IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 27 September 2022, accepted 7 October 2022, date of publication 10 October 2022, date of current version 17 October 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3213681

RESEARCH ARTICLE

Utilizing Convolutional Neural Networks and Word Embeddings for Early-Stage Recognition of Persuasion in Chat-Based Social Engineering Attacks

NIKOLAOS TSINGANOS[1], IOANNIS MAVRIDIS[1], AND DIMITRIS GRITZALIS[2]
[1]Department of Applied Informatics, University of Macedonia (UoM), 54636 Thessaloniki, Greece
[2]Department of Informatics, Athens University of Economics and Business (AUEB), 10434 Athens, Greece

Corresponding author: Dimitris Gritzalis (dgrit@aueb.gr)

This work was supported in part by the Research Grant through the Ministry of Digital Governance of Greece to the Research Center of the Athens University of Economics and Business, Greece (2021–2022).

ABSTRACT Social engineering is widely recognized as the key to successful cyber-attacks. Chat-based social engineering (CSE) attacks are attracting increasing attention because of recent changes in the digital work environment. Sophisticated CSE attacks target human personality traits, and persuasion is regarded as the catalyst to successful CSE attacks. To date, research in social engineering has mostly focused on phishing attacks, neglecting the importance of chat-based software. This paper describes the design and implementation of a persuasion classifier that utilizes machine learning and natural language processing techniques. For this purpose, a convolutional neural network was trained on a chat-based social engineering corpus (CSE Corpus), specifically annotated for recognizing Cialdini's persuasion principles. The proposed persuasion classifier network, named CSE-PUC, can determine whether a sentence carries a persuasive payload by producing a probability distribution over the sentence classes as a persuasion container. The present study is expected to contribute to our understanding of utilizing existing machine learning models and integrating context-aware information into real-life cyber security threats. The experimental application results reported in this work confirm that the approach taken can recognize persuasion methods and is thus able to protect an interlocutor from being victimized.

INDEX TERMS Cyber-security, machine learning, natural language processing, chat-based social engineering, attack detection.

I. INTRODUCTION

In recent years, there has been a significant increase in the use of electronic communication tools in small-medium enterprise environments. This rising trend can largely be attributed to the continuous development of novel communication technologies and unforeseen challenges faced by digital work environments during the COVID-19 pandemic. Consequently, a corresponding increase in the available attack surface was realized for social engineers. Social engineering is a complex phenomenon combining technology, psychology, and linguistics. Attackers treat human personality traits as vulnerabilities and use language as their weapon to deceive, persuade, and finally manipulate their victims as they wish. Human weaknesses and limitations in the context of personality traits constitute vulnerabilities which social engineers can exploit using a variety of methods.

Persuasion is a well-known method used by social engineers and the latest research regarding persuasion as an influence tactic [1], [2], confirms and emphasizes persuasion's role as a social engineering attack enabler. ISACA [3] defines enablers as the ''factors that, individually and collectively, influence whether something will work''. In [4], persuasion was identified as one of the six most critical factors that

The associate editor coordinating the review of this manuscript and approving it for publication was S. K. Hafizul Islam.

can lead to a successful chat-based social engineering (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 [5] adds 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 [6]. Therefore, from a cybersecurity perspective, it is critical to identify if a sentence in a natural communication setting contains a degree of disinformation. We define *persuasive payload* (pp) as every piece of information that purposely contains content which aims to deceive a human by altering his/her opinion or misleading him/her to act erroneously. Furthermore, we define every sentence that carries persuasive payload as a pp-container, which is to say, every sentence that carries information which corresponds to one of the seven persuasion principles defined by Cialdini will be considered a pp-container.

The main objective of this study is to guide cybersecurity defense mechanisms in detecting 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 our goal, we utilized machine learning and natural language processing (NLP) techniques, namely convolutional neural networks (CNN) and word embeddings.

In this work, 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.

While most studies have focused on email-based phishing attacks, this study investigates chat-based social engineering attacks. 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 detection and defense separately. As mentioned in [4], persuasion is only one of the various enablers, and it is critical to be able to detect 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 detected in a cyber-security context using an appropriate dataset related to digital work environments, then the expected outcome will be rather useful. The present study is expected to contribute to our understanding of the application of well-known machine learning methods to CSE attack recognition.

The remainder of this paper is organized as follows. Section 2 describes the related work in this field. Section 3 details related background information. Section 4 describes the design and implementation of the proposed CSE-PUC. Section 5 presents the experimental results and evaluation. Section 6 summarizes our findings and discusses the limitations of our study. Section 7 concludes the study and presents recommendations for future work.

II. RELATED WORK

As stated in the Introduction, although CSE attacks are increasingly attracting attention from malicious users, there is limited interest in investigating the attack techniques used and the enablers that lead to successful social engineering attacks through chat-based conversations. Current research seems to focus on phishing attacks through email. Nevertheless, the detection approaches considered are worth mentioning and presenting.

In [7], the authors explored nine different machine learning models trained on three different datasets. They extracted threat features that are compared against twenty-seven threat detectors to identify general social engineering attacks that do not focus on a specific technique. The results are promising, but the lack of focus eliminates the possibility of detecting persuasion methods in chat-based conversations.

The authors in [8] used a modern approach to classify social engineering attacks based on the technique and the influence tactic that was employed. Furthermore, the authors mapped several types of attacks to various human vulnerabilities. We utilize the ''Persuasion'' and ''Attribute and Behavior'' categories, as presented in this work.

Lumen [9] is a multi-task and multilevel learning-based framework that exposes not only persuasion cues but also framing, objectivity/subjectivity, guilt/blame, and the use of emphasis. The authors utilized a custom dataset to train the learner by combining traditional natural language processing tools such as linguistic inquiry and word count (LIWC), topic modeling, and sentiment analysis, which feed on a random forest classifier. Lumen presented satisfactory performance compared to Labeled-LDA and Long Short-Term Memory (LSTM). However, this solution lacks the ability to take advantage of modern word representation techniques and the flexibility of CNNs.

