I. G. Harris, and C. Hadnagy. 2016. 'Detection of Social Engineering Attacks Through Natural Language Processing of Conversations'. In *2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*, 262–65. https://doi.org/10/gfvb9j.

Scheeres, Jamison W. 2008. 'Establishing the Human Firewall: Reducing an Individual's Vulnerability to Social Engineering Attacks'. DTIC Document. http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA487118.

Schuster, Mike, and Kuldip K. Paliwal. 1997. 'Bidirectional Recurrent Neural Networks'. *IEEE Transactions on Signal Processing* 45 (11): 2673–81. https://doi.org/10.1109/78.650093.

Searle, John R. 1969. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge: Cambridge University Press. https://doi.org/10.1017/CBO9781139173438.

———. 2010. *Expression and Meaning: Studies in the Theory of Speech Acts*. Nachdr. Cambridge: Cambridge Univ. Pr.

Searle, John R, Ferenc Kiefer, Manfred Bierwisch, and others. 1980. *Speech Act Theory and Pragmatics*. Vol. 10. Springer.

Shelar, Hemlata, Gagandeep Kaur, Neha Heda, and Poorva Agrawal. 2020. 'Named Entity Recognition Approaches and Their Comparison for Custom NER Model'. *Science & Technology Libraries* 39 (3): 324–37. https://doi.org/10/gm3c5w.

Siddiqi, Murtaza Ahmed, Wooguil Pak, and Moquddam A. Siddiqi. 2022. 'A Study on the Psychology of Social Engineering-Based Cyberattacks and Existing Countermeasures'. *Applied Sciences* 12 (12): 6042. https://doi.org/10.3390/app12126042.

Singh, Shashank, and Shailendra Singh. 2020. 'Systematic Review of Spell-Checkers for Highly Inflectional Languages'. *Artificial Intelligence Review* 53 (6): 4051–92. https://doi.org/10.1007/s10462-019-09787-4.

Smutz, Charles, and Angelos Stavrou. 2012. 'Malicious PDF Detection Using Metadata and Structural Features'. In *Proceedings of the 28th Annual Computer Security Applications Conference*, 239–48. ACSAC '12. New York, NY, USA: Association for Computing Machinery. https://doi.org/10/gmx6nc.

Souag, Amina, Camille Salinesi, and Isabelle Comyn-Wattiau. 2012. 'Ontologies for Security Requirements: A Literature Survey and Classification'. In *International Conference on Advanced Information Systems Engineering*, 61–69. Springer.

'spaCy · Industrial-Strength Natural Language Processing in Python'. Accessed 13 October 2021. https://spacy.io/.

Spielberger, Charles Donald. 2004. *Encyclopedia of Applied Psychology*. Elsevier Academic Press.

Syafitri, Wenni, Zarina Shukur, Umi Asma' Mokhtar, Rossilawati Sulaiman, and Muhammad Azwan Ibrahim. 2022. 'Social Engineering Attacks Prevention: A Systematic Literature Review'. *IEEE Access* 10: 39325–43. https://doi.org/10.1109/ACCESS.2022.3162594.

Taylor, Ann, Mitchell Marcus, and Beatrice Santorini. 2003. 'The Penn Treebank: An Overview'. *Treebanks*, 5–22. https://doi.org/10/ft55q7.

'TensorFlow'. TensorFlow. Accessed 9 October 2021. https://www.tensorflow.org/.

'TextBlob: Simplified Text Processing — TextBlob 0.16.0 Documentation'. Accessed 11 May 2021. https://textblob.readthedocs.io/en/dev/index.html.

Thompson, Victor. 2017. 'Methods for Detecting Paraphrase Plagiarism'. *arXiv:1712.10309 [Cs]*, December. http://arxiv.org/abs/1712.10309.

Torfi, Amirsina, Rouzbeh A. Shirvani, Yaser Keneshloo, Nader Tavaf, and Edward A. Fox. 2020. 'Natural Language Processing Advancements by Deep Learning: A Survey'. *arXiv Preprint arXiv:2003.01200*.

Tsinganos, Nikolaos. 2021. 'Tsinik/Applsci-1445637: Applsci-1445637 v1.0.0'. Zenodo. https://doi.org/10.5281/zenodo.5635627.

Tsinganos, Nikolaos, Panagiotis Fouliras, and I. Mavridis. 2022. 'Applying BERT for Early-Stage Recognition of Persistence in Chat-Based Social Engineering Attacks'. *Undefined*. https://www.semanticscholar.org/paper/Applying-BERT-for-Early-Stage-Recognition-of-in-Tsinganos-Fouliras/e83dfbf2055cc0a14a60474bc9cfea33e5dae3e2.

Tsinganos, Nikolaos, Panagiotis Fouliras, and Ioannis Mavridis. 2023. 'Leveraging Dialogue State Tracking for Zero-Shot Chat-Based Social Engineering Attack Recognition'. *Applied Sciences* 13 (8): 5110. https://doi.org/10.3390/app13085110.