Duerr et al. [10] conducted nine exploratory case studies to investigate the challenges faced by writers trying to increase their persuasiveness and the complementary effect that artificial intelligence (AI) can have through natural language processing techniques. Their analysis showed that humans and AI could complement each other, as AI increases persuasiveness through the automated creation of logical coherence and conciseness. This is however a theoretical approach with no evidence of its application in real-life situations.

Dimitrov et al. [11] described the results and participating systems in the detection of persuasion techniques in text

and image tasks. Twenty-two persuasion techniques were investigated, of which 20 were applicable to both text and image communication media. The participating teams utilized a plethora of detection and analysis techniques and were presented with F1-Micro and F1-Macro metrics. The results presented are of average performance when compared with our approach, and the proposed architectures are more complex.

Wang [12] investigated conversational agents (chatbots) intended to change people's opinions. Working on an annotated dataset created from dialogues between humans, they predicted ten persuasion strategies and combined their findings with the demographic and psychological background of the interlocutors. Using a hybrid region-based convolutional network (RCNN) model with three types of features–sentence embedding, context embedding, and sentence-level features–they achieved good results in predicting persuasion strategies. The proposed solution is inefficient for CSE attacks because the persuasion methods used in CSE attacks are limited to authority and commitment.

In [13], the authors presented a study on persuasion tactics used in social network sites with the express purpose of forcing users to reveal sensitive data and consequently become victims of social engineering attacks. They focus on AI as a means to raise awareness of the presence of a nudge to use, since all persuasive tactics target people's automatic and subconscious processing systems. This approach focuses on awareness and does not detect persuasion methods that can lead to CSE attacks.

Chen [14] investigated a state-of-the-art natural language processing model, transformer-based coupled with Conditional Random Field (CRF). When comparing this architecture with several baseline systems, the model's limitations for persuasion strategy recognition become apparent. They proved the failure of the CRF component to capture persuasive label dependencies and that of the transformer part to capture sequential information in persuasive dialogues. The author concluded that neither machine learning nor deep learning algorithms can solve a problem without accounting for the problem context.

Yasin et al [15] performed a literature review on social engineering attacks, focusing on the persuasion techniques used by social engineers. They utilized thematic and game-based analysis techniques to expose human factors, principles, psychological factors, human vulnerabilities/weaknesses, and compliance principles as means of persuasion tactics. Nineteen persuasion principles have been explained and mapped to several case studies. The authors investigated different persuasion tactics in a different context than CSE attacks, and the results were inefficient for an early-stage recognition cyber defense mechanism.

Yang et al. [16] proposed a neural network architecture that quantified persuasiveness and identified persuasive strategies. The proposed learner outperformed several baseline learners, and offered increased interpretability. A semi-supervised neural network composed of a sentence encoder and a document encoder was trained using a custom dataset, and five persuasion strategies were identified. The proposed model showed an improvement in accuracy when compared with several baseline learners. The solution put forward presents comparable performance results to our approach, but with a growing complexity in architecture.

Polatidis et al. [17] proposed and demonstrated a two-stage deep learner based on natural language processing techniques to detect social engineering attacks. This model identifies the principles of persuasion based on Cialdini's work. Using a semi-synthetic dataset, they presented their approach's effectiveness by showing highly accurate results.

Dutta [18] modeled persuasive interactions between users on an online chat platform using an LSTM neural network architecture. They focused on identifying a chain of arguments that can lead to persuasion by utilizing an attention mechanism. Furthermore, their study proved that the attention mechanism can also focus on argumentative sentences in comments during the persuasion learning task. Although a promising solution, this approach requires a larger sequence of sentences to draw conclusions that are less efficient for real-time applications, such as chat-based conversations.

Hidey [19] modeled the sequence of arguments in social media discussions using a neural network architecture. The model learns the document using a long short-term memory network with an attention mechanism over sentences. An effort was made to model knowledge, dialogue, and the sequence of reasoning to predict persuasiveness. Although LSTMs can capture long-term dependencies, CNNs can independently assign weights to the words in each sentence, which is more efficient for short conversations.

Jan-Willem et al. [20] In an extensive study, [20] explored the persuasion principles used for successful social engineering attacks. They analyze seventy-four scenarios found in books written by social engineers. They break down each attack into specific steps and after dissecting them, they conclude that authority is the most common persuasion principle used. Their work is built on Cialdini's work and the corresponding persuasion principles. The authors confirm our findings that Authority is the most used persuasion method by social engineers.

Iyer et al. [21] presented a multiclass classifier trained using an unsupervised domain-independent model to detect the persuasion principles. They utilize sentence structure to detect persuasion tactics and an argument extractor to extract critical arguments in the text. Their system achieved satisfactory results in detecting reasoning, deontic/moral appeal, outcome, and empathy types of persuasion. The authors used a different persuasion classification method, which is inappropriate for CSE attack detection.

Older works worth mentioning as precursors to the interdisciplinary research field of psychology and cyber security are Matthew Edwards [22], Anand et al. [23], Bullée et al. [24], Levine [25], Ortiz [26], Putri [27], Weirich and Sasse [28], and Young et al. [29]. In these studies, traditional machine learning techniques were used along with

TABLE 1. Attributes of social engineering attacks.

Attack Attribute	Value
Actor	**Human**, Software
Approach	Physical, **Technical**
Method	Social, **Socio-technical**
Route	**Direct**, Indirect
Technique	Dumpster Diving, Shoulder surfing, Phishing, Baiting, Reverse social engineering, Water holing, tailgating, impersonation, **misleading**, grooming, pretexting, profile cloning, Quid Pro Quo, Diversion theft
Distribution method	Email, telephone, **chat**, website, popup, smshing, malware

feature extraction and selection techniques. Combinations of machine learning algorithms and features were presented, laying the foundation for current research.

III. BACKGROUND

A. SOCIAL ENGINEERING ATTACKS

In [30], social engineering is defined as ''a deceptive process in which crackers 'engineer' or design a social situation to trick others into allowing them access to an otherwise closed network, or into believing a reality that does not exist''. Furthermore, the European Union Agency for Cybersecurity (ENISA) states that social engineering ''refers to all techniques aimed at talking a target into revealing specific information or performing a specific action for illegitimate reasons'' [31]. Nevertheless, owing to the complexity of this phenomenon, several taxonomies of social engineering attacks have been proposed [32], [33], [34], [35], [36], [37].

In the most recent literature review [37] and [32], the authors classified social engineering attacks into 15 distinct categories based on the technique they employed, while in [35], the attacks were classified based on the target assets. Li et al [38] adopted a different approach, using an ontology in the context of social engineering, and utilizing it as a starting point for their proposed taxonomy. Summarizing the existing taxonomies, we identified the attack attributes that are most relevant in the context of CSE attacks. Each attack attribute can assume two or more values based on the nature of the attack. The attributes that were chosen are 'actor', 'approach', 'method', 'route', 'technique', and the 'distribution medium' used to manipulate victims. Table 1 presents the attack attributes, along with the different values that they can be assigned.

Our study focuses (shown in bold in Table 1) on human actors who take a technical approach using a direct sociotechnical method to perform a misleading technique through chat-based conversation.

B. CIALDINI'S 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 [5], [39], [40].

Cialdini, a well-known author, presents persuasive psychological principles with a remarkable ability to direct human actions. Additionally, to the six known principles of authority, reciprocity, liking, social proof, scarcity, and commitment, the author adds one more principle in his latest work: Unity. The latter work builds on his previous seminal works and acts as an entry point for most interdisciplinary scientific works on persuasion. Overall, the seven 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 our CSE corpus [41], we concluded that authority and commitment were the most common persuasion principles used by social engineers. Nevertheless, in this study, all persuasion principles were considered to be equally important.

C. CONVOLUTIONAL NEURAL NETWORKS

Convolutional neural networks [42] 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 Fig.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.

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

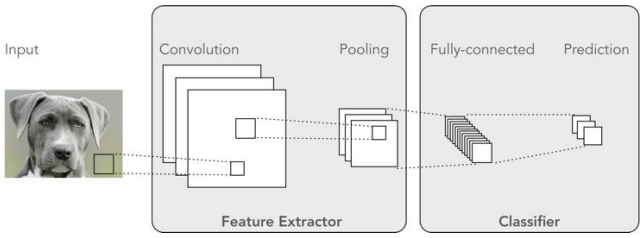

FIGURE 1. A CNN comprises two functional parts: The feature extractor part contains the convolution and pooling layers, and the classifier part contains the fully connected and prediction layers.

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 Fig.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, our 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 natural language processing (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 [43]. Using a similar representation, a CNN can learn to identify the local patterns during the training phase. Several CNN architectures [44] have been used successfully for a variety of natural language processing tasks (text classification [45], sentence classification [46], and sentiment analysis [47].

IV. THE PROPOSED CSE-PUC

A. DESCRIPTION

Cialdini [5] 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 [8], 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 our work, and thus we propose a way of identifying the 'Persuasion using authority' and 'Commitment' methods.

The proposed CSE-PUC 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 the CSE-PUC, and it was integrated at a later stage with the rest of the architecture. The full CSE-PUC network was trained end-to-end, producing the end result of the prediction task.

As mentioned in the Introduction, 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 our CSE Corpus [41]: "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.

We aim to create a deep learning network architecture, the CSE-PUC, which we will feed with sentences to identify such informative cues. CNNs with convolutional and pooling layers are used to identify such local cues [48], where they have been used to identify indicative phrases for specific tasks. Furthermore, it is not important where this pattern may appear

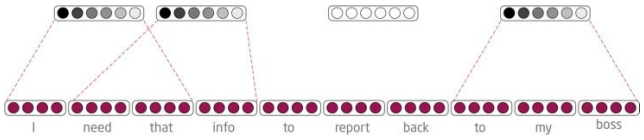

FIGURE 2. Convolution was applied over a sentence with a window of size k = 3 and an output of dimension $\ell = 6$.

in a sentence. The main point of interest is only the existence of a specific sequence of tokens of varying lengths that indicates 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 our 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 Fig.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*."

B. DATASET

The CSE-PUC architecture was trained using the CSE 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 corpus was created having the characteristics presented in Table 2.

The CSE 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 Fig.3 where 51,1% of the sentences were pp-containers and 48,9% were not (denoted in the figure as neutral).

The distribution of the number of words in the sentences in each class is shown in Fig.4. 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.

TABLE 2. CSE corpus id.

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

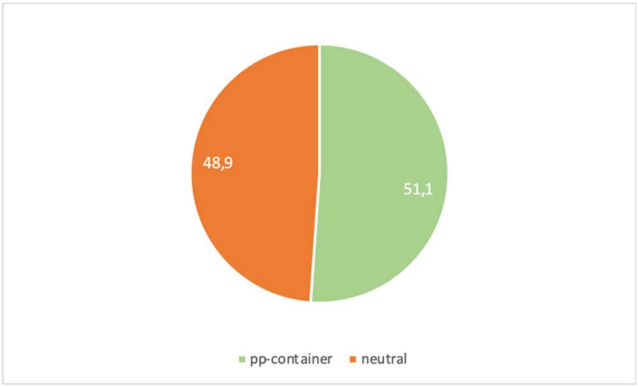

FIGURE 3. The modified CSE corpus was annotated for the pp-container classification task, and the outcome was a well-balanced corpus composed of 51,1% pp-container sentences and 48,9% neutral sentences.

C. OPERATION

The CSE-PUC 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 our tests, we used the publicly available fastText word vectors [49] 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 [50]. All words in the CSE corpus vocabulary that were not present in the set of pre-trained words were initialized randomly with the same dimension and variance.

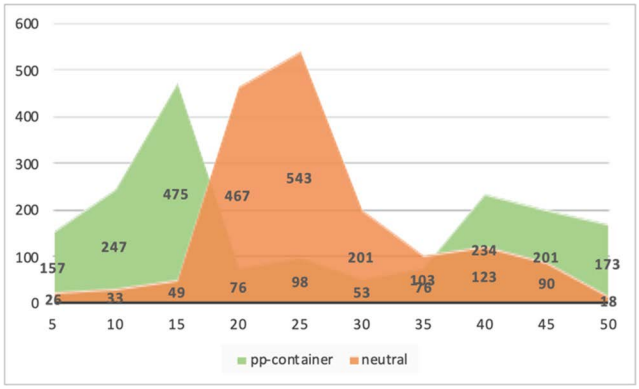

FIGURE 4. Distribution of words per sentence class in the CSE Corpus.