Tsinganos, Nikolaos, and Ioannis Mavridis. 2021. 'Building and Evaluating an Annotated Corpus for Automated Recognition of Chat-Based Social Engineering Attacks'. *Applied Sciences* 11 (22). https://doi.org/10/gnpb3z.

Tsinganos, Nikolaos, Ioannis Mavridis, and Dimitris Gritzalis. 2022. 'Utilizing Convolutional Neural Networks and Word Embeddings for Early-Stage Recognition of Persuasion in Chat-Based Social Engineering Attacks'. *IEEE Access* 10: 108517–29. https://doi.org/10.1109/ACCESS.2022.3213681.

Tsinganos, Nikolaos, Georgios Sakellariou, Panagiotis Fouliras, and Ioannis Mavridis. 2018. 'Towards an Automated Recognition System for Chat-Based Social Engineering Attacks in Enterprise Environments'. In , 10.

Uebelacker, Sven, and Susanne Quiel. 2014. 'The Social Engineering Personality Framework'. In *2014 Workshop on Socio-Technical Aspects in Security and Trust*, 24–30. Vienna: IEEE. https://doi.org/10/gf3gg9.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. 'Attention Is All You Need'. In *Advances in Neural Information Processing Systems*. Vol. 30.

Venkatesha, Sushruth, K. Rahul Reddy, and B. R. Chandavarkar. 2021. 'Social Engineering Attacks During the COVID-19 Pandemic'. *SN Computer Science* 2 (2): 78. https://doi.org/10.1007/s42979-020-00443-1.

Verizon Enterprise. '2019 Data Breach Investigations Report.' Accessed 27 June 2019. enterprise.verizon.com/resources/reports/dbir/.

Vishwanath, Arun. 2022. *The Weakest Link: How to Diagnose, Detect, and Defend Users from Phishing*. MIT Press.

Vrbanec, Tedo, and Ana Meštrović. 2020. 'Corpus-Based Paraphrase Detection Experiments and Review'. *Information* 11 (5): 241. https://doi.org/10.3390/info11050241.

Vrij, Aldert. 2014. 'Detecting Lies and Deceit: Pitfalls and Opportunities in Nonverbal and Verbal Lie Detection'. In *Interpers. Commun.*, 321–46.

Wang, Sinong, Han Fang, Madian Khabsa, Hanzi Mao, and Hao Ma. 2021. 'Entailment as Few-Shot Learner'. *arXiv:2104.14690 [Cs]*, April. http://arxiv.org/abs/2104.14690.

'What Is "Social Engineering"?' Page. ENISA. Accessed 29 October 2021. https://www.enisa.europa.eu/topics/csirts-in-europe/glossary/what-is-social-engineering.

Wilcock, Graham. 2009. 'Introduction to Linguistic Annotation and Text Analytics'. *Synthesis Lectures on Human Language Technologies* 2 (1): 1–159. https://doi.org/10/ff48b9.

Wilcox, Heidi, and Maumita Bhattacharya. 2016. 'A Framework to Mitigate Social Engineering through Social Media within the Enterprise'. In *2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA)*, 1039–44. Hefei, China: IEEE. https://doi.org/10.1109/ICIEA.2016.7603735.

Williams, Jason D., Antoine Raux, and Matthew Henderson. 2016. 'The Dialog State Tracking Challenge Series: A Review'. *Dialogue & Discourse* 7 (3): 4–33. https://doi.org/10.5087/dad.2016.301.

Williams, Jason D., and Steve Young. 2007. 'Partially Observable Markov Decision Processes for Spoken Dialog Systems'. *Computer Speech & Language* 21 (2): 393–422. https://doi.org/10.1016/j.csl.2006.06.008.

Winograd, Terry. 1986. 'A Language/Action Perspective on the Design of Cooperative Work'. In *Proceedings of the 1986 ACM Conference on Computer-Supported Cooperative Work*, 203–20.

Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, et al. 2020. 'HuggingFace's Transformers: State-of-the-Art Natural Language Processing'. arXiv. https://doi.org/10.48550/arXiv.1910.03771.

'WordNet | A Lexical Database for English'. Accessed 14 October 2021. https://wordnet.princeton.edu/.

Young, Steve, Milica Gašić, Simon Keizer, François Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu. 2010. 'The Hidden Information State Model: A Practical Framework for POMDP-Based Spoken Dialogue Management'. *Computer Speech & Language* 24 (2): 150–74. https://doi.org/10.1016/j.csl.2009.04.001.

Yu, Yong, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. 2019. 'A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures'. *Neural Computation* 31 (7): 1235–70. https://doi.org/10.1162/neco_a_01199.

Zhang, Ye, and Byron Wallace. 2015. 'A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification'. *arXiv Preprint arXiv:1510.03820*.