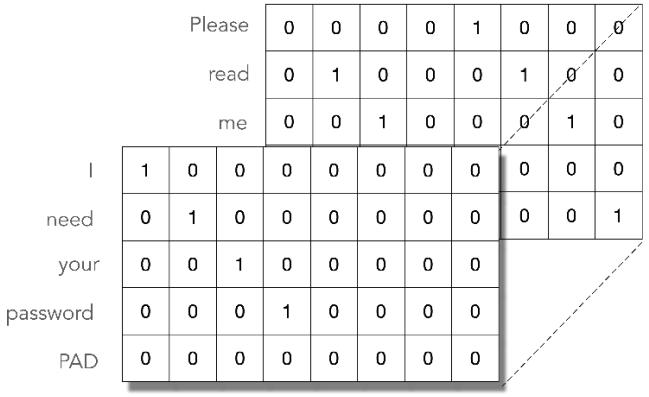

FIGURE 5. One-hot word vector representation of a corpus composed of only two sentences; thus, b = 2. The vocabulary was of size v = −8, and the largest sentence was of size n = 5.

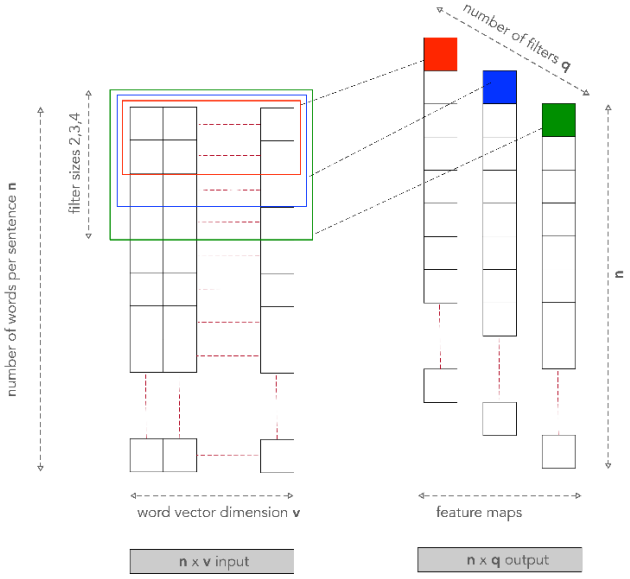

FIGURE 6. 1-D Convolution of a n*v input with a m*k filter. A number of q filters outputs a q*n matrix.

The operation of our CSE-PUC 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 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, we assume a one-hot representation. The word embeddings are shown in Figure 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 Fig.6.

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 (Fig.7).

The end-to-end CSE-PUC neural network architecture is shown in Fig.8.

D. ARCHITECTURE

Our CSE-PUC network is trained end-to-end. As mentioned earlier, we use a CNN as a feature extractor that produces a sequence of vectors that is later fed into the following parts of the network to conclude with a prediction. Every interesting pattern in each sentence that is informative for the pp-container classification task is captured by our CSE-PUC.

We use a convolutional neural network with one layer of convolution trained on top of word vector embeddings created by fastText (own trained and pre-trained). Our 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. Our 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.

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

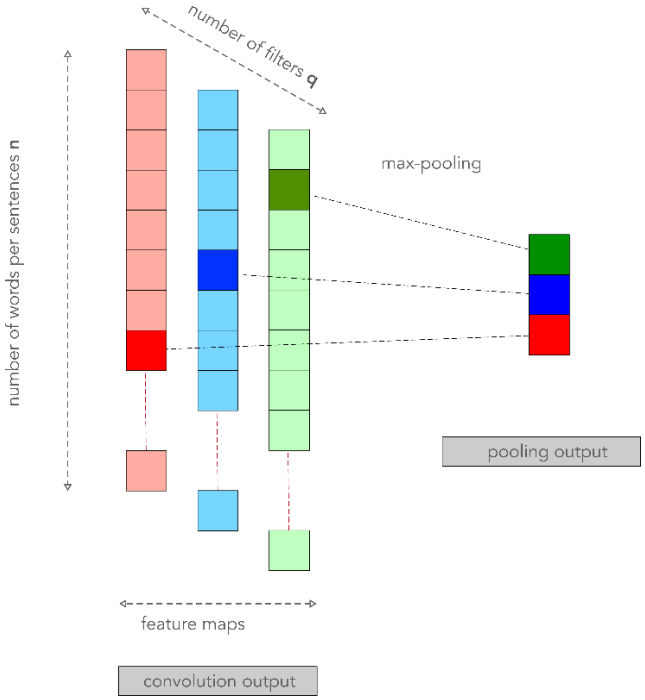

FIGURE 7. The max-pooling operation, where a q × 1 vector is produced by subsampling from the q × n matrix.

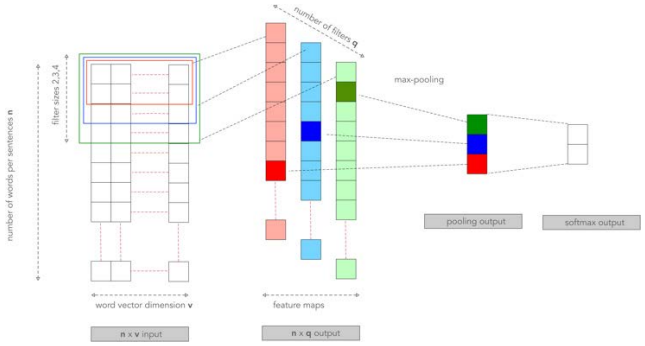

FIGURE 8. Abstract Persuasion classifier neural networks determine whether a sentence is a pp-container or not.

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

1) CONVOLUTION

Suppose that we have a sequence of n words $w_{1:n} = w_1, w_2, \ldots, w_n$, 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 our case is a rectified linear unit (ReLU) [51].

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

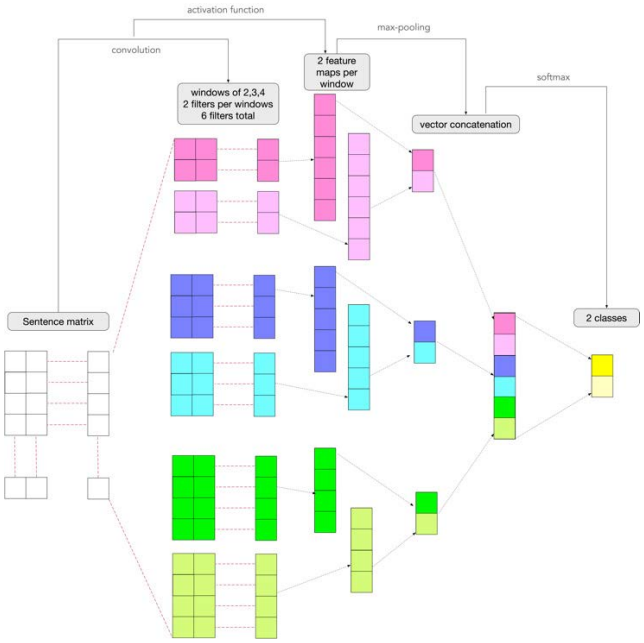

FIGURE 9. Functional end-to-end architecture of the CSE-PUC, where the labels above each layer represent the operations performed.

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 $\mathbf{U}$, 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.

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 our 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 our 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 our example sentence "I need that info to report back to boss" and adapting to the inspiring work of Goldberg [52], we illustrate in Fig.10 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. Finally, the max-pooling operation is applied, which takes the maximum over each dimension (feature), resulting in the final 3-dimensional pooled vector.

E. MODEL 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 3 presents the same process in pseudocode:

- STEP 1**:** All filter values and weights were initialized using fastText word vector representations (either own trained on the CSE corpus or pre-trained).
- STEP 2: 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.
- STEP 3: Calculate the total error at the prediction layer

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

- STEP 4: Use 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.
- STEP 5: Repeat steps 2-4 with all sentences in the training set.

When a new (unseen) sentence reaches the CSE-PUC network 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).

The above steps train the CSE-PUC network, 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.

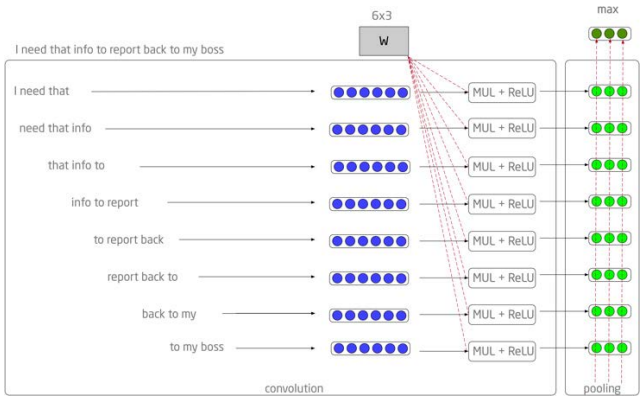

FIGURE 10. Illustration of convolution, pooling, and max-pooling operations using a window of size three. Each word vector representation is of size 6, and after the max-pooling operation a 3-dimensional vector is produced.

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 Fig.11.

V. RESULTS

PyTorch [53] was used to implement the CSE-PUC architecture. This python library is dedicated to facilitating rapid research on deep learning models by easing the implementation of innovative neural network architectures. *AllenNLP* [54], a Pytorch-based NLP library designed to support researchers who want to build novel natural language models, is also used. *Pytorch Lightning* [55] and *wandb* [56] completed our toolset to manipulate the training pipelines and Bayesian hyper-parameter sweep.

The critical difference between the different variations of CSE-PUC architectures that were tested in the word vector representation layer. Initially, we created our own trained word vectors using the cutting-edge algorithm fastText trained on the CSE corpus. fastText [57] 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, we utilized fastText pre-trained word vectors, which were treated either as fixed (they are not updated during training) or as updated parameters of the CSE-PUC network. Finally, we used randomly initialized word vectors in one test turn. Thus, the four variations of our CSE-PUC 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 corpus was made for the training/development/test sets, and a five-fold validation was conducted using the average scores across folds to compare

TABLE 3. Algorithmic steps of CSE-PUC training.

Algorithm CSE-PUC

Require labeled SOURCE corpus, unlabeled TARGET corpus, hyperparameters
Input dataset $\mathcal{D} \leftarrow$ CSE Corpus, n words $w_{1:n} = w_1, w_2, \ldots, w_n$, 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 $x_i = [w_i, w_{i+1}, \ldots, w_{i+k}] \in \mathbb{R}^{k*d}$ *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*

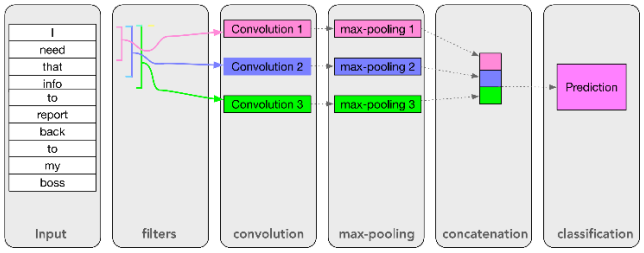

FIGURE 11. End-to-end training of the persuasion classifier for recognizing whether a sentence is a pp-container.

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.

The experimental results are presented in Table 4 and illustrated in Fig.12 and 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 on the validation set. The CSE-PUC

TABLE 4. Units for magnetic properties.

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 %

with updated pre-trained word embeddings yielded the best results among the four different CSE-PUC variations. It outperformed the fixed pre-trained word embeddings, CSE-PUC with own-trained embeddings, and CSE-PUC with randomly initialized word embeddings by a clear margin, achieving an accuracy of 71.6%. The fixed word embedding representation gave the second maximum accuracy of 66.4%, while the CSE-PUC with own-trained word embedding representation attained an accuracy of 62.2%, and the CSE-PUC with randomly initialized word embeddings 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.

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 our previous work [4].

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.

VI. DISCUSSION

The CSE-PUC architecture details and training choices are presented in Table 5.

During our tests of the CSE-PUC, several different architectures were tested (e.g., multiple layers of convolution), but no significant increase in performance was observed; thus, a simpler architecture was selected. Our 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 natural language processing 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 our case, recognizing whether persuasion methods existed in a sentence during a potential chat-based social engineering attack was cast as a sentence classification problem. Moreover, we had to tune the NLP and deep learning toolset to meet our needs for this approach to work. We made the toolset aware of the cybersecurity domain, specifically, the chat-based social

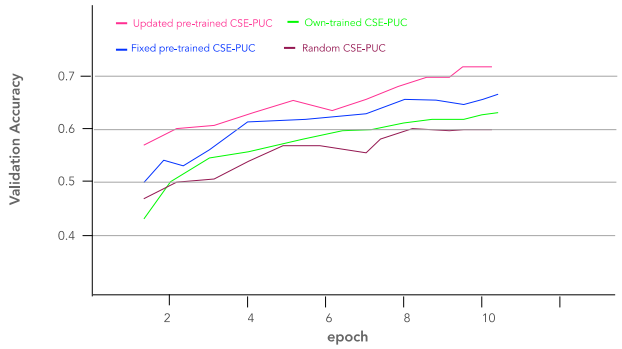

FIGURE 12. Accuracy ratio of the validation set for four different CSE-PUC variations regarding the method of word embedding initialization.

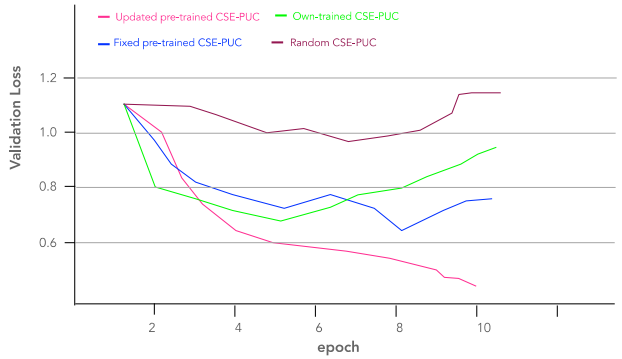

FIGURE 13. Loss on the validation set for the updated pre-trained CSE-PUC, own-trained CSE-PUC, fixed pre-trained CSE-PUC, and random CSE-PUC models.

engineering attack recognition domain. The CSE corpus and persuasion-oriented annotation played a critical role in this awareness. The annotation transforms a corpus composed of chat-based social engineering 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 CSE-PUC architecture, which encodes the written sentences of the interlocutors was trained over the CSE corpus. During this process contextual knowledge from the digital battlefield is injected in CSE-PUC. 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 Corpus gave satisfactory results, and future enrichment of the CSE corpus would be beneficial for transfer learning.

The convolutional neural network as an encoder and a feature extractor has already been tested, measured [46], and trusted for text classification. The convolution operation plays an essential role in preserving the spatial information of a sentence. In the present study, we trained a convolutional neural network with one convolution layer that uses multiple filters and filter sizes to obtain 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. However, before the training process, we still need to specify several hyperparameters, such as the number of filters and filter size, as shown in Table 5.

TABLE 5. CSE-PUC architecture & training details.

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

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 the CSE-PUC, and filter sizes of 2, 3, 4, and 5 were used with 100 feature maps each. During training, the Adadelta optimizer was selected.

Because of the small size of the training corpus, we wanted to avoid the out-of-vocabulary (OOV) problem, which was mitigated by the fastText algorithm, taking advantage of the aging subword and alleviating the OOV problem.

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 CSU-PUC. 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 CSE-PUC 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. CSE-PUC 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 CSE-PUC has only one convolutional layer keeps complexity low and makes the model efficient for implementation.

VII. CONCLUSION

Overall, our results suggest that satisfactory performance levels can be achieved while keeping a simple model architecture and low computational costs. This approach broadens our 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. Our research indicate 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, detect personality traits, or recognize speech acts by using modern text-classification machine-learning techniques.

However, the present study examined only simple CNN architectures. Therefore, further research should investigate different architectures and word-representation techniques. Although the problem of model interpretability remains, there is work in progress in this direction, and in the meantime, it is hard not to take advantage of the current trends in deep learning. Chat-based social engineering attacks are a key area to be explored further by utilizing recent developments in the deep learning and natural language processing fields. We are already working to integrate our findings and complete our CSE attack recognition system, which utilizes multiple factors (recognizers), as it was first proposed in our initial work.

REFERENCES

[1] J. Cacioppo, S. Cacioppo, and R. Petty, "The neuroscience of persuasion: A review with an emphasis on issues and opportunities," *Social Neurosci.*, vol. 13, no. 8, pp. 129–172, 2018, doi: 10.1080/17470919.2016.1273851.
[2] J. Cawood. (2021). *A Limited Literature Review of Influence and Persuasion*. [Online]. Available: http://dx.doi.org/10/gnrtt9
[3] (2018). *Information Technology—Information Security—Information Assurance|ISACA*. Accessed: Mar. 11, 2018. [Online]. Available: https://www.isaca.org/pages/default.aspx
[4] N. Tsinganos, G. Sakellariou, P. Fouliras, and I. Mavridis, "Towards an automated recognition system for chat-based social engineering attacks in enterprise environments," in *Proc. 13th Int. Conf. Availability, Rel. Secur.*, 2018, p. 10.
[5] R. B. Cialdini, *Influence, New and Expanded: The Psychology of Persuasion*. New York, NY, USA: HarperCollins, 2021.
[6] R. M. Ross, D. G. Rand, and G. Pennycook, "Beyond 'fake news': Analytic thinking and the detection of false and hyperpartisan news headlines," *Judgment Decis. Making*, vol. 16, no. 2, pp. 484–504, 2021.
[7] Z. Wang, Y. Ren, H. Zhu, and L. Sun, "Threat detection for general social engineering attack using machine learning techniques," 2022, *arXiv:2203.07933*.
[8] M. A. Siddiqi, W. Pak, and M. A. Siddiqi, "A study on the psychology of social engineering-based cyberattacks and existing countermeasures," *Appl. Sci.*, vol. 12, no. 12, p. 6042, Jun. 2022, doi: 10.3390/app12126042.
[9] H. Shi, M. Silva, D. Capecci, L. Giovanini, L. Czech, J. Fernandes, and D. Oliveira, "Lumen: A machine learning framework to expose influence cues in text," 2021, *arXiv:2107.10655*.
[10] S. Duerr, K. T. Lange, and P. A. Gloor, "What makes a message persuasive? Identifying adaptations towards persuasiveness in nine exploratory case studies," 2021, *arXiv:2104.12454*.
[11] D. Dimitrov, B. Bin Ali, S. Shaar, F. Alam, F. Silvestri, H. Firooz, P. Nakov, and G. Da San Martino, "SemEval-2021 task 6: Detection of persuasion techniques in texts and images," 2021, *arXiv:2105.09284*.
[12] X. Wang, W. Shi, R. Kim, Y. Oh, S. Yang, J. Zhang, and Z. Yu, "Persuasion for good: Towards a personalized persuasive dialogue system for social good," 2020, *arXiv:1906.06725*.
[13] N. E. D. Ferreyra, E. Aïmeur, H. Hage, M. Heisel, and C. G. Van Hoogstraten, "Persuasion meets AI: Ethical considerations for the design of social engineering countermeasures," in *Proc. 12th Int. Joint Conf. Knowl. Discovery, Knowl. Eng. Knowl. Manage.*, 2020, pp. 204–211, doi: 10.5220/0010142402040211.
[14] H. Chen, D. Ghosal, N. Majumder, A. Hussain, and S. Poria, "Persuasive dialogue understanding: The baselines and negative results," 2020, *arXiv:2011.09954*. Accessed: Dec. 08, 2021.
[15] A. Yasin, R. Fatima, L. Liu, A. Yasin, and J. Wang, "Contemplating social engineering studies and attack scenarios: A review study," *Secur. Privacy*, vol. 2, no. 4, p. e73, Jul. 2019, doi: 10.1002/spy2.73.
[16] D. Yang, J. Chen, Z. Yang, D. Jurafsky, and E. Hovy, "Let's make your request more persuasive: Modeling persuasive strategies via semi-supervised neural nets on crowdfunding platforms," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, Minneapolis, MN, USA, 2019, pp. 3620–3630, doi: 10.18653/v1/N19-1364.
[17] M. Lansley, F. Mouton, S. Kapetanakis, and N. Polatidis, "SEADer++: Social engineering attack detection in online environments using machine learning," *J. Inf. Telecommun.*, vol. 4, no. 3, pp. 1–17, Apr. 2020, doi: 10.1080/24751839.2020.1747001.
[18] S. Dutta, D. Das, and T. Chakraborty, "Changing views: Persuasion modeling and argument extraction from online discussions," 2019, *arXiv:1907.06076*.
[19] C. Hidey and K. McKeown, "Persuasive influence detection: The role of argument sequencing," in *Proc. AAAI Conf. Artif. Intell.* vol. 8, 2018, pp. 1–8.
[20] J.-W. H. Bullée, L. Montoya, W. Pieters, M. Junger, and P. Hartel, "On the anatomy of social engineering attacks—A literature-based dissection of successful attacks: On the anatomy of social engineering attacks," *J. Investigative Psychol. Offender Profiling*, vol. 15, no. 1, pp. 20–45, Jan. 2018, doi: 10.1002/jip.1482.
[21] R. R. Iyer, K. Sycara, and Y. Li, "Detecting type of persuasion: Is there structure in persuasion tactics?" in *Proc. ACM Conf.* vol. 10, 2017, pp. 1–10.
[22] M. J. Edwards, C. Peersman, and A. Rashid, "Scamming the scammers: Towards automatic detection of persuasion in advance fee frauds," in *Proc. 26th Int. Conf. World Wide Web Companion*, Perth, WA, Australia, Apr. 2017, pp. 1291–1299, doi: 10.1145/3041021.3053889.
[23] P. Anand, J. King, J. Boyd-Graber, E. Wagner, C. Martell, D. Oard, and P. Resnik, "Believe me—We can do this! Annotating persuasive acts in blog text," in *Proc. 25th AAAI Conf. Artif. Intell.* 2011, pp. 1–5.
[24] J.-W.-H. Bullée, L. Montoya, W. Pieters, M. Junger, and P. H. Hartel, "The persuasion and security awareness experiment: Reducing the success of social engineering attacks," *J. Exp. Criminol.*, vol. 11, no. 1, pp. 97–115, Mar. 2015, doi: 10.1007/s11292-014-9222-7.
[25] R. Levine, *The Power of Persuasion: How we're Bought and Sold*, vol. 1. 2015.
[26] P. Ortiz, "Machine learning techniques for persuasion dectection in conversation," thesis, Nav. Postgraduate School, Monterey, CA, USA, 2010. Accessed: Mar. 31, 2021. [Online]. Available: https://calhoun.nps.edu/handle/10945/5257
[27] H. Handoko, D. A. W. Putri, and I. Revita, "The language of social engineering: From persuasion to deception," *Lang. Civilization*, vol. 7, pp. 136–142, Aug. 2015.

[28] D. Weirich and M. A. Sasse, "Pretty good persuasion: A first step towards effective password security in the real world," in *Proc. Workshop New Secur. Paradigms*, 2001, pp. 137–143, doi: 10.1145/508171.508195.

[29] J. Young, C. Martell, H. T. Gilbert, P. Anand, and P. Ortiz. (Aug. 2011). *A Microtext Corpus for Persuasion Detection in Dialog*. [Online]. Available: http://hdl.handle.net/10945/46794

[30] B. H. Schell and C. Martin, *Webster's New World Hacker Dictionary*. Indianapolis, IN, USA: Wiley, 2006.

[31] *ENISA—Glossary*. ENISA. Accessed: Aug. 21, 2022. [Online]. Available: https://www.enisa.europa.eu/topics/csirts-in-europe/glossary

[32] K. Chetioui, B. Bah, A. O. Alami, and A. Bahnasse, "Overview of social engineering attacks on social networks," *Proc. Comput. Sci.*, vol. 198, pp. 656–661, Jan. 2022, doi: 10.1016/j.procs.2021.12.302.

[33] R. Heartfield and G. Loukas, "A taxonomy of attacks and a survey of defence mechanisms for semantic social engineering attacks," *ACM Comput. Surv.*, vol. 48, no. 3, pp. 1–39, Feb. 2016, doi: 10.1145/2835375.

[34] A. Kumar, M. Chaudhary, and N. Kumar, "Social engineering threats and awareness: A survey," *Eur. J. Adv. Eng. Technol.*, vol. 2, no. 11, pp. 15–19, 2015.

[35] T. Longtchi, R. Montañez Rodriguez, L. Al-Shawaf, A. Atyabi, and S. Xu, "Internet-based social engineering attacks, defenses and psychology: A survey," 2022, *arXiv:2203.08302*.

[36] F. Salahdine and N. Kaabouch, "Social engineering attacks: A survey," *Future Internet*, vol. 11, no. 4, p. 89, 2019, doi: 10.3390/fi11040089.

[37] W. Syafitri, Z. Shukur, U. A. Mokhtar, R. Sulaiman, and M. A. Ibrahim, "Social engineering attacks prevention: A systematic literature review," *IEEE Access*, vol. 10, pp. 39325–39343, 2022, doi: 10.1109/ACCESS.2022.3162594.

[38] T. Li, X. Wang, and Y. Ni, "Aligning social concerns with information system security: A fundamental ontology for social engineering," *Inf. Syst.*, vol. 104, Feb. 2022, Art. no. 101699, doi: 10.1016/j.is.2020.101699.

[39] R. B. Cialdini and N. J. Goldstein, "The science and practice of persuasion," *Cornell Hotel Restaurant Admin. Quart.*, vol. 43, no. 2, pp. 40–50, 2002.

[40] A. J. Resnik and R. B. Cialdini, "Influence: Science & practice," *J. Mark. Res.*, vol. 23, no. 3, p. 305, 1986, doi: 10/crkvr5.

[41] N. Tsinganos and I. Mavridis, "Building and evaluating an annotated corpus for automated recognition of chat-based social engineering attacks," *Appl. Sci.*, vol. 11, no. 22, p. 10871, 2021, doi: 10.3390/app112210871.

[42] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998, doi: 10.1109/5.726791.

[43] Y. Kim, "Convolutional neural networks for sentence classification," 2014, *arXiv:1408.5882*.

[44] A. Torfi, R. A. Shirvani, Y. Keneshloo, N. Tavaf, and E. A. Fox, "Natural language processing advancements by deep learning: A survey," 2020, *arXiv:2003.01200*.

[45] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for text classification," 2016, *arXiv:1606.01781*.

[46] Y. Zhang and B. Wallace, "A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification," 2015, *arXiv:1510.03820*.

[47] C. Dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *Proc. COLING, 25th Int. Conf. Comput. Linguistics, Tech. Papers*, Aug. 2014, pp. 69–78.

[48] R. Johnson and T. Zhang, "Semi-supervised convolutional neural networks for text categorization via region embedding," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, 2016, pp. 1–9.

[49] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin, "Advances in pre-training distributed word representations," 2017, *arXiv:1712.09405*.

[50] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, 2013, pp. 3111–3119. Accessed: Mar. 3, 2019. [Online]. Available: http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

[51] A. Fred Agarap, "Deep learning using rectified linear units (ReLU)," 2018, *arXiv:1803.08375*.

[52] Y. Goldberg, "A primer on neural network models for natural language processing," *J. Artif. Intell. Res.*, vol. 57, pp. 345–420, Nov. 2016, doi: 10.1613/jair.4992.

[53] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, and A. Desmaison, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019, pp. 8024–8035.

[54] M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. Liu, M. Peters, M. Schmitz, and L. Zettlemoyer, "AllenNLP: A deep semantic natural language processing platform," 2017, *arXiv:1803.07640*.

[55] *PyTorch Lightning*. Accessed: Jan. 13, 2022. [Online]. Available: https://www.pytorchlightning.ai/

[56] L. Biewald. (2020). *Experiment Tracking With Weights and Biases*. [Online]. Available: https://www.wandb.com/

[57] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

NIKOLAOS TSINGANOS received the B.Sc. degree in computer science and the M.Sc. degree in pervasive computing systems engineering from Hellenic Open University. He is currently a Doctoral Assistant at the Department of Applied Informatics, University of Macedonia (UoM), and a member of the Multimedia, Security, and Networking Laboratory (MSNLab), InfoSec Research Group. He has participated in several international and nationally-funded research and development projects. His current research interests include the design and performance evaluation of cyber defense mechanisms, particularly in the context of social engineering attack recognition.

IOANNIS MAVRIDIS received the Diploma degree in computer engineering and the M.A. degree from the University of Patras, Greece, and the Ph.D. degree in information systems security from the Aristotle University of Thessaloniki, Greece. He is currently a Professor in information security with the Department of Applied Informatics, University of Macedonia, Greece. He is also working as the Director of the Multimedia, Security & Networking (MSN) Laboratory. He serves as an Area Editor (*Journal of Cyber Security*) of the *Array* (Elsevier). He has published more than 100 articles in journals and conferences. He is the author or coauthor of three books on information security. He has participated as a principal investigator and a researcher for several international and nationally funded research and development projects. His current research interests include cybersecurity education on risk management, access control, cyber threat intelligence, digital forensics, and security economics.

DIMITRIS GRITZALIS received the B.Sc. degree in mathematics from the University of Patras, Greece, the M.Sc. degree in computer science from The City University of New York, USA, and the Ph.D. degree in information systems security from the University of the Aegean, Greece. He is currently a Professor in cybersecurity with the Department of Informatics, Athens University of Economics and Business, Greece, where he is also working as the Director of the Master's Program in Information Systems Security and Development and the Director of the INFOSEC Laboratory. He is the Academic Editor of the *Computers & Security* journal (Elsevier) and the Scientific Editor of the *International Journal of Critical Infrastructure Protection* (Elsevier). He has served as an Associate Rector for *Research*, the President of the Greek Computer Society, and an Associate Data Protection Commissioner of Greece. He has published more than 200 articles in academic journals and conferences. His current research interests include critical infrastructure protection, resilience, risk assessment, security education, and advanced persistent threats.